Geographic Vector Agents

from Pixels to Intelligent Processing Units

by

Kambiz Borna

A thesis submitted for the degree of
Doctor of Philosophy

The School of Surveying
University of Otago
Dunedin, New Zealand

March 2016

List of publications

Journal:

Borna, K., Moore, A. B., and Sirguey, P. (2014), 'Towards a vector agent modelling approach for remote sensing image classification'. *Journal of Spatial Science*, 59(2), 283-296.

Borna, K., Moore, A. B., and Sirguey, P. (2016), 'An Intelligent Geospatial Processing Unit for Image Classification Based on Geographic Vector Agents (GVAs) ', *Transactions in GIS*, 20, 368-381.

Conference:

Borna, K., Sirguey, P., and Moore, A. (2013), Geographical Vector Agent Modelling for Image Classification: Initial Development, *GIS and Remote Sensing Research Conference*, University of Otago, Dunedin, New Zealand.

Borna, K., Moore, A., and Sirguey, P. (2014), A Vector Agent Approach to Extract the Boundaries of Real-World Phenomena from Satellite Images, *proceedings of R@Locate14*, Canberra, Australia, pp. 142-144.

Borna, K., Moore, A. and Sirguey, P. (2014), Geographical Vector Agents for Supervised Image Classification, *GeoCart'2014 Conference*, University of Auckland, New Zealand.

Borna, K., Sirguey, P., and Moore, A. (2014), Automatic Extraction of Roof Structure from Urban Aerial Images and LiDAR data using Geographical Vector Agent, *GeoCart'2014 Conference*, University of Auckland, New Zealand.

Borna, K., Moore, A. B., and Sirguey, P. (2015), A Vector Agent-Based Unsupervised Image Classification for High Spatial Resolution Satellite Imagery, *GeoComputation 2015 Conference*, University of Texas, Dallas, pp.349-354.

Borna, K., Sirguey, P., and Moore, A. B. (2015). An intelligent vector agent processing unit for geographic object-based image analysis. *In Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International*, pp. 3053-3056.

Borna, K., Moore, A., and Sirguey, P. (2015), Automatic Extraction of Lakes from Satellite Imagery Using Vector Agents, *NZ Geospatial Research Conference*, University of Canterbury, Christchurch.

Abstract

Spatial modelling methods usually utilise pixels and image objects as the fundamental processing unit to address real-world objects (geo-objects) in image space. To do this, both pixel-based and object-based approaches typically employ a linear two-staged workflow of segmentation and classification. Pixel-based methods often segment a classified image to address geo-objects in image space. In contrast, object-based approaches classify a segmented image to determine geo-objects. These methods lack the ability to simultaneously integrate the geometry and theme of geo-objects in image space.

This thesis explores Vector Agents (VA) as an automated and intelligent processing unit to directly address real-world objects in the image space. A VA, is an object that can represent (non)dynamic and (ir)regular vector boundaries (Moore, 2011; Hammam et al., 2007). This aim is achieved by modelling geometry, state, and temporal changes of geo-objects in spatial space.

To reach this aim, we first defined and formulated the main components of the VA, including geometry, state and neighbourhood, and their respective rules in accordance with the properties of raster datasets (e.g. satellite images), as representation of a geographical space (the Earth). The geometry of the VA was formulated according to a directional planar graph that includes a set of spatial reasoning relationships and geometric operators, in order to implement a set of dynamic geometric behaviours, such as growing, joining or splitting. Transition rules were defined by using a classifier (e.g. Support Vector Machines (SVMs)), a set of image analysis operators (e.g. edge detection, median filter), and the characteristics of the objects in real world. VAs used the transition rules in order to find and update their states in image space. The proximity between VAs was explicitly formulated according to the minimum distance between VAs in image space. These components were then used to model the main elements of our software agent (e.g. geo-objects), namely sensors, effectors, states, rules and strategies. These elements allow a VA to perceive its environment, change its geometry and interact with other VAs to evolve in consistency together with their thematic meaning. It also enables VAs to adjust their thematic meaning based on changes in their own attributes and those of their neighbours.

We then tested this concept by using the VA to extract geo-objects from different types of raster datasets (e.g. multispectral and hyperspectral images). The results of the VA model confirmed that: (a) The VA is flexible enough to integrate thematic and geometric components of geo-objects in order to extract them directly from image space, and (b) The VA has sufficient capability to be applied in different areas of image analysis. We discuss the limitations of this work and present the possible solutions in the last chapter.

Acknowledgements

I would like to express my gratitude and appreciation to my first advisor Antoni Moore for his full support, patience, encouragement and immense knowledge. Without his guidance and support, the completion of this thesis would not have been possible. I am also grateful to my second supervisor, Pascal Sirguey, for his continuous support, understanding and insightful comments. His guidance helped me move through each stage of research and writing of this thesis. It was an honour and pleasure to work under their supervision. kind advises

My sincere thanks goes to Christina Hulbe, David Goodwin, Gregory Leonard and Colin O'Byrne for their support. I also want to thank my friends, Joseph Wright and Yong Chien Zheng, in the school of surveying for their kind advice. I would like to thank to Fiona Webster and Marg Newall for their attention and support during the past four years.

I would like to thank the Belgian Royal Military Academy for acquiring and providing the LiDAR DSM dataset used in this study, and the IEEE GRSS Image Analysis and Data Fusion Technical Committee for making these data available for testing. I also want to thank to DigitalGlobe Foundation for providing the remote sensing datasets of Dunedin city.

I would like to thank my mother and my brothers for their help, support and encouragement. I would not have been able to complete this study without their support and encouragement.

Contents

List of Figures

List of Tables

List of abbreviations

CA	Cellular Automata
DN	Digital Number
DSM	Digital Surface Model
FP	False Positive
FN	False Negative
GA	Geographic Automata
GAS	Geographic Automata System
GEOBIA	Geographic Object-Based Image Analysis
GIS	Geographic Information Systems
HSR	High Spatial Resolution
IOA	Image Object Agent
LiDAR	Light Detection and Ranging
MAS	Multi Agent System
ML	Maximum Likelihood
NDVI	Normalized Difference Vegetation Index
RS	Remote Sensing
SVM	Support Vector Machines
TP	True Positive
UML	Unified Modelling Language
VA	Vector Agent
VHR	Very High Resolution

In the name of the Lord of life and wisdom

(Ferdowsi)

Chapter One

Introduction

Abstract

In this chapter, we first explain remote sensing and its main components. After that, the notion of the image classification approach is explained. Then, the main issues related to automatic image classification will be discussed. The general motivations and objectives of this work are presented in the next section. Finally, the structure and organisation of this thesis is described.

1.1. Overview of remote sensing

The term 'Remote Sensing' (RS) refers to the technology of *acquiring* and *interpreting* information about an object or phenomenon without making physical contact with said object (Richards, 2006). Specifically, the term refers to the use of sensors on board airborne (e.g. aircraft) or space-borne (e.g. satellite) platforms to acquire data from objects or phenomena on the earth's surface and in the atmosphere. There are two types of sensors, active and passive, that are differentiated based on the source of energy they utilise (Richards, 2006). Both sensor types normally use electromagnetic (EM) radiation to acquire the information of an object on the earth.

Active sensors produce their own electromagnetic radiation. A man-made source of energy produced on board the sensor platform is sent towards an intended target. The quantity of energy reflected or scattered back and the time delay between emission and reception is then measured. Examples of common active sensors are Radar systems (Radio Detection and Ranging) and LiDAR instruments. For instance, in LiDAR instruments, a sensor measures the time between the transmitted and backscattered pulses of a laser light as it hits a target surface and returns.

In contrast, a passive system uses a natural source of energy usually originating from the sun or the earth. In this case, passive sensors measure the energy reflected, scattered or emitted from the earth. The measurement is usually performed over an elementary area

known as a 'pixel' (which defines the spatial resolution) in different frequency bands (which determine the spectral resolution) at a certain point in time (which defines the temporal resolution) (Gao, 2008). These measurements are then converted into electrical signals and recorded as a digital image. Such digital imagery provides a considerably wider range of information (e.g. textural) compared to the traditional methods (e.g. analogue images).

The process of extracting meaningful information from these digital images can be accomplished with the aid of human interpretation or computer algorithms. In the former case, a visual analysis of the image is performed by a human expert and interpretations made based on their personal knowledge (e.g. identifying and distinguishing a river from a lake). Manual interpretation is often limited to analysing one image at a time due to the difficulty in viewing multiple images at once. Since there is such a large quantity of content generated by multispectral and hyperspectral satellite data, along with the wide variety of sensors available, manual interpretation can be a tedious and time-consuming process. Moreover, results can be inconsistent due to the varying perspectives of each interpreter.

Computer algorithms can also be used to extract meaningful information from digital images. In this context, digital image processing is based on the manipulation of digital numbers by a computer and is thus more objective, generally resulting in more consistent results. The concept of digital image processing encompasses all the techniques used to extract meaningful information from the digital image. It consists of image correction, image enhancement, image transformation and image classification. This thesis focuses on image classification based on EM radiation, primarily in the visible/infrared spectrum, obtained from airborne/satellite remote sensing.

1.2. Image classification

Image classification is a process that groups a set of pixels into a number of categories of ground cover classes. A classification can be performed based on statistical decision rules in the multispectral domain, known either as spectral pattern recognition, or decision rules in the spatial domain, called spatial pattern recognition (Gao, 2008).

In the former case, the pixel-based approach of image classification (supervised or unsupervised) is based solely on the spectral values of pixels in the feature space. In this context, it can be difficult for conventional pixel-based approaches to differentiate between

classes that have similar spectral signature but different semantic meaning (e.g. lakes and rivers in a multispectral dataset).

In the latter case, image classification methods can apply the decision rules based on spatial information (e.g. geometry, size). To implement these rules, spatial image classification methods, such as GEOBIA, use image objects in image space instead of single pixels in multispectral space in order to classify an image. According to Hay and Castilla (2008):

> *"Geographic Object-Based Image Analysis (GEOBIA) is a sub-discipline of Geographic Information Science (GIScience) devoted to developing* ***automated methods*** *to partition remote sensing imagery* ***into meaningful image-objects****, and assessing their characteristics through spatial, spectral and temporal scales, so as to generate new geographic information in GIS-ready format."*

An image object is a group of connected pixels that is internally coherent and collectively different from its surroundings (Castilla and Hay, 2008), even if the collection of pixels corresponding to the object is heterogeneous. They are initially produced by segmentation (with or without the application of multi-scale characteristics) prior to classification (Hay et al., 2005).

Although image objects do not necessarily correspond to geographical entities, they can provide more semantic information than the spectral content, which is the sole input of pixel-based approaches. For example, textural information, such as homogeneity, similarity and contrast, as well as morphological information, such as geometry, shape and compactness, is applied in GEOBIA to support and improve the modelling process (Tian and Chen, 2007; Hay et al., 2005; Benz et al., 2004). Thus, the results of the object-based approaches are more reliable compared to the traditional per-pixel classifiers, especially when VHR images are used (VHR image, pixel size <5m) (Blaschke, 2010; Navulur, 2006; Blaschke et al., 2000).

1.2.1. VHR image and image classification

The recent abundance of high spatial resolution remote sensing data has great potential, but volume, complexity and automation challenges first need to be overcome (Baatz et al., 2008; Benz et al., 2004). One challenge is the image objects' variability at the super-pixel scale (Li et al., 2012) but GEOBIA is adaptable enough to potentially overcome the limitations of the uniform pixel unit in this regard (Lu and Weng, 2007; Baatz et al., 2008; Walter 2004; Gitas et al., 2004; Benz et al., 2004; Thomas et al., 2003). Additionally, object-based output

facilitates greater integration with vector-based geographic information systems (GIS) in comparison to pixel-based approaches (Schiewe et al., 2001). Because of these abilities, the use of GEOBIA has become popular in the past decade. As a consequence of such a success, some authors have commented that *GEOBIA* should be considered as *a new paradigm* for image classification (Blaschke et al., 2014; Hay and Castilla, 2008).

1.2.2. Limitations of the GEOBIA approach

Despite the advantages offered by the GEOBIA approach such as geographical objects with a unified identity, the results of the image classification step strongly depend on the quality of the segmentation process. This is heavily influenced by the parameters specified by the operator (e.g. scale or colour weight) for which the input image is segmented (Gao, 2008; Kim et al., 2008; Hay et al., 2005). These parameters include size and spectral homogeneity of image objects (Hay et al., 2005). However, there is no specific rule to determine the optimum value of these parameters, including how much weight they should be given to create a geographically meaningful object (Tian and Chen, 2007).

To deal with these issues, a hierarchical network of image objects segmented at different scales is usually used to address objects with different sizes (Castilla and Hay, 2008; Benz et al., 2004). Despite the advantage of a multi-scale segmentation algorithm, the quality of the segmentation step is subjective and still highly depends on the segmentation parameters (Gao, 2008). Some methods address this issue by automatically adjusting the segmentation parameters, such as Estimation of Scale Parameter (ESP) tool (Dragut et al., 2010) or a genetic algorithm approach (Feitosa et al., 2006). However, these methods only work well for certain desired object classes (e.g. homogenous objects). On the other hand, the concept of the scale and hierarchical structure between extracted objects at different scales is not clear (Hay and Castilla, 2008). In this context, formalising expert knowledge and encapsulating it into rule sets would be a time-consuming process (Mahmoudi et al., 2013), especially when using criteria such as thresholds (Baatz et al., 2000).

Once image objects are created, the GEOBIA approach uses the information of segmented objects via a set of rules in image space. In other words, the GEOBIA approach uses a sequential structure of segmentation and classification processes to classify objects in image space. In this sense, the classical approach of GEOBIA lacks the ability to take full advantage of other information that is available for segmentation and classification. This information includes:

1) thematic meaning (including vague phenomena);
2) geometric character of a given class (shape, size, boundary complexity);
3) neighbour relationships;
4) scale-based limits of the object being created;
5) the prevalent case of incomplete information (e.g. vague boundaries);
6) the procedural knowledge being generated during this process.

To address this issue, Baatz et al. (2008) proposed an object-oriented workflow whereby the object primitives are created through a segmentation process (Figure 1.1).

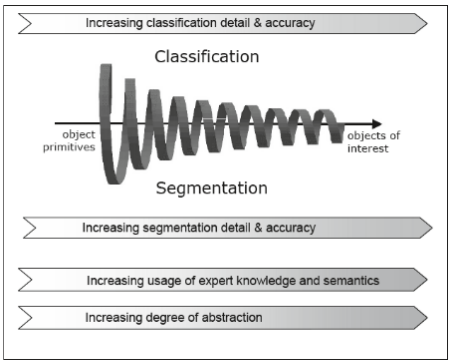


Figure 1.1. Object-oriented workflow: the generic procedure (from Baatz et al., 2008).

In contrast to the object-based workflow, the object-oriented approach employs these objects not only as information carriers but also as building blocks for any further shape modification, *merging* or *segmentation procedures*. As the analysis progresses, more expert knowledge and domain knowledge can be used to address the object of interest (Baatz et al., 2008). Blaschke et al. (2014) proposed an advanced workflow in which a cycle of classification and segmentation can be applied (Figure 1.2).

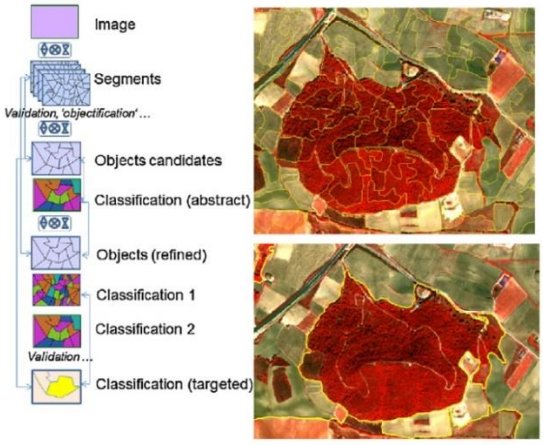


Figure 1.2. Idealised GEOBIA workflow proposed by Blaschke et al. (2014), which illustrates how classified objects are extracted through a cycle of segmentation and classification.

In Figure 1.2, the term 'objectification' refers to the integrated spatial and thematic object definition (Blaschke et al., 2014). In this case, the results of a classification process are used to take advantage of the domain-dependent knowledge of the real-world objects (Figure 1.2), where the *geometry of objects* is determined via *a multiscale segmentation* process.

To enable a dynamic geometry for image objects, Hofmann et al. (2015) used agents to give power to the image objects in order to change their geometry (Figure 1.3). Here, after *initial segmentation and classification* steps, objects can *re-segment* themselves or *merge* with a neighbouring Image Object Agent (IOA) during the classification process based on the rules formulated on the characteristics of 3D roofs (e.g. slope).

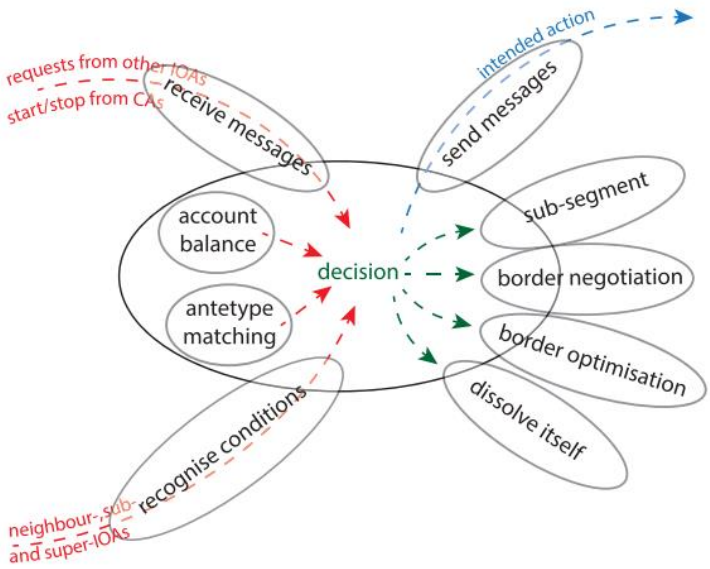


Figure 1.3. Image Object Agent (IOA) for image classification: the IOA can send messages and change its own shape (from Hofmann et al., 2015).

Despite the advantages of the proposed methods, such as contextual information, these methods still rely on a *geometry* formulated through *a segmentation process*. In other words, the geometric and thematic 'states' of real-world objects are determined separately. This points to an implicit assumption that real-world objects have predictable behaviours in terms of their geometry. Thus, a function exists (e.g. Equation 2.7 in Chapter 2) to determine the initial geometry of real-world objects. In this case, every object merged against the homogeneity criteria leads to a meaningful object, whereas there is no unique solution for image segmentation (Castilla and Hay, 2008).

Considering the above discussion, an unanswered question needs to be addressed: If geographic objects or ***geo-objects***– "*a bounded geographic region identified for a period of*

time as the real-world object of a geographic term" (Castilla, 2003) – are supposed to be a basis for the GEOBIA processes (Blaschke, 2014; Castilla and Hay, 2008), ***how can we dynamically link the segmentation and thematic meaning in a unified classification process?***

Although these approaches allow the geo-objects to change their geometry during the classification process, geometric changes only include *re-segmenting* or *merging* of image objects. In other words, image objects lack the ability to tune their geometry at pixel level in the classification step (e.g. absorbing or removing a pixel). *These methods often ignore the local geometric changes between objects*. In these cases, the interactions between objects are only performed at the object level by using geometric operators, such as merging or splitting. Since these operators cannot support the interaction between geo-objects with indeterminate boundaries at pixel level, their boundaries cannot be directly sampled. Hence these approaches assume that objects in the real world have crisp boundaries.

1.3. Problem statement

From the above, it can be concluded that *both processing units, namely pixels and image objects, lack the necessary abilities to simultaneously model the geometry and thematic meaning of real-world objects in image space*. Although image objects provide more information compared to pixels, it is not always possible to link image objects to real-world objects due to the absence of semantics.

This research attempts to address this limitation by developing the VA model in image space. A VA is an automated processing unit that has the ability to intelligently control and alter its shape and attributes in order to evolve in accordance with the nature of the phenomena being represented (Moore, 2011; Hammam et al., 2007). Each VA has seven components, that enables it dynamically change its state, geometry, neighbourhood and the associated rules (Moore, 2011; Hammam et al., 2007). Within this setting, a geo-object can simultaneously identify its geometry and state, and directly interact with its environment and also other geo-objects. In contrast with the VA model, pixel-based and object-based approaches utilise a sequence of segmentation and classification processes (or vice versa) to extract geo-objects from image space (see Section 2.2.3 and 2.3.2).

In this way, an image classification method is enabled to directly identify geo-objects in image space. In this kind of classification, the agents are closely coupled with their

corresponding objects in the real-world in both representation and behaviour (see Section 2.10.2 and 3.1), and the model is more understandable than pixel-based and object-based approaches.

1.4. Research questions

To achieve the above aim, the main research questions addressed by this thesis are:

1) What is the most suitable dynamic geometric data structure that would allow the VA model to represent real world phenomena captured in a raster image?

2) How can the VA be parameterised to find and update their thematic meaning of real world objects based on an elastic geometry?

3) How can VAs interact with each other?

4) How can the VA control geometry, state and neighbourhood relations, and evolve over time in image space?

5) What are the advantages of using the VA model for image analysis?

1.5. Research objectives

Considering the above research questions, the main objectives of this research are as follows:

The first objective of this thesis is to develop a generic structure based on VAs that can address real-world objects in image space. This will address the following issues:

i. Representing the geometry of real-world objects, as well as the rules and methods for evolving and expressing this geometry in the modelling space.

ii. To analyse and explore transition rules and their effect on the class of real-world objects, and how the VA uses those transition rules to identify their classes.

iii. To formulate the interactions between VAs in the modelling environment based on neighbourhood rules.

The challenge in integrating these components was resolved through a flexible agent architecture in which each component was implemented with various classes and sub-classes (see Chapter 3).

The second objective of this thesis is to validate this model in comparison with per-pixel classifiers in terms of classification accuracy achievable. To validate the proposed model against pixel-based classification types, which are mainly unsupervised and supervised approaches, the VAs have been tested for the following applications in image analysis:

i. An unsupervised image classification: The ability of VAs in implementing a self-training algorithm for an unsupervised image classification is evaluated (see Chapter 4).

ii. A supervised image classification: The ability of VAs in creating reliable training samples is evaluated (see Chapter 5).

The third objective of this thesis is to assess the capabilities of the VA model in addressing the main limitations of the GEOBIA approach. VAs have been tested for the following applications in image analysis:

i. To implement an object-based approach to classify a satellite image. The aim of this research is to show the capability of VAs, as an automated processing unit, in addressing the main issues of the conventional GEOBIA method, including geographic objects (see Chapter 6).

ii. To test how satisfactory the model simulation outputs are in terms of their accuracy and the quality of the classification maps when compared with a conventional GEOBIA approach (see Chapter 6).

The fourth objective of this thesis is to explore the proposed VA model and its ability to extract and identify real-world objects from raster datasets in a specific area. In this context, this study will assess the ability of the VA to identify, extract and classify 3D roofs from LiDAR datasets (see Chapter 7).

1.6. Motivation and approach

In order to achieve the noted objectives in the above section, for the first time, this thesis proposes a new dynamic geometry based on the VA model to directly extract geo-objects from image space. This geometry allows geo-objects to automatically change their shape and affect the geometry of each other in image space over time. The use of this geometry along with the transition rules enables the geo-objects to simultaneously find and extract their attributes and geometry in image space. This approach is in contrast to GEOBIA

approaches that utilise an iterative process of segmentation and classification to address geometry of geo-objects in image space.

These agents are also taught how to interact with each other and with their environment, thus enabling the concept of image interpretation by association. In a VA context, the main contributions of this research can be summarised as follows, using an autonomous object to:

i. expand geometric possibilities to model indeterminate boundaries.

ii. affect the definition of states through geometry (boundary character), and

iii. create the potential for joining an object's state with that of its neighbours to create a higher-level state definition that is semantically meaningful (one that is also capable of containing the original states as internal structure).

In the area of image classification, the proposed VA model contributes in two ways: intelligent spatial classification and spectral classification.

1.6.1. Spatial classification

Conventional object-based approaches use the image object to classify an image. At a fundamental level, image objects are a set of regular/irregular polygons based on a collection of user-defined parameters (e.g. scale) in image space. In this sense, two main assumptions are implicit within the conventional object-based approaches. First, in the classification step, the process of merging image objects leads to a new meaningful object. To tackle this limitation, geographical image objects can use a dynamic geometry that allows them to constantly change their geometry during the classification step. The second assumption is that geographical image objects have a crisp boundary. To cope with this issue, the dynamic geometry should also be flexible. In this sense, geographic image objects can change their geometry at pixel level even where there is incomplete spatial information or vague boundaries. Chapters 3, 6 and 7 of this thesis explore the VA as a way to support a dynamic and flexible geometry in close association with the shape, size and attributes of the corresponding class to which they are expected to belong.

1.6.2. Spectral classification

Conventional pixel-based approaches rely solely on the spectral values of pixels in order to classify an image in a multispectral domain through a set of statistical rules. In this context,

there is an implicit assumption that all the classes of interest to be mapped have a unique distribution of values in all the multispectral bands used (Gao, 2008). The dynamic structure of the VA allows them to address spectral patterns in a multi spectral domain in close association with spatial patterns. In this way, the VA can find the class of each pixel based on the characteristics of real-world objects (e.g. geometry, texture) during an evolutionary process in image space. Chapters 4 and 5 explore these aspects.

1.7. Organisation of the thesis

The next chapter provides an overview of the conventional methods of spatial modelling to capture real-world objects in an image. The concept of set theory and graph theory are then reviewed to formulate the geometry of the VA. To address the class of the VA in image space, the concept of thematic meaning of spatial objects is explored. To integrate the geometric and thematic definition of spatial objects, a review of object-oriented approaches is then presented. The following section explores the key concepts of time. The characteristics of a self-organising system for spatial modelling will be studied by reviewing the techniques of the geosimulation, GAS and VAs. Finally, Chapter 2 concludes by stating the basis for the work which has been developed in the context of this thesis.

Chapter 3 presents the concept of vector agent modelling for remote sensing image analysis. After that, the main components of the VA in the context of image classification are formulated. The framework to model the relationship between geometry, transition (i.e. of states, or attributes) and neighbourhood rules will be introduced. Finally, the implementation architecture of the proposed model is described through UML.

In Chapter 4, the application of VAs will be discussed within the context of pixel-based approaches. First, unsupervised pixel-based approaches are briefly reviewed. Then the structure of a VA-based unsupervised method is presented. After that, the proposed method is tested and evaluated against the conventional unsupervised method.

Chapter 5 discusses the application of VAs for a class of supervised image classifications called ‘semisupervised image classification’. First, the current semisupervised algorithms are reviewed. Then the structure of a semisupervised VA-based method is defined, formulated and tested. Finally, the results are compared with a conventional semisupervised algorithm.

In Chapter 6, VAs are compared with a conventional object-based approach. First, GEOBIA are reviewed. Then, the limitations of the conventional GEOBIA approaches are discussed. After that, the structure of the VA is defined according to an object-based approach. The proposed VA-based image classification is then tested and evaluated in the last section.

Chapter 7 describes the application of the VAs in extracting real-world objects (e.g. 3D roofs) from raster datasets (e.g. LiDAR). The conventional 3D roof extraction approaches are first reviewed. The elements of the VA are then formulated to extract 3D roofs. The proposed VA-based 3D roof extraction process is then tested and evaluated.

Chapter 8 summarises and discusses the work done in this thesis and presents the results and major findings of the research. It will establish whether these conclusions answer the research questions raised earlier in this chapter. With these concluding remarks, limitations and possible directions for future research are discussed, bringing this thesis to a close.

Chapter Two

Literature review: image Classification, spatial modelling and agents

Abstract

Geo-objects in image space are usually addressed by pixel-based or object-based approaches. These approaches lack the ability to simultaneously address the geometric and thematic components of real-world objects. This chapter reviews spatial modelling techniques in order to develop a new processing unit for extracting directly real-world objects from image space. To construct the geometry of the new processing unit, the taxonomy of vector data structures is first discussed. This chapter then examines the thematic component of the processing unit. Through the review of different computational approaches for the integration the theme and geometry of geo-objects, it is concluded that the vector agent model is an appropriate solution due to its capability in assimilating both thematic and geometric characteristics in order to directly address real-world objects in image space.

2.1. Geo-objects

The real world can be seen as a collection of spatially interacting entities, called geo-objects. The term ‘geo-object’ describes an object which is georeferenced in a coordinate system and/or a geodetic reference system on earth (geo-). The ‘object’ itself refers to an identifiable, relevant (of interest), and describable (has characteristics) entity (Mattos et al., 1993).

By means of the modelling process, geo-objects in a source domain are represented by corresponding objects in the target domain (Worboys, 2004). A geo-object in such a model is usually defined by its spatial (shape and size in real-world), graphical (cartographic form at the generalization level), temporal (when it is created in the real world and in the database)

and textual (attributes) components (Worboys, 2004). In a general context, Figure 2.1 shows the basic components of a geo-object. Here, the geometric aspect relates to the shape and size of the geo-object. It also describes the location and spatial relationships of the geo-objects in real-world. Attribute or non-spatial aspects consist of information that usually describes the thematic properties of geo-objects (e.g. class). From this perspective, a geo-object can be modelled via its components. These components are first modelled for each geo-object. Then, these elements are linked via a unique identification to model a geo-object.

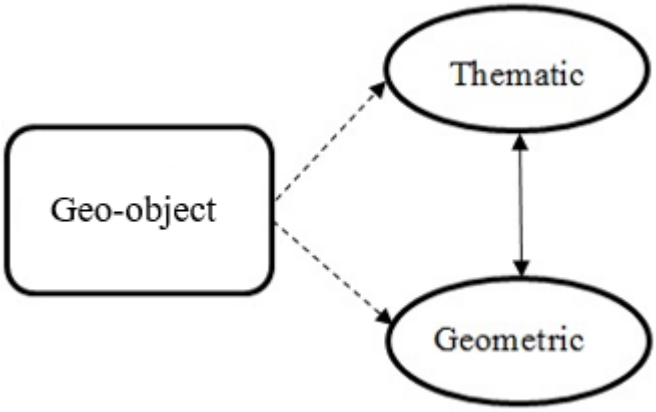


Figure 2.1. Components of geo-objects: geometric data, thematic data, and a link identification (ID) for the geometric and the thematic components (from Abdul-Rahman and Pilouk, 2007).

A geo-object can also be modelled through a primitive form of geographic information. In this method, a geo-object can be modelled via aggregation of fundamental building blocks. For example, to model geo-objects, the GEOBIA approach uses image-objects as the fundamental processing unit. These are a set of regular or irregular polygons formulated based on the parameters specified by a user (Hay and Castilla, 2008), *without direct link to the geo-objects in the real-world*. To define the geo-objects, Goodchild and Cova, (2007) used the concept of the geo-atoms. A geo-atom is associated with a point location in space–time and a property. Based on this concept, a geo-object is defined by aggregation of points in space–time which have specified values for certain properties. In this case, a geo-object can be modelled via the changes of its elements- geometry, state and neighbourhood *in direct connection to the real-world*. In this thesis, the notion of the geo-objects as formalised by Goodchild and Cova (2007) are applied to define geo-objects, more specifically automata types (see Chapter 3).

In remote sensing, the process of abstraction is usually carried out via image classification. ***To model geo-objects in image space***, image analysis methods usually employ image classification process. This process can be performed using pixels or image objects. To do

this, both pixel-based and object-based approaches typically employ a linear two-staged workflow of segmentation and classification. Pixel-based methods segment a classified image to address geo-objects in image space. In contrast, object-based approaches classify a segmented image to determine geo-objects. In the following sections, we review the structure of pixel-based and object-based approaches to extract geo-objects from raster images in more detail.

2.2. Pixel-based approaches

In a pixel-based approach, pixels are the main processing unit used to classify an image. A pixel is the smallest element of an image, and refers to the ground area from which the reflected or emitted electromagnetic radiation is integrated and recorded as a single value in the image (Gao, 2008). In the classification process, the pixel is labelled using the Digital Numbers (DNs), which represents the amount of radiation received at the sensor, based on a set of statistical decision rules in the feature space.

- **Feature space**: This is known as an abstract space in which each pixel is represented as a point in n-dimensional space. Its dimension is determined by the number of features used to describe the patterns within the feature space. Euclidean distance is the shortest length between any two points in a Cartesian space. The dimension of the feature space is specified according to the number of spectral bands used. The spectral distance between two points is often measured in n-dimensional Euclidean space by the following equation (Gao, 2008):

$$d\big((x_1, x_2, x_3, \dots, x_n), (y_1, y_2, y_3, \dots, y_n)\big) = \sqrt{\sum_{i=1}^{n} (\boldsymbol{x_i} - \boldsymbol{y_i})^2}\,, \tag{2.1}$$

where $\boldsymbol{xi}$ and $\boldsymbol{y_i}$ are specified by the digital numbers (DNs) of two pixels corresponding to the spectral band, and n is the number of spectral components of raster datasets (e.g. the number of bands in multispectral image). Such spectral image classification can be carried out through either an unsupervised or supervised approach.

2.2.1. Unsupervised image classification

An unsupervised classification or clustering algorithm uses the DNs of pixels in the feature space to group them into certain categories according to the similarity of their spectral values

(Gao, 2008). In this approach, an analyst usually determines the number of clusters, and then every pixel in the input data is assigned to one of those groups specified by the analyst. Therefore, prior to classification, the image analyst does not need to know about the scene or the thematic meaning of the objects in the real world (Gao, 2008). The classes produced have no thematic meaning. The image analyst labels each cluster after the clustering process is completed. Accordingly, unsupervised methods do not require as much intervention or priori information to classify an image as compared to supervised approaches (Duda et al., 2012; Tso and Olsen, 2005). There are different ways to implement an unsupervised method (e.g. ISODATA, K-means). The K-means algorithm applied for a vector agent-based unsupervised image classification in Chapter 4 is reviewed in more detail below.

- **K-means**

K-means is an iterative algorithm that uses the mean values of DNs in each cluster to classify pixels in the feature space (Figure 2.2). The process is performed as follows (Gao, 2008):

1. Candidates' cluster centres are initialised using the statistical information of the DNs in feature space and the number of clusters (e.g. k) specified by an operator.
2. The Euclidean distance between each pixel and all cluster centres is calculated based on Equation 2.1, where $\boldsymbol{x_i}$ and $\boldsymbol{y_i}$ are specified by the DNs of a candidate pixel and a cluster centre, respectively. A pixel belongs to the candidate cluster to which the spectral distance is shortest.
3. The sum of square error (SSE) is computed as follows:

$$\text{SSE} = \sum_{j=1}^{k} \sum_{i=1}^{t} \left[DN(i,j) - m_j \right]^2 , \tag{2.2}$$

where $DN(i,j)$ is the value of the *i*th pixel in *j*th cluster, m_j is the mean of each cluster and *t* is the number of pixels in each cluster, which varies from cluster to cluster.

4. The class centres coordinates in feature space are updated at each iteration and SSE values are computed.
5. The process terminates if the permitted number of iterations is reached, or the class centres do not change significantly from one iteration to another. An operator determines thresholds.

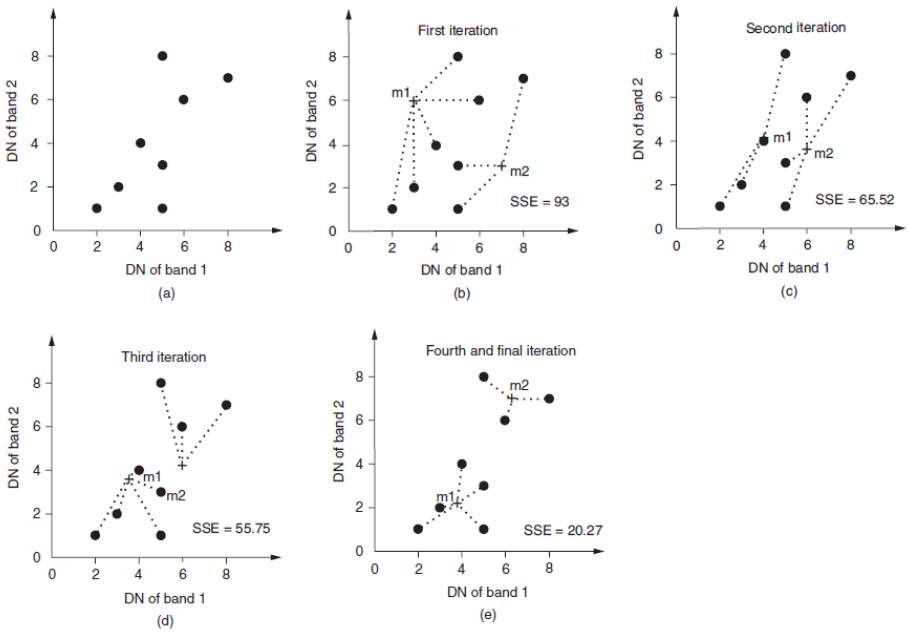


Figure 2.2. (a) K-means clustering in a 2-dimensional feature space. (b) The pixels are classified in the two classes. (c) Clustering results after the pixels are reassessed using the updated cluster centres during the second iteration. (d) Clustering results after the pixels are reassessed using the updated cluster centres during the third iteration. (e) Final clustering results (from Gao, 2008).

After the clustering process, the image analyst determines the thematic class of each cluster. This means that an unsupervised approach can be applied even where the ground truth and ancillary information is not sufficient. Despite the advantage that an unsupervised approach offers, the extracted clusters generally differ from meaningful ground covers identified by the user in the area of the study.

2.2.2. Supervised image classification

A supervised image classification approach uses training samples to identify the class of each cell or pixel in image space. The process of supervised classification is based on the following steps:

1. The first step is the development of a classification scheme in order to determine the thematic classes and a classifier algorithm used to classify the image.
2. In the second step, representative samples for each thematic class and learning process are selected. The selection of training samples is usually performed by ground surveys or by the interpretation of the image. After that, the selected classifier is trained based on the labelled samples.

3. In the final step, a classifier (e.g. Maximum Likelihood (ML)) determines the class of each pixel.

In this thesis, the VA uses the ML (an example of parametric classifier) and Support Vector Machines (SVM) (an example of non-parametric classifier) to identify the class of objects in the real world.

2.2.2.1. Maximum likelihood

Parametric classifiers (e.g. ML algorithm) assume that the data for the classes of interest is distributed normally. The ML classifier is one of the most widely used parametric algorithms for image classification. The applications of ML algorithms have already been addressed in several studies (Sisodia et al., 2014; Srivastava et al., 2012; Foody et al., 1992). The ML algorithm employs the cluster centres and covariance matrix of the clusters, which are determined on a set of training samples to evaluate a candidate pixel $\boldsymbol{x}$. In this case, the statistical distance is a probability value computed through the following algorithm (Richards, 2006):

$$g_i(\boldsymbol{x}) = \ln P(\omega_i) - \frac{1}{2}\ln|\boldsymbol{\Sigma}_i| - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{m_i})^t\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{m_i}), \quad (2.3)$$

where i is the class, $P(\omega_i)$ is the probability that class ω_i occurs in the image assumed the same for all classes, $|\boldsymbol{\Sigma}_i|$ is the determinant of the covariance matrix of the data in class ω_i , $\boldsymbol{\Sigma}_i^{-1}$ is the inverse matrix and $\boldsymbol{m_i}$ is the mean vector.

2.2.2.2. Support vector machine

Non-parametric classifiers (e.g. SVM algorithms) are a group of classifiers that make no assumptions about the statistical nature of the raster datasets (Srivastava et al., 2012). The theory of SVM algorithms was originally proposed by Vapnik and Chervonenkis (1971). Over the last few years, the applications of SVM algorithms have received increasing attention in the remote sensing area (Mathur and Foody, 2008; Bruzzone, 2006). In the SVM algorithm, the decision rules are formulated on the function sgn[f(**x**)], where a discriminant function f(**x**) is usually expressed as follows (Melgani and Bruzzone, 2004):

$$f(\mathbf{x}) = \sum_{i \in S} \alpha_{\mathbf{i}}\, y_i K(\mathbf{x_i}, \mathbf{x}) + b, \quad (2.4)$$

where $\mathbf{x_i} \in \mathfrak{R}^d$ $(i = 1,2,\dots,N)$ consists of N training samples, d is the dimension of the feature space. Training data are represented by $\{\mathbf{x_i}, y_i\}$, and $y_i \in \{-1, 1\}$ for a binary classification. $K(\mathbf{x_i}, \mathbf{x})$ is a kernel function, $\alpha_i{}'$ s $(i = 1,2,\dots,N)$ is the Lagrange multipliers, S is the subset of training samples corresponding to the non-zero Lagrange multipliers $\alpha_i{}'$ s and a bias $b \in \mathfrak{R}$. In this thesis, the VA uses the Radial Basis Function (RBF) kernel (Equation 2.5) to implement the SVM algorithm.

$$K(\mathbf{x_i}, \mathbf{x}) = \exp(-\gamma\|\mathbf{x_i} - \mathbf{x}\|^2), \tag{2.5}$$

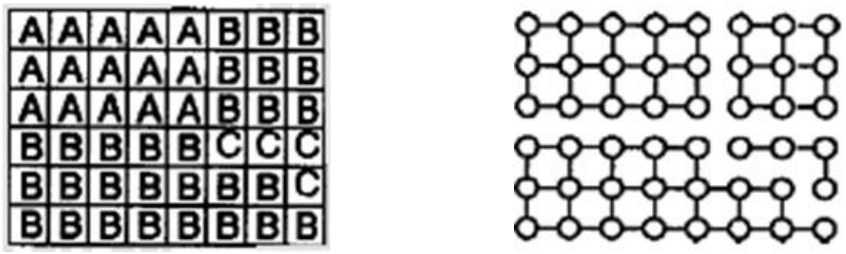
where γ is a parameter inversely proportional to the width of the Gaussian kernel. To train the SVM model for use in classification, two parameters, namely γ and *C*, should be chosen. *C*, regularisation parameter, determines the level of the trust to the training data. These parameters are often not known. To identify the optimum values of *C* and γ, an *n*-fold cross-validation algorithm is usually applied. The algorithm uses a grid search to automatically select these parameters. To do this, the algorithm divides the training data into *n* subsets of equal size. At each iteration, one of the subsets from the training samples is first removed. The remaining subsets are then trained in terms of different values of *C* and γ. For each(C, γ), the algorithm uses the omitted subset or test samples to compute the accuracy of data. After *n* iterations, the algorithm chooses the *C* and γ values with the maximum accuracy for learning process.

2.2.3. Spatial objects in a pixel-based approach

In pixel-based approaches, the geometry of spatial objects is determined after image classification. This process is often performed in two main steps: post classification filtering and geometric extraction. In the first step, a post classification filtering process is applied to improve the classification results, either by eliminating isolated pixels or using majority filters (Gao, 2008). Then the classified image is segmented based on the topology of the thematic raster data to address the geometry of the spatial objects (Figure 2.3). There are two ways to link object information to the cells (Molenaar, 1998):

i. Each cell has a label based on attribute information. The complete geometry of an object can be found by inspecting the labels of all elements of the raster to check whether it belongs to the required object. Thus, objects can be identified as contiguous sets of cells with the same label.

ii. Each object points to the cell representing it. This can be done in the form of a list or linked list. The geometric structure of the object is then directly realised in the data model.



(a) (b)

Figure 2.3. (a) Area object in raster space and (b) topologic structure (from Molenaar, 1998).

The structure of pixel-based approaches shows that these methods use a sequential process of classification and spatial segmentation to address the spatial objects in image space. In this context, the elements of spatial objects, namely thematic and geometric, are independently determined (Figure 2.1). In other words, this structure shows that the spatial modelling algorithms based on ***pixel-based approaches lack the ability to directly address the geo-objects in image space***. Thus, real-world objects are modelled regardless of their nature in the real world. This can lead to poor results for modelling geo-objects, especially when these objects are heterogeneous (e.g. forest).

2.3. Object-based image analysis

In object-based approaches, image objects are the main processing units. Image objects are a set of regular/irregular polygons created in image space through a process known as segmentation. The image objects are then labelled in the classification step.

2.3.1. Image segmentation

Image segmentation is a process that groups sets of connected pixels of a given image into a collection of homogenous areas that supposedly depicts a homogeneous thematic meaning, even if the collection of pixels corresponding to the object is heterogeneous. It can be formulated as follows (Mylonas et al., 2013):

$$\bigcup_{i=1}^{N_S} S_i = \mathbf{T} \tag{2.6}$$

where **T** denotes the image space, S_i is a connected set of **T,** $i \in \mathbb{N}$ in which $S_i \cap S_j = \emptyset,\ \forall\, i, j \in \mathbb{N}^2, i \neq j.$

Segmentation methods are generally divided into three main categories: pixel-based, edge-based and region-based (Blaschke et al., 2014). Object-based classification methods usually use region-based algorithms to segment an image, especially when there is a HSR image. They are less sensitive to noise compared to pixel-based and edge-based methods (Schiewe, 2002). In this context, the multiresolution image segmentation proposed by Baatz and Schape (2000) is one of the most popular image segmentation methods. This approach uses a region-based growing algorithm formulated on spectral and spatial information of geo-objects in order to segment an image.

In this case, the segmentation algorithm applies the following function *f* to control the heterogeneity of objects in image space (Benz et al., 2004):

$$f = w_{colour}.\Delta h_{colour} + w_{shape}.\Delta h_{shape}, \tag{2.7}$$

where $w_{colour} \in [0,1], w_{shape} \in [0,1]$ are the weight parameters applied to adapt heterogeneity definition to the application of image analysis and $w_{shape} + w_{colour} = 1$. Δh_{colour} and Δh_{shape} are calculated as follows:

$$\Delta h_{colour} = \sum_c w_c(n_{merge}.\sigma_{c,merge} - (n_{obj-1}.\sigma_{c,obj-1} + n_{obj-2}.\sigma_{c,obj-2})), \tag{2.8}$$

where n_{merge} is the number of pixels within merged object, n_{obj-1} and n_{obj-2} are the number of pixels in objects 1 and 2, respectively. $\sigma_{c,}$ is the standard deviation within an object of channel *c*. w_c allows multi-variant segmentation. Δh_{shape} is computed as follows:

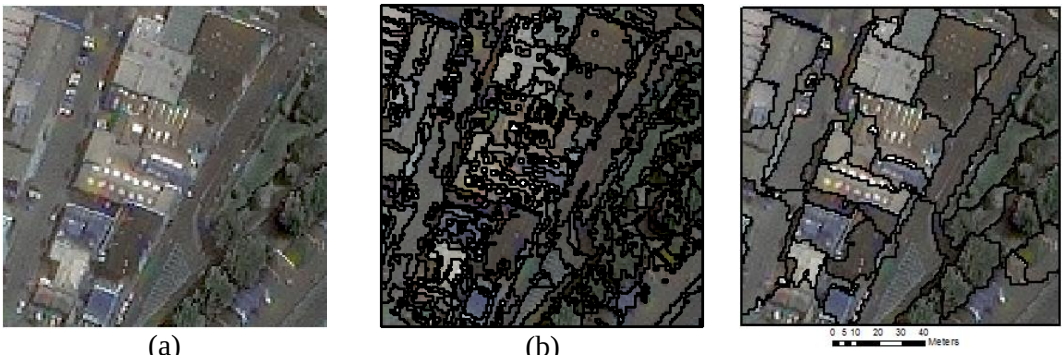
$$\Delta h_{shape} = w_{compt}.\Delta h_{compt} + w_{smooth}.\Delta h_{smooth}, \tag{2.9}$$

where Δh_{shape} controls the smoothness and compactness of an object's shape and Δh_{compt} and Δh_{smooth} are defined as follows:

$$\Delta h_{compt} = n_{merge}.\frac{l_{merge}}{b_{merge}} - \left(n_{merge-1}.\frac{l_{merge-1}}{b_{merge-1}} + n_{merge-2}.\frac{l_{merge-2}}{b_{merge-2}}\right), \tag{2.10}$$

$$\Delta h_{smooth} = n_{merge}.\frac{l_{merge}}{\sqrt{b_{merge}}} - \left(n_{merge-1}.\frac{l_{merge-1}}{\sqrt{b_{merge-1}}} + n_{merge-2}.\frac{l_{merge-2}}{\sqrt{b_{merge-2}}}\right), \quad (2.11)$$

where l is the perimeter of the segmented object and b is the perimeter of the object's bounding box. The weights w_c, w_{shape}, w_{colour}, w_{smooth}, and w_{compt} are parameters defined by a human expert to get suitable segmentation results for a certain image datasets and a given application. As there is no specific rule to determine these parameters, they are defined based on trial and error (Hay et al., 2005). For example, Figure 2.4 shows the results of the segmentation process based on different values of these parameters.



(a) (b) (c)

Figure 2.4. (a) A subset of IKONOS image from an urban area, Dunedin, New Zealand. (b) and (c) segmented images with scale: 30, compactness: 0.5 and shape 0.1 and shape 0.9, respectively.

The product of the segmentation process is a set of connected pixels known as image objects, which can satisfy Equation 2.6. The thematic class of image objects as main processing units is determined in the classification step using a set of rules in image space via a GEOBIA algorithm.

2.3.2. Spatial objects in an object-based approach

Compared to pixel-based approaches, using image objects enables GEOBIA algorithms to apply and analyse more informative data such as geometry, shape, context, content and spectral information (Tian et al., 2007; Hay et al., 2005; Benz et al., 2000). The label of the extracted segmented regions in a segmentation step is determined by a classification method, such as the nearest neighbour or fuzzy rules (Benz et al., 2004). Rule-based systems (e.g. GEOBIA) belong to knowledge-based methods that simulate the human reasoning mechanism and translate knowledge through decision rules (Mather and Tso, 2009). The use

of a classification method based on knowledge provides an opportunity for the classification algorithm to use more information for image analysis compared to a pixel-based approach. This allows for better differentiation between object classes and an efficient extraction of objects (Campbell, 2007).

To classify the image objects, there are two main strategies: parallel and sequential. In sequential methods, GEOBIA determine the class of image objects belonging to each class one at a time. This allows the GEOBIA methods to use information from procedural knowledge in order to label image objects. For example, shadow objects are usually found next to elevated objects (e.g. buildings). When the buildings are already classified, the shadow objects can be identified not just with spectral information but also the neighbourhood rules. Roads or rivers can be identified using the width of the objects in addition to spectral information. However, the labelling process can be difficult and time consuming because of the difficulty in formalising expert knowledge and encapsulating it into rule sets (Mahmoudi et al., 2013).

In contrast, parallel methods use a fast architecture to classify an image. In this case, all image objects are labelled at once. However, these methods cannot use the procedural knowledge in the classification step. To address this issue, the application of agent-based modelling has already been addressed in several studies (Hofmann et al., 2015; Zhong et al., 2014; Mahmoudi et al., 2013). The agent-based approach not only speeds up the processing of (remotely sensed) data analysis tasks by exploiting parallelism but also allows the agents to share their information.

Despite the advantages that parallel methods offer, both parallel and sequential methods have two common limitations: they use a static geometry to address geo-objects in image space. In this case, *objects cannot change their geometry once they are created* (Baatz et al., 2008). This means that the process of merging image objects can lead to a meaningful object during the classification step, whereas the geometric elements of geo-objects are determined *via a set of user-defined parameters based on trial and error* (Hay et al., 2005) *without direct connection to the real-world environment* (Benz et al., 2004).

The above example also illustrates that object-based image classification uses a sequential process of segmentation and classification to address geo-objects. In contrast to the pixel-based approaches, the geometry of geo-objects is first determined by a segmentation process. Then, the thematic meaning of the spatial objects is addressed using a classification method.

Similar to pixel-based approaches, the main components of spatial objects, namely thematic and geometric, are independently identified from the image space. In other words, ***object-based approaches lack the ability to directly address geo-objects in the image space***. The advanced object-based method uses an iteration strategy of image segmentation and classification (see Figures 1.1, 1.2 and 1.3) to address this issue. However, all studies are established on *the static geometry of real-world objects*. In this context, object-based approaches are performed based on two main assumptions:

- The parameters of the segmentation process can be determined accurately.
- Geo-objects have crisp boundaries.

In light of the above, we assume in object-based approaches that geo-objects have predictable behaviours in a complex real-world environment.

So far, we have seen that *pixels lack the ability to directly address geo-objects* in image space. Segmented objects show that they can be a proper solution to address this issue. However, a gap between reality and vector representation still remains. In our research, we will present a new automated processing unit ***to directly address geo-objects in image space***.

As the main objective of this thesis is to extract and represent real-world objects directly from raster data (such as remote sensing imagery), it is necessary to perform a review of vector geometry and the thematic concept of spatial objects. The aim of this review is to analyse the abstract formalisms that have been used to present a vector model in image space.

2.4. Space

The concept of space is important in understanding and modelling real-world phenomena or objects. There is always an implicit model of space underlying every spatial representation (Takeyama, 1997). Geographic space can be expressed in raster format as a field of measurement (e.g. temperature) or in vector space as a collection of geometric discrete objects (e.g. houses) (Takeyama, 1997; Couclelis, 1992). Accordingly, two main data structures, namely raster and vector, can be used to represent real-world phenomena.

2.4.1. Raster space

The raster data is structured via tessellations for geometric modelling. Each tessellation in a raster space is a partition of space connecting disjointed areas of a certain size or resolution.

Tessellations can be categorised as regular (e.g. grid, hexagon or triangle) or irregular (partitions with varying sizes and shapes). In this thesis, raster data refers to a regular tessellation, consisting of an array of pixels that are regularly spaced with a common shape (square). Each cell in the raster space is defined by its position, organised into rows and columns (or a grid), and a value that represents information, such as height. The coordinates of each cell are usually calculated using origin coordinates (generally lower-left) and the resolution of raster data.

Hence, raster data has two main roles in our model. Firstly, the values of a cell provide the necessary information for the VAs to identify the thematic information of the geo-objects. Secondly, a predefined geometric structure of raster data, as a base map, allows the VA model to formulate the geometric rules for extracting geo-objects. In raster space, the geometry of objects is represented via a cell or a group of cells. For example, a point may be represented by a cell, a line by a sequence of neighbouring cells, and a polygon by a collection of connected cells (e.g. image objects in Figure 2.4 or thematic layer in Figure 2.3 (a)).

2.4.2. Vector space

Despite the advantages that raster data offer, such as simple data structure, they lack the ability to provide an actual geometry for geo-objects in real-world space. Since geo-objects are abstractions of entities in simulation domain, the vector data (e.g. point, line or polygon) is more suitable than raster data to represent geo-objects in real-world space (Cova and Goodchild, 2002). According to Hay and Castilla (2008):

> *"GEOBIA relies on RS (remote sensing) data, and generates GIS (Geographic Information Systems) ready output, it represents a critical bridge between the (often disparate) raster domain of RS, and the (predominantly) vector domain of Geographic Information System."*

As VAs are implemented in a vector or discrete space, the concept of vector data structure is used to define the VA model. Here, vector space is a subset of spatial space associated with the occurrence of geographic phenomena on the earth. In this context, vector space can be regarded as a container or framework of discrete, identifiable units, namely geo-objects.

2.5. Geometry of spatial objects

At a basic level, the geometry of vector data can be regarded as a collection of points with geographic coordinates. The geometry of geo-objects can also be expressed by the relative position of features (e.g. points, lines or polygons) or a mathematical function.

2.5.1. Geometric primitives

In vector space, each geo-object can be represented by point or line or polygon. These features, geometric primitives, are defined as follows (David et al., 1996):

i. **Point objects**: A point is a 0-dimensional geometric primitive associated with one set of coordinates (x, y), based on a georeferenced system. This coordinate represents the distance from the origin in the direction of each axis.
ii. **Line objects**: A line is a one-dimensional geometric primitive, which may or may not be closed. A line can also be a segment (a finite line which begins and ends at two defined points), a string (an ordered sequence of sets of coordinates and the shortest connection between them), or an arc (an ordered sequence of sets of coordinates and connections between them that are defined by a set of mathematical functions).
iii. **Area objects**: An area object is considered a bounded continuous two-dimensional geometric primitive, delimited by one outer non-intersecting boundary and zero or more non-nested non-intersecting inner boundaries.

2.5.2. Structure primitives

Structure primitives are applied to describe the relative position of features. These primitives are defined as follows:

i. **Node**: A node is a 0-dimensional structure primitive. It can be an isolated node (not related to any edge) or a connected node (related to one or more edges).
ii. **Edge**: An edge is a one-dimensional structure primitive, specified as connecting a start node and end node.
iii. **Face**: A face is a structure primitive with a minimum of two dimensions, defined by one outer ring and zero or more inner rings.

The geometry of spatial objects can also be addressed indirectly in vector space. In this way, a mathematical function is usually applied to divide the vector space into a set of discrete objects. A Voronoi diagram is a special kind of decomposition of a metric space determined by distances to a specific set of objects (like a discrete set of points) within the space (Okabe

et al., 2009). A Voronoi diagram divides a 2D vector space into a set of regions. Each region corresponds to one point (site), and all points in this region are closer to the corresponding site than to any other (Figure 2.5).

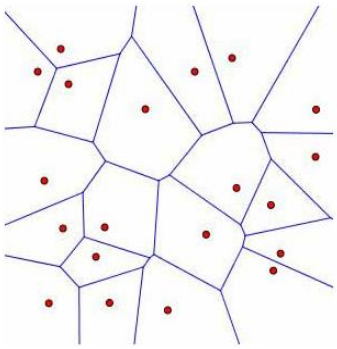


Figure 2.5. Voronoi diagram with irregular points distribution.

The geo-objects can also be modelled via their geometric elements (e.g. edges). For example, on a raster data structure (e.g. images), sharp discontinuities can be extracted via an edge detection algorithm, such as Laplacian. The Laplacian $L\ (x,y)$ for a pixel with intensity values $I\ (x,y)$ is calculated as follows (Gao, 2008),

$$L(x,y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}, \tag{2.12}$$

As an input image is represented by a collection of discrete pixels, we usually use a set of convolution kernels (e.g. Figure 2.6) to approximate the second derivatives in Equation 2.12.

0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

−1	2	−1
2	−4	2
−1	2	−1

Figure 2.6. The commonly used the Laplacian kernel with a window size of 3 × 3 pixels.

The extracted discontinuities can then be applied to model the boundaries of geo-objects in a scene.

2.5.3. Set theory

Spatial objects can be defined from a set theory point of view. A set is a collection of objects, such as people or points in 2-dimensional plane (Kainz, 2004). In a discrete space, a set X =

$\{x_1, x_2, \ldots, x_n\}$ can be specified by enumeration. In set theory, four operations, including union, intersection, difference and complement, are often applied between sets.

iv. ***Union***: the union of two sets *A* and *B*, written as $A \cup B$, is the set containing all the elements that belong either to *A* or to *B*. It is expressed by $A \cup B = \{x: x \in A \; or x \in B\}$. The union of a collection of sets is written as $\bigcup_{i\in\{1,\ldots,n\}} X_i$.

- ***Intersection***: $A \cap B$ displays the intersection of two sets *A* and *B*, written as $A \cap B = \{x: x \in A \; and \; x \in B\}$. The intersection of a collection of sets is written as $\bigcap_{i\in\{1,\ldots,n\}} X_i$.
- ***Difference***: $A - B$ shows the difference between two sets *A* and *B*, written as $A - B = \{x: x \in A \; and \; x \notin B\}$.
- ***Complement***: the complement of a set *A* in the universe *U* is the set $A^c = U - A = \{x: x \notin A\}$.

The concept of a set can also be applied to define a topological space in which a collection of neighbourhoods fulfils certain conditions. In Figure 2.7, three types of points can be distinguished in set A:

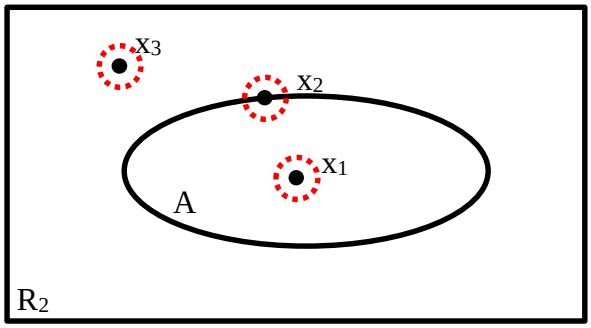


Figure 2.7. The topology of point set R_2 and the neighbourhood of points x_1, x_2 *in set* A *with dimensions of 2, where R_2 is the real plane defined on the Euclidean distance (adapted from Molenaar, 1998).*

i. For points of type x_1, there are neighbourhoods U_x where all points (x) belong to x_1 and are defined by $U_x \subset A$. These points are called interior points of A, written as $^{\circ}A = \{x \in A | \exists \; U_x \Rightarrow U_x \subset A\}$.

ii. For points of type x_3, points (x) do not belong to A. They have a neighbourhood U_x, which consists of points that do not belong to A. These points are known as exterior points of A, written as $^{-}A = \{x \notin A | \exists \; U_x \Rightarrow U_x \cap A = \emptyset\}$.

iii. For the third group represented by x_2, there are neighbourhoods U_x that consist of points that belong to A and A^c (complement of set A). These points are known as frontier or boundary points of A, written as $\partial A = \{x \in U | \forall\, U_x \Rightarrow U_x \bigcap A \neq \emptyset \; and \; U_x \bigcap A^c \neq \emptyset\}$.

2.5.4. Graph theory and planar graph

The use of graphs is a simple way to define an object in image space. A graph formulates the relationships between structure primitives (node, edge and face). A graph *G* is an incident relation between two disjoint sets *N* and *E* (Rahman and Pilouk, 2007; Molenaar, 1998), where *N* and *E* are defined as follows:

i. *N* is a non-empty set of *i* nodes, $N = \{n_1, n_2, n_3, \dots, n_i\}$ where the position of nodes are specified based on the coordinates.

ii. E is a set of j edges, $E = \{e_1, e_2, e_3, \dots, e_j\}$.

iii. An edge is a connection of two nodes, $e_k = \{n_p, n_q\}$, where . $e_k \in E$ and $\{n_p, n_q\} \in N^2$.

Considering the above, we can define the concept of direction, chain, segment and polygon in a graph as follows (Rahman and Pilouk, 2007; Molenaar, 1998):

- Two nodes are adjacent if an edge connects them.
- An edge is directed if $e_i = \{n_p, n_q\}$, with n_p and n_q are the start and end node of e_i, respectively.
- Two edges can be adjacent if there is a common node between them.
- If a node n_p occurs in *m* edge, the degree of node is equal to *m.*
- A sequence of edges forms a polyline or path, if an edge only occurs once and the degree of nodes within the polyline is equal to 2.
- If all possible pairs of nodes are connected, a graph is known as a connected graph.
- A chain is a sequence of vertices and edges in which each edge's endpoints are the preceding and following vertices in the sequence.
- A graph segment or g-segment is a polyline, when the degree of all nodes in the chain is equal to 2.
- A node of a g-segment with a degree of 2 is often known as vertex.
- A polygon is a closed polyline, when the degree of all nodes is equal to 2.

2.5.5. Planar graph

In graph theory, a planar graph is a graph that can be drawn on the plane in such a way that no two edges intersect (Goodrich and Ramaiyer, 1998). A planar graph can always be embedded in the plane so that all its edges are straight line segments. Such a planar graph is called a planar straight line graph (PSLG). In the planar graph, the area segments are formed by polygons based on the nodes and edges. Faces are regarded as a special type of area segments. The data structure of PSLG can be defined through a collection of polygons. However, this representation is not flexible enough to support the traversal of edges around a vertex. This issue is often addressed by a winged edge structure or doubly connected edge list (DCEL) (Sack and Urrutia, 1999).

2.5.5.1. Winged-edge data structure

The winged-edge data structure developed by Baumgart (1975) explicitly describes the geometry and topology of faces, edges, and vertices when three or more surfaces come together and meet at a common edge. It is assumed that the faces in objects do not have internal loops and are composed of a single shell or exterior ring. Each face is represented as a sequence of the edges it includes (i.e. a face-edge relation), while the edges are specified as ordered pairs of their constituent vertices (i.e. an edge-vertex relation) (Figure 2.8).

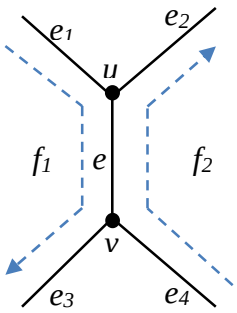


Figure 2.8. Winged-edge data structure: e has a reference to e_1, e_2, e_3, e_4, u, v, f_1 and f_2 (from Čomić and de Floriani, 2012).

In other words, for each edge *e* with two vertices, there are two faces incident to it, and four adjacent edges representing the boundary of the two faces incident to *e:* For each face f_1, there is a reference to one edge on the boundary of f_1: For each vertex *v*, there is a reference to one edge incident to *v*, which efficiently supports the retrieval of all topological relations (Samet, 2006). The cells in the star of a vertex or on the boundary of a face can be traversed in either a clockwise or counter-clockwise direction.

2.5.5.2. Doubly connected edge list (DCEL)

Muller and Preparata (1978) proposed the DCEL representation for embedded planar graphs. In a DCEL, each edge is regarded as a directed edge. In fact, the DCEL structure can be regarded as a simplified version of the winged-edge data structure representation (Čomić and Floriani, 2012). The DCEL data structure stores only two edges, namely e_2 and e_3, for each edge, one for each of the two faces incident in e (Figure 2.9).

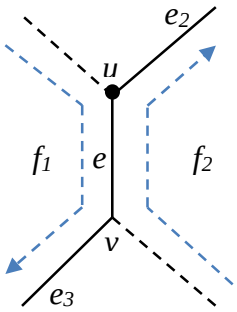


Figure 2.9. DCEL data structure: e has a reference to e_2, e_3, u, v, f_1 and f_2 (from Čomić and de Floriani, 2012).

The DCEL can also support the traversal of all topological relationships, similar to the winged-edge data structure. However, the DCEL can only address the edges around faces in a clockwise direction and around a vertex in a counter-clockwise direction (Čomić and Floriani, 2012). A comparison between two data structures can be performed based on Figure 2.8 and Figure 2.9.

2.6. Spatial Relations between spatial objects

Spatial relationships between geographical objects can be grouped into three main classes, namely topology, order and metric, based on their function or relationship with a set of objects (Egenhofer, 1989).

2.6.1. Topology

Topology is a branch of Euclidean geometry concerned with the set of geometric properties that remain invariant under topological transformation, such as scaling or rotation (Egenhofer, 1989). Hence topology relationships describe the relationship between objects. In the context of set theory, these relations can be based on the three components of a spatial object, namely interior, boundary, and exterior, as described in Section 2.5.3. The intersection of these components can be organised into a 3×3 matrix (Figure 2.10). Through

this matrix, 9 logical rules can be interpreted (Pullar and Egenhofer, 1988). In Euclidean space, there are only eight topological relationships between two spatial objects or two sets *A* and *B*, summarised and shown as follows:

- ***A*** disjoint ***B***: there is no common boundary or interior between both objects.
- ***A*** meets ***B***: two polygons share at least one common boundary.
- ***A*** equals ***B***: two objects are equal if they have the same boundary and interior.
- ***A*** inside ***B***: ***A*** and ***B*** share a boundary, but not an interior.
- ***A*** contains ***B***: if ***B*** is inside ***A***.
- ***A*** covers ***B***: a polygon ***A*** covers ***B*** if both polygons share part of a common boundary as well as interior ***B***.
- ***A*** covered by ***B***: the same definition as ***A*** covers ***B***, rewritten as ***B*** covers ***A***.
- ***A*** overlaps ***B***: two polygons overlap if they share a common interior part and the boundaries intersect.

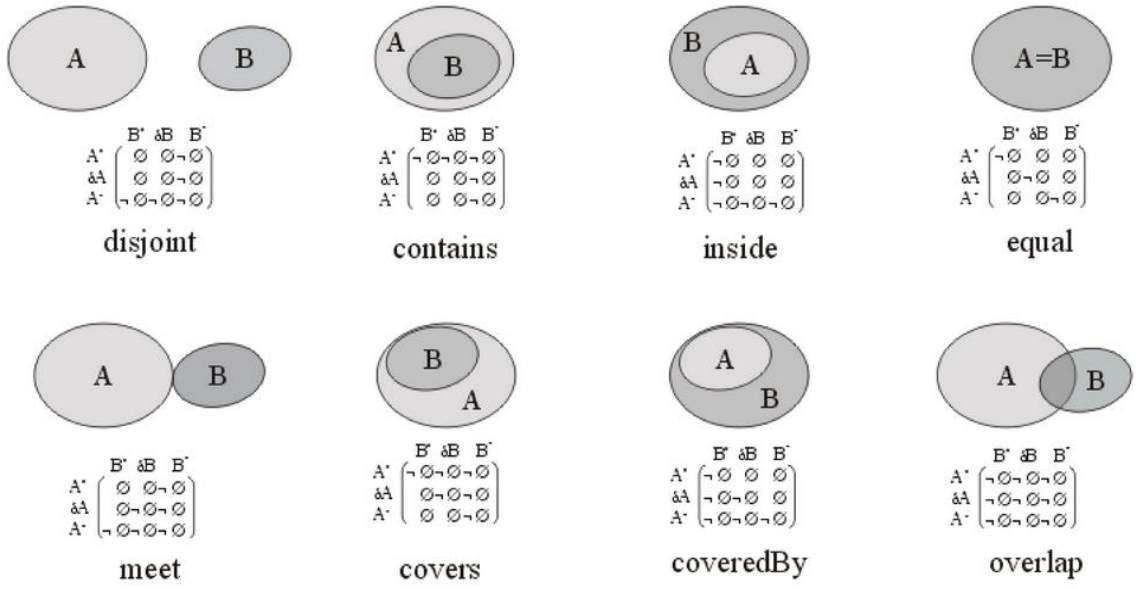


Figure 2.10. Topological relationships (from Egenhofer, 1989).

2.6.2. Order

Spatial order relationships rely on the definition of order based on a preference (e.g. behind) (Egenhofer, 1989). There are two types of order relations: strict order and partial order. In

the former case, namely strict order (e.g. <), the order relations are transitive, if $x < y$ and $y < z$, then $x < z$. In the latter case, order relations are reflexive, $x \leq x$, antisymmetric, if $x \leq y$ and $y \leq z$ then $x = z$, and transitive (Abdul-Rahman and Pilouk, 2007). Each order relation generally has a converse relationship. For instance, the converse relation of a<b is determined by b<a.

2.6.3. Metric

Metric relationships exploit the existence of measurements (e.g. distances and directions) in a metric space (Egenhofer, 1989). A metric space is defined according to an ordered pair (M, d), where M is a set and d is a function determined by $d : M \times M \rightarrow \mathbb{R}$, called distance function, in which for any $x, y, z \in M$, d can satisfy the following conditions (Choudhary, 1992):

1. $d(x, y) \geq 0$, the distance between x and y is more than or equal to zero,
2. $d(x, y) = 0 \Leftrightarrow x = y$, the distance from x to itself is equal to zero,
3. $d(x, y) = d(y, x)$, the distance from x to y is equal to the distance y from x, and
4. $d(x, y) \leq d(x, y) + d(y, z)$, the sum of the distances of (x, y) and (y, z) is more than or equal to the distance between x, y.

The distance function is defined in Equation 2.1 for the n-dimensional Euclidean space.

So far we have reviewed the geometry of geo-objects and the spatial relationships between them in vector space. In the next section, the thematic component of spatial objects (Figure 2.3) will be explored in more detail.

2.7. Thematic meaning of spatial objects

The thematic meaning or class of a geo-object is represented by a label or a class name. In the real world, geo-objects may belong to the same class if they have a common structure of attributes that are similar to the characteristics of the class of interest. In this case, geo-objects compare their attributes with those of the class of interest through a classification algorithm to find their class or thematic meaning. For example, in remote sensing, an image may be defined based on the spectral characteristics of the class of interest or on a set of spatial rules. To address the thematic meaning of real-world objects, the concept of ontology can also be applied. Gruber (1995) defined ontology as a formal, explicit specification of a shared conceptualisation. The conceptualisation is an abstract and simplified view

representation of the world specified for a certain purpose (Gruber, 1995). The conceptualisation should be explicit, shared and formal to be considered ontology. These characteristics can be defined as follows (Arvor et al., 2013):

i. **Explicit**: Concepts and constraints can be defined precisely.
ii. **Shared**: The ontology can capture consensual knowledge.
iii. **Formal**: The ontology can be defined by axioms in a formal language with the aim of providing an unbiased view of reality. It is also machine understandable.

In the design step, the method to represent objects in ontology is determined (Arvor et al., 2013). For example, Hofmann et al. (2015) used ontology to define the characteristics of classes of interest, namely the roof shapes of buildings, in order to classify an image.

The previous discussion reviewed the main elements of spatial objects, including thematic and geometric. In a static structure, these components are usually linked through a unique identification (ID) (see Figure 2.1) to model a real-world environment. In a dynamic structure, all spatial objects are interacting not only with the environment, but also with others, whilst all components within each spatial object are linked. In this context, spatial modelling should be able to simultaneously address the components of geo-objects together with the connections and processes between objects in the simulation domain. In this way, a logical design (e.g. object-oriented structure) is needed to assimilate the elements of spatial objects and model the interactions between objects.

2.8. Object-oriented approach

Object orientation can be described as a strategy for organising a system as a collection of interacting objects that can combine data and behaviour (Blaha & Premerlani, 1997). The processes and connections between and within objects are usually described through a modelling language called the UML. UML is a general-purpose visual modelling language usually applied in object-oriented modelling (Arlow and Neustadt, 2005).

The object orientation approach is established based on the assumption that real world objects can be modelled as distinguishable entities based on their identity, state and behaviour. In an object oriented system, each object has a unique identity. The state of each object is described based on its attribute values at any one moment in time. The behaviour of objects can be regarded as a set of synchronisation constraints that define how objects execute their methods in relation to one another (Firesmith and Eykholt, 1995), or a

metaphor referring to the way objects change over time within a defined structure (Martin and Odell, 1992). The behaviour of an object is encapsulated in its methods or operations performed by the object itself or by another object. Some characteristics of an object-oriented approach are summarised as follows:

i. **Abstraction**: In the object-oriented environment, abstraction refers to the capture of essential characteristics of an object or group of objects while temporarily ignoring the unessential details (Abdul-Rahman and Pilouk, 2007; Sellers and Edwards, 1994). In the context of an entity, abstraction denotes a model that includes the most important, essential or distinguishing aspects of something while suppressing or ignoring less important, immaterial or diversionary details (Firesmith and Eykholt, 1995). There are four common abstraction mechanisms applied in an object-oriented system, including classification, generalisation, association and aggregation.

   - ✓ *Classification*: Objects should be placed in the same class if they have the same kind of properties and behaviours.
   - ✓ *Generalisation*: There is a general parent class as if two or more classes have many properties and behaviours in common.
   - ✓ *Association*: This defines linked objects of two different classes, allowing one object to cause another to perform an action on its behalf. Association can be one to one, one to many, many to one and many to many.
   - ✓ *Aggregation*: Different classes may be aggregated to build up an aggregated class.

ii. **Inheritance:** The inheritance mechanism allows the propagation of properties and behaviours to lower level objects in the same hierarchy.

iii. **Polymorphism:** This ability allows objects to have multiple forms, so that multiple types of objects can use the same attribute name without confusion about which class the attribute belongs to.

iv. **Encapsulation:** The encapsulation ability allows each object to access its data only through its methods.

As mentioned before, the entities of geographic phenomena usually are dynamic in nature, and their physical characteristics are subject to changes over time. In this sense, geo-objects

should be capable of perceiving time during the simulation process. The following section will review some key concepts of time for spatial modelling in more detail.

2.9. Time

In a dynamic environment, time is an essential dimension for understanding and modelling spatial objects. The following is a brief overview in which some concepts of time are reviewed.

2.9.1. Definition

2.9.1.1. Measurement of time:

This can be a discrete (e.g. years, months, or weeks) or continuous variable, whereby there are an infinite number of points between any two points (Worboys and Duckham, 2004). A chronon is the smallest, non-decomposable unit of time. This is the time that is supported by a temporal database. A finite number of chronons in the database is applied to describe the lifespan of an object. The transition from one state of an object to another is called an event, and is shown by a point on the timeline, whereas a state is regarded as an interval on the timeline.

2.9.1.2. Models of time:

There are three mathematical models of time: linear time (where time moves from the past to the future), branching time (where there are single past and many futures), and cyclic time (where time repeats itself).

2.9.1.3. Types of time:

This can be world or event time (the time at which an event actually occurred in the real world); observation or evidence (the event is observed); or database or transaction time (the time at which an event is recorded in the database).

2.9.2. Space-time representation

The passage of time for spatial objects is normally understood through changes occurring to objects in space (Peuquet, 1994). These changes can be based on geometry (e.g. urban expansion), position (e.g. vehicle movement), attribute (e.g. traffic volume) or a combination of these components (Figure 2.11) (Goodchild et al., 2007).

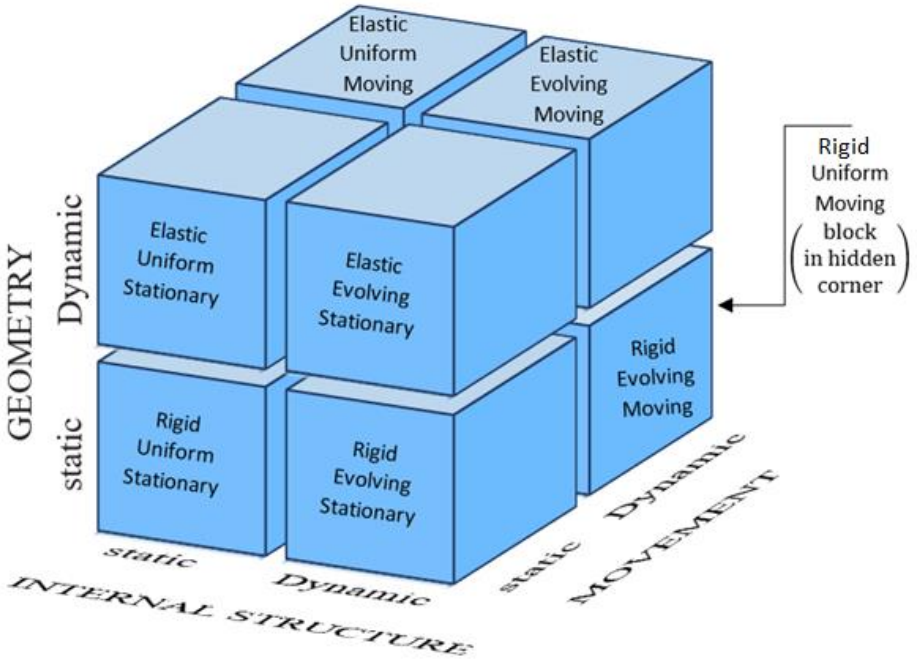


Figure 2.11. Three dimensions of temporal variability in geo-objects (from Goodchild et al., 2007).

For example, objects (e.g. airplanes) can be regarded as uniform, moving and rigid objects, in which the changes are only determined by tracking their location. A moving object can be an evolutionary object if its changes are subject to its state (e.g. speed of an airplane). A spatial object is an evolutionary, stationary and elastic object if it can only change its geometry and state (e.g. vegetation cover during desertification). However, these changes are usually addressed via a static geometry (Hammam, 2008). There are different ways to model these changes. Some of the main models (Pelekis et al., 2004; Raza, 2001) are briefly introduced here.

i. **Snapshot approach**: If a change occurs, it stores all versions of the map as a series of snapshots. It does not represent the events that cause changes.
ii. **Update model**: Only the changed objects are stored. It is not a complete snapshot of the data.
iii. **Space-time composite**: This model stores all past and present features by overlapping all timestamped layers to produce a space-time composite layer (Langran and Chrisman, 1988).
iv. **3D/4D temporal GIS**: In this approach, time is considered as a dimension of spatial objects that can lead to a true temporal GIS system. This approach cannot be implemented without a major software engineering effort (Pelekis et al., 2004; Raza, 2001).

When comparing the above models, it can be concluded that the notion of 3D/4D temporal GIS is most optimal for the representation of real-world objects. In this sense, geo-objects can be treated individually in a simulation domain that can change and update their elements in relation to its environment and other objects.

In the real world, a geo-object is usually complex and composed of several parts. To model geo-objects, a spatial modelling method should be able to support not only mereology (a theory of part-whole relations) and topology but also qualitative geometry (Smith and Mark, 1998). In a dynamic system, such as geo-simulation, this is usually addressed via automata. These methods model a complex system (e.g. the real world) via its elements (e.g. geo-objects). This complex system generally exhibits the characteristics of emergent properties and self-organisation. Emergence means that the repeated actions of a simple element in a complex system can produce spatial patterns. Depending on the initial configuration of the simple elements, the system can also provide unique spatial patterns (e.g. fractals). In this sense, a complex system is known as a self-organisation system.

Therefore, the structure of the real world (e.g. spatial patterns) can be extracted through the local and recursive actions of geo-objects in a self-organised fashion. In the following sections, some examples will be reviewed in greater detail.

2.10. Geosimulation

The term ‘geosimulation’ is generally applied to describe both object-based and pixel type spatially-explicit modelling of dynamic systems (Benenson and Torrens, 2004b). The main aim of geosimulation is to model a spatial system via its elemental objects, such as individuals in human systems or automata in computer systems (Marceau and Benenson, 2011). There are two major classes of geosimulation models: Cellular Automata (CA) and Multi Agent System (MAS) (Benenson and Torrens, 2004b).

2.10.1. Cellular automata and geosimulation

In the CA model, space is represented by a regular 2D lattice of cells. Each cell is characterised by a finite set of states that is updated via transition rules, taking into account the cell’s previous state and the states of the neighbouring cells. White and Engelen (2000) used CA for ecological impact assessment, land-use and social planning. The application of the CA model to reproduce patterns of the land-use dynamics of cities and regions was presented by Batty (2007). For visualising land use patterns in urban partitions, Shen et al.

(2009) used a CA data structure for simulation that can be edited using irregular polygon layers that represent blocks and parcels in an urban area. Despite the advantages that the CA model offers, there has been extensive debate regarding the strict formalism of this method. The points of contention are as follows:

i. In CA, regular grids are the standard grid structure defined in a finite space. Both the idea of uniformly regular shapes and an infinite spatial plane are unrealistic for most urban contexts (Torrens, 2000).
ii. The state of a cell is only influenced by its neighbouring cells (von Neumann and the Moore neighbourhood). In other words, in formal CA, actions are performed locally (Batty, 2007).
iii. Transition rules are usually applied in an overly simplified way (e.g. the same neighbourhood for all cells) (Wu, 1996).

The limitation of rigid geometry and uniform neighbourhoods of the CA model are usually addressed via an irregular tessellation such as Voronoi diagrams (Shi and Pang, 2000), Delaunay triangles (Semboloni, 2000), and spatial graphs (Moreno et al., 2009). To deal with the strict CA transition rules, utilising methods are usually applied, such as genetic algorithms (Colonna et al., 1998), spatial optimisation (Goldstein, 2004), and neural networks (Li and Yeh, 2001).

2.10.2. Agents and geosimulation

The idea of an agent was proposed by John McCarthy in the mid-1950s, and a few years later, the term ‘agent’ was applied by McCarthy and Selfridge (1954). They introduced an agent as a soft robot and goal oriented system which can fulfil its goal by using appropriate computer operations (Kay, 1984).

With the appearance and development of software agents in the 1990s, the range of domains that utilised agents grew. Thus, various terminology and definitions were developed to define what an “agent” is. For example, Brustoloni (1991) defined an agent as autonomous and goal oriented. Smith et al. (1994) defined agents as simulation objects. Here, agents were used to develop a tool for Apple Inc. that allows children to build a symbolic simulation. Russell and Norvig (1995) defined an agent as anything with the ability to perceive its environment through sensors and acting upon that environment through effectors (Figure 2.12).

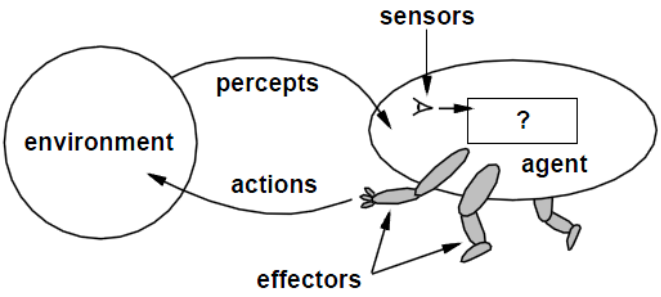


Figure 2.12. Agents interact with environments through sensors and effectors (from Russell and Norvig, 1995).

Wooldridge (1995, 2009) considered an agent as an autonomous decision-making system, which senses and acts in certain environments. Since agents are utilised in such a wide range of domains, it seems complicated to extract a consistent and conclusive definition of their characteristics from literature (Bonabeau, 2002). Some authors believe that it is not possible to produce a universally accepted definition of the term 'agent' (Macal and North, 2009; Raubal, 2001; Russell and Norvig, 1995). However, most definitions do tend to share some common characteristics (Macal and North, 2009; Wooldridge and Jennings, 1995). These properties were studied, extended and explained by Franklin and Graesser (1996), Epstein (1999) and Macal and North (2009). The following characteristics were considered by Crooks and Heppenstall (2012):

i. **Autonomy**: Agents are autonomous units which are capable of processing information and exchanging this information with other agents in order to make decisions independently. Agents are also free to interact with each other, at least over a limited range of situations, and this does not (necessarily) affect their autonomy.

ii. **Heterogeneity**: The development of autonomy can be performed individually by agents. A collection of agents can exist through a bottom-up structure. Thus, they can be amalgamations of similar autonomous individuals.

iii. **Active**: In the simulation domain, agents are active. The following features can be identified:

   ✓ *Pro-active/goal-directed*: Agents are often regarded as goal-directed or having goals to accomplish, which are not solely driven by objectives to maximise, with respect to their behaviours.

- ✓ *Reactive/Perceptive*: The reactivity property allows the agents to be aware of their surroundings. Agents use a 'mental map' of their environment supplied via prior knowledge in order to perceive other entities, obstacles, or required destinations within their environment.
- ✓ *Bounded Rationality*: This feature allows agents to make inductive, discrete, and adaptive choices that move them towards achieving goals.
- ✓ *Interactive/Communicative*: The interactive feature enables agents to communicate.
- ✓ *Mobility*: Agents can move around in space within the simulation domain. Alternatively, agents can also be fixed in space.
- ✓ *Adaptation/Learning*: Depending on its previous state, an agent can alter its current state, thus, permitting it to adapt, akin to a form of memory or learning. Agents can adapt at an individual level or population level.

Considering the above, three main components —environment, sensors and actuators, can be assigned to the agents:

- **Environment**: Every entity that surrounds the agent and affects the agent.
- **Sensors**: Every component of an agent which allows the agent to understand its current state and changes in the environment.
- **Actuators**: The elements that enable the agent to act on the environment autonomously.

2.10.2.1. Spatial agent

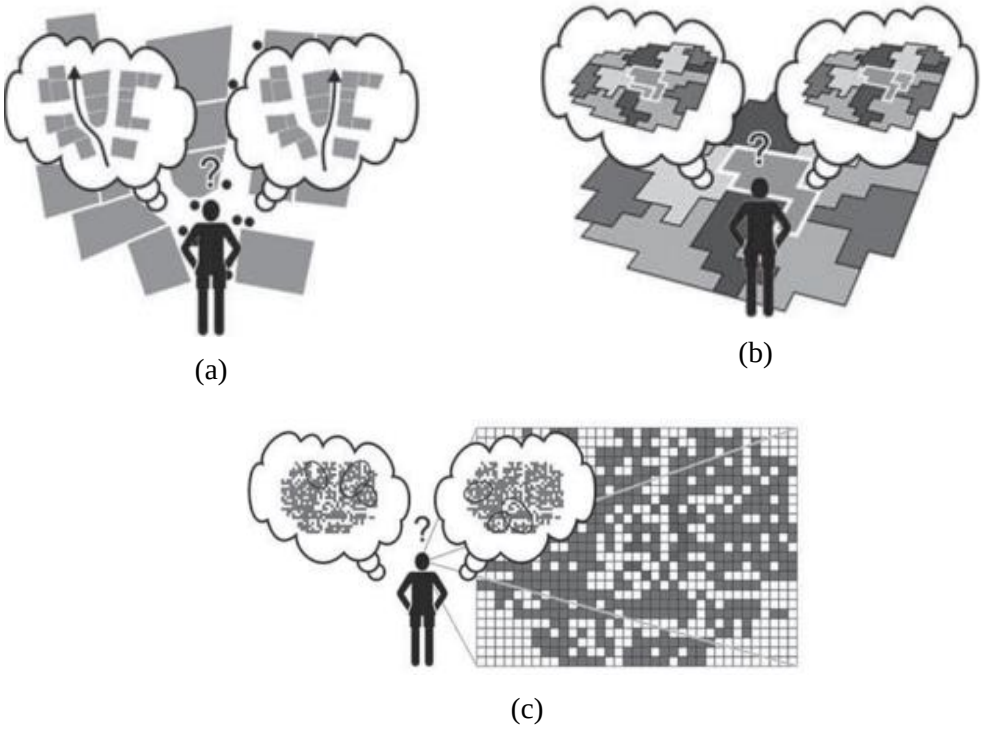
The use of agents with a spatial awareness is a relatively old concept (Rodrigues et al., 1995). Rodrigues et al. (1998) considered spatial agents as a type of agent that can understand either physical or non-physical real-world phenomena. The structure of these models is usually formed based on an environment characterised through a set of individuals, who are defined by their behaviour and attributes, and their interactions. Reynolds (1987) argues that some individual-based models are spatially explicit. In other words, individuals are associated with a location in space. A framework of spatial agents can be expressed as follows (Figure 2.13) (O'Sullivan et al., 2012):

i. Agents can be considered as spatial agents if each has a different relationship with its spatial environment, specifically in terms of a location (e.g. pedestrian

agent) (Figure 2.13 (a)). In the case that all agents have the same spatial relationship with the environment, agents should have an equal capability to alter every location in the model irrespective of their location.

ii. Agents can be considered as spatial agents if they change their spatial relationship with the environment over time by means of movement, alteration, acquisition or disposal of locations (e.g. agent for land use change).

iii. Agents can evaluate spatial configurations (e.g. a property developer agent in an urban growth) (Figure 2.13 (c)).



(a) (b)

(c)

Figure 2.13. Schematic illustration of the choices facing agents in three different types of model: (a) Pedestrian agent, (b) land use change agent and (c) property developer agent (from O'Sullivan et al, 2012).

It is possible for spatial agents to have a complicated scenario based on a combination of the above cases.

2.10.2.2. Spatial agent toolkits

There are many agent development toolkits (e.g. Swarm, Repast, NetLogo) influenced by the individual-based model that is spatially explicit. These systems are usually capable of developing spatially explicit models and integration with GIS functionality. The Recursive Porous Agent Simulation Toolkit (Repast) is an open-source, cross-platform, agent-based modelling toolkit developed at the University of Chicago's Social Science Research

Computing Lab (Altaweel et al., 2006). Swarm and Repast are quite similar in philosophy and appearance. They both provide a library of code for creating, running, displaying and collecting data from simulations (Allan, 2009). However, Swarm appeals to users who have strong programming skills. Repast allows basic models to be developed by users with limited programming experience using the built-in graphical user interface (GUI), unless there are models to be developed with more complex functionality. Variation between toolkits can be evaluated based on the purpose of the modelling, level of development, and modelling capabilities. In this thesis, the Repast was used to model geo-objects in image space.

2.10.2.3. Multi agent system

A Multi Agent System (MAS) consists of multiple agents that model complex and heterogeneous components of a system within a computer model through the use of a virtual copy of the real system. The relationships between the agents can be modelled through the agent's behaviours, rules or goals. Rules are typically defined based on 'if-else' statements which allow the agents to perform an action once a specified condition has been satisfied. Agents can interact amongst themselves and with the environment. There are different ways to model the relationships between the agent and its environment, including the reactive (i.e. agents only perform actions when triggered to do so by some external stimulus, such as actions of another agent) and the goal-orientated (i.e. seeking a particular goal) (Crooks and Heppenstall, 2012). The behaviour of agents can also be scheduled to take place synchronously or asynchronously. The ability to model diverse interactions between agents and their environment allows the MAS to realistically simulate the processes and their impacts (Crooks and Hailegiorgis, 2014; Crooks and Heppenstall, 2012). In this regard, the use of agents with spatial awareness is a powerful approach for evaluating and analysing real-world phenomena.

2.10.2.4. Agents in geosimulation model

The application of MAS is addressed in several studies to simulate the interactions of social actors in various contexts, including urban dynamics (Jackson et al., 2008; Benonson, 1988), spatial planning (Ligtenberg, 2001), land-use changes (Lim et al., 2002), and natural resource management (Gimblett, 2002). Despite the advantage of MAS on aspects like movement, the geometry of agents is static, in contrast to CA models where objects can geometrically evolve through transition rules. Considering the above properties of CA and MAS, the geosimulation model can be based on three main characteristics:

i. **Management of Spatial Entities**: Geosimulation modelling is an object-based approach. Simulation models can directly address spatial building blocks (e.g. land, road).
ii. **Management of Spatial Relationships**: The spatial relationships between building blocks are explicitly modelled.
iii. **Management of Time**: Geosimulation models are dynamic. Objects' temporal behaviour can be considered as either synchronous or asynchronous.

To combine CA and MAS in geographic space, Benonson and Torrens (2004b) proposed a general framework known as GAS.

2.11. Geographic automata system

Geographic Automata System (GAS) refers to a variety of types of geographic automata. GAS automata are considered as a direct representation of real-world objects (Marceau and Benenson, 2011) characterised through the following components:

$$\mathrm{GA} \sim (\mathbf{K};\ \mathbf{S}, \mathbf{T_S};\ \mathbf{L}, \mathbf{M_L};\ \mathbf{N}, \mathbf{R_N}), \tag{2.13}$$

where **K** determines the automata types in GAS and three pairs of symbols refer to the spatial or non-spatial characteristic of automata and the rules applied by automata to change their characteristics. The first pair denotes a set of states, **S**, and state transition rules, **T**$_S$, that are associated with GAS. In the second pair, **L** dictates the location of automata in the system and **M**$_L$ denotes the movement rules for automata. In the third pair, **N** represents the neighbours of automata and **R**$_N$ represents the rules that govern changes of the automata relative to other automata.

2.11.1. Automata types

At an abstract level, GAS is composed of two types of automata: fixed and non-fixed geographic automata. Fixed geographic automata represent objects that are static and do not change their location over time (e.g. parks). Non-fixed geographic automata (e.g. vehicles) identify entities that can move over the simulation space over time.

A geographic system is usually formed based on both fixed and non-fixed automata types. For example, in the implementation of an object-based environment for urban simulation (OBEUS) software (Benenson and Torrens, 2004b), OBEUS consists of the two above automata types, an urban class with different object types (GIS layers), and a population

agent class. The actual geo-referenced data (e.g. parcels) can be extracted from the GIS vector layers of the urban class. The classes of agents (e.g. social, economic, and demographic) can generally be extracted from census attribute data.

2.11.2. State and transition rules

The state, **S**, describes the characteristics of the automata in the real-world. It can be formulated based on the characteristics of fixed and non-fixed automata. For example, in the urban environment, the value of an apartment can be specified based on the characteristics of the property and its neighbours and on the structure and neighbourhood population. Any variable or attribute can be used to characterise **S**. In this way, the transition rule describes the conditions allowing the value of real estate to change.

In the framework of the GAS, CA is artificially closed; cell state transition rules are driven only by cells (Benenson and Torrens, 2004a). In contrast, the states of objects represented by means of geographic automata are totally subject to their surrounding objects which can also be formulated on mobile geospatial automata (i.e. agents) that are responsible for controlling object states.

2.11.3. Location and movement rules

L describes the location of geographic automata and how they are placed in space. For fixed geographic automata, the geo-referencing process is performed by recording their position coordinates. These coordinates do not change over time. In contrast to the fixed automata, non-fixed geographic automata use a dynamic method to move. They need specific conventions regarding **L**. Moreover, the geo-referencing process can be performed directly or indirectly (Figure 2.14).

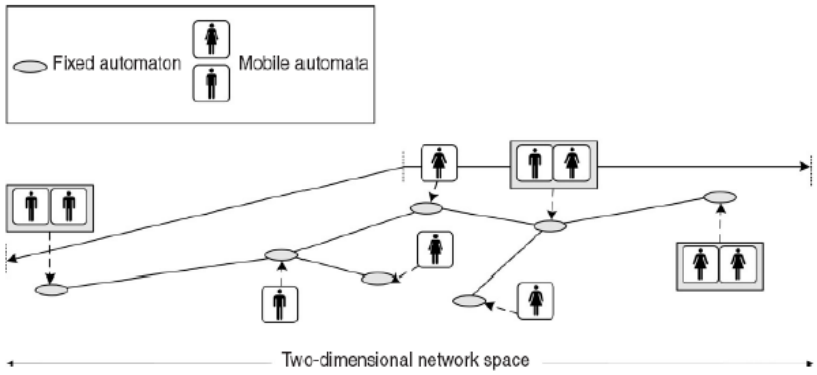


Figure 2.14. Two different geo-referencing rules: direct and indirect of fixed and non-fixed GA (from Benenson and Torrens, 2004b).

In the former case, automata use a list of coordinates to find their initial location in space. In the latter case, automata can be geo-referenced based on the location of the other automata. $\mathbf{M_L}$ rules allow the non-fixed automata to move in space. Different formulations of $\mathbf{M_L}$ can be used by automata, offering great potential for encoding the motion of traveling objects (e.g. vehicles or pedestrians).

2.11.4. Neighbours and neighbourhood rules

Neighbours **N** are determined based on a collection of objects which can affect geographic automata. Fixed geographic automata can use adjacency rules formulated on regular or irregular tessellations, and by the connectivity of network nodes or proximity. Moreover, these rules can be determined in terms of human-like measures such as accessibility or visibility. In contrast to the fixed automata, rules are usually dynamic in space and time for non-fixed geographic automata. $\mathbf{R_N}$ rules are usually formulated on geographic automata positions.

Despite the advantage that the OBEUS offers, it remains geometrically static in the repast vector extension in terms of its implementation of the specific residential segregation model. It can be argued here that the dynamic aspect of real-world entities in GAS is denied. That means the formalisation and exemplification of represented entities are essentially static in the GAS. This issue has already been addressed with the introduction of geographical vector agents VA (Moore, 2011; Hammam et al., 2007).

2.12. Geographical vector agents

In general, Vector Agents (VA) are a distinctive type of geographic automata (GA) (Torrens and Benenson, 2005), which can find their geometry and state, and interact with each other and their environment in a dynamic fashion (Moore, 2011; Hammam et al., 2007). The main properties of the VA are as follows:

i. The VA can represent any discrete geographic phenomena through an irregular (or regular) vector data structure.
ii. The VA can extract their own geometry and find their own location in vector space.
iii. The VA are born with a nondeterministic shape boundary that subsequently changes based on geometric rules and methods.

iv. The VA can find, store and manipulate their attributes through transition rules. They can change and update their states in terms of states, neighbourhood and geometry.

v. The VA have a dynamic neighbourhood structure that allows them to perceive objects in a simulation domain.

Hammam et al. (2007) used the VA for an urban scenario. They applied the Brownian motion (BM) algorithm to formulate the geometry and geometry rules for modelling the geometry of real-world phenomena. As the BM algorithm can only represent a small subset of real-world objects (e.g. star-shapes), they formulated a set of geometric methods to create a dynamic geometry in order to model a wide range of shapes for both real-world objects and phenomena. They used the following geometric methods to model the geometry of real-world objects:

i. **Midpoint displacement**: This operator allows the VA to create a new point at a new point based on a line segment as follows:

$$P_{new-md} = \frac{1}{2}\,(P_1 + P_2) + \mu\sigma_0 2^{-lh} \tag{2.14}$$

where P_1 and P_2 are the start and end points of the line segment being divided, σ_0 is the standard deviation of the Gaussian curve, l is the level of recursivity and h is the Hurst exponent which controls the roughness of an object.

ii. **Vertex displacement**: a new point can be created based on a random amount of bearing, α_{vd}. This amount is derived from a Gaussian distribution.

$$P_{new-vd} = P + \mu\sigma_0 \tag{2.15}$$

iii. **Edge displacement**: this method applies the above equation in a vertex displacement operator for two consecutive existing points. The magnitude and direction α_{vd} are equal for both points.

Figure 2.15 displays how the VA use these algorithms to evolve in vector space. The neighbourhood rules allow the VA to perceive other agents in vector space.

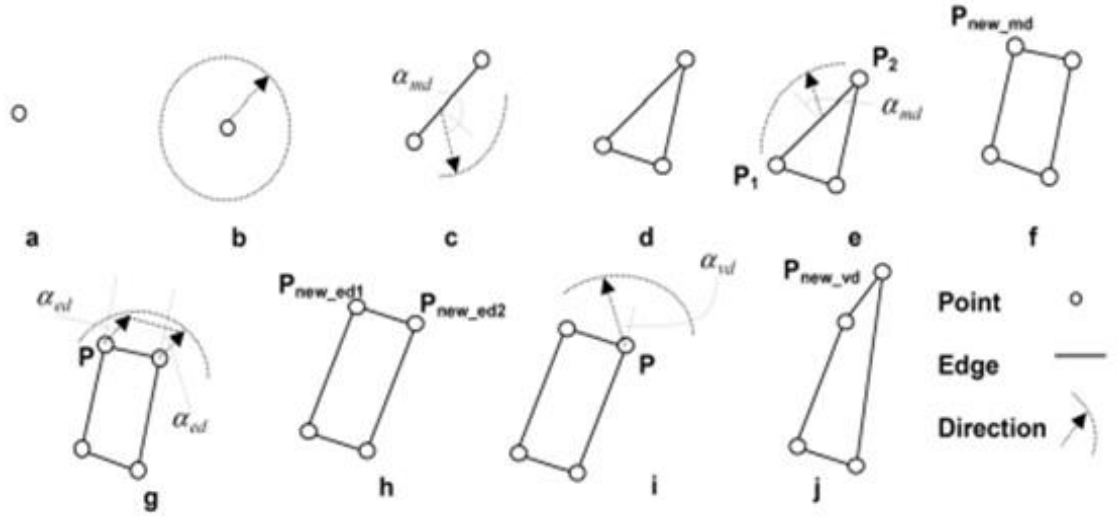


Figure 2.15. The geometry of VAs based on the mentioned methods. (a) Initialising by a random point, (b) allocating a second point by random displacement, (c, d) applying the random new point displacement and accomplishing a closed polygon, (e, f) choosing any edge randomly and applying the new point displacement, (g, h) edge displacement, (i, j) vertex displacement (from Moore, 2011).

Moore (2011) developed a generic Vector Agent library for agricultural implementation. Here, VAs were used to model a theory of agricultural land use. The proposed method was a more sophisticated model that implemented states (land use categories), state rules (the geographic and economic processes governing agricultural land use change) and neighbourhoods (based on a Delaunay triangular network connecting the centroids of the farm polygons), but not neighbourhood rules.

Moreno also implemented vector-based automata et al. (2009, 2010), with an emphasis on a dynamic neighbourhood for an object in their VecGCA model. This neighbourhood is not restricted by distance but defined by the state of neighbouring objects, and where neighbours can also affect the geometry of an object (hence, neighbourhood rules can be seen to be in effect).

2.13. Spatial agents and image analysis

Agents have also been used in remotely-sensed image processing such as image classification or feature extraction. Zhou and Wang (2008) used agents to extract the impervious surface areas from high resolution satellite images. Here, agents were used to improve the results of the segmentation process. After that, a modified classifier formulated the relationship between spectral bands and the variability in the training objects, and those objects to be classified were taken into account to classify the segmented objects. This

structure allowed a feature extraction algorithm to use more information for extracting impervious surface areas in image space. Samadzadgan and Mahmoudi (2010) proposed an agent-based method for automatic building recognition. In this research, spatial agents, including trees and buildings, were used to extract buildings from the LiDAR DSM at feature level based on the textural information. Espínola et al. (2012) used MAS and CA for image classification. In the proposed algorithm, three different groups of agents, namely training-based, spectral and contextual agents were applied to simultaneously take advantage of contextual and spectral information to classify an image.

Currently, Object Based Image Analysis (OBIA) is being recognised as a new paradigm in image analysis because it allows an enhanced description of meaningful objects and their context (Blaschke et al., 2014). In an OBIA framework, agents proved to be a competitive alternative in classifying segmented objects (Mahmoudi et al., 2013; Zhou and Wang, 2007). Zhong et al. (2014) then proposed to use agents to optimally control the merging of image objects, thus giving the agent some ability to control the geometry of the initially segmented objects. The ability of agents to perceive object geometry as just another attribute to be optimised was also recently tested by Hofmann et al. (2015), who gave agents the power to adjust the boundaries of segmented objects, thus enabling a more dynamic linkage between the semantic definition of a geographical phenomenon and its geometry and context.

The results of these approaches show that agents can be considered as an effective tool for remotely-sensed image processing (e.g. image classification or feature extraction). These methods use different strategies based on spatial agents to address the objects in a raster space (e.g. satellite image or DSM). However, these approaches have one characteristic in common: they focus on the procedure of an image analysis to model objects in image space. In other words, these methods lack the ability ***to directly address real-world objects in image space***.

2.14. Bridging the gap between image objects and geo-objects

All the image-based spatial modelling methods including pixel-based, object-based and agent-based in the previous sections have two main limitations in common when addressing real-world objects in image space:

- They use a static geometry to address real-world objects. They ignore the dynamic change of the physical characteristics of real-world-objects. Even with

the introduction of object-based approaches (Section 2.3), the formalisation represented objects are essentially static. For example, the geometry of image objects or segmented areas are determined based on a set of parameters (e.g. scale) specified by the human expert (see Equation 2.7-12).

- They use a sequential process to address real-world objects. In this way, the geometry and the state of objects are separately determined. For example, in the pixel-based approach, first we determine the thematic meaning of each pixel, then the geometry of the objects can be delineated (Section 2.2). In an object-based approach, the class of the objects is determined after finding the initial geometry of the real-world objects (Section 2.3). Although geographical objects with a unified identity are considered in object-based approaches, these objects are not related directly to the real-world objects due to the sequential process of segmentation and classification.

In this situation, the classical image classification algorithms solely rely on evidence derived from the image itself to address geo-objects. This means that these methods lack the ability to take full advantage of the other information (e.g. association or location in conjunction with other image elements) usually applied by the human interpreter to find and identify real-world objects in the image space.

These issues can be addressed through a geosimulation model called vector agents or VA. In this model, agents have the ability to control and alter their shape and attributes (Moore, 2011; Hammam et al., 2007) and evolve in accordance with the nature of the phenomena being modelled, as well as interact with other agents to capture spatial relationships and context. This thesis demonstrates the application of the VA presented by Moore, (2011) and Hammam et al. (2007) for remote sense image analysis. In the next chapter, we will define, formulate and test the main elements of VAs to address real-world objects in image space.

Chapter Three

Vector agent development for image classification

Abstract

In this chapter, the main elements of vector agents used to formulate and achieve image classification are examined. We expand the range of automata types based on the characteristics of geo- objects in the real world. We define and formulate a set of new geometric methods that allow the VAs to evolve in image space. In order to assimilate the thematic meaning and geometry of real-world objects, we define transition rules and how VAs or geo-objects can use them to find and update the set of attributes characterizing their state. To model the relations between geo-objects in image space, we propose a dynamic neighbourhood structure formulated on Euclidean distance. We test the VA model to identify and extract directly real-world objects from image space.

3.1. Introduction

VAs for image analysis are formulated on the same six basic elements defined by Hammam et al. (2007) and Moore (2011), which are geometry, geometry methods, state, transition rules, neighbourhood and neighbourhood rules (see Section 2.12). From a VA perspective, each object is considered to be an abstraction of a real-world phenomenon (Hammam et al., 2007). A VA is part of a collection of independent units that implement some functionalities of a geographic automata system (GAS) (Torrens and Benenson, 2005) (see Section 2.11). The elements of VAs for image analysis can be written as follows (adapted from Torrens and Benenson, 2005):

$$\text{VA} \sim (\mathbf{K};\ \mathbf{L}, \mathbf{M_L}; \mathbf{S}, \mathbf{T_S};\ \mathbf{N}, \mathbf{R_N}), \tag{3.1}$$

where,

- **K** is a set of automata types. VAs are considered to be objects that are evolutionary (its internal structure can change), static (they are not mobile), and elastic (their boundary can change) (Goodchild et al., 2007, see Figure 2.10).
- (**L**, $\mathbf{M_L}$) are the pair of terms that define the geometry of the automaton (e.g. point, line, and polygon), and the rules and methods that allow this geometry to evolve. The notions of set theory and planar graphs (see Section 2.5.4, Section 2.5.5 and Section 2.12) are applied to the geometry of the VAs.

$$\mathbf{M_L}: (\mathbf{S_t}, \mathbf{L_t}, \mathbf{N_t}) \rightarrow \mathbf{L_{t+1}}, \tag{3.2}$$

- **(S**, $\mathbf{T_S}$**)** are the pair of terms that define the automaton's state attributes (e.g. thematic meaning and the set of characteristics that allow such meaning to be determined). Each geographic object can find, evaluate and update its state based on transition functions in image space. These functions are formulated on a classifier such as the SVM or ML (see Section 2.2.2 and Section 2.7).

$$\mathbf{T_S}: (\mathbf{S_t}, \mathbf{L_t}, \mathbf{N_t}) \rightarrow \mathbf{S_{t+1}}, \tag{3.3}$$

- (**N**, $\mathbf{R_N}$**)** are the pair of terms that define the automaton's neighbourhood and its behaviour with respect to it (e.g. merging). Neighbourhood rules allow VAs to alter the state of other VAs in their neighbourhood. Geographic objects can perceive and interact with each other through metric spatial relations (see Section 2.1.3 and Section 2.9).

$$\mathbf{R_N}: (\mathbf{S_t}, \mathbf{L_t}, \mathbf{N_t}) \rightarrow \mathbf{N_{t+1}}, \tag{3.4}$$

In this sense, meaningful objects can dynamically change their own geometry and states, as well as directly perceive each other. This enables the real-world environment captured in the image to be modelled in a dynamic fashion similar to a human interpreter (Figure 3.1). In this way, VAs simultaneously segment and classify image objects from initial pixels to the meaningful objects (in an object-based approach or a feature extraction process) or training samples (in a pixel-based approach) based on an evolving process, enabled by an iterative scheme that involves constant interactions between all VA components. In Section 3.2, the main elements of the VAs are explored. The conceptual model of the VA model is then presented in Section 3.3.

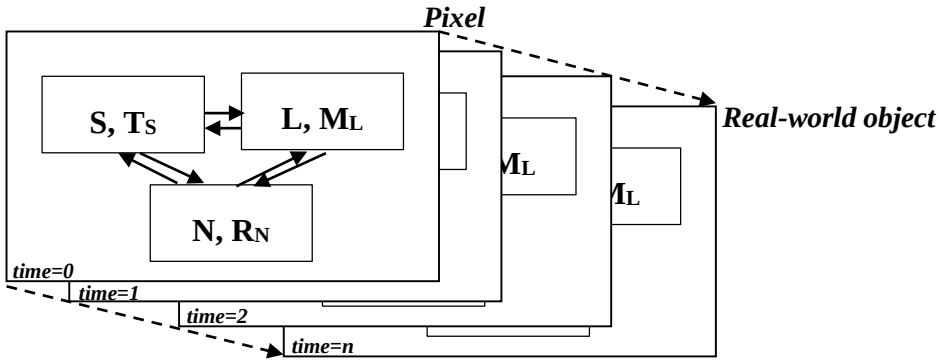


Figure 3.1. The components of each VA including state, transition rules, geometry, geometry rules, neighbourhood and neighbourhood rules, and their evolution through iteration.

3.2. VA Elements

3.2.1. Automata type

For automata types (**K**), Hammam et al. (2007) explored the scope of anchored automata with varying boundaries, which can be formalised as another type. The range of types can be expanded further and has itself been formalised by Goodchild and Cova (2007) in another context – the 'temporal variability in geo-objects' (see Figure 2.10). Focusing on the geometry and movement dimensions (ignoring internal structure regardless if there is variation within the geo-object or not) we can see the fixed ('stationary'), non-fixed ('moving') and varying boundary elastic elements (see Section 2.9). Using the full scope of the conceptual diagram, we can see each of the eight categories as belonging to an automata type (see Section 2.9).

3.2.2. Geometry

In accordance with the properties of raster datasets (e.g. satellite images) and the geo-objects within this space, we try to model geo-objects in a vector space with polygons. These polygons can then be applied to extract a specific object, generate training samples for a pixel-based classification and complete a full object-based image classification. Let $x = (\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \cdots, \boldsymbol{x_n}) \in \mathbb{R}^d$ denote a multispectral *d*-dimensional image with *n* pixels, $y = (y_1, y_2, y_3, \cdots, y_n) \in k$ an image of labels, $K = \{1,2, \cdots, k\}$ a set of *k* class labels, $\text{VA} = \{\boldsymbol{x}_i\}_{i=1}^{l}$, with *l* the number of pixels within VA. With this notation, **L** stores the vertices of the polygon that defines the boundary ∂X_{VA} of the VA. **L** uses an adjacency list algorithm to store the coordinate of each vertex. In this case, for each vertex *v* in polygon (graph), **L** is defined via all points of *v* to *v*. Thus, X_{VA} is a connected subset of the image of labels

formed by the pixels belonging to the VA. The coordinate data for each polygon is contained within the polygon's agent.

3.2.3. Geometry methods

Geometry methods ($\mathbf{M_L}$) are rules that can define a change of boundary coordinates of the polygon (elastic object, Chapter 2, Figure 2.12) in either a localised (acting only on specific boundary points) or global (scaling, shearing of the entire polygon) manner. These methods can control the movement of the polygon (translation, rotation of the polygon) and/or define the geometric nature of geo-objects. An example of locally applied geometric rules in a vector agent context is the fractal/Brownian motion process governing the elastic geometries of urban land units (Hammam et al., 2007) and agricultural land use units (Moore, 2011) (see Section 2.12). In the case of image classification, VAs use the following rules that are consistent with the raster data model:

1) The geometry of the VA is initialised as a vertex in the pixel centre;
2) Each vertex can generate another new vertex along four cardinal directions at a set distance corresponding to the raster cell size r (see Figure 3.2(a));
3) Each new vertex has three reference points;
4) An edge can be created if there is a vertex in the local neighbourhood (eight cardinal directions) of the new vertex and if the characteristics of the corresponding pixel meets some pre-defined criteria allowing it to belong to the VA as per state and transition rules;
5) Each edge has a maximum length of $r\sqrt{2}$, (the diagonal distance of a pixel).

Figure 3.2 illustrates the possible evolution of a VA defined as a triangle. Point *d* can be placed in seven different locations. Three resulting polygons from these seven cases are shown. In Figure 3.2, reference points are vertices 'a', 'b' and 'c'. The position of each point is specified by its *x*- and *y*-coordinates. The reference points allow the VA to specify the boundary of the VA in the order in which they are numbered. Thus, **L** will always be a sequence of the ordered coordinates.

Assume that 'a' is the reference point applied to create a new point by the VA. Thus, in the next iteration, **L** can be expressed by one of [a d_1 b c], [a d_2 b c] or [a b c d_7] configurations based on the above rules (Figure 3.2 (e), Figure 3.2 (f) and Figure 3.2 (j)). The VA uses qualitative shape relations (Schlieder, 1996) to define the above configurations. For

example, in Figure 3.2 (c) the configuration of points can be expressed by [a d_2 b c], [a b d_2 c] and [a b c d_2] based on point d_2.

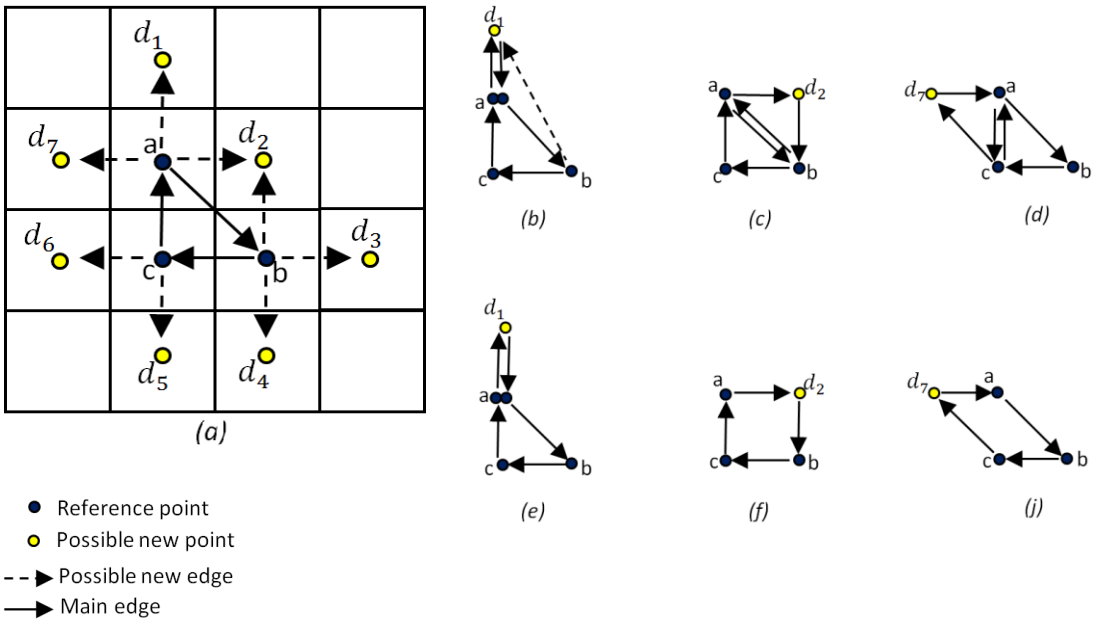


Figure 3.2. (left) The different possibility of a new vertex 'di' for the VA given existing vertices, 'a', 'b' and 'c', (right) the resulting geometry of seven of the possibilities.

The orientation of each face (e.g. [a b d_2] in [a b d_2 c]) is computed through the following equation:

$$[\mathrm{a\ b}\ \ d_2] = sgn \begin{vmatrix} x_\mathrm{a} & x_\mathrm{b} & x_{d_2} \\ y_\mathrm{a} & y_\mathrm{b} & y_{d_2} \\ 1 & 1 & 1 \end{vmatrix}, \tag{3.5}$$

[a b d_2] shows a counter-clockwise orientation if it is positive. The negative value of [a b d_2] corresponds to a clockwise numbering. For example, [a b d_2] shows a clockwise direction, in contrast to the orientation of the reference points [a b c]. The VA tests different configurations to find the faces, edges and vertices in terms of a winged-edge data structure (see Section 1.3.6.1). To enable a dynamic geometry for objects, a set of individual methods are defined, formulated and applied in terms of the above geometry rules.

3.2.3.1. Individual methods

To support the above process and changes dynamically, a set of methods are defined and implemented as follows:

- **Vertex displacement**: this places a new vertex in vector space (Figure 3.3(a), (b)) and connects two vertices together by two half-edges.

- **Half-edge joining**: this constructs a new edge based on a twin edge that is formed by two half-edges (Figure 3.3(c), (e)).
- **Converging vertex displacement**: two new edges are constructed to create a single new neighbouring vertex from two existing vertices (Figure 3.3(d)).
- **Edge removal**: this forms a new polygon by merging two existing polygons (Figure 3.3(f)).

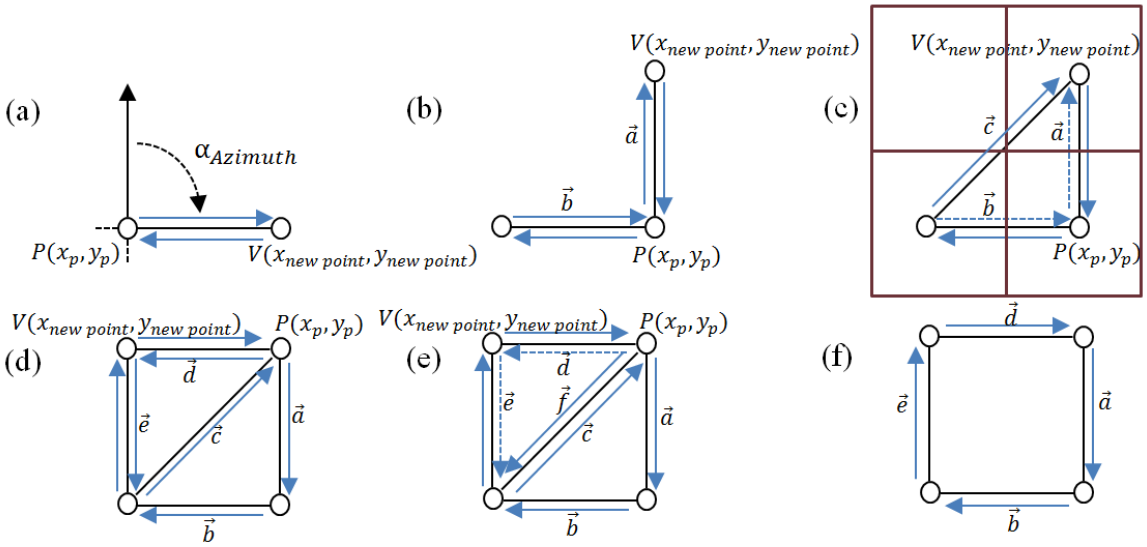


Figure 3.3. Four individual operations are required to change the image objects geometry: (b) vertex displacement, (c) converging vertex displacement, (d) edge joining, and (e) edge removal. The spatial relationship of the lattice point to the raster cell it represents is also made clear in 1(a).

These methods allow a VA to constantly change its geometry. Figure 3.4 illustrates how a VA implements these methods, and consequently produce its iterative evolution. First, a point coinciding with the centre point of a cell in the image being classified (all points in the vector object based classification are part of a lattice of all cell centre points) is automatically initialised in space. Next, the first edge is randomly constructed by finding a second point in the local neighbourhood. The new edge created consists of two half-edges. After that, one or more of the four strategies specified above are implemented, as applicable. For example, as a result of adding a vertex (1), it may become necessary to implement diagonal edge construction (2), convergence (3) and/or edge removal (4).

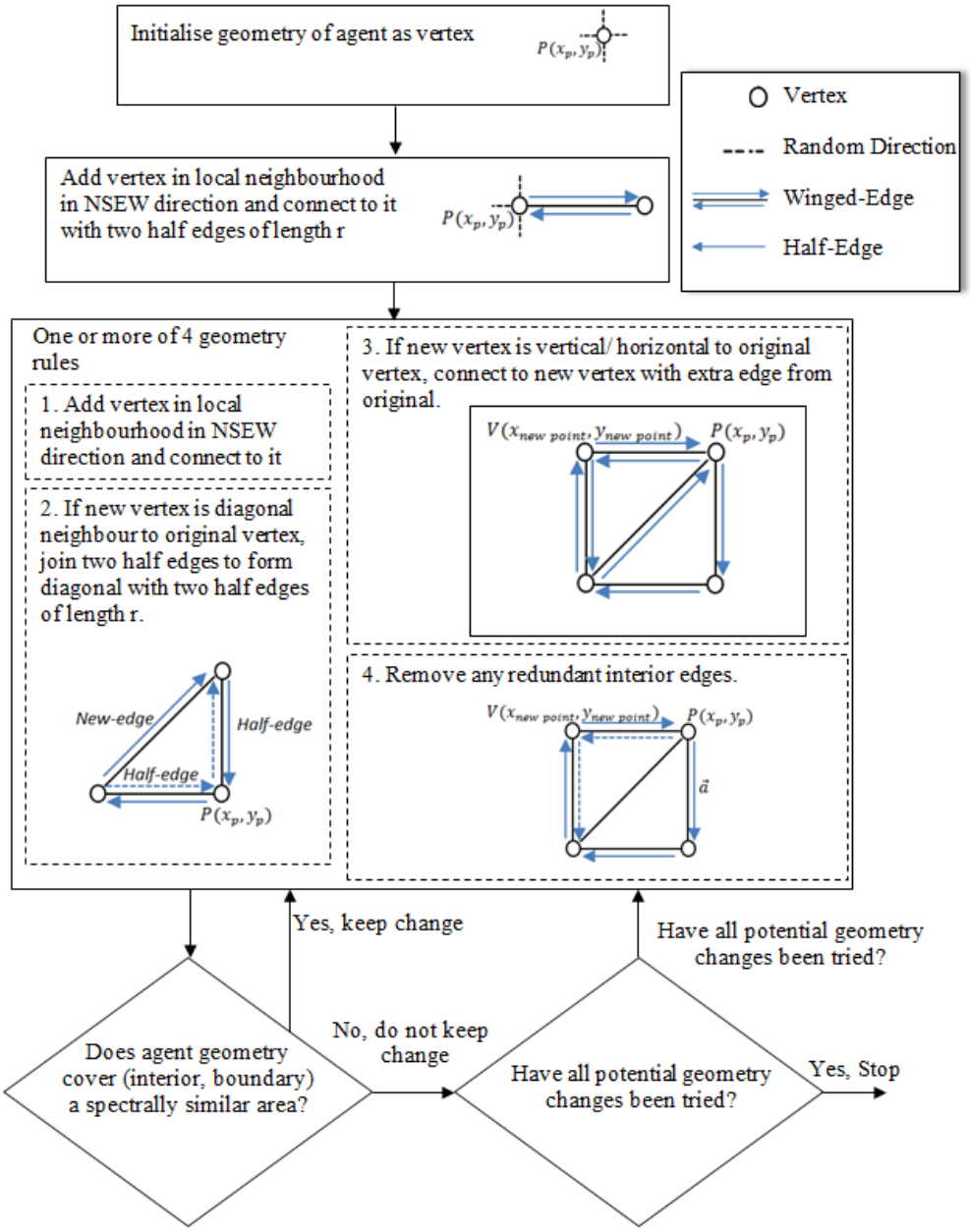


Figure 3.4. How an image object is born and changes through time in image space. Note that each point corresponds to the centre of a raster cell in the remotely sensed image being classified and is thus part of a regular lattice.

At each step, there is a check to ensure that the geometry boundary and interior cover spectrally similar pixels to suggest a homogeneous thematic class in the underlying raster image. An example of the evolutionary process of an image object from an initial point over 1000 iterations is illustrated in Figure 3.5. The appearance of the results reflects the stochastic nature of the process generating the geometry, as the polygons generated here are similar to polygons generated by geometric methods formulated on fractals (e.g. Hammam et al., 2007).

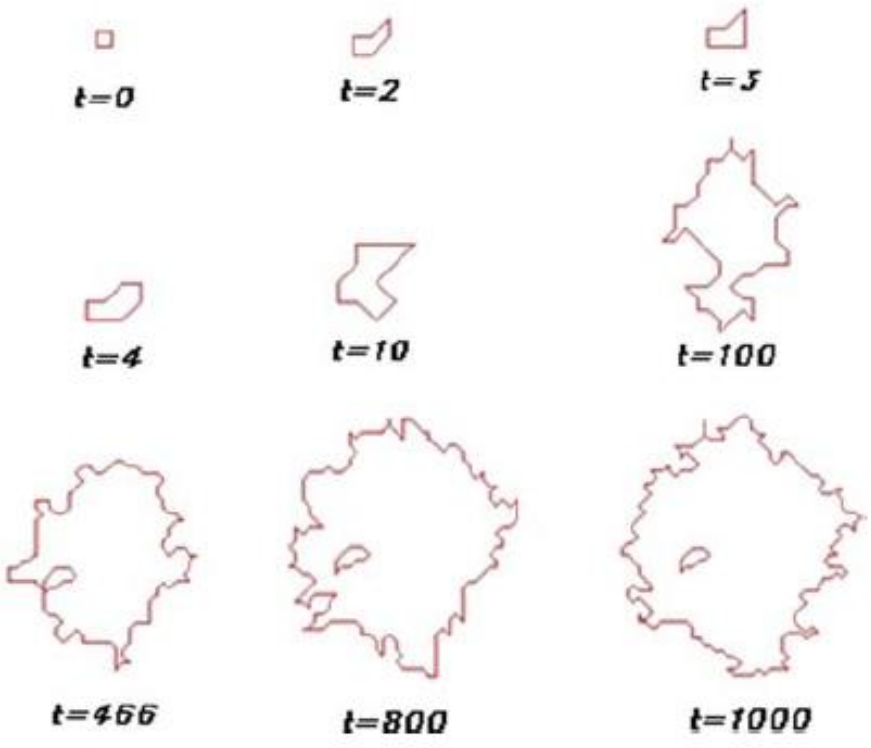
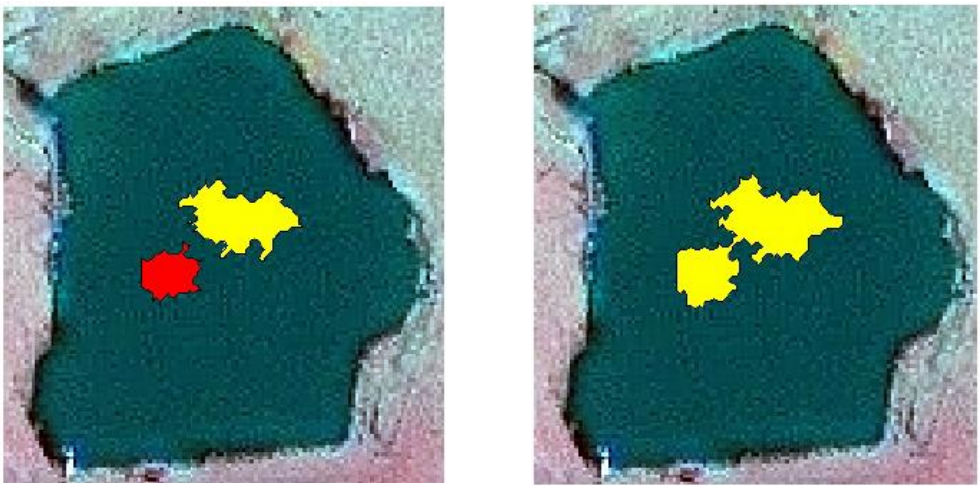


Figure 3.5. Simulation result for first 1000 time steps in the agent modelling shell Repast Simphony representing how an image object uses the four aforementioned operations to transform its geometry.

3.2.3.2. Interaction methods

Interactive methods are applied when VAs can affect the geometry of each other. These methods are activated based on the neighbourhood rules $\mathbf{R_N}$ that themselves depend on the state of the VAs. The interaction between VAs can lead to the birth, death or geometric change of two interacting VAs.

1) **Joining/Killing**: Two VAs becoming neighbours may involve by the merging of their respective geometry **L** and attributes **S** to form a single VA. This involves the killing of the second VA and happens when neighbourhood rules $\mathbf{R_N}$ for each VA given their respective **S** and $\mathbf{T_S}$ allow such interaction. Figure 3.6 shows an example of this joining/killing process, whereby two VAs with the similar state **S** become neighbours, and their transition rules $\mathbf{T_S}$ allow them to join. This yields a single VA with updated components **S**, **L** and **N** and the death of the other VA.



(a) (b)

Figure 3.6. Subset of an IKONOS image showing a water body. Evolution of two VAs classified as water (a), until they become neighbours and join to form a single water VA (b).

2) **Growing/Shrinking**: When two VAs become neighbours but their respective neighbourhood rules do not allow them to join given their state **S** and transition rules **Ts**, then one VA may grow into the other one when pixels at the boundary are found to be more likely to belong to the other VA according to its **Ts.** This operator removes and transfers a vertex from one VA to the other VA. Figure 3.7 illustrates a growing/shrinking process.

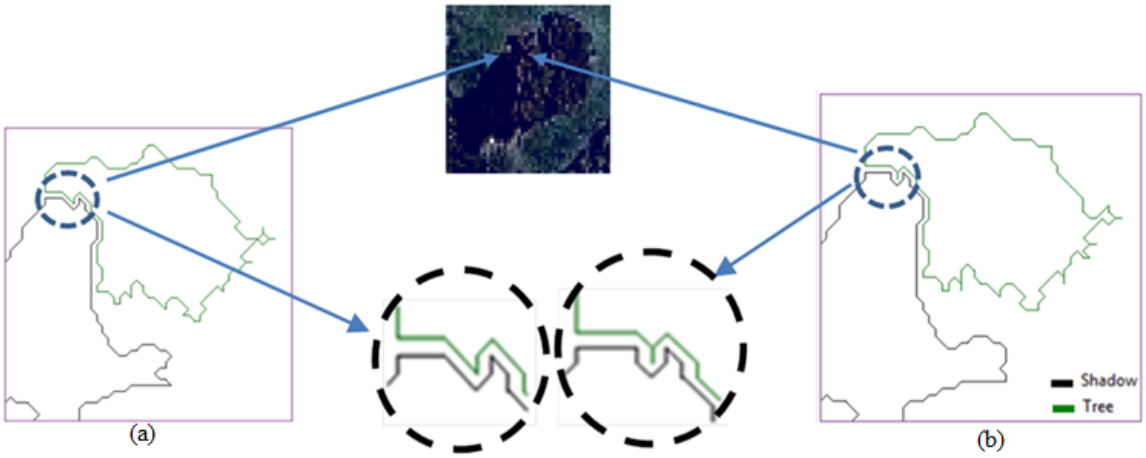
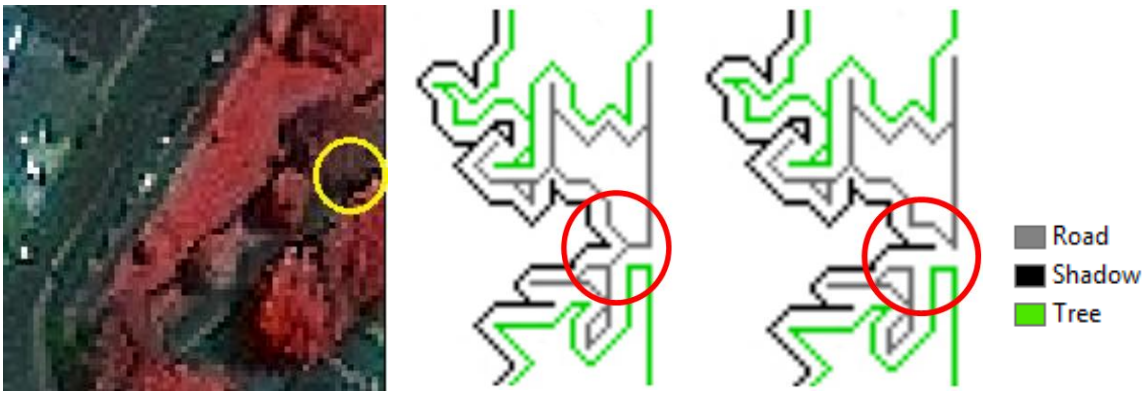


Figure 3.7. The growing of a shadow VA into a shrinking tree VA. This happens when a relatively dark pixel initially found to match TS of the tree VA is reconsidered in view of the nearby shadow VA and found to be more likely to belong to the latter.

This process might lead to one VA splitting into two VAs. In this case, a self-intersecting polygon is either broken into two polygons or a set of polygons (Figure 3.8).



(a) (b) (c)

Figure 3.8. An example of splitting where the road VA is divided into two. A relatively dark pixel initially found to match the road VA (b) changes to shadow, leading to the birth of a road VA (c).

3.2.4. State

The state of the vector agent is stored alongside the polygon geometry data within the agent architecture. In the context of image classification, this would normally be the classification label, but may also include the following parameters:

i. **Spectral descriptors:** These can be expressed based on the mean values of each band as well as a covariance matrix. Spectral indices, such index ratios, can also be used to establish the foundation of classification (Mahmoudi et al., 2013).

ii. **Textural descriptors:** Textural descriptors can be measured based on the grey value relationships between each pixel and its neighbours in the polygon of the VA.

iii. **Contextual relationships:** Definitions of contextual relationships between neighbouring objects can improve the accuracy of object based image analysis results. This is performed based on the neighbourhood rules. Context based methods operate at the level of image understanding, analysing the whole image to retrieve the required information (Peets and Etzion, 2010).

iv. **Structural descriptors:** Calculating suitable structural descriptors based on the geometry of a VA, namely **L**, provide a tool for refining the results of the classification process at object level.

There is also potential to formulate states based on ontology of the real-world objects. In this case, an object can be described based on an integrated collection of facts from the above parameters. For example, a lake state may be constrained to a smoothly curved boundary geometry with a specific neighbour (e.g. wet sand). A state can also be associated with a

learning algorithm during the evolving process. In this situation, a state is determined by performing a particular task.

These state descriptions allow image objects to convey more information compared to individual pixels. The descriptors form new features by which the classification can be relied upon to discriminate classes that would otherwise be similar based solely on spectral description. Being dynamic in nature, each VA updates its attributes at each iteration and possibly changes its class when transition rules allow. This procedure is performed through transition rules.

3.2.5. Transition rules

VAs use transition rules (**T**s) to find, evaluate and update their classes and attributes. Depending on the application of VAs in image analysis and available datasets, there are three main strategies used to define transition rules:

1) The spectral signatures of the classes of interest are the only available information. In this case, spectral descriptors are the fundamental information used by VAs to implement the transition rules based on a classifier algorithm such as the SVM or ML (see Chapter 4 and Chapter 5, the application of VAs for unsupervised and semisupervised image classification). As the SVM classifier does not make any assumptions on the underlying data distribution, in this thesis, the VA model uses the SVM classifier to formulate the transition rules.
2) The information available is the spectral signatures of the classes of interest and also a set of facts about those classes. Here, different combinations of spectral, textural, contextual and structural descriptors can be applied to address objects in image space (see Chapter 6, the application of VAs for object-based image classification).
3) The ontology of the classes of interest can be applied to describe geo-objects. For example, a building is defined as an elevated object with a very steep slope at its edge (see Chapter 7, the application of VAs to extract 3D roofs).

To find the initial state, VAs can either use the spectral signature of the classes of interest, a set of specified rules based on the characteristics of real-world objects or a combination thereof. Then, VAs use the evaluation process to assess whether a neighbouring pixel is eligible to be captured. Depending on whether the pixel already belongs to a VA or not, VAs use different transition rules and geometric methods. In the former case, where a pixel belongs to another VA, VAs utilise interactive geometric methods (e.g. growing/shrinking

(Figure 3.6(b)). In the latter case, where the candidate pixel is an isolated pixel, VAs utilise only the individual geometric methods (e.g. Figure 3.6(a)). In updating the process, VAs use the membership functions and/or various reasoning rules to update their attributes based on new attributes. These attributes can also be obtained from other VAs or their environment. Contextual descriptors apply the knowledge that the VA has of its neighbours via the **N** component. Specific contextual rules may allow the transition of VAs from one class to another on the basis of its context. For example, in an IKONOS image (a 4 band multi-spectral dataset), water and shadow are two classes spectrally similar but can be discriminated based on contextual information, since shadows require the proximity of an elevated class, such as trees. As such, a contextual rule will allow a water VA to transition to a shadow VA if tree VAs are among its neighbours.

3.2.6. Neighbourhood

The proximity between VAs is explicitly defined based on the metric relationships (Chapter 2, Section 2.4.3). This metric is formulated on the minimum distance between two planar sets in a Euclidean space. Thus, the distance between $VA_{1,t}$ and $VA_{2,t}$ is defined as:

$$d\left(VA_{1,t}, VA_{2,t}\right) = \inf\left\{d(x_1, x_2): x_1 \in X_{VA_{1,t}}, x_2 \in X_{VA_{2,t}}\right\}, \tag{3.6}$$

where $d(x_1, x_2)$ is the Euclidean distance in space between pixels x_1 and x_2. For each VA, the pair-wise distance with all other VAs is stored in its neighbourhood component **N**. Alternatively, neighbourhoods could be defined by Delaunay triangulation of polygon centroids (Hammam et al., 2007).

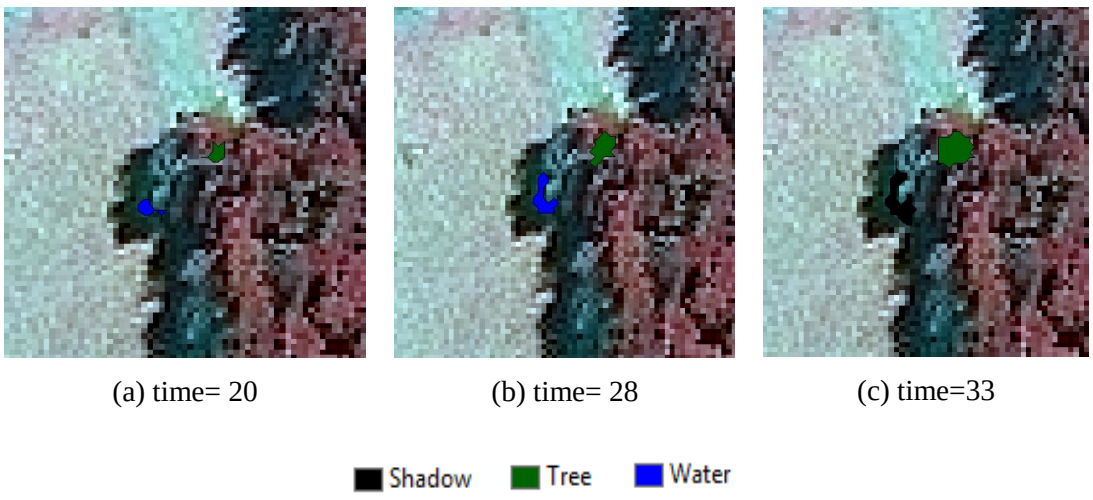
3.2.7. Neighbourhood rules

Neighbourhood rules can alter the neighbourhood of an agent from time step to time step. There are different ways to define neighbourhood rules. For example, Moore (2011) used a Delaunay triangular network to form the basis for neighbourhood operations. In this thesis, neighbourhood rules are explicitly defined by the Euclidean distance between VAs in the modelling space. This means that there are two kinds of neighbourhood rules, namely adjacency rules and remote rules. The set of VAs adjacent to $VA_{i,t}$ is defined as:

$$d\left(VA_{i,t}, VA_{j,t}\right) \leq r\sqrt{2}, \tag{3.7}$$

where $VA_{j,t}, j \in \mathbb{N}$. A basic adjacency rule triggers the joining/killing geometry method when two adjacent VAs have the same class. Alternatively, two adjacent VAs of different classes may enter a state of negotiation for pixels at the boundary. This may lead to a growing/shrinking process (Figure 3.7). Furthermore, adjacency rules allow a VA to change the state of its direct neighbours to a class that is judged more consistent with its own state at iteration *t*. For example, a VA of the forest class can change the class of the water VA if it is next to the forest VA.

Remote rules are unrestricted in terms of Equation 3.7. In this case, remote rules enable VAs to alter the state of other VAs in an extended neighbourhood despite the absence of a shared boundary. These rules allow, for example, a VA of class building to reclassify all VAs of class pasture into class recreation park (within a given distance) if there are many other VAs of class building in the same area. Another example is shown in Figure 3.9, where a VA of class tree is growing next to a VA of class water up to the point of becoming adjacent. In this case, a $\mathbf{R_N}$ rule has been defined to change the water VA into a shadow VA because the latter exhibits a similar signature, but is more likely to be close to an elevated feature such as a tree. The notion of dynamic neighbourhood proposed by Moreno et al. (2009, 2010) can also be applied.



(a) time= 20 (b) time= 28 (c) time=33

Figure 3.9. The interaction between two VAs in terms of adjacency rules whereby a tree VA becomes adjacent to water VA, forcing the water VA to change to a shadow (c).

Both adjacency and remote neighbourhood rules allow for the explicit coding of rules related to the process of association, whereby knowledge of the meaning of an image object informs that of another connected, or nearby object. However, the dynamic segmentation/classification approach enabled by the VA departs from traditional

classification schemes whereby such contextual information is used and applied a posteriori during a re-classification step, rather than dynamically in a 'classify as you go' approach. The elements of the VA are summarised and displayed in Figure 3.10.

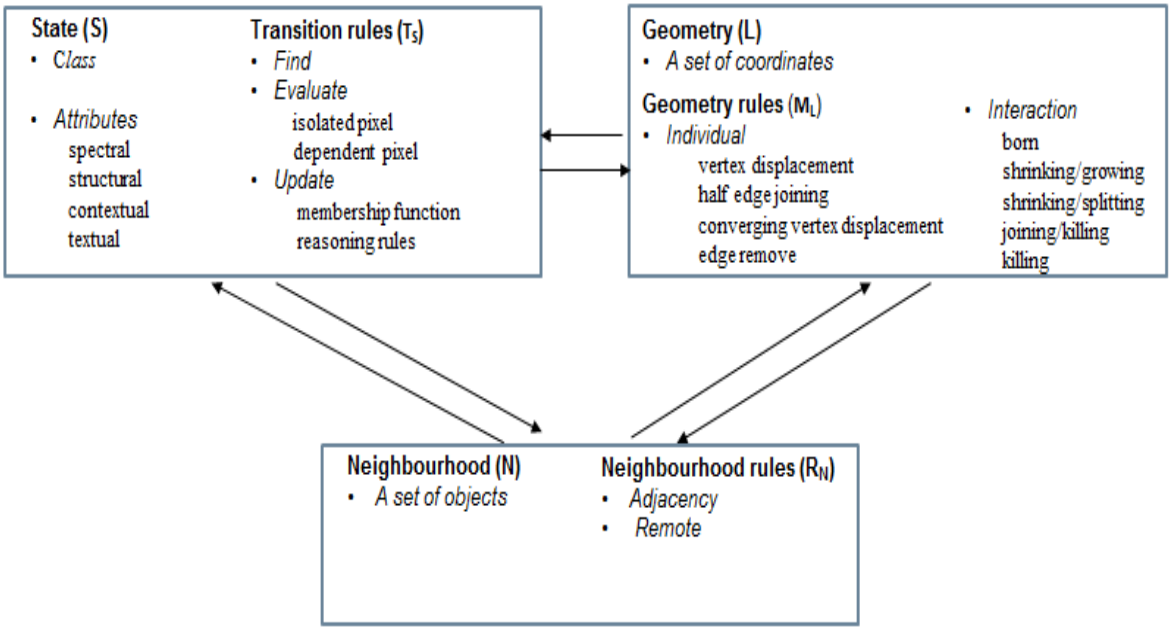


Figure 3.10. The detailed components of each VA from Figure 3.1.

3.3. Model Architecture

The VA applied in this thesis is a goal-oriented agent. Figure 3.11 displays the main components of an agent. In this context, VAs can perceive information from feature space, passive objects and other agents in the simulation environment, make decisions about actions based on their goals, and subsequently perform those actions. In fact, the agent's behaviour is to find a way to achieve its goal. As shown on Figure 3.11, agents are not capable of total autonomous actions because they do not have the means to learn from experience yet.

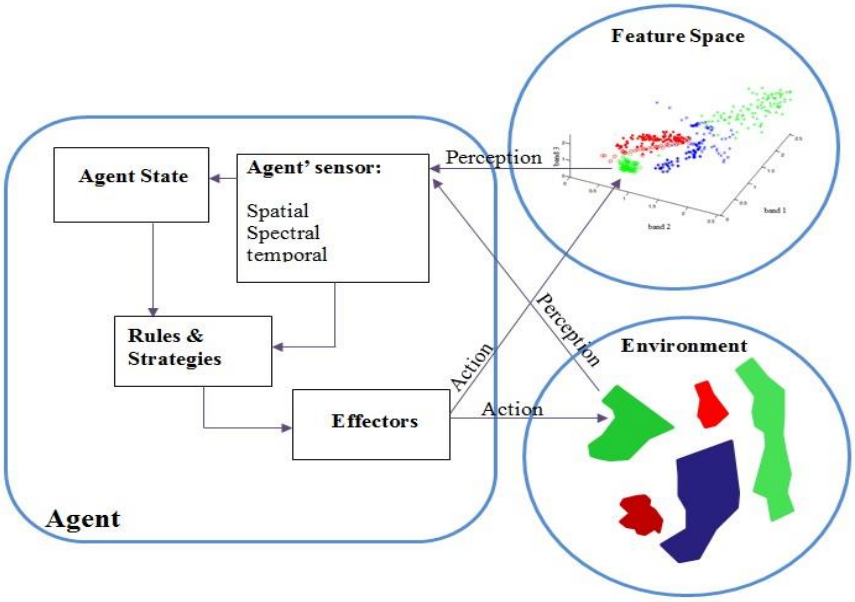


Figure 3.11. Schematic diagram of the model architecture of the VA including state, sensor, rules and strategies, and effectors.

3.3.1. Agent's sensor

VAs perceive their environment through sensors. The agent's sensor – its measuring instruments – includes spectral, spatial and temporal information that is collected when the observations are made. These observations are collected from the simulation environment and feature space. Moreover, VAs are capable of perceiving other VAs in the environment. To perceive the simulation environment and feature space, VAs use the following classes:

i. **VATransitionRule**: This class has methods that VAs use to evaluate pixels in a feature space. For example, in Figure 3.12, VAs use a SVM to evaluate the class of an isolated pixel.
ii. **VANeighbourhoodRule**: This class includes methods that enable VAs to perceive other VAs in a simulation space.
iii. **VAGeometryFactory**: This class has a method that allows the VA to evaluate a candidate point based on geometric rules in a simulation space.

3.3.2. Agent's state

The internal state of each VA is determined by the class and geometry of real-world objects. Depending on the application of VAs for image analysis, VAs also maintain additional information. For example, in the image classification process, VAs store their signatures and update it at each increment. The signature of a VA is defined by the pixels within the VA.

In this case, the VA simply uses a table to store this information (see Chapter 4, Chapter 5 and Chapter 6).

3.3.3. Rules and strategies

Rules and strategies keep the agent behaving in a structured way and allow it to perform a task based on a set of decision rules. These rules can be divided into two groups: internal rules and external rules. In the former case, rules are defined by the characteristics of the vector agent, such as the creation of dynamic geometry with different operators (see Section 3.3). External rules include those defined by a human expert based on the application of VAs. These rules are usually embedded within VANeighbourhoodRule, VATransitionRule or VecAgent classes. For example, to create the training objects for supervised image classification, the area of VA should be less than the specific threshold.

3.3.4. Effectors

Effectors allow the VA to act upon the simulation environment and the feature space. In the simulation space, actions of a VA can include changing its geometry or class. VAs can also affect the geometry and state of other VAs. Furthermore, VAs can affect the pixels in the feature space through their effectors. The main classes compromising the vector agent's effector (the actions of the VAs) are as follows:

i. **VANeighbourhoodRule**: There are some methods within this class that enable VAs to affect the geometry or state of other VAs.
ii. **VATransitionRule**: This class contains the method that VAs use to update their classes during a simulation process.
iii. **VAPolygon**: A VA uses this class and its method to change its geometry.
iv. **VALineString**: This class and its method are applied by VAs to implement the half-edge joining method on polygons.
v. **VAPoint**: VAs use this class and its method to create a random point based on one of four main directions.
vi. **VAGeometry**: This class and its methods provide essential geometric operators (e.g. merging/killing) for the VAs.

3.3.5. Environment

The environment of any simulation for vector agents is a continuous vector space with predefined x, y coordinate limits determined on the input raster datasets. In this thesis, the

simulated environment is static and cannot change while the vector agent is deciding on an action. The environment is accessible, and therefore, VAs have access to the complete, accurate and up-to date state of other agents in the environment at each time step. Moreover, VAs can perceive the feature space. In this context, this means that VAs can extract spectral patterns in the feature space through their actions in the modelling space.

Figure 3.12 shows the UML class diagram of the main components constructing the VA model, which is an example of the application of the VA model for the GEOBIA approach. Figure 3.12 only includes the main methods of each class (see Appendix A for more details). There are three types of agents in the proposed approach: SimpleAgent, MakerAgent and VecAgent.

i. **SimpleAgent**

SimpleAgent provides initial information on input data for all agents. This information includes: the signature of the classes of interest, the input raster datasets, and any other information (e.g. the structural information of real-world objects). There are two offshoots of SimpleAgent: MakerAgent and VecAgent.

ii. **MakerAgent**

MakerAgent exists in a hierarchical structure, with SimpleAgent as the top level agent, which does not have a geometry in itself. MakerAgent first creates the VAs, placing them into context through the generateVA method. MakerAgent is also responsible for facilitating the coordination between VAs. To do this, MakerAgent applies the trackVA method.

iii. **VecAgent**

The instances of the VecAgent class, namely VAs, use the VAGeometry, VAPoint, VALineString, VAPolygon and VAGeometryFactory classes (Figure 3.12) to determine and change their geometry stored in the py variable. At each increment, VAs update the polygon variable.

iv. **ContextCreator**

VecContext is the class for running a simulation in the vector agent model. The class has the functionality to display and represent the environment in a dynamic fashion with all agent's geometrics. There is one context, VecContext, to which all agents belong.

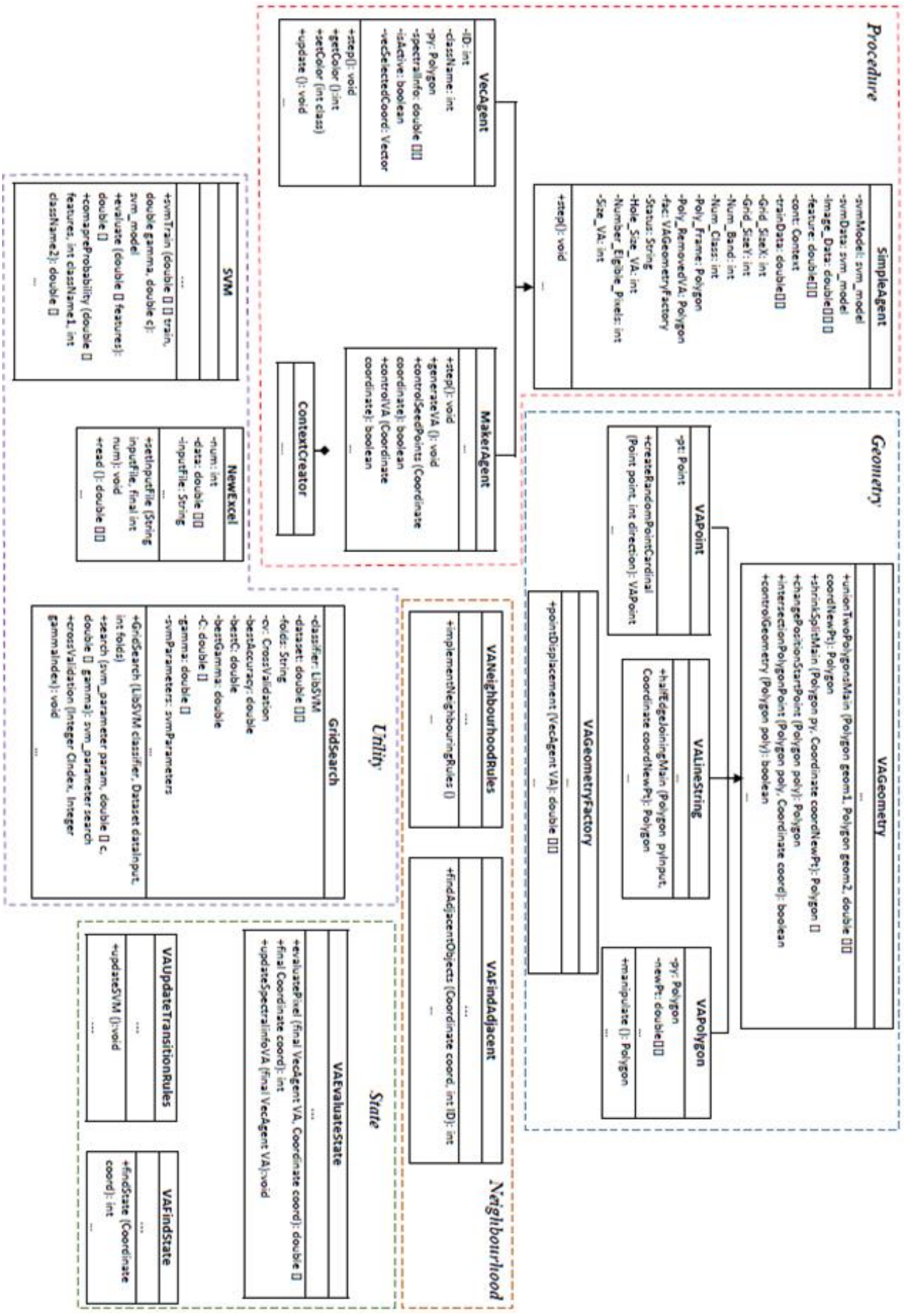


Figure 3.12. A UML diagram shows the abstract level of the application of VAs for object-based image analysis.

3.3.6. Processes

The intelligent agent-driven classification process encompasses adaptation, updating image objects over time based on feedback from the vector and raster space, their internal knowledge and knowledge of their neighbours. By adaptation, image objects can continually alter classes based on the contextual information in the simulation domain, pixel value in real-world space and class of image objects. In addition, a parallel mechanism is applied to perform and control a network of the image objects in an image analysis process.

This modified method, which is also enabled by iteration and adaptation, can simultaneously cope with image segmentation and classification in tandem. In this case, image objects are efficient and robust than they would be if created by a sequential process. Other important aspects of the action of vector agents in segmentation and classification that impacts geometry, states and neighbourhoods include the potential ability to work with vague phenomena (i.e. modelled by fuzzy logic) and incomplete information (some examples are given above). Another aspect gleaned from the above is the ability to implement an adaptable (spatial and possibly temporal) scale through rules. In the next chapters, we will test the VA model for different applications of image analysis.

Chapter Four

Vector agent model for unsupervised image classification

Abstract

Unsupervised image classification methods usually use the DNs of pixels to classify an image in the spectral domain without using training samples. This chapter presents a novel unsupervised approach based on the VA model to classify a High Spatial Resolution (HSR) image without human supervision. The proposed method can be summarised in four main steps: selection, creation, identification and classification. In the selection step, we first utilises an unsupervised algorithm (e.g. K-means) to cluster the image, assuming that the desired number of clusters is known. The algorithm then uses the mean and standard deviation of each cluster to select a set of reliable samples for each cluster. The spectral information of these samples allows the VA model to apply an SVM classifier for formulating the transition rules. In the creation step, the method employs the VA model to identify a set of reliable samples from the image space. The dynamic structure of the VA enables them to recapture the spectral information of each cluster through a set of polygons in the image space. In the identification step, the algorithm then applies the properties of the VA-generated polygons (e.g. shape, size, and covariance) to automatically rename the VA-generated samples. In the classification step, the spectral information of the updated and selected VAs (e.g. covariance) allows the method to utilise a classifier algorithm, such as Maximum Likelihood (ML), to label the remaining pixels. The preliminary results–Overall Accuracy (OA) and precision– show the enhanced capability of VAs to classify a HSR satellite image.

4.1. Introduction

Image classification refers to the process of converting a set of pixels into a number of thematic classes or meaningful objects (see Chapter 1 and Chapter 2). When considering pixels as the underlying unit for image classification, there are two main types of

classification: supervised and unsupervised. In the former case, supervised methods use training data to classify an image. As the process requires human input, it can be time consuming, error-prone and expensive (Chi et al., 2008).

In contrast, unsupervised methods (e.g. K-means or Fuzzy c-means) simply use the DNs of the pixels to classify an image without the need of training data. The K-means is an iterative algorithm that starts with K cluster centres in the feature space. Cluster centroids can be initialised to random values or derived from a priori information (Omran and Engelbrecht, 2006). Each pixel in the image is then assigned to the closest cluster. At each iterative step, the centroids are recomputed according to its associated pixels in feature space. This process is repeated until convergence is achieved (see Section 2.2.1).

Compared to supervised approaches (which need labelled samples), unsupervised K-means methods do not require as much intervention (Tso and Olsen, 2005; Gao, 2008) and a priori information (Duda et al., 2012) to classify an image. However, K-means methods have the following drawbacks:

i. Results depend on the initial values and suboptimal partitions are frequently found.
ii. The results depend on the value of K.
iii. The K-means algorithm is also an ill-posed problem when compared to the supervised approaches, due to the absence of training samples (Banerjee et al., 2015; Alajlan et al., 2011).

The above restrictions can cause difficulty when the clusters have poor contrast, overlapping intensities and noise (Ghaffarian, 2014). This limitation is usually addressed by using a soft clustering algorithm, in which pixels belong to multiple clusters and are characterised by varying degrees of membership values (Dopido et al., 2012). In this case, instead of making a hard assignment of labels, the density models or clusters are analysed as probabilistic distributions (Kearns et al., 1998).

Several studies have already addressed the applications of soft clustering algorithms, such as Fuzzy c-means (FCM), to reduce noise (Yang and Hung, 2012), improve the degree of automation (Ghaffarian, 2014), increase accuracy (Hung et al., 2011), or find the optimum number of clusters (Zanaty and Afifi, 2013). FCM is a method of clustering for which each pixel is defined by a fuzzy membership, rather than a crisp value. Processes of an FCM method, including initialisation, iteration, and termination, are the same as the K-means

algorithm (see Section 2.2.1) except FCM methods use a weighted centroid based on probabilities during the iteration process.

In the context of soft clustering algorithms, several studies show that the use of spatial information can not only improve the results of conventional soft clustering methods, but also remove the noise. There are various strategies available, such as utilizing a local window in image space (Zheng et al., 2014; Liu et al., 2006), segmentation process (Tyagi et al., 2008) or features (e.g. edges) (Li et al., 2013). For example, Tyagi et al. (2008) used the results of a segmentation process to take advantage of spatial-context information via an FCM algorithm to identify the initial clusters. They employed an Expectation-Maximisation (EM) algorithm within a Bayesian framework to estimate the statistical parameters of each cluster and then to classify the image. Li et al. (2013) proposed a modified FCM method, whereby the weights of pixels within local neighbour windows were formulated on the edges extracted by Canny edge detection. Zheng et al. (2014) argued that the use of spatial distances and membership values of neighbouring pixels, along with the quality of the centre pixel in a local window could affect the fuzzy membership values of pixels within clusters. They demonstrated that their method could improve the robustness and noise insensitiveness of conventional FCM methods.

These methods show that the use of spatial information, along with FCM and EM algorithms can reduce not only the effect of crisp boundaries between clusters but also salt and pepper effects and the noise within clusters. However, their results are subject to an implicit assumption that all clusters to be mapped have predictable behaviours in feature or image space. For example, in EM, the clustering algorithms use Gaussian distribution in the feature space for the clustering process (e.g. Tyagi et al., 2008). To take advantage of the spatial information, conventional FCM methods use a neighbourhood system of fixed size (e.g. Li et al., 2013; Zheng et al., 2014), but there is no specific rule to determine what shape should be used and what size should be given in order to obtain optimum results. Thus, pixels may be assigned to an erroneous cover identity in the output results if the above assumption is violated.

Additionally, conventional soft clustering algorithms are sensitive to initial conditions (e.g. the cluster centroids in the FCM method or Gaussian components) (Tao et al., 2016; Ghaffarian, 2014). In this case, an improper initialisation may lead to suboptimal solutions in FCM algorithms (Banerjee et al., 2015; Omran and Engelbrecht, 2006), overfit the data or a reduction of model flexibility in EM -based algorithms (Tao et al., 2016).

From the above discussion, it can be concluded that the use of some information about clusters is necessary to improve the clustering results. In this case, unsupervised methods are utilised via a two-stage process to label pixels in the feature space. These methods first apply clustering algorithms to select a set of reliable training samples from raster dataset. In contrast to supervised methods, these samples are generated without human intervention. Next, a classifier (e.g. ML), which is trained based on the selected samples, is used to label remaining pixels in the classification step. For example, Mukhopadhyay and Maulik (2009) used a multiobjective fuzzy clustering to identify the reliable samples for each cluster in the feature space. The authors then applied an SVM classifier trained based on the selected samples to label the remaining pixels in the classification step. To select the reliable samples in the spectral domain, Banerjee et al. (2015) proposed an advanced method based on the ensemble cluster method. They then used an EM algorithm to find the optimum parameters for each cluster, such as covariance. To classify the image, the authors utilised an ML algorithm. The results show that the use of some information (e.g. covariance) about the clusters can improve the clustering result.

In this chapter, the flexibility of VAs is applied to extract a set of training polygons for each cluster from the image space, just as a human interpreter would do in a supervised approach. ***In contrast to the conventional above approaches, the proposed method extract training samples based on not only the spectral information (e.g. covariance) in the feature space but also the spatial information (e.g. location, shape and size) in the image space***. The algorithm then employs the spectral information of the eligible VAs to train an ML algorithm for labelling the remaining pixels in the classification step.

In section 4.2, the proposed method is presented. The results are discussed in Section 4.3. A short summary and ideas for further work are presented in Section 4.4.

4.2. Proposed method

The proposed approach consists of four main steps: selection, creation, identification and classification. First, a set of reliable samples for each cluster is selected from the image clustered by a K-means algorithm. This method uses the mean and variance of the clusters to identify initial samples. These samples allow the VA to formulate its transition rules and find their state in image space (see Section 3.2.5). Next, the method employs the VAs to extract a collection of training objects for each cluster from the image space. The dynamic structure of the VA enables them to identify the training objects (polygons) based on the

rules which are usually applied by a human interpreter in a supervised approach (Gao, 2008, pg. 270; Richards and Jia, 2006, pg. 199). These objects are then applied to select the eligible labels from the initial labels in the identification step. This is performed based on the geometric and spectral information of the VA-generated samples via a separability matrix. Finally, in the classification step an ML classifier is trained on the spectral information of the updated VAs and employed to label the remaining pixels.

4.2.1. Selection process

Let $X = (\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \cdots, \boldsymbol{x}_n) \in \mathbb{R}^d$ denote a multispectral *d*-dimensional image with *n* pixels (or vectors) and $Y = (y_1, y_2, y_3, \cdots, y_n) \in K$ denotes an image of labels where $y_i \in K$, $K = \{1,2, \cdots, k\}$ is a set of *K* class labels. Let us assume that *K* is already known, whereas the labels are not. Let $Z = (z_1, z_2, z_3, \cdots, z_n) \in C$ represent an image of labels where $z_i \in C$, $C = \{1,2, \cdots, c\}$, $|C| > |K|$, $|C| = K \times w$, $K \subset C$ and $w \in [1.5, 2.5]$. Thus, the size of *C* is always greater than the number of desired clusters *K*. By partitioning the feature space in more detail, the VA model can test a number of different starting points. This can reduce the level of sensitivity of the proposed method to the results of K-means algorithm. This strategy also allows the method to use different partitional clustering algorithms (e.g. K-means), regardless of their results. Additionally, the method can use a multimodal distribution for each cluster to label pixels in the classification step.

A K-means clustering is first used to cluster pixels in which the number of clusters is set to *C*. The following rule is applied to identify the initial training samples or labelled samples for each cluster.

$$\bar{p}_l - \lambda \times \sigma_{p,i} < p_{c,i} < \bar{p}_l + \lambda \times \sigma_{p,i}\,, \tag{4.1}$$

where $\bar{p}_l$ and $\sigma_{p,i}$ are the mean reflectance and standard deviation in band *i* of all pixels in each cluster, respectively. λ is a constant set between [0.01,2]. To select the reliable samples, $X_L = \{(\boldsymbol{x_i}, z_i)\}_{i=1}^{l}$, which are very close to the centroid of each cluster, the algorithm uses a minimum distance algorithm. l is the number of initial training samples, $z_i \in Z$ and $\boldsymbol{x_i} \in X$. VAs use these samples to implement transition rules and geometrically evolve in order to find and update their classes.

4.2.2. Creation process

In the creation step, VAs are applied to extract training objects from the image space. To classify an image, the main elements of VAs, including geometry, state and neighbourhood together with their rules along the lines of GA are formally defined in Chapter 3.

4.2.2.1. Geometry and geometry methods

The geometry component (**L**) of a VA stores the vertices that define the boundary ∂X_{VA} of the VA (see Section 3.2.2). The geometric methods $\mathbf{M_L}$, defined in Chapter 3 (see Section 3.2.3.1), enable VAs to change the boundary ∂X_{VA} and interact with other VAs in the simulation domain. However, in the proposed unsupervised method, VAs do not use the interaction geometry rules, including growing/merging, growing/shrinking and shrinking/splitting.

4.2.2.2. State and transition rules

The state **S** of a VA is the class of a VA and the set of attributes (e.g. spectral signature rectangularity, homogeneity) that form the feature space involved in the classification process. **T**s rules allow the VAs to find and update their classes and evaluate pixels in the image space. To initialise in image space, the candidate pixel $\boldsymbol{x_c}$ and its immediate neighbours should be found to be members of the same class. VAs use the SVM classifier, which is trained based on a reliable sample set, namely X_L, to evaluate such membership. These candidate points, namely VAs, will evolve via an iterative process into polygons by capturing nearby pixels based on **T**s rules. To do this, VA_t uses the following rules to assess a candidate pixel $\boldsymbol{x}_c$(e.g. *V* in Figure 1 (b)) at time *t+1*:

$$VA_t.S.class = \boldsymbol{x}_c.class \tag{4.2}$$

$$\left[P_a\left(VA_{t+1}\right) - P_b(VA_{t+1})\right] \geq \beta \tag{4.3}$$

To evaluate Equation 4.2, VA_t uses the SVM classifier trained according to the labelled training X_L. $P_a(VA_{t+1})$ and $P_b(VA_{t+1})$ are the largest and the second largest probability of VA_{t+1} to belong to classes *a* and *b*, where *a* and *b* belong to *k*, computed by the SVM classifier, which is based on the spectral descriptors (mean value in each spectral band) of pixels within X_{VA} and pixel $\boldsymbol{x}_c$. These two rules allow the VAs to evaluate pixel $\boldsymbol{x}_c$ at two different levels, regardless of the spectral distribution in the feature space. At the first level, the VA_t locally evaluates pixel $\boldsymbol{x}_c$ based exclusively on the labelled training samples X_L. At the second level, Equation 4.3 allows the VA_t to evaluate pixel $\boldsymbol{x}_c$ in terms of its own

signature at each iteration. β is a parameter set between [0,0.8]. β is applied by each VA to control the effect of its signature at each iteration. This structure improves the confidence level of the extracted samples for the learning process. If $\boldsymbol{x}_c$ is found to be eligible to belong to VA_t, its attributes in **S** are updated according to the new geometry $\mathbf{L_{t+1}}$ and the corresponding set of pixels $X_{VA_{t+1}}$. The features characterising the VA_t, such as spectral, textural, and structural descriptors are re-evaluated. This yields a new signature for the VA_t in the next iteration, namely $t+1$.

4.2.2.3. Neighbourhood and neighbourhood rules

The neighbourhood component **N** is a collection of objects within the neighbour distance of a VA. Neighbourhood rules $\mathbf{R_N}$ are based on a set of rules applied by an object when it interacts with each other in a simulation domain. As DNs are the only available information, VAs cannot affect each other's geometry and state. In our case, this is the main neighbourhood rule applied by VAs. Thus, each VA should have knowledge of the subset of VAs adjacent in the image space. The set of VAs adjacent to $VA_{i,t}$ is defined as,

$$d\left(VA_{i,t}, VA_{j,t}\right) \leq r\sqrt{2}\ , \tag{4.4}$$

where $VA_{j,t}, j \in \mathbb{N}$. This structure simultaneously allows VAs to address the geometry and class of objects from initial pixels to the final training objects based on an evolving process, enabled by an iterative scheme that involves constant interactions between all VA components (Equation 4.5 adapted from Torrens and Benenson (2005)). Equation 4.5 describes the elements of each VA (see Section 3.2):

$$\text{VA} \sim (\mathbf{L, M_L}; \mathbf{S, T_S}; \mathbf{N, R_N}) \tag{4.5}$$

4.2.2.4. Implementation of VAs for unsupervised classification

To ensure the highest quality of the training samples and to maximize the representativeness of the training samples, a set of rules is considered by the VAs during the evolving process. The criteria of these rules are based on quantity, size, location, number and uniformity. In conventional supervised algorithms, a human expert usually performs this process. In our experiment, the following set of rules is used to automatically accommodate the prescriptions as discussed by Gao (2008, pg. 270), Richards and Jia (2006, pg. 199), and also when selecting training samples in supervised image classification:

1) A VA is killed if the total area of the VA cannot grow beyond 40 pixels.

2) A VA stops growing once it reaches 60 pixels, although this can be adjusted on the number pixels of each cluster. The size of the training dataset should be at least 10 to 30 times the number of features for each class (Gao, 2008). In our experiments, we use a 4 band multispectral IKONOS satellite image.

3) The maximum number of training polygons for each cluster is equal to five. Gao (2008) recommended a minimum of five to ten polygons per class.

4) A VA is removed if it has an interior ring (see Figure 3.5). The geometry of the VAs allows them to control their internal structure.

It is worth mentioning that in the proposed method, the above rules can automatically be updated based on different images (datasets) without human supervision. The process starts by seeding a desired number of VAs as points in a vector agent space whose coordinates correspond to the centre of image pixels. This seeding process can obey a specific sampling scheme (e.g. fully random, stratified random, systematic unaligned random, etc.). In this chapter, a systematic unaligned random sampling scheme is chosen so that VAs are seeded for every class at various locations throughout the image as described above. At the beginning, VAs are born with a threshold β of 0.8 (the highest confidence level).

These points or VAs will evolve via an iterative process into polygons by capturing nearby pixels based on transition and neighbourhood rules via their geometry methods. In the event that all VAs are passive and β is not zero, the algorithm automatically reduces the threshold β based on an interval of 0.1. The new VAs are born and activated on the new value of the threshold β. The simulation process continues until all VAs become passive with the threshold β of zero. This structure allows the VA model to extract training objects at different confidence levels. The output of the creation step, namely VA-generated samples, can be expressed as follows:

$$\mathcal{D} = \{VA_i | VA_i \cap VA_j = \emptyset, i \neq j\}_{i=k}^{N}, \tag{4.6}$$

where $N \in [k, 5c]$ is the number of extracted VAs. We set the maximum number of training polygons to 5 (see rule 3 in the above). The size, shape and the class of VAs may vary according to the observed data. Figure 4.1 indicates that there are no training objects for the cluster 8. Because the objects in the cluster 8 lack the necessary geometric properties (see the above rules) to exist in the image space. Figure 4.1 also displays how VAs use the above rules to extract training objects at different confidence levels from the image in Figure 4.1(f).

For example, in the case that the threshold β is 0.6, only the objects in clusters 4 and 5 are determined (Figure 4.1(a), 4.1(b) and 4.1(c)). When the threshold β becomes 0.5, the objects in clusters 9 and 2 are identified from the image space. The β values is utilised by the VA-based approach to rename the clusters in the identification step (see Section 4.2.3).

As can be seen in Figure 4.1(e), the number of cluster labels is greater than the desired number of clusters, *k*. To reduce the number of cluster labels to *k*, the algorithm uses a separability matrix formulated on the VA-generated samples in an iterative manner. In supervised approaches, a human expert usually employs a separability matrix to evaluate, merge, delete or rename the signature of training polygons.

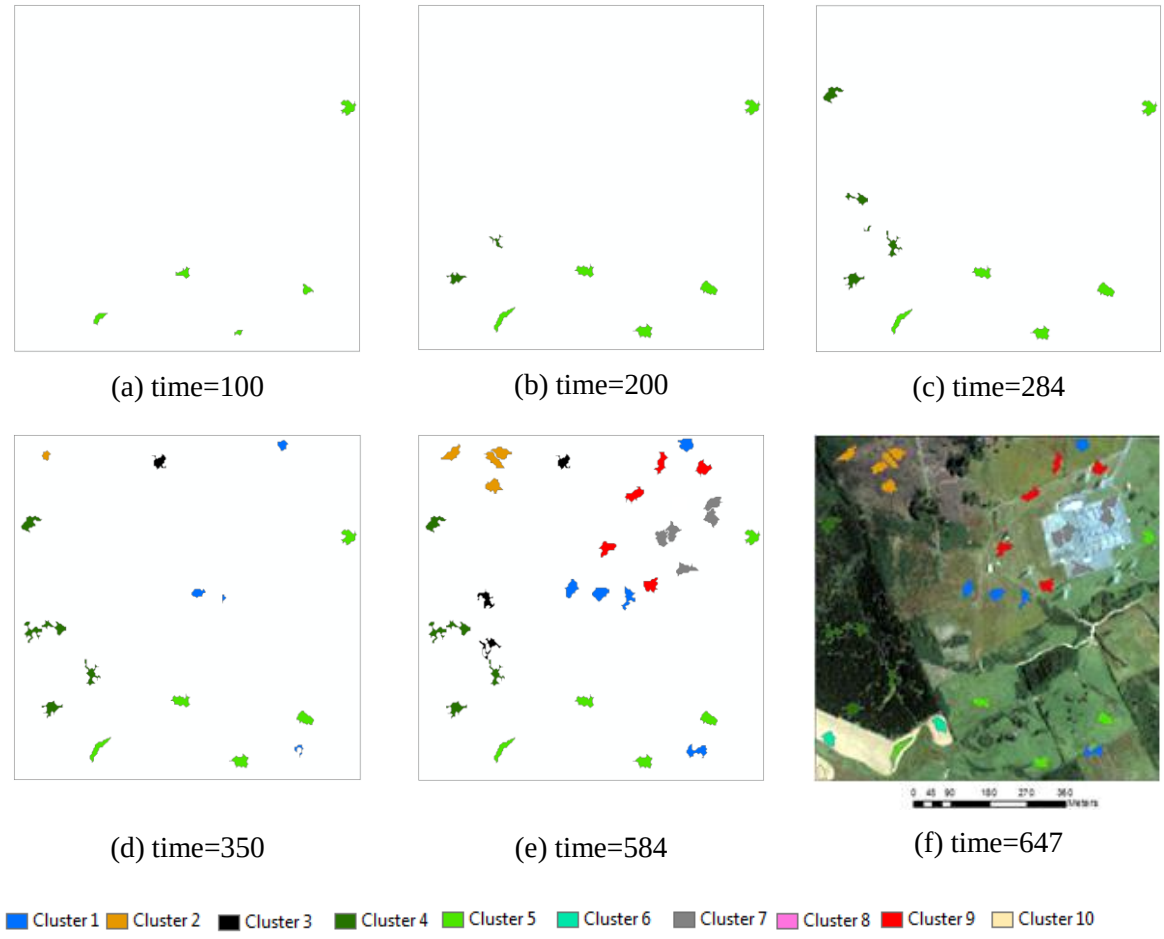


Figure 4.1. (a), (b) and (c) are the extracted VAs at a confidence level of 0.6. (d) and (e) are the VA-generated samples at a confidence level of 0.5. (f) Projected VAs on a subset of a multispectral IKONOS image, pixel size 4 meter, at confidence level of zero.

4.2.3. Identification

In the identification step, VAs are iteratively refined via a separability matrix, computed by a Transformed Divergence (TD) algorithm (Mather and Koch, 2011; Richards, 2006).

$$d_{ij} = \frac{1}{2} tr\{(S_i - S_j)(S_j^{-1} - S_i^{-1})\} + \frac{1}{2} tr\{(S_j^{-1} + S_i^{-1})(m_i - m_j)(m_i - m_j)^t\}, \quad (4.7)$$

$$\mathrm{TD_{ij}} = 2000\left(1 - \exp\left(\frac{-d_{ij}}{8}\right)\right), \quad (4.8)$$

where *i* and *j* correspond to the classes being compared. The symbol *tr(.)*, trace, is computed by the sum of the diagonal elements of the indicated matrix. S_i and S_j are the variance-covariance matrices for two classes computed by the spectral information of the VAs belonging to the class labels *i* and *j* . m_i and m_j are the corresponding mean vectors. The value 2000 is used as an exponentially decreasing weight to increase distances between the classes. Considering the above formula, the matrix $\breve{D}$ can be expressed as follows:

$$\breve{D} = \begin{bmatrix} d_{11} & \cdots & d_{1n} \\ \vdots & \ddots & \vdots \\ d_{n1} & \cdots & d_{nn} \end{bmatrix}, \quad (4.9)$$

where d_{ij} is the distance between two clusters *i* and *j* computed by Equations 4.7 and 4.8 according to the variance-covariance matrix and mean vectors of the whole VAs that belong to class labels *i* and *j*. At each iterative step, the algorithm first finds the minimum value in matrix $\breve{D}$, in which d_{ij} is not zero. It converts the elements of the matrix corresponding to the minimum value into zero. The clusters *i* and *j* are then declared to form an identical pair. To reduce the number of clusters, the proposed method uses not only the covariance matrix of each VA-generated samples but also the number of the VAs in each cluster.

The algorithm then uses the triplet $(\vartheta_i, \bar{\beta}_i, \delta_{ij})$ to rename one of these clusters, where ϑ_i and $\bar{\beta}_i$ are the number of polygons and the average of the β values in cluster *i*, respectively. δ_{ij} indicates how many times in matrix $\breve{D}$, d_{in} values in cluster *i* are greater than the element corresponding d_{jn} in cluster *j*. If a pair, e.g. $(\vartheta_i, \bar{\beta}_i)$, in cluster *i* is greater than the corresponding pair in cluster *j*, the algorithm removes the row and column in matrix $\breve{D}$, which linked to index *j*. Thus, the class of all VAs in cluster *j* is renamed as cluster *i*.

The iteration continues until the number of cluster labels reaches *k*, namely the desired number of clusters. Figure 4.2(b) displays the results of the identification. In this sense, the VA approach can model clusters according to different distributions in the feature space. For example, in Figure 4.2, the algorithm changes the VA samples within cluster 1 and cluster 9 into cluster 5. First, the algorithm finds that cluster 1, cluster 5 and cluster 9 can be

converted into each other based on TD distance (Equation 4.8). To rename the samples within each cluster, the algorithm uses $(\vartheta_i, \bar{\beta}_i, \delta_{ij})$. For example, although the number of training polygons for cluster 1, cluster 5 and cluster 9 are the same (5 polygons per cluster, as shown in Figure 4.2(a)), Figure 4.1(c) and Figure 4.1(e) show that the average β value obtained from the VAs in cluster 5 is more than the average β values of VAs belonging to cluster 1 and cluster 9. This method uses the spectral information of these polygons to train a classifier (e.g. ML) in order to label the remaining pixels in the classification step.

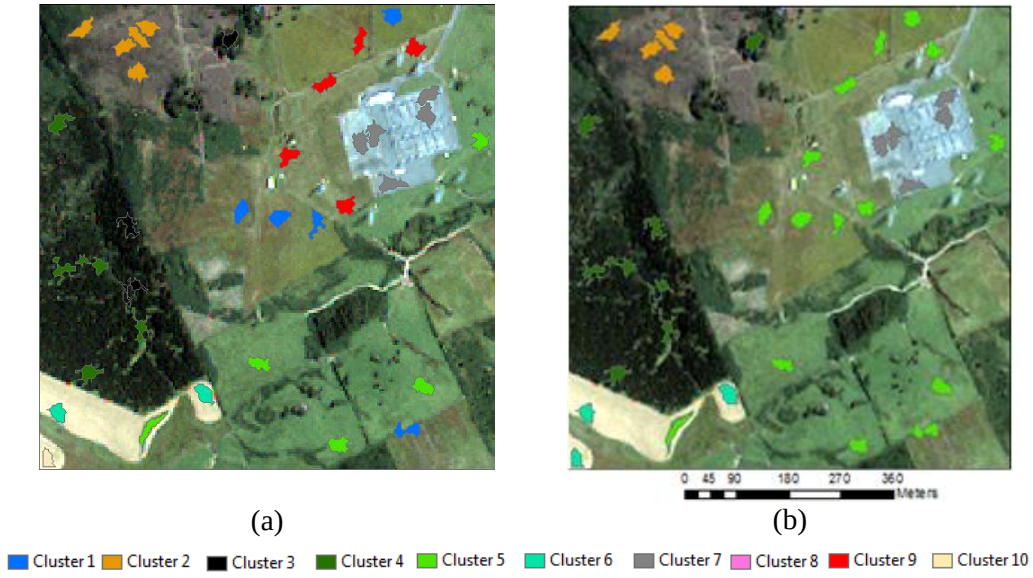


Figure 4.2. (a) The training polygons before the identification step. (b) The results of the identification step.

4.3. Experimental results

4.3.1. Data

The proposed approach was tested on two subsets of a multispectral IKONOS image (blue, green, red and near infrared bands) from a rural area near Dunedin, New Zealand (Figure 4.3(a) and Figure 4.6(a)). A Java implementation of the Repast (Recursive Porous Agent Simulation Toolkit) modelling framework (Howe et al., 2006), along with a generic Vector Agent library developed by Moore (2011), was used to develop a Vector Agent-led training and labelling process. The VA model uses the geometric rules and methods described in Chapter 3. This solution was used to implement unsupervised image classification.

To implement the transition rules, VAs employ the SVM classifier. The LIBSVM classification library for support vector machines developed by Chih-Chung Chang and

Chih-Jen Lin (2011), is applied. In our case, the SVM classifier is trained according to the Radial Basis Function (RBF) kernel. The RBF kernel is defined by $K(x, x') = \exp(-\gamma\|x - x'\|^2)$, where x and x' are two samples represented as feature vectors in spectral space (Camps-Vallas et al., 2007). γ and *C* parameters are tuned via a 10-fold cross-validation algorithm where *C* is a regularisation parameter.

4.3.2. Results and discussion

4.3.2.1. Dataset 1

In the first experiment, the image dataset utilised is a 200 by 200 pixel (4meter spatial resolution) scene containing five land cover classes: bare soil, gray area, meadow, soil and tree(Figure 4.3(a)) (Mathieu et al., 2007). These classes are specified according to a preliminary K-means algorithm where *C* is set to 10. In this experiment, λ is set to 1 in Equation 4.1, yielding a candidate set of labelled training samples for each cluster. 10 pixels are selected for each cluster according to a minimum distance algorithm.

For the quantitative evaluation of the classification maps, some areas of the classes of interest are manually classified as a ground truth map (Figure 4.3(b)). Figure 4.3(d), 4.3(e), 4.3(f) display the classification maps generated by K-means, K-medoids and Fuzzy c-means algorithms, respectively. In contrast to the K-means, K-medoids chooses the DNs of pixels as centers (medoids) in the feature space. To produce the maps in Figure 4.3(d), 4.3(e) and 4.3(f), we first cluster the image into 10 clusters. The clusters are then merged into each other.

A visual assessment of the classification maps based on the above algorithms (Figure 4.3(d), 4.3(e), 4.3(f)) and VA model (Figure 4.3(g)) reveals that the VA-based approach generates a smoother classification map compared to the conventional unsupervised methods. As it can be observed from Figures 4.3(g) the VA-based map has more homogenous regions (or patch-like areas) than the K-means, K-medoids and FCM algorithms. A comparison between Figure 4.3(c), which is provided by a Normalized Difference Vegetation Index (NDVI), and Figure 4.3(g) also indicates the satisfactory results of the VA model in separating vegetation from non-vegetation clusters.

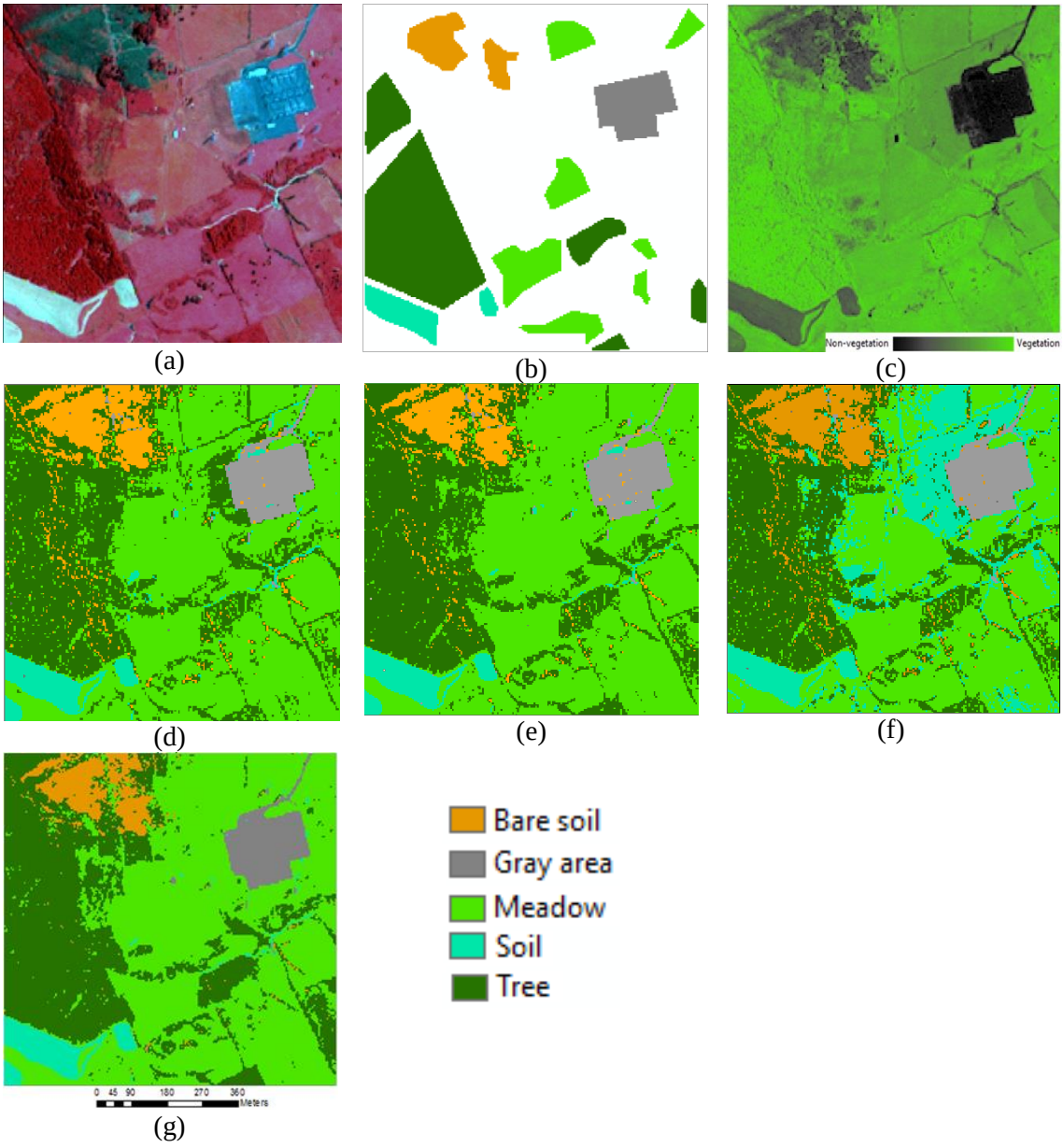
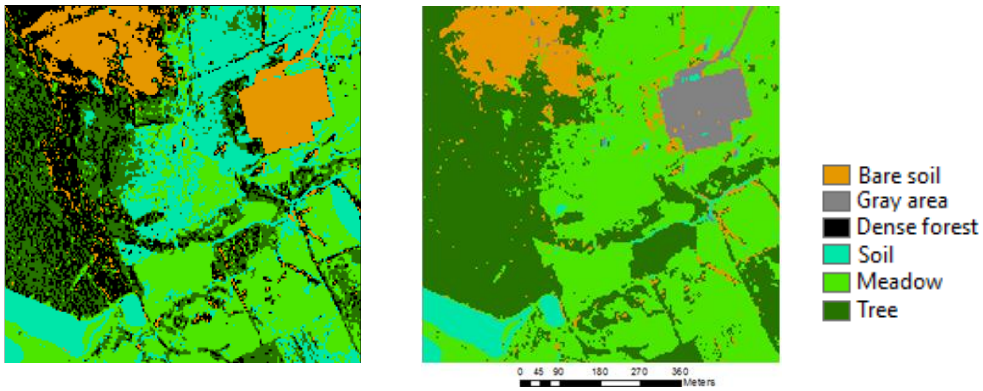


Figure 4.3. (a) 4m resolution false colour IKONOS image. (b) Ground truth map. (c) NDVI map. (d), (e) and (f) the classified maps based on K-means, K-medoids and Fuzzy c-means methods, respectively. (g) The classified maps based on the VA model.

Figure 4.4(a) displays a classified map obtained by using a traditional K-means algorithm in which the number of clusters is set to 5. As can be observed in Figure 4.4(a), the classified map differs from that provided by the VA model (Figure 4.3(h)). This may be due to the fact that traditional K-means methods are dependent on initial conditions. For example, a visual assessment of Figure 4.4(a) and 4.3(a) shows that some pasture pixels are classified as bare soil. The algorithm also divides the class of forest into two subclasses, namely tree and dense forest.



(a) (b)

Figure 4.4.(a) the classified map based on a traditional K-means in which the number of clusters is set to 5. (b) the classified map by using the ML algorithm.

To evaluate the results of the identification step, we manually create a set of training polygons (1 polygon per cluster) to implement an ML algorithm (Figure 4.4(b)). For example, the meadow areas in Figure 4.4(b) and Figure 4.3(g) are similar. However, the VA-based approach uses the spectral information of different clusters (see cluster 1, cluster 5 and cluster 9 in Figure 4.2(a) and Figure 4.2(b)). This shows the ability of the VA model to deal with the limitations of mono-modal Gaussian distribution, as applied in the clustering algorithms that use the EM algorithm. Using this strategy also allows us to measure the ability of the VA model to deal with issues such as overlapping intensities and noise originating from the crisp boundaries between clusters. Figures 4.3(g) and 4.4(b) indicate that the classified maps produced by the VA-based and ML supervised approaches are similar, whereas in the ML method, an expert operator manually defines the training objects.

To evaluate the performance of the proposed approach, the clusters are compared with their corresponding reference clusters in Figure 4.3(b). Table 4.1 displays the outcome of this comparison in terms of True Positive (TP), False Positive (FP) and False Negative (FN) analysis.

i. TP is the number of pixels that have the same class in both datasets (ground truth map and classified map).
ii. FP or commission error represents pixels that belong to another class, but are classified belonging to the class in the classification process.
iii. FN or omission error represents pixels that belong to the class, but are wrongly identified in the classification process.

For assessing the accuracy of the proposed method, three indices, including completeness, correctness (precision) and quality, are computed through the following equations:

$$Correctness = \frac{True\ Positive}{True\ Positive+False\ Positive} \tag{4.10}$$

$$Quality = \frac{True\ Positive}{True\ Positive+False\ Negative+False\ Positive} \tag{4.11}$$

$$Completeness = \frac{True\ Positive}{True\ Positive+False\ Negative}, \tag{4.12}$$

Table 4.1. Comparison between the results of K-means, Fuzzy c-means, K-medoids and the VA-based approaches. TP, FP and FN values are divided by the number of pixels within each ground truth class.

Classification method	Criteria	Object class (%)				
		Bare soil	Gray area	Meadow	Soil	Tree
K-means	TP	92.56	98.68	99.85	**97.73**	85.42
	FP	24.69	**0.37**	24.48	0.61	1.5
	FN	7.44	1.32	0.15	2.27	14.58
Fuzzy c-means	TP	93.45	99.12	89.87	97.25	87.18
	FP	23.32	**0.37**	22.98	35.06	1.23
	FN	6.55	0.88	10.13	2.75	12.82
K-medoids	TP	88.75	97.65	99.93	97.85	92.58
	FP	15.17	0.23	12.93	0.86	2.06
	FN	11.25	2.35	0.07	2.15	7.42
VA model	TP	**94.42**	**99.85**	**99.89**	97.25	**97.30**
	FP	**0.56**	1.69	**6.89**	**0.12**	**0.64**
	FN	5.58	0.15	0.11	2.75	2.70

Figure 4.5 indicates that the VA model outperforms the K-means, K-medoids and FCM methods. For example, a sharp improvement (more than 15%) in the quality and correctness indices of the bare soil and meadow classes can be observed. There are two reasons for this. Firstly, the conventional unsupervised algorithms only use the mean or medoids values to cluster an image. Secondly, due to the heterogeneous structure of the tree cluster (as can be seen in Figure 4.3(d)), some tree pixels are classified as bare soil or meadow. This increases the omission errors for these classes (see Table 4.1).

The results show the capability of the VA model in dealing with the heterogeneous classes that are composed of pixels with different classes (e.g. tree objects). The values of the above indices in Figure 4.5 show that the proposed algorithm provide more reliable results compared to the K-means, K-medoids and Fuzzy c-means methods.

From Figure 4.5 (d), it can also be observed that the completeness, correctness, quality and OA values of the ML approach are slightly better than the VA-based method. For example, the overall accuracy of the VA method is 97.89%. compared to 98.91% for the ML method. However, the ML approach uses human-provided training samples to classify the image. In contrast, the VA-based method generates training samples without human supervision.

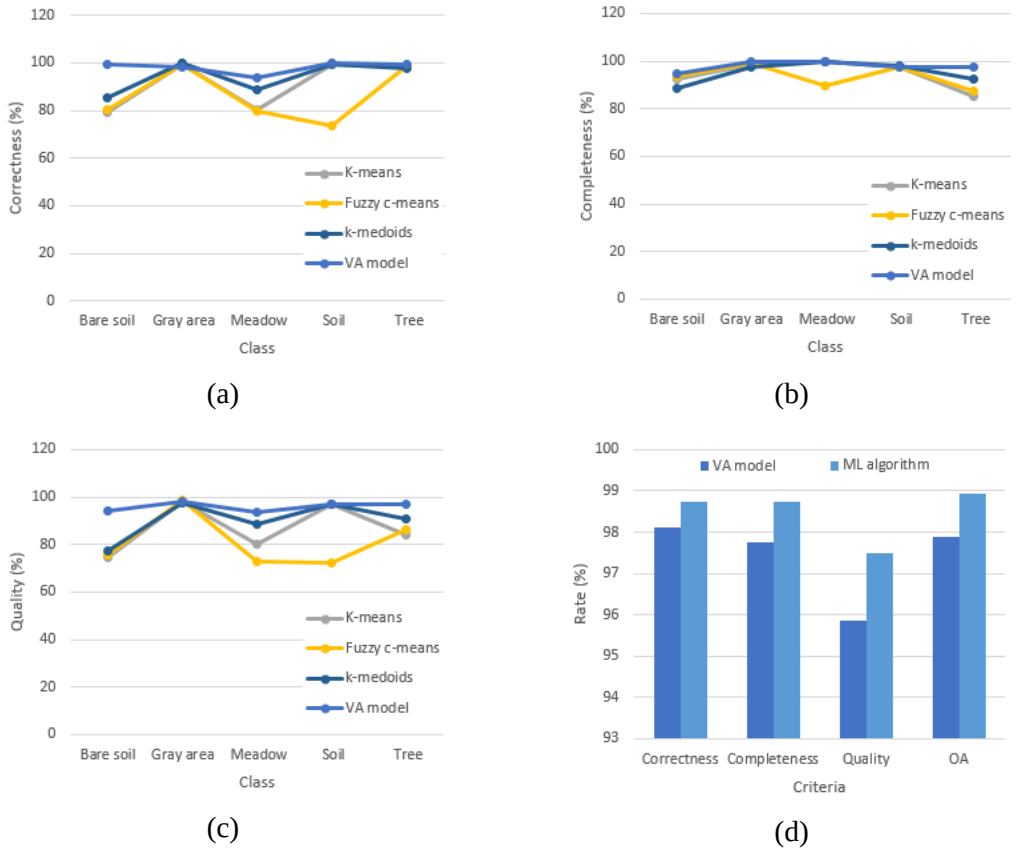


Figure 4.5. Comparison between K-means algorithms, Fuzzy C-means and VA-based methods based on correctness (a), completeness (b) and quality (c) indices. (d) The average rate of these indices in terms of the ML and VA-based approaches.

4.3.2.2. Dataset 2

In the second experiment, a K-means algorithm is used to cluster a subset of multispectral IKONOS image (240×240 pixels) into six land-cover classes that include bare soil, building, bush, grey area, pasture and shadow (Figure 4.6 and Figure 4.7(a)).

The image benefited from a pan-sharpening process, yielding four spectral bands, namely blue, green, red and near infrared, with a 1×1m spatial resolution. In this example, λ is set to 1 in Equation 4.1. 10 pixels are selected for each cluster from the eligible pixels obtained from Equation 1. A K-means algorithm is applied to extract 13 initial clusters from the image

in Figure 4.7(a). Figure 4.6(a) and 4.6(b) show the spatial distribution of the VA samples before and after the identification step, respectively.

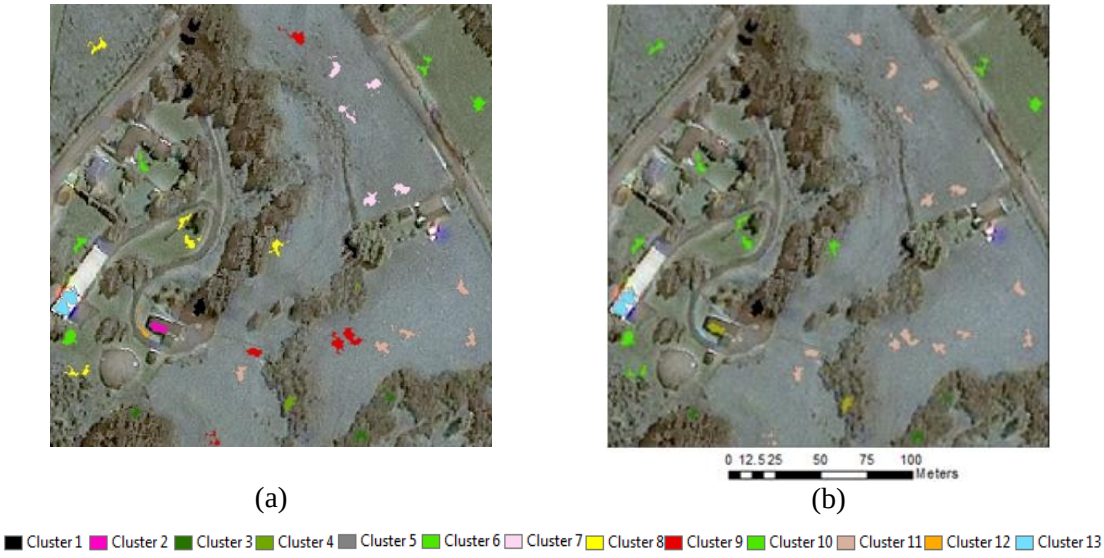


Figure 4.6. (a) The result of the creation step. (b) The results of the identification step.

Figure 4.7(b) shows some areas manually identified as a ground truth map. Figure 4.7(d), (e), (f) and (g) display the classification maps provided by the K-means, K-medoids and the Fuzzy c-means algorithms. It can be observed from Figure 4.7 that the unsupervised algorithms produce maps with a salt and pepper effect. In contrast, the VA-based method (Figure 4.7(h)) provides more homogenous regions compared to the unsupervised methods. The results also indicate that the VA-based method accurately separates vegetation from non-vegetation areas.

Similar to the previous example, we use a supervised approach to evaluate the extracted VA-generated samples. This also allows us to assess the capability of the proposed method in dealing with the limitations of mono-modal Gaussian distribution. To do this, firstly, a set of training samples, one training polygon for each cluster, are manually selected. These samples are then used by an ML algorithm to classify the image. The results indicate that the ML method lacks the ability to classify the image accurately, despite both the VA-based approach and the ML method using the same statistical rules to classify pixels. This can be explained by the fact that the dynamic VA approach allows variation in the signature of a thematic class to be found. This may cause the signature of a class to depart from a Gaussian distribution, thus compromising the ML classifier. In this example, the SVM classifier, as a supervised approach, is used to classify the image (Figure 4.7(I)). The SVM classifier exhibits better performance compared to the ML method. However, the SVM classifier

produces a map with salt and pepper compared with the VA approach (see Figure 4.7(h) and 4.7(I)).

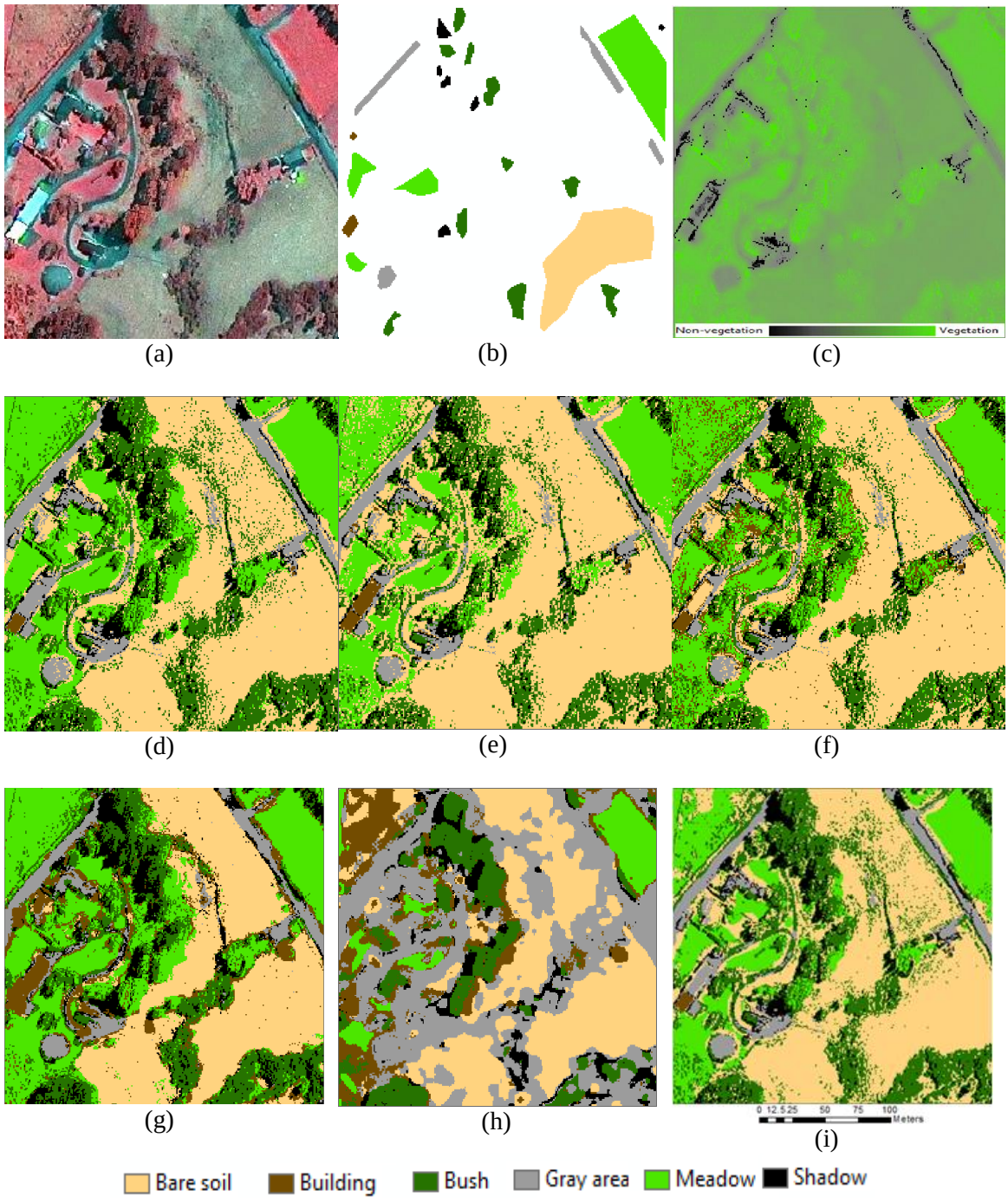


Figure 4.7. (a) A 1m resolution false colour IKONOS image. (b) Ground truth map. (c) NDVI map. (d), (e) and (f) classified maps based on the K-means, K-medoids and Fuzzy c-means methods, respectively. (g), (h) and (i) display the results of the VA-based method, the ML and the SVM classifiers.

Table 4.2 shows the results of the different methods based on the TP, FP and FN errors. These errors are applied to calculate the correctness, completeness and quality indices. As

can be seen in Figure 4.8, the VA model improves the correctness, completeness and quality of the bare soil cluster by a rate of 1% to 7%. Because of the salt and pepper effect, the K-means algorithms exhibits poor results when compared with the VA model. Figure 4.8(c) also shows that the VA model increases the quality of the gray area cluster by more than 7%. The completeness index of the building cluster shows an improvement of more than 30% based on the VA approach. However, the correctness value of the building cluster decreases by 30% based on the VA model. As the spectral reflectance of the gray area and building clusters are similar, some gray area pixels are classified as bush (see Figure 4.7(h)). When considering the meadow cluster, the correctness, completeness and quality indices of all approaches are similar. However, these indices for the bush class are entirely different for each respective approach. This could be due to the high spatial distribution of the shadow and meadow pixels within the bush areas.

Table 4.2. Comparison between the results of VA model, K-means algorithms and Fuzzy C-means and K-medoids approaches. TP, FP and FN values are divided by the number of pixels within each ground truth class.

Classification method	Criteria	Object class (%)					
		Bare Soil	Building	Bush	Gray area	Pasture	Shadow
K-means	TP	98.18	63.28	**72.37**	86.26	98.91	96.69
	FP	3.87	**0.00**	12.76	6.23	5.69	58.56
	FN	1.82	36.72	27.63	13.74	**1.09**	3.31
Fuzzy c-means	TP	98.95	69.53	68.47	86.26	98.91	96.69
	FP	6.39	89.89	7.09	2.22	4.97	56.27
	FN	1.05	30.47	31.53	13.74	1.09	3.31
K-medoids	TP	99.6	82.03	67.52	88.05	96.01	85.29
	FP	10.57	1.90	10.85	2.17	3.55	52.16
	FN	0.4	17.97	32.48	11.95	3.99	14.71
VA model	TP	**99.35**	**99.22**	63.85	**90.92**	**99.15**	**99.26**
	FP	**0.03**	83.46	**3.87**	**0.39**	6.06	97.78
	FN	0.65	0.78	36.15	9.08	0.85	0.74

Figure 4.8 shows a comparison between the VA-based method and the SVM algorithm. The aim is to evaluate the quality of the VA-generated samples in terms of a supervised approach, such as the SVM algorithm. The correctness, completeness, quality and OA values of the VA-based method are similar to the SVM algorithm. In contrast to the SVM algorithm, the proposed method classifies the image without the need of training data.

Finally, Table 4.3 shows the CPU processing time (in seconds) spent on the different steps of the proposed method for the cases illustrated in Figure 4.3 and Figure 4.6. The method is implemented using Repast and powered by an Intel CPU running at 3.40 GHz with 16 GB of memory.

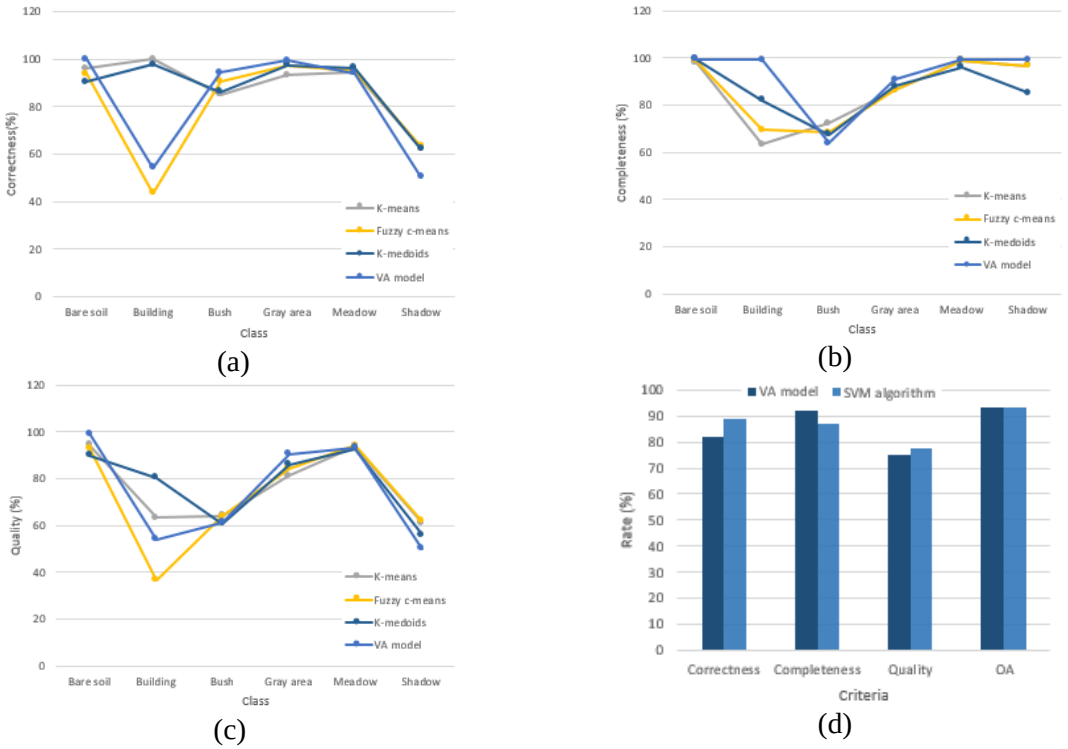


Figure 4.8. Comparison between K-means algorithms, Fuzzy C-means and VA-based approach method based on (a) correctness, (b) completeness, and (c) quality indices. (d) The average rate of these indices in terms of the VA-based method and the SVM approach.

Table 4.3. Computational time spent in each step of the VA model.

Name	Creation step	Identification step	Classification step	Sum
Dataset 1	160.0 Sec.	2.0 Sec.	2.0 Sec.	164.0Sec.
Dataset 2	210.0 Sec.	2.0 Sec.	2.0 Sec.	214.0Sec.

4.4. Conclusions

Traditional K-means methods use pixels in isolation to cluster an image. As a result, they lack the ability to deal with certain limitations such as poor contrast, overlapping intensities and noise, especially where there are HSR images. These limitations are usually addressed through the use of a soft clustering algorithm (e.g. FCM) or an EM-based method along with the spatial information. These methods usually employ a neighbourhood system of fixed shape and size or a Gaussian function to cluster an image. Hence these methods assume that

the clusters have a predictable behaviour in the feature or image space. When this assumption is violated, clustering results may be imprecise.

From the above discussion it can be concluded that in order to overcome the aforementioned limitations, some information about clusters is required. This information is usually obtained by utilising clustering methods, such as the proposed methods by Banerjee et al. (2015) and Mukhopadhyay and Maulik (2009). These methods usually ignore the spatial characteristics of clusters (e.g. geometry) in image space. That means, this information about clusters is only provided based on the spectral characteristics of clusters. In this chapter, we investigated the capabilities of VAs as a new processing unit to extract a set of training objects in image space, just as a human interpreter would do in supervised approaches. The spectral information of these training polygons was then applied by the proposed method to classify the image.

To reach this aim, we first used a traditional K-means algorithm to identify the initial clusters. We assumed that the number of initial clusters is known. We then described the components of VAs and how VAs can extract a set of training objects from image space. To control the quality of the training polygons, VAs employed a set rules usually applied by a human interpreter in supervised approaches. To reduce the number of clusters to the desired number clusters K, the method used an iterative approach. This was formulated on a separability matrix, based on the spectral information of the VAs in the feature space and the geometric characteristics of the VA polygons in the image space. In the classification step, the updated signatures were employed by an ML algorithm to label pixels. The experimental results demonstrate the desirable performance of this new approach. VAs prove able to classify an image without using a predefined shape or setting-specific distribution. Moreover, the results of the labelling step indicate that the accuracy of the classified images delivered by VAs are better than both the traditional K-means method and the supervised approaches, even in complex scenes.

In our examples, the VA model is implemented with the assumption that the number of clusters is known. To resolve this issue, we can use the characteristics of the TD algorithm. According to Jensen (1996), a TD value of 1900 or more indicates that two signatures can be separated, and a value between 1700 and 1900 means that the separation is relatively good. If TD is less than 1700, the separation is poor. In this sense, the identification process can be based on the quality of the clusters, without setting the number of clusters. For example, the algorithm terminates the identification process if all TD values are more than

1900. In this case, the method can identify the optimum number of clusters based on the quality of clusters. Consequently, we do not need to set an exact number of the clusters. Developing an intelligent identification process can also significantly reduce the processing time of the creation step. The use of a full scope of VA model to classify an entire image would also be interesting to study.

In the next chapter, we will explore the application of the VA model for supervised image classification.

Chapter Five

Vector agent model for supervised image classification

Abstract

Supervised approaches use training samples to classify an image. This method produces a more reliable result compared to unsupervised approaches. Despite promising results, the process of generating training samples is usually time consuming and expensive, especially when using hyperspectral remotely sensed images. This limitation is usually addressed via semisupervised approaches. The main aim of semisupervised methods is to generate reliable labelled samples from the limited subset of labelled samples without significant effort/cost. In this chapter, we explore the capabilities of VAs in classifying hyperspectral images within the context of a semisupervised SVM classification. The proposed method consists of three main steps: creation, selection and classification. In the creation step, VAs are employed to extract a set of training objects from the image space. The dynamic structure of VAs allows them to capture the spectral information of each specified class of the initial labelled training samples through a set of polygons in the image space. In the selection step, a similarity metric is applied to identify the most reliable samples from the VA-generated samples. Finally, in the classification step, the samples selected from the previous step are used by the SVM classifier in order to label the remaining pixels. We have validated this concept by implementing the VA method to classify two different hyperspectral remote sensing datasets. The preliminary results show that the VA-based method can yield high classification accuracy with only a small number of labelled training samples.

5.1. Introduction

Image classification is a fundamental process in image analysis and is typically addressed in either unsupervised or supervised manner. In the latter case, the algorithms involve two main steps: training and classification. The training step generates statistics for each class

that describe its distribution in a defined feature space. These results are then used in a classification process, such as the ML method.

Alternatively, unsupervised approaches rely on a weaker formulation that involves the prior definition of the number or the typical variance of clusters to be identified in the feature space (e.g. the K-mean and ISODATA algorithms). It has been well established that when both methods are compared, supervised methods, with its reliance on training samples, yields more accurate classification, but has the downside of needing a higher level of expertise (Gao, 2008).

Nonetheless, the quality of the classification depends ultimately on a suitable and consistent definition of the training samples usually collected by a human expert (Tuia et al., 2011; Chi et al., 2008; Foody and Mathur, 2004). Thus, the training process is affected by the availability of training samples and the experience and skills of the human operators. This makes the process time consuming, relatively complex, and costly (Jun and Gush, 2013; Chi el al., 2008).

Semisupervised Learning (SSL) techniques have allowed some of these limitations to be mitigated. SSL techniques belong to a group of supervised classification algorithms that only need a limited quantity of labelled data to classify an entire dataset (Chapelle et al., 2008). This is based on the assumption that a suitable set of unlabelled training samples (Shahshahani and Landgrebe, 1994) can be obtained from the limited subset of labelled samples without significant effort/cost (Chapelle et al., 2008; Bruzzone et al., 2006). Figure 5.1 shows how the use of unlabelled training samples can affect the performance of a classifier, such as SVM.

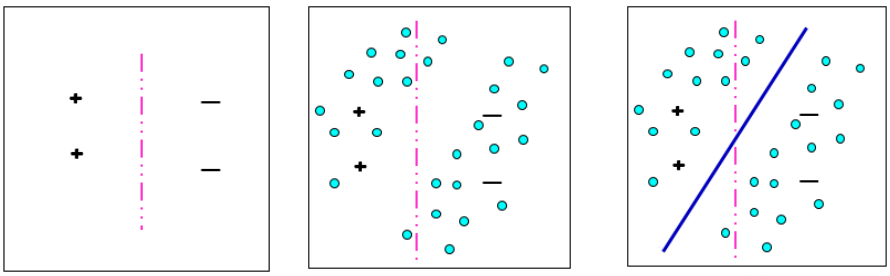


Figure 5.1. (a) the SVM model based on only the labelled data. (b) the SVM model projected on the labelled and unlabelled datasets. (c) the effect of using unlabelled samples on the SVM model.

Because of this ability, the use of SSL methods in remote sensing image analysis has become popular over the past decade, especially in connection with hyperspectral images that have a limited number of training samples. In hyperspectral data processing with a limited number

of training samples, increasing the dimension of feature space can significantly reduce the accuracy of the classified map (Bruzzone et al., 2006; Hughes, 1968).

To date, many different methods have been proposed for semisupervised classification of hyperspectral images, such as Transductive Support Vector Machines (TSVMs) (Bruzzone et al., 2006), graph-based methods (Camps-Valls et al., 2007), generative models (Krithara et al., 2011) and self-learning methods (Dópido et al., 2013).

For example, Bruzzone et al. (2006) utilised the SVMs through a transductive process to execute a semisupervised method. Here, the authors employed both labelled and unlabelled samples (e.g. Figure 5.1 (c)) to search for a reliable separating hyperplane in the training phase. Some studies also propose the use of graphs (Di and Crawford, 2010; Camps-Valls et al., 2007). For example, Camps-Valls et al. (2007) employs graphs to exploit reliable unlabelled samples based on their neighbours in the image. Krithara et al. (2011) employed a generative model to measure mislabelling errors generated by adding unlabelled samples during the training process. Here, the algorithm iteratively assigned class labels to unlabelled examples using the current model and re-estimating the probabilities of the mislabelling errors. Dópido et al. (2013) presented a self-learning method. In their method, unlabelled samples are selected with an algorithm applied a first-order spatial connectivity on a map provided by a probabilistic SVM. They then employed an active learning algorithm to identify the most informative samples. In contrast to passive learning algorithms (e.g. Figure 5.1), the learner is able to interactively select unlabelled samples for the learning process.

Based on these methods, the main challenges of a semi-supervised learning approach are the issue of selecting the most helpful unlabelled samples and subsequently determining the class label of these new selected samples. Several studies show that the use of spatial information can improve the results of the semisupervised image classification (Tan et al., 2015 and 2014; Wang et al., 2014; Dópido et al., 2013; Bruzzone and Persello, 2009). In this framework, the SVM algorithms have proven to be an effective tool for combining spectral and spatial information (Bruzzone and Persello, 2009).

For example, Bruzzone and Persello (2009) presented a novel context-sensitive semisupervised method to reduce the effect of mislabelled training samples. The method employed an updated cost function formulated on labelled and unlabelled samples to minimize the total misclassification costs. To take advantage of the spatial information, authors used a local window centred on labelled samples. Wang et al. (2014) used a

neighbourhood system of fixed size to identify unlabelled samples. Here, the authors employed a probability map created by an SVM algorithm to identify the most informative unlabelled samples and then label them through an active learning algorithm. Tan et al. (2014) proposed a segmentation-based algorithm to select unlabelled training samples. They then applied a similarity metric formulated on labelled samples and segmented regions to select reliable unlabelled training samples for each cluster.

The above methods show that the use of spatial information along with spectral characteristics is key towards inferring a suitable set of training pixels from limited initial sampling. The aforementioned algorithms use different strategies to improve the SSL methods. However, all methods have one property in common: they use a neighbourhood system of a fixed size specified by a human expert. For example, to segment an image, we need to specify some parameters (e.g. scale or kernel size). To take advantage of the spatial information, conventional semisupervised approaches employ a local window of fixed shape, but there is no specific rule to determine what shape should be used and what size should be given to it in order to obtain optimum results.

This chapter demonstrates that the flexibility of the VA approach to image analysis can also be used to support the selection of numerous and consistent training samples towards a semisupervised method. ***The proposed method uses spatial information without setting shape parameters or using a specific shape in order to select unlabelled samples in the context of semisupervised approaches***. In the next section, the proposed method is presented. The algorithm will be discussed in Section 5.3 and 5.4. The experimental results of the VA model are presented in Section 5.5. A short summary and ideas for further work are represented in Section 5.6.

5.2. Proposed method

Let $x = (\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \cdots, \boldsymbol{x_n}) \in \mathbb{R}^d$ denote a multispectral *d*-dimensional image with n pixels, $y = (y_1, y_2, y_3, \cdots, y_n)$ an image of labels, $K = \{1,2, \cdots, k\}$ a set of *k* class labels, $X_L = \{(y_i, \boldsymbol{x_i})\}_{i=1}^{l}$ labelled samples, *l* number of labelled samples, and $X_U = \{\boldsymbol{x_i}\}_{i=l+1}^{n}$ unlabelled samples. Let $\widetilde{X}_L = \{(j, \overline{\boldsymbol{x}}_{\boldsymbol{j}})\}_{j=1}^{k}$ denote the feature labelled samples, where $\overline{\boldsymbol{x}}_{\boldsymbol{j}}$ is calculated on the mean value in each band of all $\boldsymbol{x_i} \in X_L$ with the same class. The proposed approach is based on three main steps: creation, selection and classification. In the creation step, VAs

are applied to extract training objects from image dataset. The output representation can be described as follows:

$$I_P = \bigcup_{i=1}^{N} VA_{i,c\in K}\,, VA_i \cap VA_j = \emptyset\,, i \neq j\,, \tag{5.1}$$

where I_p is the VA-based image model (or VA map), $VA_{i,c}$ is regarded as a set of labelled connected pixels of the class $c \in K$, and N is the number of VAs. The size, shape and the class of VAs may vary according to the observed data. In the selection step, a similarity metric is applied to extract reliable VA samples from VA-generated samples and candidate sets for the learning process. In the classification step, the selected VAs are then used by the SVM classifier to classify the image.

5.3. Vector agents

For semisupervised applications, VAs are used to identify a set of training objects. In the following sections, the elements of VAs are described in terms of a semisupervised algorithm.

5.3.1. Geometry and geometry methods

The geometry component **L** determines the shape and size of the training objects. **L** stores the vertices that determine the boundary of the VA at each increment. Geometry methods $\mathbf{M_L}$ are a collection of individual geometric methods (see Section 3.2.2.1) allowing the geometry **L** to evolve in the simulation space.

5.3.2. State and transition rules

Similar to the VA-based unsupervised approach, the state **S** of a VA is the class of a VA and the set of attributes (e.g. spectral descriptor) that form the feature space involved in the classification process. Since it is dynamic in nature, each VA updates its attributes at every increment. A VA is initialised in the image space if a candidate pixel $\boldsymbol{x_c}$ and its neighbouring pixels should have the same class. To assess the class of neighbouring pixels, VA_t uses the SVM classifier.

These points or candidate pixels will evolve into polygons via an iterative process by capturing nearby pixels based on $\mathbf{T_S}$ rules (see Section 4.2.2.2). If $\boldsymbol{x}_c$ is found eligible to belong to VA_t, its attributes in **S** are updated based on the new geometry $\mathbf{L_{t+1}}$ and the corresponding set of pixels $X_{VA_{t+1}}$. The features characterising the VA_t such as spectral,

textural, and structural descriptors are re-evaluated. This yields a new signature for VA_t in the next iteration, namely *t+1*.

5.3.3. Neighbourhood and neighbourhood rules

As VAs cannot affect the geometry of each other, VAs use the same neighbourhood rules $\mathbf{R_N}$ described in the previous chapter (Chapter 4, section 4.2.2.3). In this context, each VA only has knowledge of the subset of VAs that are adjacent.

This structure allows VAs to address simultaneously the geometry and class of objects from initial pixels to the final training objects based on an evolving process enabled by an iterative scheme (see Section 3.3.6) that involves constant interactions between all VA components. The output of the creation step is the VA-generated samples (Equation 5.1). Once candidate sets are inferred, the proposed approach uses the following selection algorithm to automatically identify the most informative samples from the VA-generated samples.

5.4. Selection of VA-generated samples

As the use of all VA-generated samples in the creation step for the learning process can drastically increase computational cost, the Nearest-Neighbour (NN) algorithm is applied to find the most informative candidate sets of each class. Let $D = \{VA_i\}_{i=1}^{N}$ be the VA-generated samples set, $\widetilde{D} = \{(\overline{\boldsymbol{z}_i}, S_i)\}_{i=1}^{N}$ the feature set, *N* the number of VAs, $\overline{\boldsymbol{z}_i}$ a vector based on the mean value in each spectral band of all pixels within each VA, and $S_i \in K$ specified on the state of each VA, **S** (see Section 3.2.4). Thus, the similarity metric θ of the two samples ($\overline{\boldsymbol{z}}$,$\overline{\boldsymbol{x}}$), $\overline{\boldsymbol{z}} \in \widetilde{D}$ and $\overline{\boldsymbol{x}} \in \widetilde{X}_L$, can be calculated via the spectral angle algorithm as follows:

$$\theta = \cos^{-1}\left(\frac{\sum_{i=1}^{d} \overline{\boldsymbol{z}}_{i,S=k} \cdot \overline{\boldsymbol{x}}_{i,y=k}}{\left(\sum_{i=1}^{d} \bar{z}^2{}_{i,S=k}\right)^{\frac{1}{2}} * \left(\sum_{i=1}^{d} \overline{\boldsymbol{x}}^{\,2}{}_{i,y=k}\right)^{\frac{1}{2}}}\right), \tag{5.2}$$

$M = \{\theta_i\}_{i=1}^{N}$ and $R = \{(\delta_j, S_j)\}_{j=1}^{k}$ are the similarity metric set and the reference set, respectively, where δ_j is equal to the minimum θ for each class in *M* and $S_j \in K$. Thus, VA_i in *D* is a selected sample if its corresponding θ_i in *M* can satisfy the following rule:

$$\theta_i - \delta_{j,S=VA_i.\mathbf{s}} \leq \alpha, \tag{5.3}$$

where α is a threshold manually determined. The selected VAs can be expressed as follows:

$$\mathcal{D}_s = \bigcup_{i=k}^{P} VA_i\ , \tag{5.4}$$

P is the number of selected VAs. *P=k* means there is only one selected VA for each class. In the classification step, the selected VAs, along with the initial label training samples, are applied via an SVM algorithm to identify the class of the remaining pixels.

5.5. Experimental results

5.5.1. Data

The proposed approach was experimentally tested on two well-known datasets, ROSIS Pavia University and AVIRIS Indian Pines (doi:10.4231/R7RX991C).

1) Indian Pine: the first dataset used in our experiments was a forest/agricultural region in India and collected by the Airborne Visible Infrared Imaging Spectrometer (AVIRIS) sensor with a size of 145×145 pixels. 200 spectral dimensions were used and 20 spectral bands were removed due to the noise and water absorption. We used eight main classes with the largest numbers of samples including corn-notill, corn-mintill, grass-trees, hay-windrowed, soybeans-no till, soybeans-notill, soybeans-clean, and woods (Figure 5.2(a)).

2) University of Pavia: the second dataset was an urban area in Pavia, Italy and was collected by the reflective optics system imaging spectrometer (ROSIS) sensor with a subset image of size 250×250 pixels and spatial resolution of 1.3m. It covers nine classes including, asphalt, meadows, gravels, trees, metal sheets, bare soil, bitumen, bricks, and shadows. The original 115 bands were reduced to 103 bands; water absorption bands affected by the atmosphere were removed. Figure 5.3(a) shows the studied area.

The proposed method uses the Repast modelling framework discussed in Section 4.3.1 to run the VA model. The threshold α also is set to 0.01 and 0.008 for the Pavia University and Indian Pines, respectively (Tan et al., 2014).

5.5.2. Experimental design

To assess the performance of the proposed approach, we used three different sizes of labelled training samples X_L for each class: 5, 10 and 15 pixels. To create the classification maps, different pixel numbers of the selected VAs in $\mathcal{D}_s$ were randomly applied to classify the images. This permitted us to analyse the effect of the size of VAs on the classification accuracy. Moreover, the parameter β is tested to evaluate the effect of the proposed algorithm to identify the proper VAs.

To analyse the performance of the proposed approach, the results of the VA-based semisupervised classification methods were compared with a controlled classification whereby the SVM algorithm used only the initial label training samples to classify images. Since data acquired from remotely sensed imagery often have unknown distributions, the use of methods, such as ML, may lead to misclassifications, especially when there is a hyperspectral image (Mountrakis et al., 2010). The purpose of this comparison was to measure the benefit of the VA-generated samples on the classification. In all cases, the overall accuracy (OA) and Kappa coefficient (Kappa) were used to measure the classification accuracy.

5.5.3. Results and discussion

Figure 5.2(c) shows the VA map for the experiments conducted using 15 labelled training samples on the image dataset. Figure 5.2(d) displays the spatial distribution of the selected VAs on the ground truth map (yellow polygons). As can be seen, most selected polygons lie on the corresponding ground truth class. A visual assessment of Figures 5.2(e) and (f) shows that the proposed method suggests a better depiction of the ground truth map compared to the Supervised SVM (SSVM).

To assess the performance of the proposed approach, we randomly selected three different sizes of labelled training samples of 5, 10 and 15 pixels for each class from the ground truth map (Figure. 5.2(b)). To create the classification maps, the method used different pixel numbers of the selected VAs within $\mathcal{D}_s$. This permitted the analysis of the effect of the size of VAs on the classification accuracy. To analyse the performance of the proposed approach, the results of the VA-based method were compared with a controlled classification where the SVM algorithm used only the initial label training samples to classify images.

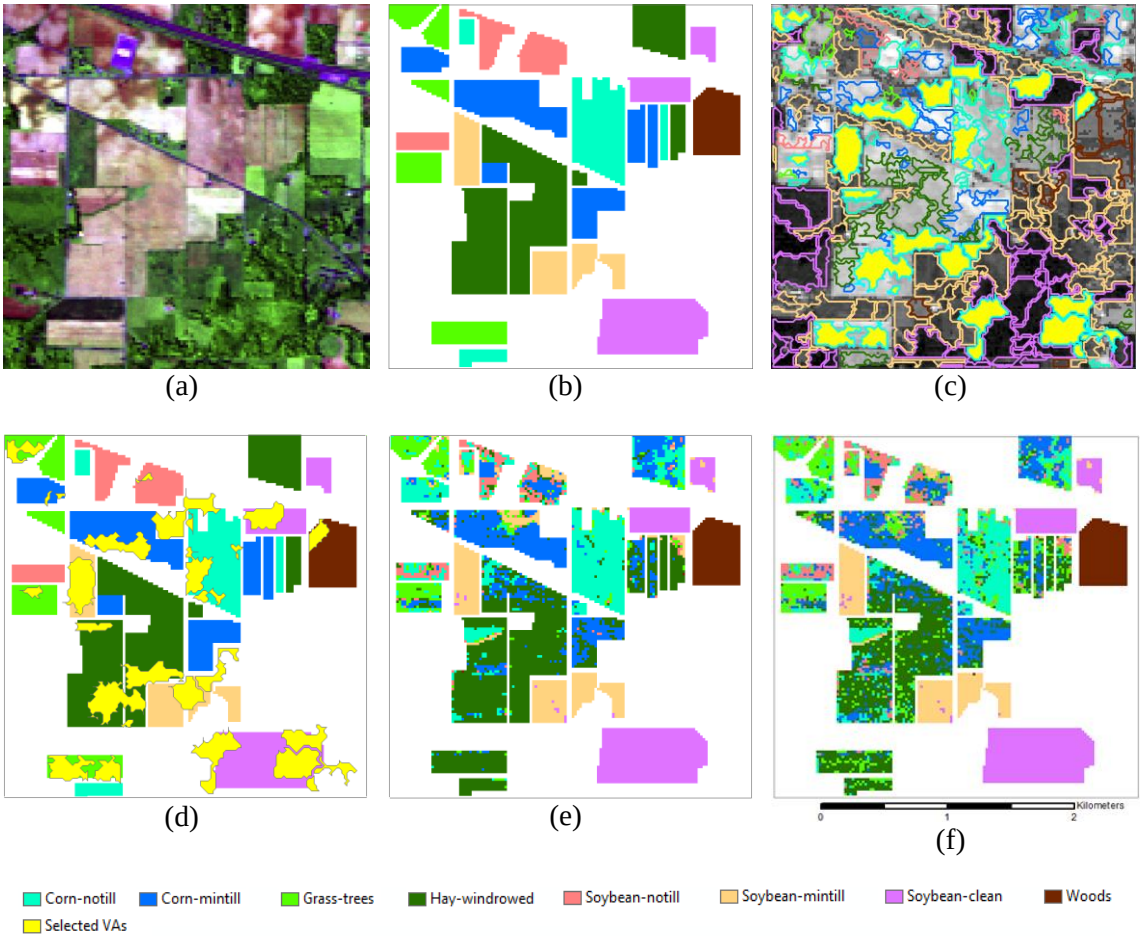


Figure 5.2. (a) The Indian Pine datasets. (b) Ground truth map of the Indian Pine datasets. (c) VA-generated training samples. (d) The selected VAs (yellow polygons) based on 15 initial training samples. (e) and (f) classification maps based on the VA approach and the supervised SVM algorithm by using 15 label training samples, respectively.

Table 5.1 lists the overall accuracy (OA) and Kappa values of the classified images based on the different configurations described above.

Table 5.1. OA values and Kappa values of the Indian Pine dataset based on different methods: the supervised SVM (0%), and the VA-based algorithm where β is set to 0.1.

Percentage of each selected VA			0	10	20	30	40	50	100
Number of labelled training samples	**5**	OA	54.94	55.10	56.39	56.52	54.95	56.34	**57.71**
		Kappa	46.96	47.40	48.73	48.92	46.96	48.69	**50.65**
	10	OA	56.75	59.70	60.54	59.94	60.27	60.84	**61.09**
		Kappa	49.81	52.74	53.51	52.8	53.8	53.15	**54.07**
	15	OA	64.75	68.20	68.95	70.01	70.05	69.4	**71.43**
		Kappa	58.43	62.18	62.95	64.27	64.26	63.53	**65.92**

As can be observed from Table 5.1, in all cases the VA-based approach exhibits better performance than the SSVM. For example, in the case with 15 training samples per class, the OA values are 64.75% and 71.43% based on the SSVM and the VA-based approach, respectively. The use of the VAs shows an improvement of more than 7% for the OA value. The OA accuracy and Kappa values are slightly increased when the VA-based approach uses more samples.

Figure 5.3(c) shows the VA map for the experiments conducted with 15 labelled training samples on the Pavia University image (Figure 5.3(a)). Figure 5.3(d) displays the spatial distribution of the selected VAs on the ground truth map. As can be seen, most selected polygons lie on the corresponding ground truth class (Figure 5.3(d)). Figure 5.3(e) and 5.3(f) display the classification maps based on the VA semisupervised SVM and supervised SVM (SSVM), respectively.

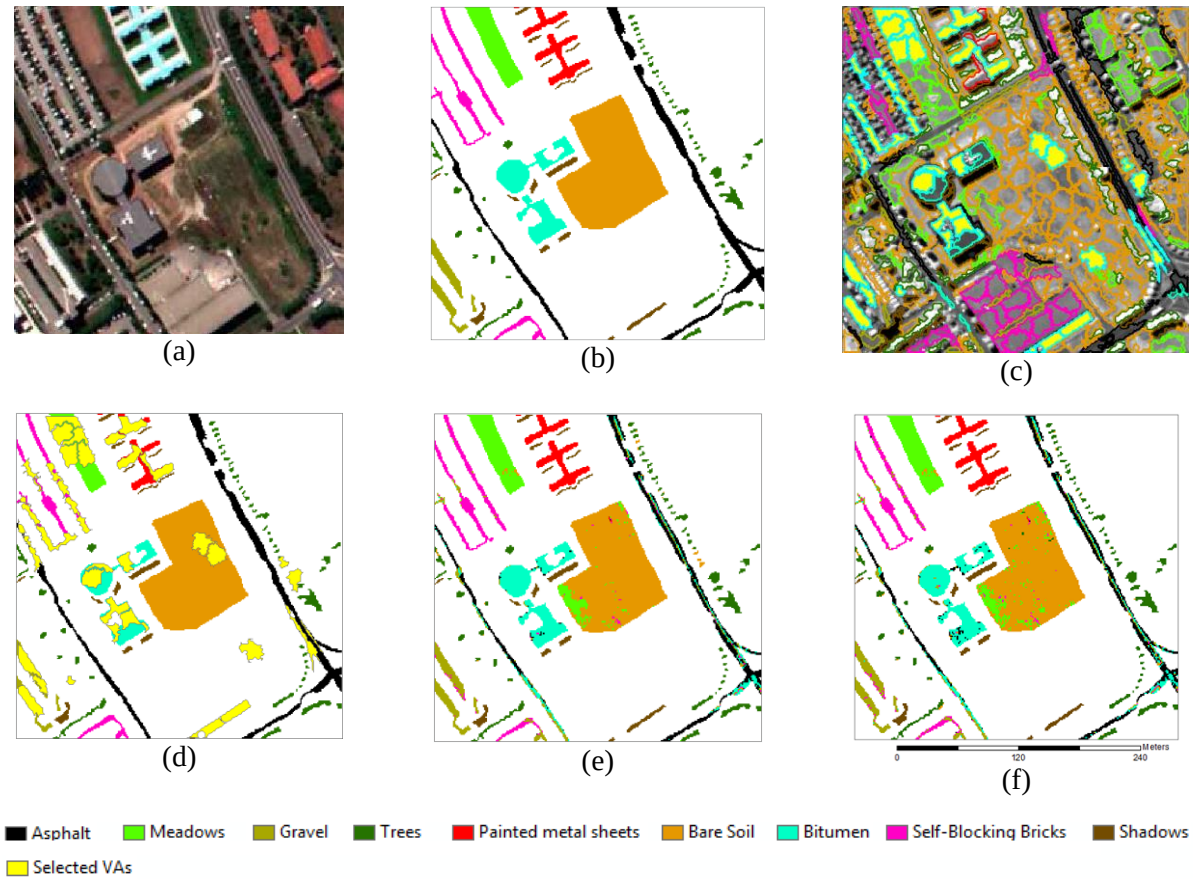


Figure 5.3. (a) The Pavia University datasets. (b) Ground truth map. (c) VA-generated training samples showing the selected VAs by the algorithm 1 (represented as yellow polygons) based on 15 initial training samples (d). (e) and (f) are the classification maps based on VA approach and the SSVM by using 15 label training samples.

Table 5.2 lists the overall accuracy (OA) and Kappa values of the classified images based on the different configurations described in Section 5.5.2. Table 5.2 shows that the VA-based approach exhibits better performance than the SSVM for all cases. The OA accuracy and Kappa values are increased when the VA-based semisupervised approach uses more samples. Tables 5.3 and 5.4 list the classification accuracy with 15 labelled training samples based on different values of β. As can be seen from Tables 5.3 and 5.4, the OA values are slightly improved when threshold β increases.

Table 5.2. OA values and Kappa values of the Pavia University dataset based on different methods: the supervised SVM (0%), and the VA-based algorithm where β is set to 0.1.

Percentage of each selected VA			0	10	20	30	40	50	100
Number of labelled training samples	**5**	OA	77.58	82.43	84.41	83.3	84.80	**84.47**	82.46
		Kappa	73.12	78.64	80.94	79.65	**81.40**	81.05	78.36
	10	OA	84.01	86.93	86.5	85.91	86.86	86.28	**86.79**
		Kappa	80.05	83.67	83.16	82.41	83.52	82.83	**83.43**
	15	OA	85.15	88.15	88.41	87.94	87.76	88.02	**89.25**
		Kappa	51.65	85.33	85.66	85.11	84.89	85.2	**86.67**

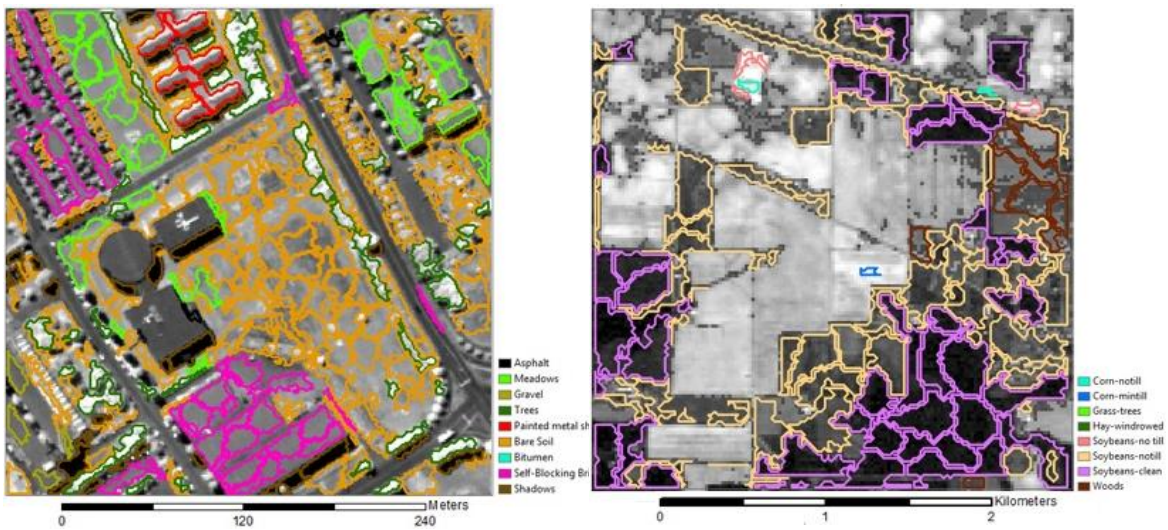
Table 5.3. The classification accuracy of the Indian Pine dataset with 15 labelled Training samples based on different values of threshold β.

Percentage of each selected VA			10	20	30	40	50	100
β	**.000**	OA	67.58	68.81	70.27	69.95	68.55	**71.33**
		Kappa	61.63	62.96	64.54	64.23	62.61	**65.87**
	0.15	OA	67.71	69.32	69.39	70.23	71.05	**72.42**
		Kappa	61.84	63.59	63.62	64.49	65.52	**67.04**

Table 5.4. The classification accuracy of the Pavia University dataset with 15 labelled Training samples based on different values of threshold β.

Percentage of each selected VA			10	20	30	40	50	100
β	**0**	OA	88.03	88.19	87.42	87.89	88.42	**89.36**
		Kappa	85.16	85.39	84.49	85.04	85.67	**86.82**
	0.15	OA	88.87	86.57	88.13	**89.32**	87.95	89.04
		Kappa	86.20	83.52	85.33	**86.74**	85.14	86.43

It is important to mention that increasing the β value not only improves the confidence level of the extracted VAs, but also helps to reduce the computational cost. However, increasing β can result in some classes being ignored by the VAs (Figure 5.4). For example, Figure 5.4(b) shows that VAs cannot identify the grass-trees and hay-windrowed classes when β is set to 0.5.



(a) (b)

Figure 5.4. The extracted VAs based on 15 labelled training samples when β is equal to 0.5: (a) the Pavia University dataset, (b) Indian Pine dataset.

Finally, Table 5.5 shows the CPU processing time (in seconds) spent on the different steps of the proposed method for the case illustrated in Figure 5.2(d).

Table 5.5. Computational time spent in each step of the VA-based approach based on 100% of the samples from the selected VA.

Creation step	Selection step	Classification step	Sum
135.0 Sec.	2.0 Sec.	6.0 Sec.	143.0Sec.

It should be noted that this configuration was designed to address the difficult case of the Indian Pine image. The image has more than 136 VA polygons with a range of 43 to 200 pixels, representing up to 70% of the pixels in the image. The processing is carried out using Repast running on an Intel CPU at 3.40 GHz with 16 GB of memory.

5.6. Conclusions

In supervised image classification, the process of selection label samples is often expensive and time consuming, especially for hyperspectral satellite images. This issue is usually addressed via the SSL algorithms in which a large number of unlabelled data along with the

available labelled data are utilised to build classifiers. Thus, SSL methods require less human efforts in sample collection. However, the quality of the selected unlabelled samples is important in order to obtain accurate classification results. To ensure the highest quality of the selected unlabelled samples, several studies have proposed using spatial information (Tan et al., 2015 and 2014; Wang et al., 2014; Dópido et al., 2013; Bruzzone and Persello, 2009). These methods usually utilise a predefined geometry (e.g. a fixed size window) or parameters (e.g. kernel size) to identify the most helpful unlabelled samples. This chapter presented a new approach, based on an actual dynamic geometry to generate new labelled samples from relatively few initial training samples, without setting predefined segmentation parameters or shapes.

To reach this aim, we defined the components of VAs, including geometry, geometry methods, state, transition rules, neighbourhood and neighbourhood rules. The geometry of VAs is modelled as a directional planar graph, using a set of spatial reasoning relationships and geometric operators to implement a set of dynamic geometric behaviours, such as growing or joining. The geometry and geometry rules enable VAs to implement a dynamic, irregular geometry to segment and classify concurrently new training samples. State and transition rules were applied to specify the classes and membership criteria allowing VAs to determine their class dynamically as their geometry evolves. To update their own class, VAs can rely on a membership function as well as a crisp test related to spectral descriptors. By being spatially aware via their neighbourhood components and neighbourhood rules, VAs can explicitly perceive each other in image space. To identify the most effective VA-generated samples, we used the NN algorithm formulated on the spectral angle function. The selected VAs were applied by the SVM classifier to label the remaining pixels in the classification step. The experimental results demonstrate the desirable performance of the new approach.

The process to find, extract and evaluate the training samples is an automatic process based on a fully random sampling scheme. In the future, we will explore the possibility of developing an intelligent initialisation process in image space based on the characteristics of the classes of interest in the real-world that could significantly reduce processing time. Two thresholds, α and β, were determined manually. To increase the degree of automation of the proposed method, some information of the initial labelled samples can be applied to automatically determine β. The selection process is performed through the threshold α. The

use of the geometric information of VAs along with the spatial relationship between VAs instead of the threshold α would also be interesting to study.

So far, we have seen some examples of the VA model to improve the results of the pixel-based approaches. In Chapter 4 and Chapter 5, we used the VA model to create training samples for supervised and unsupervised approaches. In the next chapter, we demonstrate the application of the VA model for image classification in the context of the GEOBIA approach.

Chapter Six

Vector agent model for GEOBIA image classification

Abstract

The aim of this chapter is to show the capabilities of VAs in extracting geo-objects under different conditions. This is tested in the context of GEOBIA image classification. We use VAs to detect and extract real-world objects from different raster datasets, including a WorldView-3 image subset, an IKONOS image subset and a LiDAR DSM. The results show the flexibility of VAs to incorporate various transition rules in order to address a complex system composed of a number of heterogonous components, exhibiting adaptive properties through space and time. The obtained results are compared to those obtained using a traditional GEOBIA approach. Experimental tests indicate that VAs outperform GEOBIA, marking them out as viable new processing units for remote sensing image classification.

6.1. Introduction

In simple terms, the Geographic Object-Based Image Analysis (GEOBIA) approach refers to a group of image analysis techniques that use the spatial and thematic knowledge of meaningful image objects in geographic space to analyse an image (Hay and Castilla, 2008). The core idea of GEOBIA for an object-based classification system (OBC) is to use expert knowledge for image classification. This use of expert knowledge allows the object-based approaches to rely on more information and produce more reliable maps compared to the conventional pixel-based approaches.

Despite the advantages that the GEOBIA approach offers such as spatial knowledge, classical GEOBIA lacks the necessary abilities to directly address real-world objects in as dynamic a fashion as human interpreters do. To address real-world objects, the GEOBIA approach usually uses a sequential process of image segmentation and classification (see Section 2.3). In this context, the classical GEOBIA approach lacks the ability to take full advantage of the spatial and non-spatial information (e.g. shape or thematic meaning) of real-world objects during an image classification process. To tackle this limitation, the

GEOBIA approach uses a cycle of classification and segmentation (see Figures 1.1, 1.2 and 1.3). In this way, the GEOBIA approach employs the results of a pre-classification step in order to take advantage of the domain-dependent knowledge of real-world objects.

Although the above strategy provides the necessary contextual information, it still relies on a static geometry obtained through a segmentation process. In this sense, objects cannot change their geometry during the classification process (see Section 2.1.2.2). This causes problems for the subsequent classification stage, which depends upon a correspondence of segmented object and real-world object to be effective.

In this framework, it would also be difficult to find the source of errors; that is, whether the error is from the segmentation or the classification steps. In other words, the GEOBIA approach separately models the geometry and theme of geo-objects, but there is no specific rule to determine the accuracy of the segmentation. According to Hay and Castilla (2008),

> *"The visual appeal of image-objects, their easy GIS-integration and their enhanced classification possibilities and information potential have attracted the attention of major RS image processing vendors, who are increasingly incorporating new segmentation tools into their packages. This provides a wider choice for practitioners, but promotes confusion (among different packages, options, syntax, etc.) and makes it more difficult to develop a cohesive GEOBIA community.* ***Will a lack of protocols, formats, and standards lead to a segmentation of the field rather than a consolidation?****"*

To address the challenges highlighted thus far, this chapter assesses the capabilities of VAs as a new dynamic processing unit that improves on current GEOBIA methods.

In the next section, the proposed method is presented. The experimental results will be discussed in Section 6.3 and Section 6.4. A short summary and ideas for further work are discussed in Section 6.5.

6.2. Vector Agent

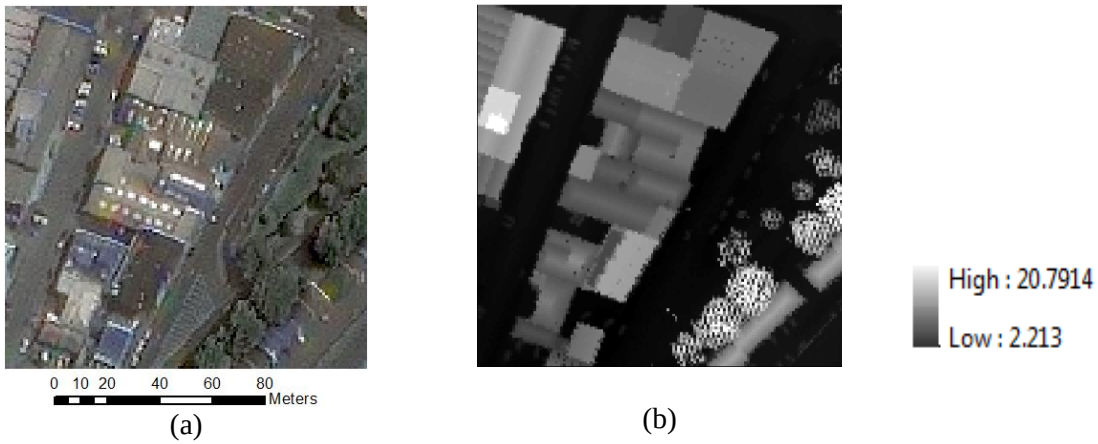
In the context of the VA model, meaningful objects can dynamically change their own geometry and state, as well as directly perceive each other, thus allowing the real world environment captured by the image to be modelled in a dynamic fashion similar to a human interpreter. VAs simultaneously segment and classify image objects from initial pixels to the final classified objects based on an evolving process enabled by an iterative scheme that

involves constant interactions between all VA components. The aim of the VA model is to provide an autonomous processing unit for image analysis that is implemented using VAs.

Supporting this general account of the model are the following examples that demonstrate the various stages of processing. To implement the VA model, the proposed method uses the Repast modelling framework discussed in Section 4.3.1. To describe the VA model, we use a UML. The main components and methods of constructing the vector agent-based image classification are shown in Figure 3.12 in Chapter 3.

6.3. Dataset 1

The first example model includes a collection of spatial goal-oriented agents to extract five different classes, building, meadow, road, shadow and tree, from a multispectral IKONOS image and LiDAR DSM (Figure 6.1). The image benefited from a pan-sharpening process yielding four spectral bands, namely blue, green, red and near infrared, with 1×1m spatial resolution.



(a) (b)

Figure 6.1. (a) Subset of an IKONOS multispectral image with a size of 140×140 pixels obtained from an image fusion process (panchromatic pixel size 1m × 1m; multispectral pixel size 4m × 4m for blue, green, red and near infrared bands) and (b) LiDAR DSM with 1m spatial resolution.

6.3.1. Proposed methodology

As can be observed from Figure 3.12 in Chapter 3, there are two main types of agents in the proposed approach: MakerAgent and VecAgent, which both inherit from the top level agent in the object hierarchy, SimpleAgent.

6.3.1.1. MakerAgent

MakerAgent creates the VAs in the first step, putting them into context through the generateVA method. To create and add VAs to the vector space, MakerAgent uses the rules in Table 6.1.

Table 6.1. Initialisation rules.

Name	Description
Rule 1:	The candidate pixel $\boldsymbol{x_c}$ and its immediate neighbours should be found to be members of the same class.
Rule 2:	An elevated VA can exist in vector agent space if its height is more than T_L.
Rule 3:	The standard deviation of pixel $\boldsymbol{x_c}$ and its immediate neighbours should be less than $\sigma_h \leq T_H$.

To implement *Rule 1*, MakerAgent uses the SVM classifier, which is trained by a limited number of training samples, to evaluate membership values of candidate pixels for each of the classes (e.g. meadow, road, shadow and tree). To initiate elevated VAs (e.g. tree), MakerAgent applies *Rule 1* and *Rule 2*. In this case, T_L is set to 2.70m computed by the lowest elevation on DSM (in this case, 2.20m) plus the minimum height of tree objects (e.g. 0.5m). To create building VAs, MakerAgent uses *Rule 2* and *Rule 3*. In our case, T_L is equal to 5 meters calculated according to the lowest value on DSM plus the standard height of buildings (e.g. 2.80m) in an urban area. In fact, T_L values divide pixels into three main groups: ground pixels, non-ground pixels and intermediate pixels (Figure 6.2).

For the ground pixels, T_L is less than or equal to 2.70m (see Figure 6.2). These pixels obviously belong to the meadow, road and shadow classes. As the terrain is quite flat in our example, a candidate pixel belongs to the building or tree classes if its elevation is more than 5m. Pixels between 2.70m and 5m are regarded as intermediate pixels. In the case that the terrain is not flat, these thresholds can be updated according to average slope on LiDAR DSM datasets. Because the tree objects produce heterogeneous elevation information in contrast to buildings (Figure 6.1(b)), Maker agents also employs standard deviation values to separate buildings from trees via *Rule 3*. In this case, T_H can be defined by the accuracy of the DSM datasets or a human interpreter.

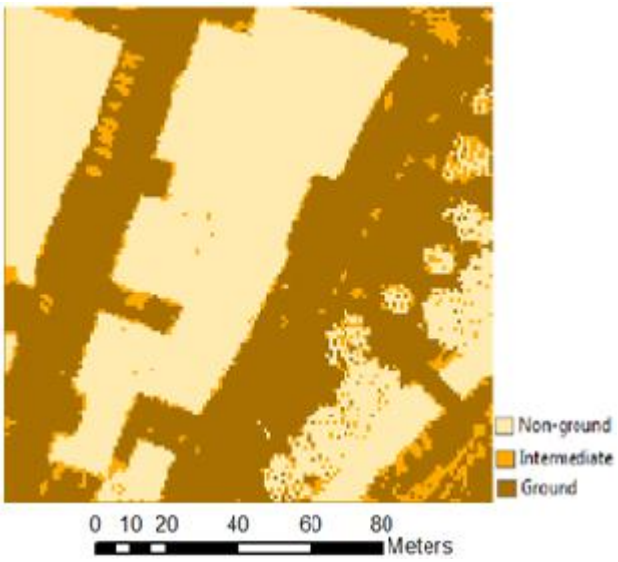


Figure 6.2. The classification map based on the elevated information including 3 different classes: non-ground, intermediate and ground classes provided by ArcGIS software according to the T_L.

The initialisation process can obey a specific sampling scheme (e.g. fully random, stratified random, systematic unaligned random, etc.). In our example, MakerAgent uses a fully random schema to initiate VAs in vector space. MakerAgent is also responsible for facilitating the coordination between VAs. To do this, MakerAgent applies the trackVA method. For example, the MakerAgent retrains the SVM classifier based on the VA-generated signatures if all VAs are passive.

6.3.1.2. VecAgent

The instances of the VecAgent class, namely VAs, use the VAGeometry, VAPoint, VALineString, VAPolygon and VAGeometryFactory classes (Figure 3.12 in Chapter 3) to determine and change their geometry stored in the polygon variable. At each increment, VAs will update the polygon variable. VAs can also generate a new VA or set of VAs (see Figure 3.8). In the shrinking/splitting process, the shrinking of a VA may lead to the initialisation of a new VA. Each new VA has its own attributes specified in terms of its geometry on the input raster datasets.

Through the class variable, VAs can determine the thematic meaning of the real-world objects. In this case, VAs employ VATransitionRule and VANeighbourhoodRule classes to find and update their classes. VASignature and VAFeatureSignature variables allow the VAs to store and update their own spectral information at each iteration. To implement the neighbourhood rules, VAs calculate the spatial distance in the vector space. VAs use these

variables and classes via the Evolving method to find and extract their geometry and classes during the simulation process.

Time is considered as discrete time steps supported through the Schedule class in Repast Simphony. The polygon and VASignature variables are the elements of the VAs updated at each increment. Each VA uses a snapshot approach to model its changes. At each time increment, this model simply provides a new map which is composed of a temporally homogeneous unit (Figure 6.3).

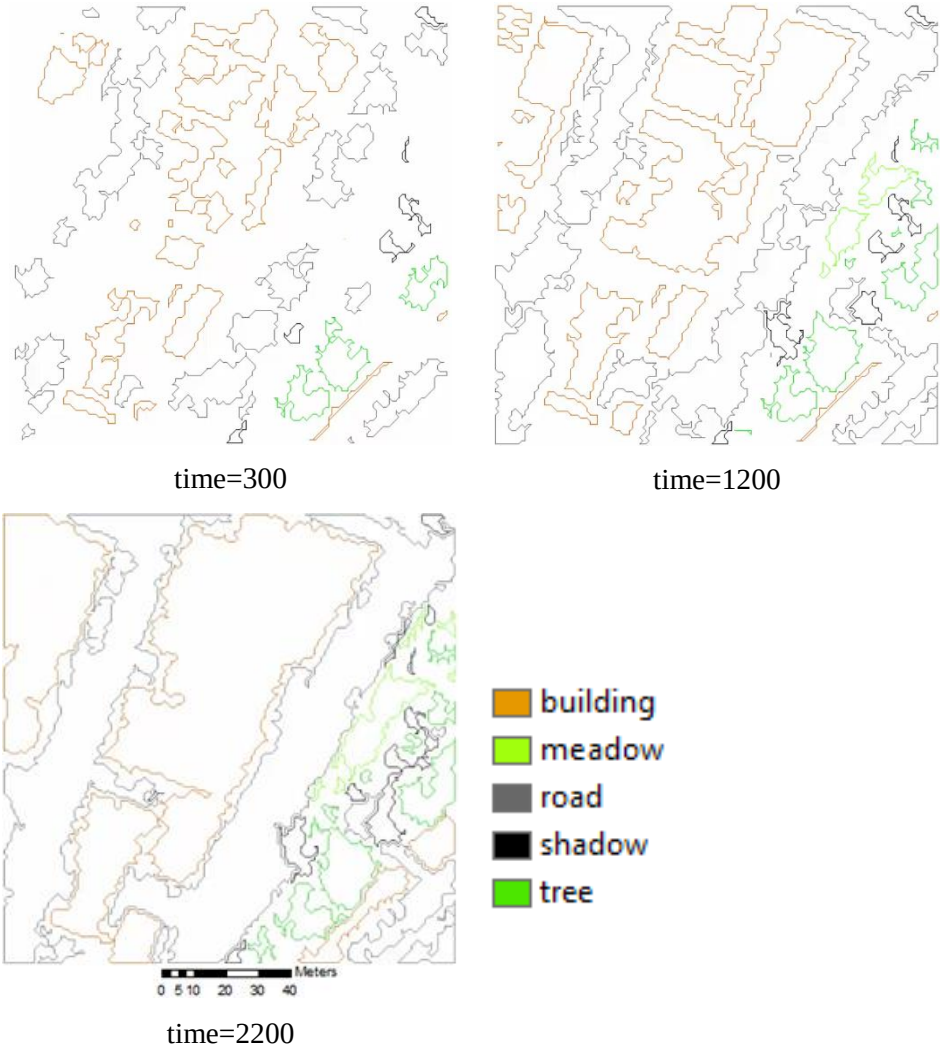


Figure 6.3. Simulation result over 2200 time steps in the agent modelling shell Repast Simphony.

6.3.1.3. Process overview and scheduling

The model simulates the interactions between VAs and their environment over time. The processes running during each time step and their respective relations are illustrated in UML sequence diagrams (Figure 6.4). This diagram presents an outline of the sequence of

processes and the schedule of methods applied by agents at each discrete time step. Each process is defined according to a certain class of agents from the UML class diagram.

The proposed model works in five main steps: initialisation, growing, developing, construction and production. First, the MakerAgent uses the initialisation rules (see Table 6.1) and the generateVA method to create and add VAs to the vector space. Furthermore, the MakerAgent checks the VAs as a coordinator agent via the trackVA method. A parallel mechanism allows the MakerAgent and VecAgent classes to perform simultaneously their own respective methods.

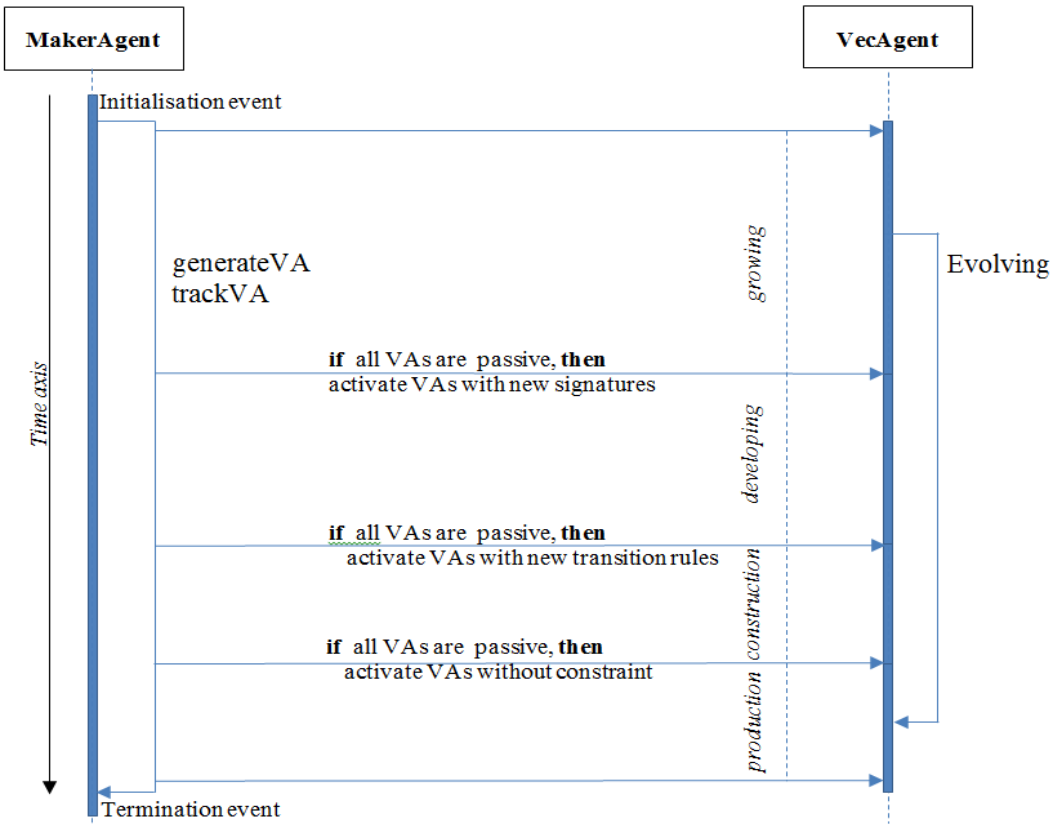


Figure 6.4. UML sequence diagram of the MakerAgent and its methods including generateVA and trackVA; and VecAgent and its main method, namely Evolving executed through an iterative mechanism.

In the growing step, VAs automatically change their geometry in terms of geometric methods and transition rules based on initial training samples. To do this, VAs employ a mechanism of adaptation and updating through the Evolving method (e.g. Figure 6.3). When all VAs are passive (e.g. Figure 6.5(a)), MakerAgent uses the VA-generated samples to retrain the SVM classifier in the developing step.

As the use of all VA-generated samples can greatly increase computational cost, MakerAgent applies a Nearest-Neighbour (NN) algorithm to identify the most informative VA-generated samples. Let $VA_j, j = 1, \cdots, m$ denote the j-th VA of the k class (e.g. meadow) generated via the growing process, and m be the number of VAs. A similarity metric to measure the distance between the samples in VA_j and the initial labelled samples X_L of the k class is defined as:

$$D\ (VA_j, X_L) = min\{d(x_i, x)|x_i\ \in VA_j, x\ \in X_L\ \}, \tag{6.1}$$

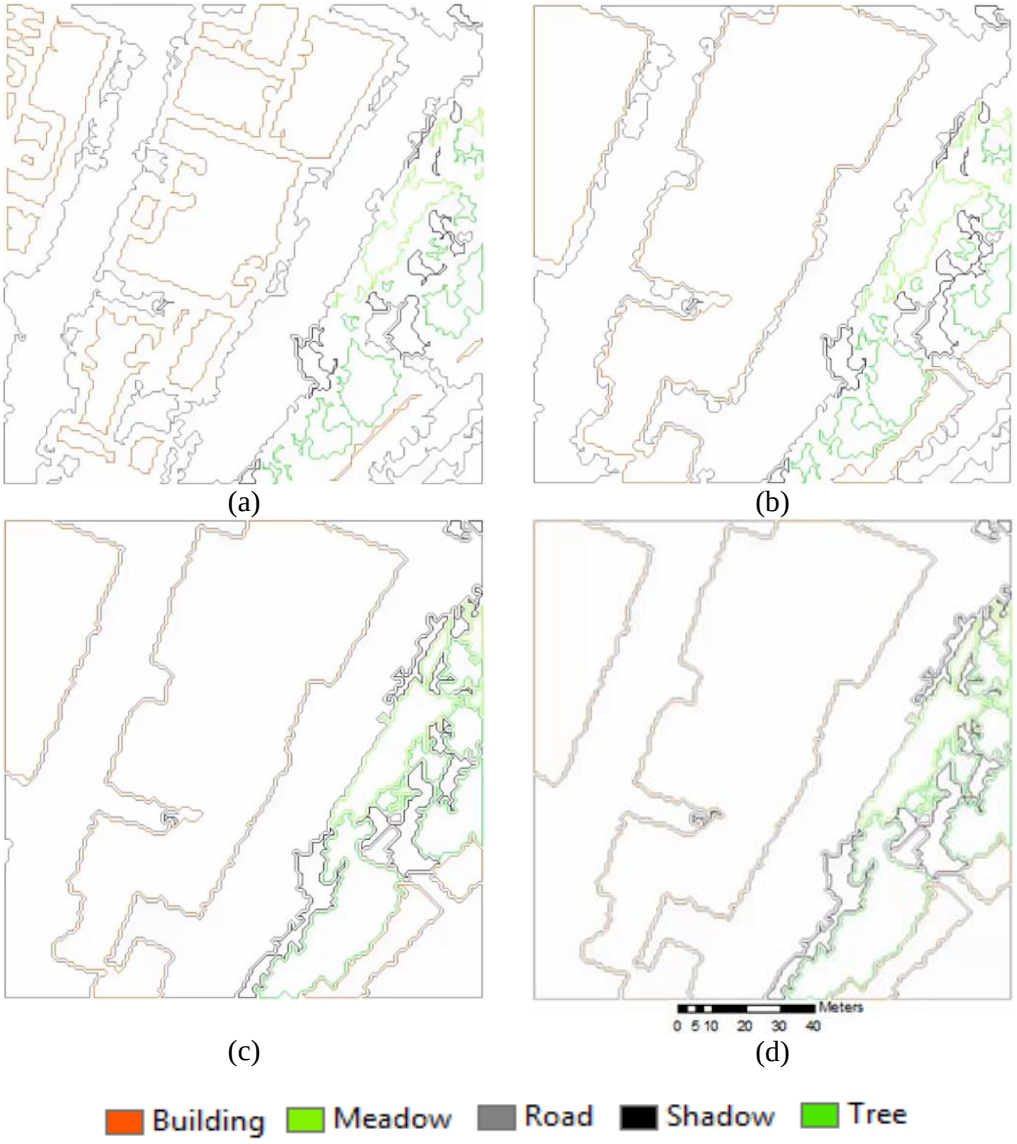


Figure 6.5. Simulation results of each step: (a) growing step and (b) developing step where there is only interaction between a VA and its environment (see Section 7.3.1.4). (c) construction step. (d) production step where there is interaction between VAs and between VAs and their environment.

where x_i and x are the vectors specified by the mean value in each spectral band, and $d(x_i, x)$ is the distance of two samples (x_i, x) formulated on spectral angle distance. The selected VAs, namely $\mathcal{D}_s$, can be expressed as follows:

$$\mathcal{D}_s = \cup_{i=1}^{K} VA_i, \tag{6.2}$$

where *K* is the number of classes. After that, random selection is applied to select a smaller number of pixels (e.g. 40 pixels) within each eligible VA for the learning process. VAs are then activated by a new SVM model by the MakerAgent. Figure 6.5(b) shows the results of the developing step.

In the construction step, VAs change their behaviour based on new transition rules. In this case, all VAs except the building VAs continue the developing process even where there are intermediate pixels. Thus, a VA can capture a candidate pixel $\boldsymbol{x_c}$ only if it is an adjacent pixel. If pixel $\boldsymbol{x_c}$ belongs to another VA, VAs use the interaction rules (see Section 6.3.1.4) to negotiate with each other (e.g. Figure 6.5(c)). In the production step, pixel $\boldsymbol{x_c}$ belongs to a VA only if it is an adjacent pixel. If pixel $\boldsymbol{x_c}$ is a dependent pixel, VAs (except building VAs) employ the interaction rules (see Section 6.3.1.4) to determine the boundaries of real-world objects in the image space (e.g. Figure 6.5(d)). In the production step, all VAs are regarded as active agents (see Appendix B).

6.3.1.4. Interactions

The interactions among VAs, and between VAs and their environment are the primary basis for the dynamics of the model. Table 6.2 summarises the rules employed by VAs through the Evolving method.

Table 6.2. Interaction rules.

Name	Formula
Rule 4:	$VA_t.S.class = \boldsymbol{x}_c.class$
Rule 5:	$P_a\,(VA_{t+1}) - P_b(VA_{t+1}) \geq \beta$
Rule 6:	$DN_{DSM}\,(\boldsymbol{x_c}) \geq DN_{DSM}\,(\boldsymbol{x_b})$
Rule 7:	$P_{VA_{2,t}}(\boldsymbol{x_c}) \leq P_{VA_{1,t}}(\boldsymbol{x_c})$

i. Vector agent-environment interaction

In the event that a candidate pixel $\boldsymbol{x_c}$ is an isolated pixel, all VAs except the building VAs use *Rule 3* and *Rule 4* to evaluate pixel $\boldsymbol{x_c}$. The VA uses the SVM classifier to evaluate the class of pixel $\boldsymbol{x}_c$. $P_a(VA_{t+1})$ and $P_b(VA_{t+1})$ are regarded as the largest and second largest

probability of VA to belong to the classes *a* and $b \in K = \{1,2,\cdots,k\}$, a set of *k* class labels. $P_a(VA_{t+1})$ and $P_b(VA_{t+1})$ are computed through the SVM classifier and the mean value in each spectral band. The mean values are calculated based on the assumption that pixel $\boldsymbol{x_c}$ belongs to the VA at time *t+1*. β is a parameter applied by each VA to control the effect of its signature at each iteration. This structure allows the VA to evaluate pixel $\boldsymbol{x_c}$ at two different levels. At the first level, the VA locally evaluates pixel $\boldsymbol{x_c}$ based on the labelled training samples. At the second level, the above function allows the VA to evaluate pixel $\boldsymbol{x_c}$ in terms of its own signature at each iteration. Pixel $\boldsymbol{x_c}$ belongs to the VA if it can satisfy *Rule 4* and *Rule 5*.

In the case that pixel $\boldsymbol{x_c}$ is a non-ground pixel, building VAs use *Rule 3* and *Rule 6* to capture pixel $\boldsymbol{x_c}$. In the growing step, the VA building applies *Rule 3* to evaluate pixel $\boldsymbol{x_c}$. This rule allows the VAs to implement a region growing algorithm to find their initial boundary (Figure 6.5(a)). As the standard deviation of pixels lying on the boundary of roofs is high, building VAs use the transition rules, namely Rule 6, formulated on the Gaussian kernel, to delineate their boundary in the developing step (Figure 6.5(b)).

In this case, VAs simultaneously change their behaviour from region-based to edge-based via *Rule 6*. $DN_{DSM}\ (\boldsymbol{x_c})$ and $DN_{DSM}\ (\boldsymbol{x_b})$ are convoluted output pixel values of pixel $\boldsymbol{x_c}$ and pixel $\boldsymbol{x_b}$ on DSM computed via Equation 6.3 and Equation 6.4. Pixel $\boldsymbol{x_b}$ is the adjacent pixel of pixel $\boldsymbol{x_c}$ that the VA uses to create a new vertex centered on pixel $\boldsymbol{x_c}$.

$$DN_{out} = \frac{1}{W}\sum_{i=1}^{d}\sum_{j=1}^{d}\omega_{ij}\,DN(i,j)\quad , \tag{6.3}$$

$$W = \sum_{i=1}^{d}\sum_{j=1}^{d}\omega_{ij}, \tag{6.4}$$

where *d* is set to 3. $DN(i,j)$ is the pixel value in the input DSM at location *(i, j)* and DN_{out} is the convoluted output pixel value. ω_{ij} and *W* are the values of the element at location *(i, j)* in the kernel and sum of all kernel elements, respectively.

ii. Vector agent- vector agent interaction

In the case that pixel $\boldsymbol{x_c}$ belongs to another VA, all VAs except building VAs use the interaction rule, *Rule 7*. The interaction between VAs can explicitly be carried out based on the geometry and the state of involved VAs (Equation 6.5).

$$l(VA_{i,t}, VA_{j,t}) \leq r\sqrt{2}, \tag{6.5}$$

where $VA_{j,t}, j \in \mathbb{N}$, l is the Euclidean distance and r is the resolution of raster datasets. In the event that pixel $\boldsymbol{x}_c$ lies on the common boundaries between two VAs such as $VA_{1,t}$ and $VA_{2,t}$ having different classes k_1 and k_2, respectively, $VA_{1,t}$ can capture $\boldsymbol{x}_c$ during a growing/shrinking (Figure 6.6) or shrinking/splitting (see Figure 3.8) process if it can satisfy *Rule 7*, where $P_{VA_{2,t}}(\boldsymbol{x_c})$ and $P_{VA_{1,t}}(\boldsymbol{x_c})$ are calculated by the SVM classifier. These operators remove and transfer a vertex from one to the other VA.

(a) (b) (c)

Figure 6.6. An example of growing/shrinking process where pixel xc lies on the shadow VA. (b) is more likely to belong to the tree VA based on Rule 7 (c).

The spatial and spectral space provides all of the vector agents with their required resources and the spatiotemporal dynamics of interactions are influenced by the object's characteristics at each location. Accordingly, the interactions vary within the study area and the VAs will behave heterogeneously in terms of their elements.

6.3.2. Implementation and results

6.3.2.1. Initialisation

VecContext class initiates the MakerAgent. In this event, the normalised input images and DSM dataset along with the ground truth (GT) map (Figure 6.7(d)) are read in raster format. The initial training samples are randomly selected from the GT map using 15 pixels for each class except the building class. The number of VA-generated samples is set to 40 pixels. In other words, the MakerAgent only selects VAs with a size of more than 40 pixels to retrain the SVM classifier in the developing step. In this experiment, the Radial Basis Function (RBF) kernel is chosen for classification (Camps-Vallas et al., 2007). To find the appropriate kernel parameters for the SVM classifier, MakerAgent uses a 10-fold cross-validation

algorithm (see Section 2.2.2.2). In our example, β is set to 0.1. As already mentioned in the previous chapter, the increasing β value can improve the confidence level of the extracted VAs. However, some classes may be ignored by the MakerAgent if the VA model uses a high value for threshold β (e.g. Figure 5.3 in Chapter 5).

6.3.2.2. Discussion

Figure 6.7(g) displays the classification map produced by the VA-based method. To evaluate the accuracy of the VA model, a conventional GEOBIA method is applied. First, the image and the DSM datasets are segmented through a multiresolution segmentation process using eCognition software (Figure 6.7(f)). Figure 6.7(b) displays the image created by a Normalized Difference Vegetation Index (NDVI).

As Figure 6.7(b) shows, it is difficult to separate the meadow class from the tree class. This issue can be resolved by using the height information from the LiDAR DSM. Figure 6.7(c), produced by a Laplacian edge detection operator, shows that the height information from the LiDAR DSM dataset is key to identifying elevated objects (e.g. buildings and trees). As the information contained in the LiDAR DSM and in the image is important, all datasets are applied for the segmentation process. The segmented image is then classified via a set of structural (e.g. elevated information), spectral (e.g. NDVI) and contextual rules (e.g. neighbourhood relationships) in the classification step (Figure 6.7(f)). A visual assessment of the classification maps (Figure 6.7(f) and Figure 6.7(g)) shows that the VA-based approach provides more satisfactory classification results than the conventional GEOBIA approach (e.g. building objects). A comparison between Figure 6.7(c) and Figure 6.7(g) indicates the VA buildings are very close to the real structure of the buildings in detail, while the VAs do not impose any constraint on building shape (e.g. scale). As can be seen in Figure 6.7(c), the LiDAR DSM image shows the jagged edges, especially where there are features such as chimneys and windows. These results indicate the high potential of the VA model in dealing with noise caused by features such as windows or chimneys on LiDAR DSM images (Figure 6.7(c)). Figure 6.7(g) shows that in terms of meadow and tree classes, the VA model can separate the meadow objects from tree objects without setting a threshold. Figure 6.7(d) displays the ground truth map manually created by an expert operator to evaluate the performance of the VA model. Classified objects in Figure 6.7(g) and 6.7(f) are compared with their corresponding reference objects in Figure 6.7(d).

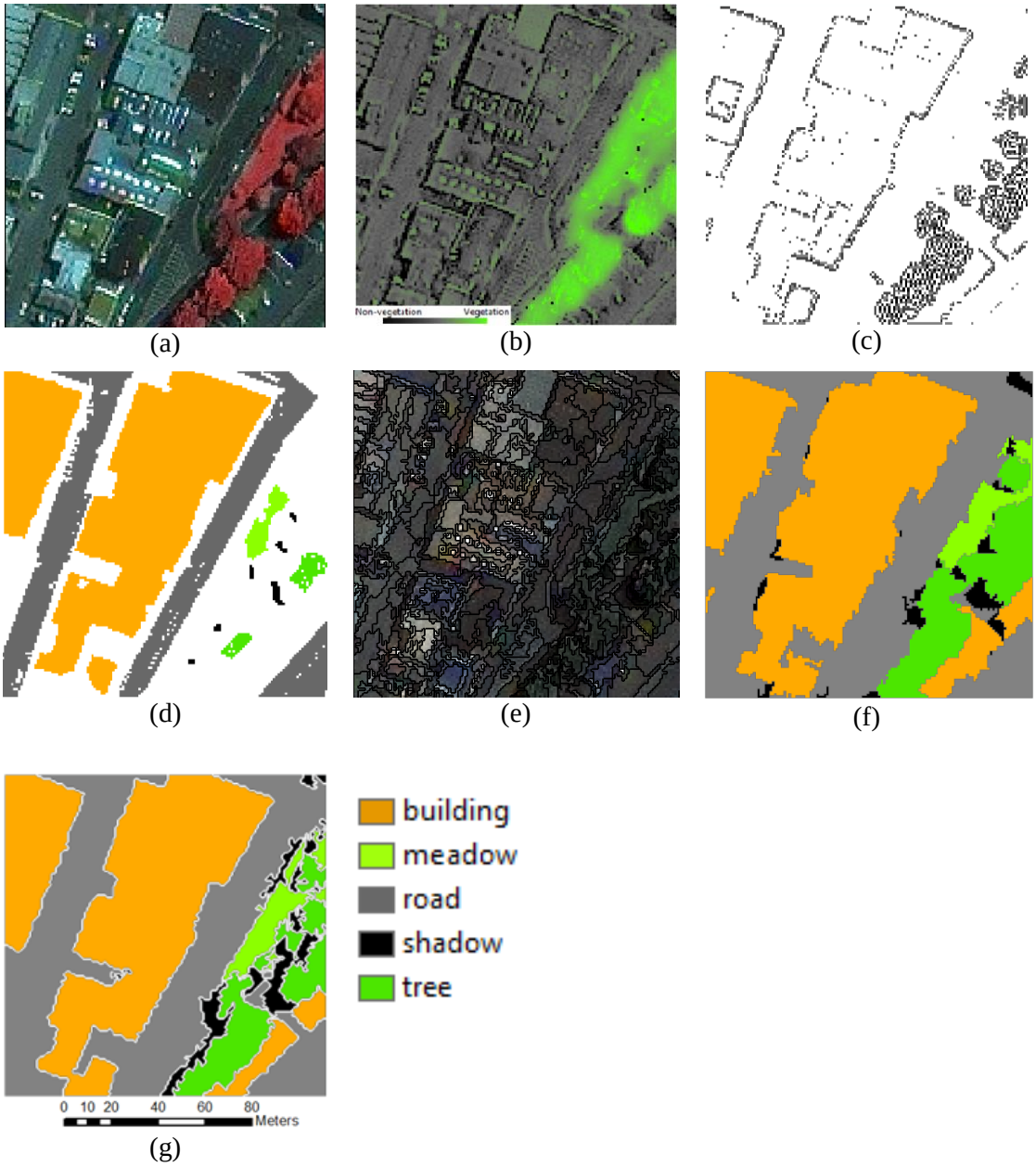


Figure 6.7. (a) A false colour combination of an IKONOS image. (b) NDVI map. (c) Extracted edges in terms of a Laplacian edge detection kernel. (d) Ground truth map manually selected by an expert operator. (e) Segmented image based on scale=14, shape=0.5, compactness=0.5 and layer weights=1(f) Classification map based on the GEOBIA method. (g) VA map.

Table 6.3 displays the results of this comparison based on TP, FP and FN values. For the assessment of the accuracy of the proposed method, three indices, completeness, correctness and quality, are computed through Equations 4.10, 4.11 and 4.12 (see Chapter 4). From Figure 6.8, it can be observed that the completeness, correctness and quality rates of the VA-based and the GEOBIA approaches for the building class are similar. However, the geometric structure of the extracted boundaries indicates the VA buildings are more similar to the true buildings compared to the GEOBIA approach. The main reason for this error in

the GEOBIA approach is related to the segmentation step. As the segmented objects are formulated on the spectral and elevated information, it is difficult to distinguish buildings in relation to their neighbourhood (Figure 6.7(e)).

Table 6.3. Comparison between the results of the VA-based and object-based approach. TP, FP and FN values are divided by the number of pixels within each ground truth class.

Classification method	**Criteria**	**Object class (%)**				
		Building	Meadow	Road	Shadow	Tree
GEOBIA	TP	99.48	**98.98**	98.75	74.29	**100.00**
	FP	0.57	**0.00**	1.09	11.43	8.20
	FN	0.52	1.02	1.25	25.71	0.00
Vector Agent	TP	**99.88**	**98.98**	**100.00**	**98.57**	98.36
	FP	**0.00**	**0.00**	**0.17**	**7.14**	**1.64**
	FN	0.12	1.02	0.00	1.43	1.64

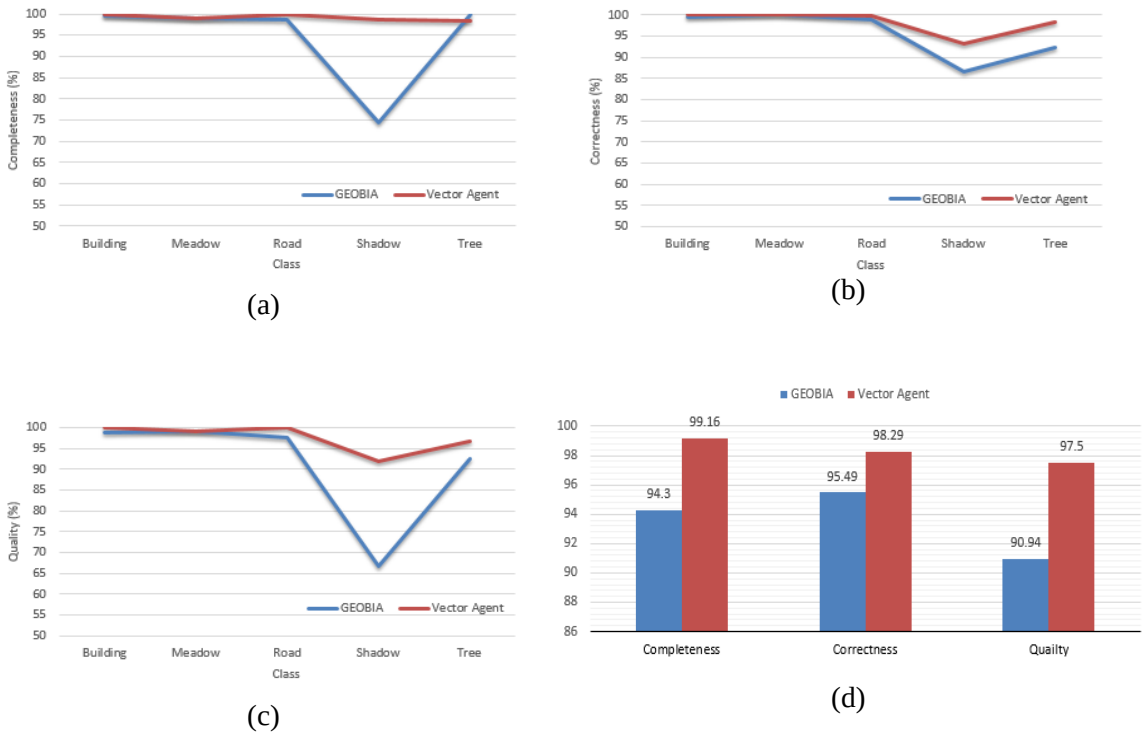


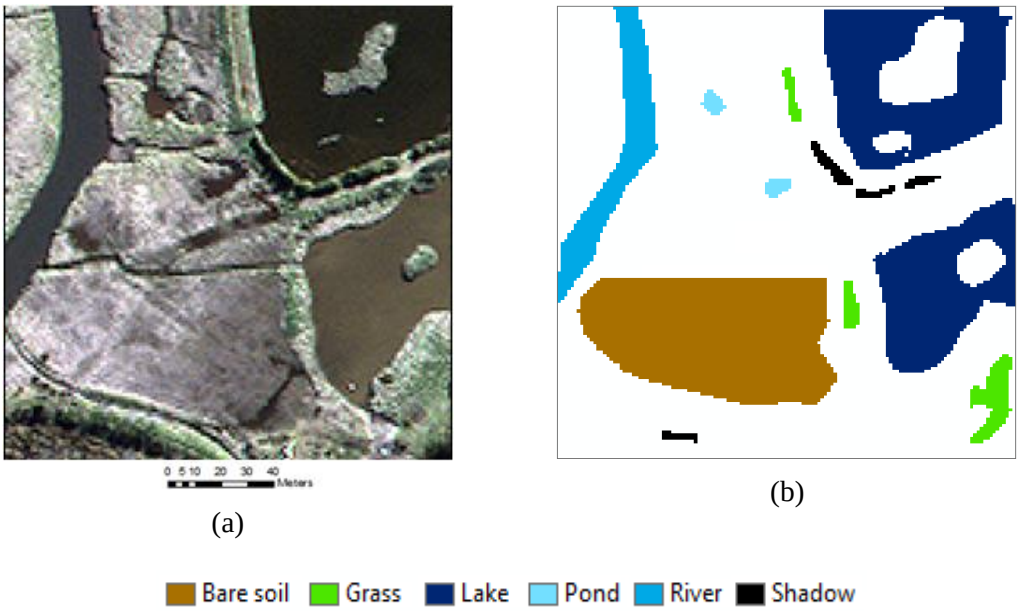
Figure 6.8. Comparison between the VA-based approach and the GEOBIA method based on the completeness (a), correctness (b) and quality (c) indices. (d) The average rate of these indices in terms of VA and object-based approaches.

The results show a considerable improvement for the shadow class (more than 20%). There are two issues to consider when evaluating this result. Firstly, the segmented objects may lack the necessary characteristics of the shadow class. Secondly, there is an unbalanced spatial distribution of the shadow objects on the ground truth map. During photo-interpretation, it was difficult to separate shadow areas from other classes because of the low contrast between objects. When considering the road class, the results of the VA-based approach show more reliable results when compared to the GEOBIA method (Figure 6.8(c) and 6.8(d)). Figure 6.8(d) shows that the VA model achieves an improvement of more than 5% for completeness, 3% for correctness, and 6% for quality compared to the GEOBIA model. The results from the proposed method confirm the high potential of VAs in integrating the segmentation and classification step in order to directly address real-world objects in image space. The framework of this method is developed in Repast using an Intel CPU running at 3.40 GHz with 16 GB of memory. The VA model took 432 seconds to find the optimum values for the RBF kernel, train the SVM algorithm and classify the image.

6.4. Dataset 2

In the second experiment, the VA model is experimentally tested on a subset of WorldView-3 image 8 multispectral bands (red, red edge, coastal, blue, green, yellow, near-IR1 and near-IR2) from a rural area near Dunedin, New Zealand (Figure 6.9(a)). The image dataset is a subset with a size of 140×140 pixels (1.20m pixel size) of a scene containing six land cover classes: bare soil, grass, lake, pond, river and shadow (Figure 6.9(a)).

Figure 6.9(b) displays the ground truth (GT) map manually extracted by an expert operator. In the initialisation step, 15 pixels are randomly identified from GT map as training samples. It is important to mention that all water bodies, including pond, river and lake, use the same training samples, 15 pixels for all water objects. Similar to the previous experiment, the algorithm uses a 10-fold cross-validation algorithm to identify the parameters of the RBF. β is set to 0.1. The MakerAgent only uses 40 pixels from the eligible VA samples to train the SVM classifier.



(a) (b)

Figure 6.9. (a) Subset of a WorldView-3 multispectral image with a size of 140×140 pixels from a rural area in Dunedin, New Zealand, Digital Globe Foundation, www.digitalglobefoundation.org. Ground truth map manually created by an expert operator.

6.4.1. Implementation

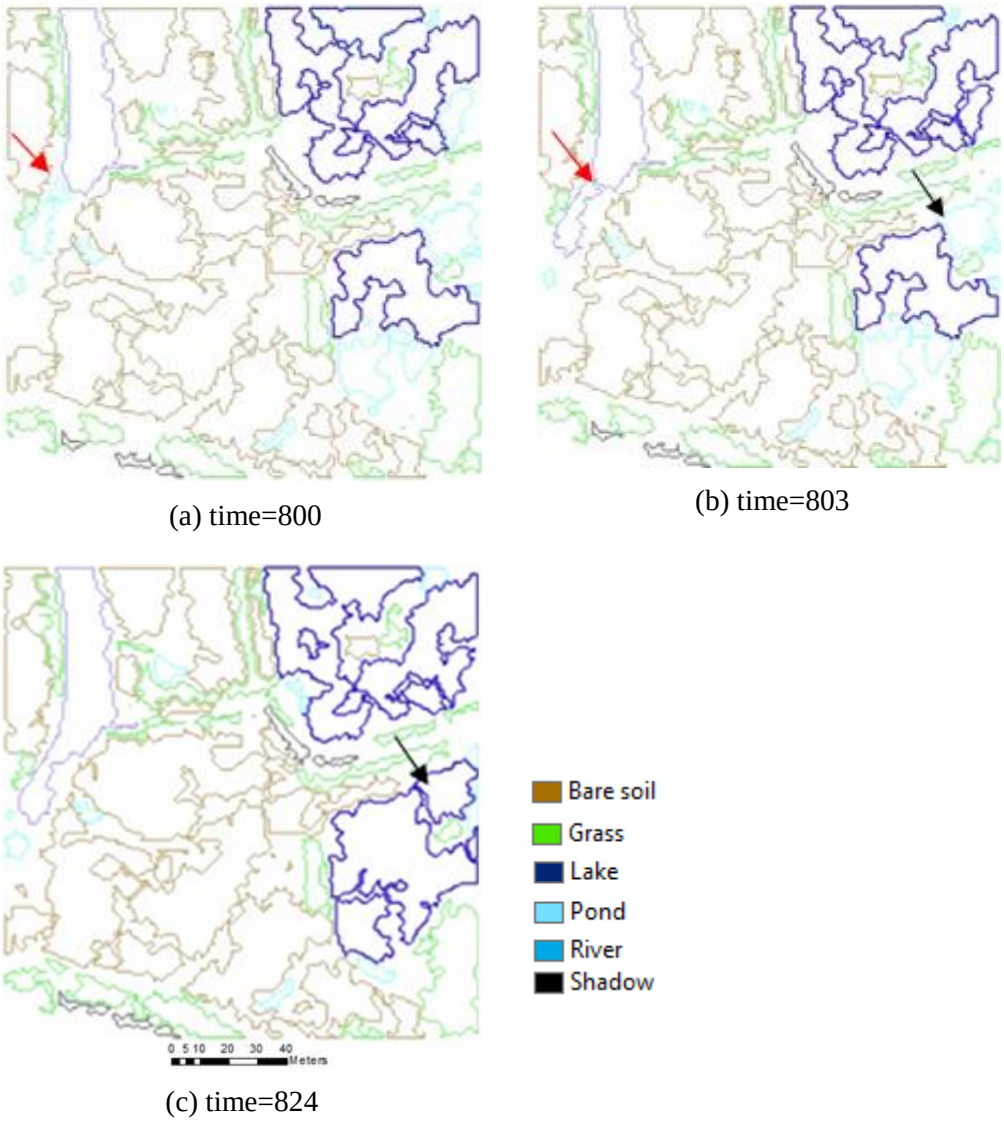
In this example, the VA model is implemented in four main steps: initialisation, growing, developing and production. In the first step, the MakerAgent uses *Rule 1* in Table 6.1 to create and add VAs to the vector space. The MakerAgent also employs a parallel mechanism to check the VAs as a coordinator agent via the trackVA method.

VAs change their geometry in terms of geometric methods and transition rules based on initial training samples. As the pond, river and lake classes have similar DNs and use the same training samples, VAs employ the following structural rules in order to distinguish each of those classes:

i. Water bodies are born as pond objects.

ii. If the area of a pond is more than threshold T_a and the elongation index is more than threshold T_e, it is considered a river object. The elongation index refers to the ratio of the major axis of the bounding polygon to its minor axis. Figures 6.10(a) and 6.10(b) show how the pond VA is converted into river VA according to this rule (see red arrows in Figures 6.10(a) and 6.10(b)).

iii. A pond becomes a lake, if its area is more than threshold T_a and its elongation index is less than threshold T_e. The black arrows in 6.10(b) and 6.10(c) illustrate the transition from pond VA to lake VA.



(a) time=800

(b) time=803

(c) time=824

Figure 6.10. The red arrows in (a) and (b) display the transition from pond VA to river VA. The black arrows indicate the transition from pond VA to lake VA.

In the event that all VAs are passive, MakerAgent retrains the SVM classifier based on the VA-generated samples in the developing step. MakerAgent employs One-Nearest-Neighbour (NN) algorithm to identify the eligible VA. The algorithm randomly selects 40 pixels for each class from the eligible VAs. VAs are then activated by the new SVM model (Figure 6.11(b)). In the production step, VAs can capture pixel $\boldsymbol{x_c}$ if it is an adjacent pixel. In the case that pixel $\boldsymbol{x_c}$ belongs to another VA, VAs employ interaction rules (see Section 6.3.1.4) to negotiate with each other (Figure 6.11(c)).

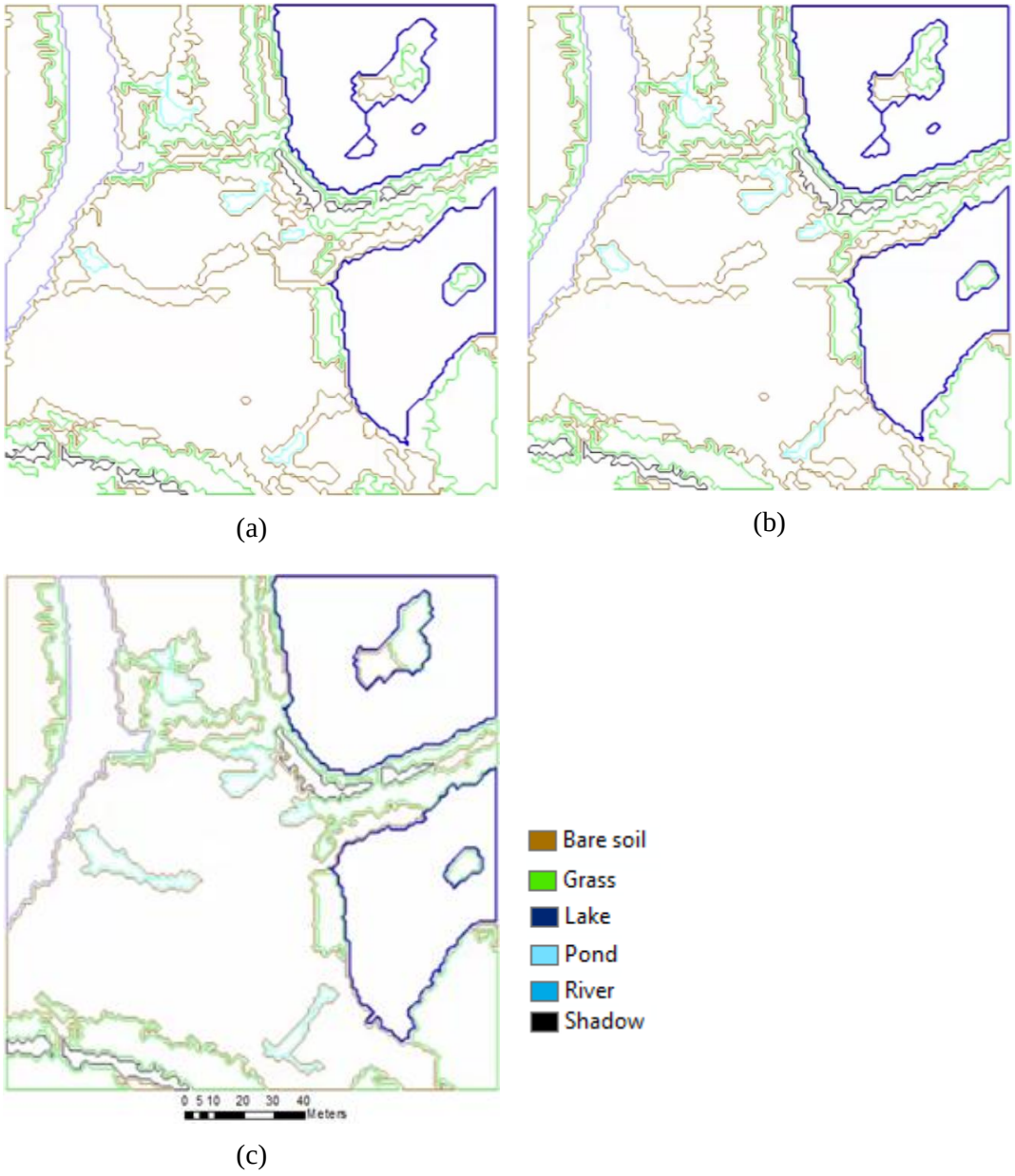


Figure 6.11. Simulation results of each step: (a) growing step, (b) developing step, (c) production step.

6.4.2. Discussion

Figures 6.12(a) and 6.12(b) show the maps created by the VA model and the GEOBIA, respectively. To create the GEOBIA map, we use a sequential GEOBIA approach (see Section 2.1.2.2). This is performed using eCognition software according to spectral features (to determine the initial class of image objects), NDVI (to identify the grass objects), structural information (to classify water bodies) and neighbourhood information (to refine mislabelled classes). A visual assessment of the classified map shows that the results of the

VA model are similar to the GEOBIA approach (Figure 6.12 (b)). However, in the GEOBIA approach we manually define thresholds and rules to label image objects.

Figure 6.12(c) and (d) display the map provided using an SVM classifier, similar to the VA-based approach. In this case, the object-based method first uses initial training samples to classify different segmented images by using the SVM classifier. Then, the method uses the rules, which is applied by the VA model, to classify waterbodies into three classes, namely Lake, Pond and River. As can be observed from Figure 6.12(c) and (d), the object-based method misses some objects belonging to the pond class. However, the object-based and the VA-based methods use the same rules to classify waterbody objects. Because image objects corresponding to the pond class lack the necessary characteristics of the pond class, some pond objects in Figure 6.12(c) are classified as shadow objects.

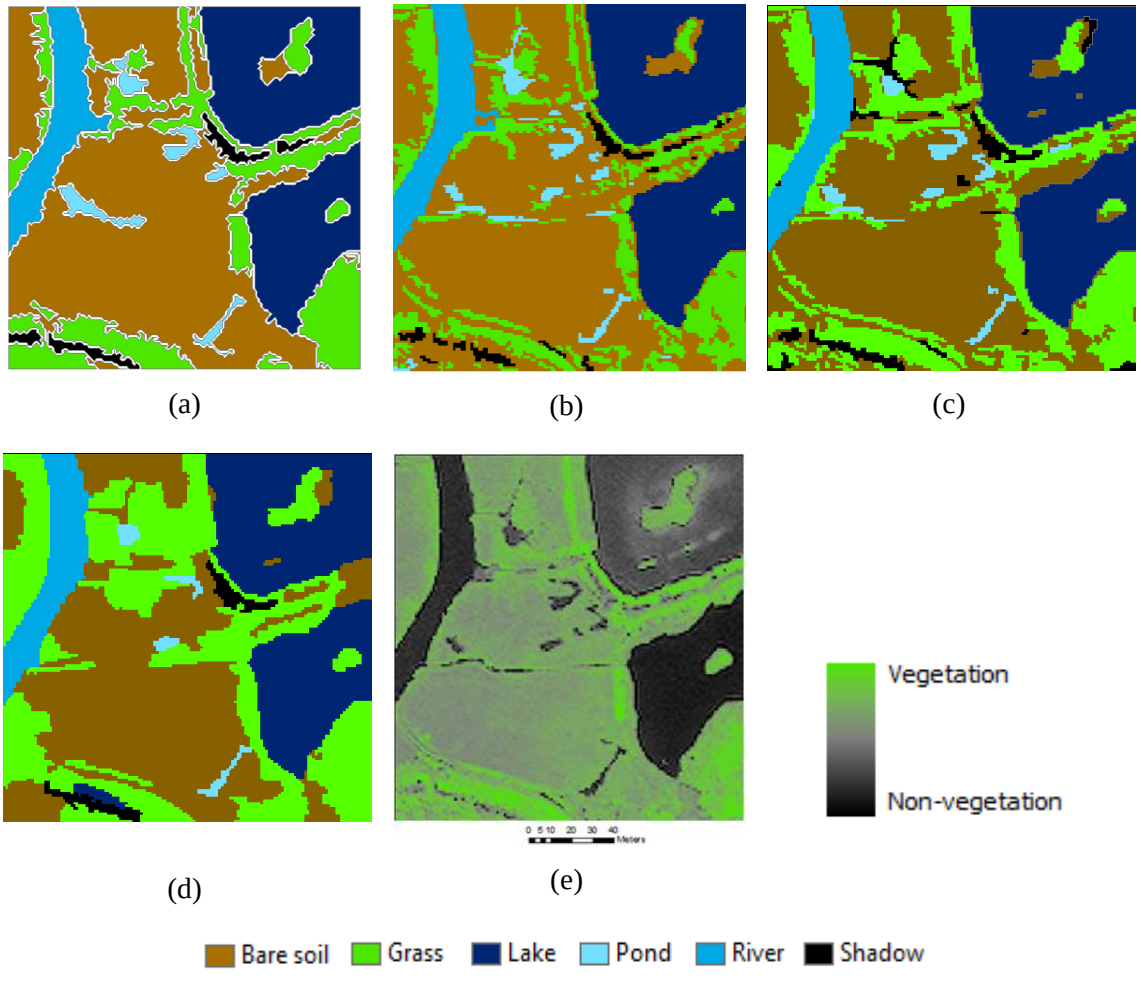


Figure 6.12. (a) the results of the VA model (b) the extracted geo-objects based on the GEOBIA approach; the method uses the image which is segmented based on scale=15, shape=0.5, compactness=0.5, layer weights= 1 except band7 weight=2, (c) the GEOBIA method uses the SVM classifier to classify the segmented image parameterized in (b), (d) the GEOBIA method applies the SVM classifier to classify the segmented image specified based on scale=15, shape=0.9, compactness=0.5, layer weights= 1 except band7 weight=2 and (e) NDVI map.

Figure 6.12(e) shows the image created by the NDVI. A comparison between Figures 6.12(a) and 6.12(e) indicates that the VA model separates vegetation from non-vegetation areas without using NDVI or setting any thresholds. Figure 6.12(e) shows that the GEOBIA approach cannot detect some grass objects (e.g. the south west area in Figure 6.12(b)). For water bodies, Figure 6.12(a) shows how the VA model can accurately classify them based on different criteria. However, the VA model fails to identify few pond objects. This is due to the small area of some of those pond objects.

Table 6.4 shows TP, FP and FN values calculated according to the GT map in Figure 6.9(b).

Table 6.4. Comparison between the results of VA-based and object-based approach in Figure 6.12(b). TP, FP and FN values are divided by the number of pixels within each ground truth class.

Classification method		**Object class (%)**					
Criteria		Bare soil	Grass	Lake	Pond	River	Shadow
GEOBIA	TP	**99.92**	74.43	**100.00**	**97.40**	**100.00**	76.42
	FP	4.38	**1.62**	**0.00**	**0.00**	**0.00**	**0.00**
	FN	0.08	25.57	0.00	2.60	0.00	23.58
Vector Agent	TP	99.68	**91.58**	**100.00**	90.91	**100.00**	**84.28**
	FP	**1.30**	7.11	**0.00**	10.39	**0.00**	**0.00**
	FN	0.32	8.42	0.00	9.09	0.00	15.72

The results of the VA model in Figure 6.13 reveal that the correctness, completeness and quality indices of bare soil, lake and river classes are similar for both the VA model and the GEOBIA method. Figure 6.13 indicates there is a considerable improvement of results for the pond class when using the GEOBIA approach (more than 15% for the quality index). As can be seen from Figure 6.12(b) and Figure 6.9(b), some soil pixels are wrongly classified as pond pixels. For grass objects, the VA model shows an improvement of more than a 10% in the quality index. Figure 6.13(d) indicates the GEOBIA approach has an average value correctness index that is higher than the VA model. This is due to the effect of the pond objects. However, the completeness and quality indices improved by more than 3% and 1%, respectively, compared to the GEOBIA method. This shows that the VA model is successful in achieving image classification, whereas in the VA model, user-defined parameters do not need to be set (e.g. scale, thresholds) to segment the image or classify the segmented objects. The VA model took 780 seconds to classify the entire image.

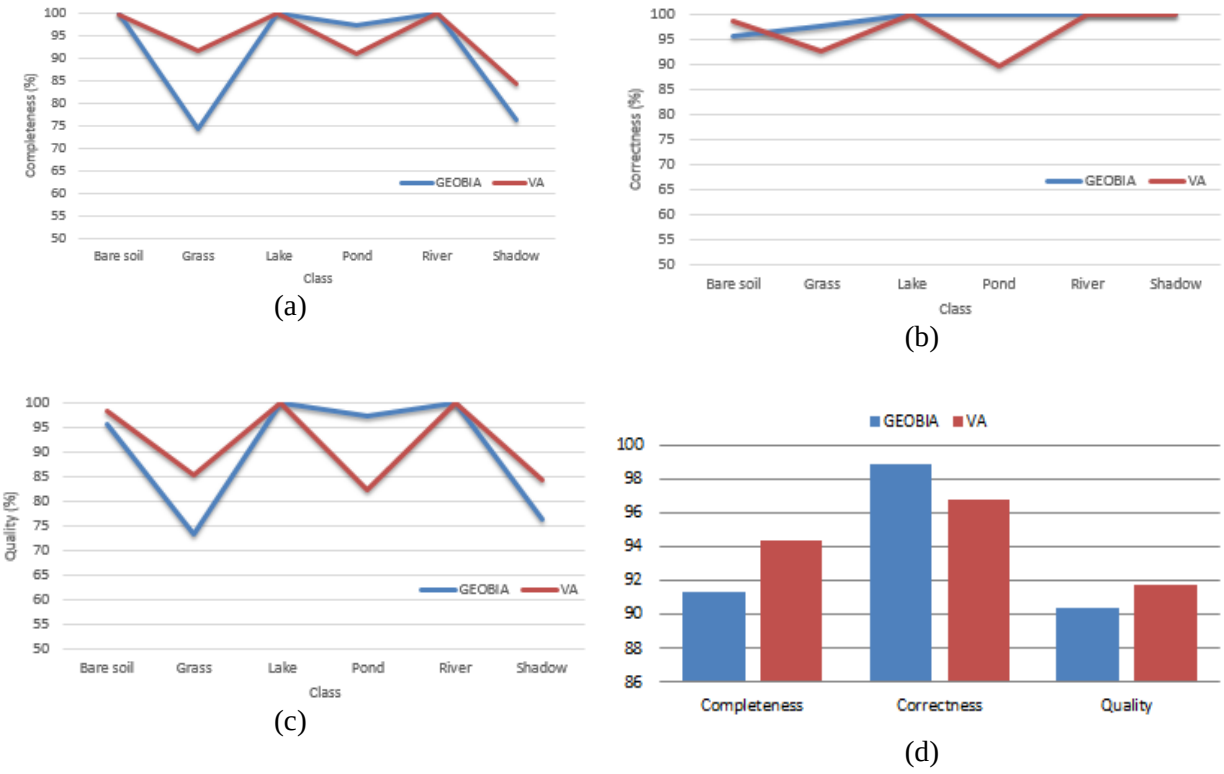


Figure 6.13. Comparison between the VA-based approach and the GEOBIA method based on completeness (a), correctness (b) and quality (c) indices. (d) The average rate of these indices in terms of VA and object-based approaches.

6.5. Conclusions

In GEOBIA methods, image segmentation, thematic meaning and spatial knowledge of real world objects are the fundamental elements used to analyse a satellite image (Hay and Castell, 2008). Accordingly, GEOBIA outputs are strongly dependent on the results of a segmentation process. However, such image segmentation is highly subjective in terms of scale, which is defined based on trial and error (Hay and Castell, 2008; Benz et al., 2004).

To address the above limitations, GEOBIA approaches usually utilise an iterative procedure of segmentation and classification to classify an image (Baatz et al., 2008; Blaschke et al., 2014; Hofmann et al., 2015). This workflow allows GEOBIA methods to change and update geometry and class of geo-objects during the classification process. This means that, the adaption of each object's geometry is triggered externally based on an image analysis, such as classification.

In this chapter, we investigated the capabilities of VAs as a new processing unit to classify an image. In this regard, a given image is classified through its elements, namely geo-

objects, in the real-world. In contrast to the above methods, here, a geo-object autonomously evaluates its current situation and decides to change its geometry.

To reach this aim, we first defined the main components of VAs, including geometry, state neighbourhood and their rules. To describe the structure of the proposed method, we employed a UML diagram. We tested the VA model by modelling a set of objects on a subset of WorldView-3 images, IKONOS images and LiDAR DSM datasets. The experimental results demonstrate the desirable performance of the VA model to extract real-world objects from raster datasets. For example, the extracted 3D roofs exhibit a high degree of similarity with corresponding 3D roofs in the real world when the VA model only uses the LiDAR DSM dataset. In the example implemented on the WorldView-3 image, the results show that the VA model can accurately classify objects which have similar DNs (e.g. rivers and lakes) based on different criteria, such as area. The obtained results also indicate that the VA model can identify geo-objects from remote sensing satellite images and LiDAR DSM datasets without setting predefined segmentation parameters and user-defined classification thresholds, as compared to conventional GEOBIA approaches. Moreover, results of the classification step demonstrate that the accuracy of the VA map is better than the map provided by the GEOBIA method.

As there are several stochastic processes within the VA model, such as initialisation, results may be slightly different in each run especially where the VAs only use the image datasets. Further research needs to formalise the initialisation process. In our example, the building VAs only use the elevated information to find their own boundaries, therefore, there is no interaction between the elevated VAs (e.g. buildings and trees). The use of spectral information along with elevated information allows the building VAs to interact with the tree VAs, which can improve the accuracy of the extracted boundaries between the elevated VAs. It would also be interesting to study the concept of combining the spatial semantics and thematic semantics of objects in order to enhance the aspect of the neighbourhood rules.

So far, we demonstrated the capabilities of the VA model for the pixel-based and object-based image classification methods. In the next chapter, we will discuss the application of the VA model in identifying, extracting and classifying geo-objects (e.g. 3D roofs) from raster datasets (e.g. LiDAR datasets) without using the classification process.

Chapter Seven

Vector agent models for extraction of 3D roofs

Abstract

This chapter addresses the application of VAs in extracting real-world objects from raster datasets. In this context, a variety of geo-objects in a real-world environment (e.g. lakes, roads, 3D roofs) can be addressed via the use of VAs. This chapter shows the capability of the VAs in extracting geo-objects from raster datasets without the use of training samples. Here we use VAs to identify, extract and classify 3D roofs from LiDAR DSM datasets. To achieve this, we define the transition rules based on the characteristics of 3D roofs in the real world. In contrast to conventional roof extraction methods, VAs detect, extract and classify 3D roofs based on different elevations without the need of static geometric primitives (e.g. edges), predefined shapes, or user-defined parameters (e.g. scale).

7.1. Introduction

Recent advancements in Light Detection and Ranging (LiDAR) systems (more generally known as Airborne Laser Scanners (ALS)) have provided new opportunities for improving the accuracy and the level of automation in spatial modelling, such as automatic roof extraction. To date, a wide range of algorithms for automatically extracting buildings from LiDAR datasets have been proposed. These algorithms are usually categorised into two main classes: model driven and data-driven (Maas and Vosselman, 1999).

Model-driven approaches employ a predefined catalogue of 3D roof forms to reconstruct 3D roofs. In this context, primitive building shapes (e.g. edges) are initially identified from the input datasets. The most appropriate model for each roof is then selected from the predefined roof library according to the properties of the primitive shapes. In the modelling step, the 3D roofs are reconstructed by updating the parameters (e.g. scale or rotation) of the selected models. For example, Schwalbe et al. (2005) used vertical profiles as the primitive building features to reconstruct 3D roofs. To model more complex building shapes, Lafarge and Mallet (2010) proposed an advanced method based on regular and irregular components

of roofs. Here, a combination of geometric primitives (e.g. planes) and mesh-patches were used to address the primitive features. Hung et al. (2013) developed a generative statistical method via a segmentation process to reconstruct 3D roofs.

Despite the advantages that model-driven approaches offer, such as consistent geometry, their results strongly depend on the predefined catalogue of roof forms (Song et al., 2015; Kim and Shan 2011; Tarsha-Kurdi et al., 2007). In other words, the model-based methods cannot address a 3D roof if it is especially complex or not included in the predefined roof library (Song et al., 2015; Kim and Shan 2011; Tarsha-Kurdi et al., 2007). In contrast, data-driven approaches can address any building shape (Song et al., 2015; Kim and Shan 2011; Tarsha-Kurdi et al., 2007).

Data-driven algorithms use planar roof primitives to model 3D roofs. Planar roof primitives are groups of points or pixels that have homogenous properties. They are created through a segmentation process in feature space or image space (e.g. raster LiDAR datasets) (Lari and Habib, 2014). For example, Ma (2005) used the texture information (e.g. variance) of LiDAR points to extract planar roof surfaces. Filin and Pfeifer (2006) developed a cluster-based segmentation based on slope to identify roof primitives from laser point datasets. To create planar roof segments, Awrangjeb et al. (2014) proposed a novel approach based on pure non-ground points (not including wall points) via a clustering algorithm. Sampath and Shun (2007) applied a region-growing algorithm formulated on a predefined local window to segment LiDAR datasets. Sun and Salvaggio (2013) used a region growing segmentation to identify the roof primitives. After the segmentation process, 3D roofs are reconstructed by assimilating the planar roof primitives based on their topology in the real world.

Although most data-driven approaches show promising results, they still rely on a two-stage linear segmentation and reconstruction process. In this context, these methods lack the ability to simultaneously identify, extract and classify 3D roofs based on different criteria (e.g. elevation or slope). Furthermore, the formulation of topological relations among primitive roofs during the reconstruction process is still a challenging task (Kwak and Habib, 2013; Kim and Shan, 2011; Tarsha-Kurdi et al., 2007; Brenner, 2005). To address these issues, we propose a novel data-driven approach based on the VA model to directly extract roofs from LiDAR DSM datasets and classify them based on elevation attributes.

Agent modelling is an artificial intelligence technique whereby elementary objects with a dynamic nature evolve in parallel to achieve pre-defined goals. Each agent is viewed as an

autonomous processing unit that can communicate, cooperate and negotiate with other agents and its environment to achieve objectives through a Multi Agent System (MAS) (Wooldridge, 2009). Samadzadegan et al. (2010) proposed MAS to extract 3D roofs from LiDAR datasets. Here, textural and spatial information was applied by two different groups of agents, including tree and building types, to recognise and extract buildings from LiDAR datasets. The ability of agents to perceive the object geometry as just another attribute to be optimised was also tested by Hofmann et al. (2015). In their work, agents had the capability of adjusting the boundaries of roof objects and classify roofs based on different criteria (e.g. slope). Here, the initial geometry of agents was defined based on the segmentation process. During the subsequent adaptation process, class ontology was used to change agent geometry.

This chapter shows that the flexibility of the VA approach on image analysis can also be used to identify, extract and classify 3D roofs from LiDAR DSM, even when there are no training samples. ***The VA model allows 3D roof extraction methods to directly identify roofs from raster datasets without using a secondary process, such as segmentation or edge detection.*** The main elements of the VA model are described in Section7.2. The processes of the proposed method are presented in Section 7.3. The experimental results are discussed in Section 7.4. A short summary and ideas for further work are discussed in Section 7.5.

7.2. Vector Agents

From a VA perspective, each object is considered to be an abstraction of a real-world phenomenon (e.g. 3D roofs) (Hammam et al., 2007).

7.2.1. Geometry and geometry rules

Through the geometry component, **L**, a 3D roof VA stores its vertices that define the boundary ∂X_{VA} of the VA. X_{VA} is the coordinate of a connected subset of the raster datasets formed by the pixels on LiDAR DSM belonging to the VA. The geometric methods, $\mathbf{M_L}$, allow the VAs to address any shape of 3D roof and reconstruct the topological relations between 3D planar roofs with minimum human intervention.

7.2.2. State and transition rules

The state **S** of a VA is defined based on the elevation of VAs on the DSM. Being dynamic in nature, each VA uses the transition rules $\mathbf{T_S}$ to update its attributes at each iteration through the following rules:

$$\sigma_h(\boldsymbol{x_c}) \leq T_H \ , \tag{7.1}$$

$$DN_{DSM}\ (\boldsymbol{x_c}) \ \geq DN_{DSM}\ (\boldsymbol{x_b}), \tag{7.2}$$

where, $\sigma_h(\boldsymbol{x_c})$ is the standard deviation of the elevations of a candidate pixel $\boldsymbol{x_c}$ and its neighbours. A human interpreter defines threshold T_H. $DN_{DSM}\ (\boldsymbol{x_c})$ and $DN_{DSM}\ (\boldsymbol{x_b})$ are convoluted output pixel values of pixel $\boldsymbol{x_c}$ and pixel $\boldsymbol{x_b}$ on DSM computed via Equation 7.3 and 7.4. Pixel $\boldsymbol{x_b}$ is the adjacent pixel of pixel $\boldsymbol{x_c}$ that the VA uses to create a new vertex centred on pixel $\boldsymbol{x_c}$.

$$DN_{out} = \frac{1}{W}\sum_{i=1}^{d}\sum_{j=1}^{d}\omega_{ij}\,DN(i,j)\ , \tag{7.3}$$

$$W = \sum_{i=1}^{d}\sum_{j=1}^{d}\omega_{ij}, \tag{7.4}$$

where d is the kernel size set to an odd number ranging from 3 to 9. Here, d is set to 3. $DN(i,j)$ is the pixel value in the input DSM at location *(i, j)* and DN_{out} is the convoluted output pixel value. ω_{ij} and W are the value of the element at location *(i, j)* in the kernel and sum of all kernel elements, respectively. In the case that there is homogenous elevation information (e.g. within a roof boundary), VAs use Equation 7.1 to evolve geometrically. As the standard deviation of pixels on the boundary of 3D roofs is high, building VAs use transition rules, namely Equation 7.2, formulated on the Laplacian edge detection kernel to delineate their initial boundaries.

In the case that pixel $\boldsymbol{x_c}$ lies on the common boundaries between two VAs, such as $VA_{1,t}$ and $VA_{2,t}$ with their classes k_1 and k_2, respectively, $VA_{1,t}$ can capture $\boldsymbol{x_c}$ during a growing/ shrinking method if it can satisfy the following rules:

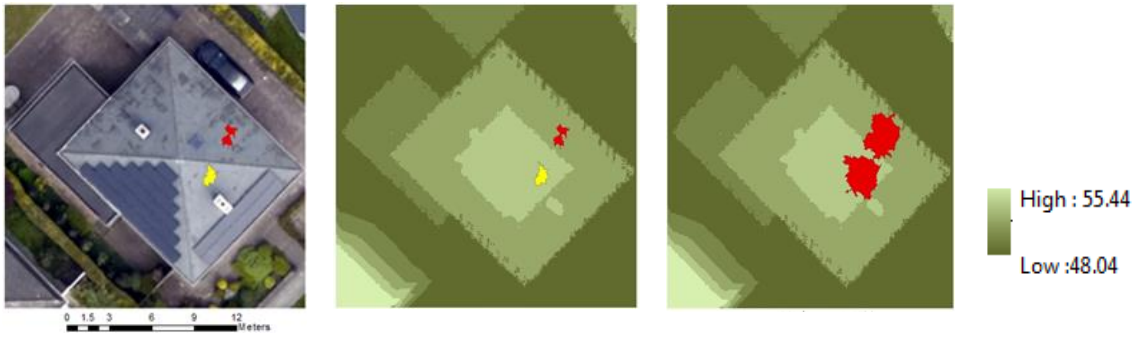
$$|\Delta H_{VA_{1,t}}(\boldsymbol{x_c})| \ \leq |\Delta H_{VA_{2,t}}(\boldsymbol{x_c})|, \tag{7.5}$$

$$|\overline{H}_{VA_{1,t}} - \overline{H}_{VA_{2,t}}| > T_H, \tag{7.6}$$

where $\Delta H_{VA_{1,t}}(\boldsymbol{x_c})$ and $\Delta H_{VA_{2,t}}(\boldsymbol{x_c})$ are computed on the elevation of $\boldsymbol{x_c}$ and a local average elevation of $VA_{1,t}$ and $VA_{2,t}$, namely $\overline{H}_{VA_{1,t}}$ and $\overline{H}_{VA_{2,t}}$. VAs use a local 3×3 window centred on pixel $\boldsymbol{x_c}$ to calculate $\overline{H}_{VA_{1,t}}$ and $\overline{H}_{VA_{2,t}}$.

7.2.3. Neighbourhood and neighbourhood rules

Neighbourhood component **N** is a collection of objects that lie in the adjacent distance of a VA. To interact with other objects in the simulation domain, VAs use neighbourhood rules $\mathbf{R_N}$ specified on an adjacent distance. For example, a basic adjacency rule triggers the merging/killing geometry method when two adjacent VAs have the same class (Figure 7.1).



(a) (b) (c)

Figure 7.1. Subset of an RGB image and LiDAR DSM. Evolution of two VAs classified as roof (a) (b), until they become neighbours and merge to form a single roof VA (c), resulting in the 'killing' of the yellow VA.

7.3. Process overview and scheduling

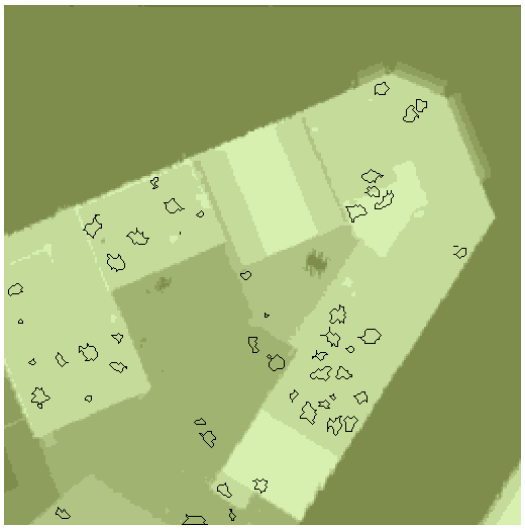
The model simulates interactions among VAs and between VAs and their environment. The proposed model has five main steps: initialisation, growing, developing, construction and production. To initialise in the simulation space, VAs use the following rules:

i. *Rule 1:* The elevation of pixel $\boldsymbol{x_c}$ and its neighbouring pixels should be more than T_L. Threshold T_L is determined on the lowest elevation on the DSM plus the standard height of buildings (e.g. 2.80 m) in an urban area.

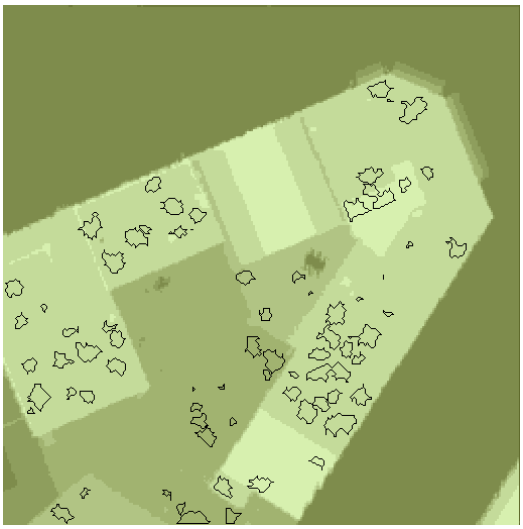
ii. *Rule 2:* The standard deviation of candidate pixel $\boldsymbol{x_c}$ and its immediate neighbours should be less than $\sigma_h \leq T_U$. Threshold T_U is defined based on the accuracy of the LiDAR DSM dataset.

VAs use *Rule 1* to identify the initial location of the buildings on the LiDAR DSM datasets. *Rule 2* allows the VAs to lie only on the homogenous areas of 3D roofs (e.g. within the boundary of roofs). In the growing step, VAs automatically change their geometry through

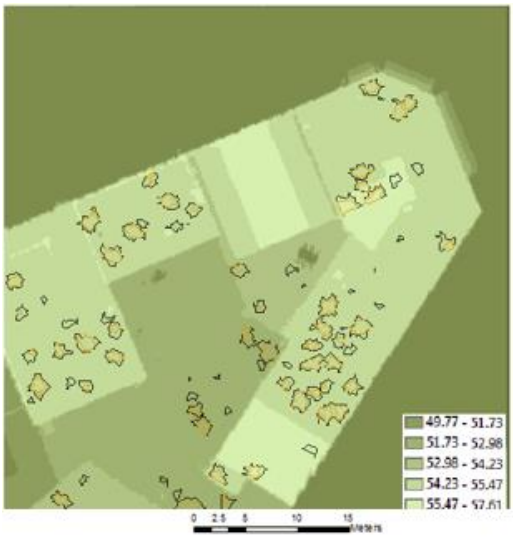
the growing and merging/killing methods and transition rule, namely Equation 7.1(e.g. Figure 7.2). To do this, VAs employ a mechanism of adaptation and update (see Section 3.3.6). The process is continuously repeated until all VAs become passive. A VA is passive if it cannot change its geometry based on the current transition rules. In this neutral state, it is possible for a VA to be active again with different transition rules. Once transition rules are updated, all VAs within the image space simultaneously to become active. In this event, VAs use the new transition rules (e.g. Equation 7.2) to change their geometry until all VAs become passive again.



(a) time=100



(b) time =200



(c) time=200

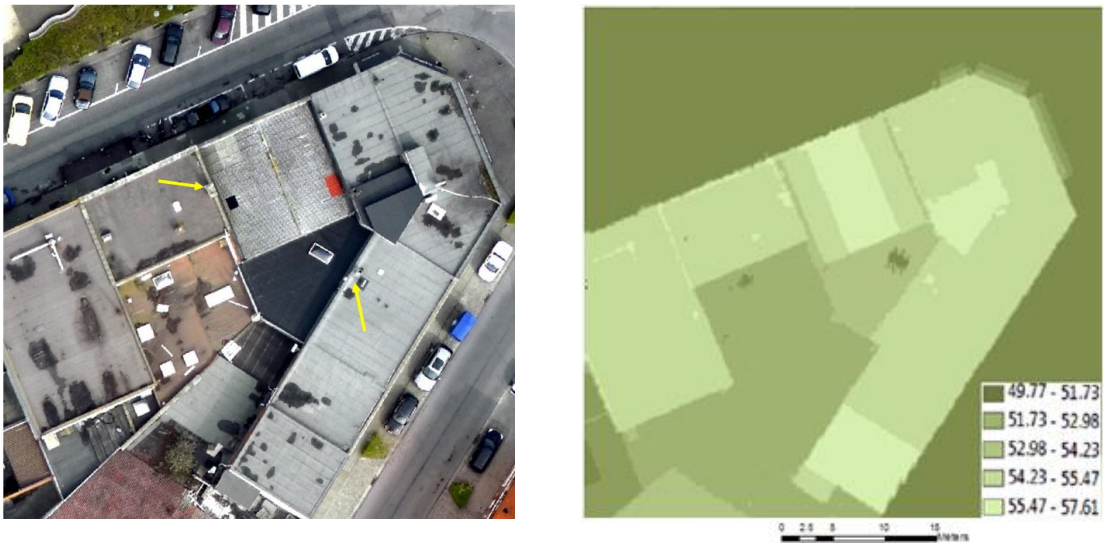
Figure 7.2. (a) and (b): Simulation results over 200 time steps in the agent modelling shell Repast Simphony to identify 3D roofs from a subset of LiDAR DSM datasets. (c) 3D roof VAs at time=100 (brown polygons) projected on 3D roof VAs at time=200 (black polygons).

In the developing step, VAs only employ the growing method. If all the VAs are passive in the construction step, a VA can evolve if the elevation of an adjacent pixel calculated on a low pass filter is more than T_L. The use of a low pass filter allows the VA to reduce the effect of noise pixels that lie on the edges. In the construction step, VAs only use the growing method to evolve geometrically. In the production step, the boundaries of the 3D roofs are tuned through the interaction between VAs. In this step, VAs apply growing/shrinking methods to determine their boundaries via Equations 7.5 and 7.6 (see Appendix C).

7.4. Experimental results

7.4.1. Dataset 1

In the first experiment (Figure 7.3(b)), the VA model approach was tested on a subset (251×251 pixels) with 20cm resolution LiDAR DSM which covers an urban area in Zeebrugge, Belgium (GRSS, 2015). The test area contains various buildings of complex shapes (Figure 7.3(a)). The LiDAR DSM shows jagged edges especially where there are features such as chimneys and windows.



(a) (b)

Figure 7.3. (a) A subset RGB colour image from an urban area in Zeebrugge, Belgium. (b) A subset of LiDAR DSM datasets.

The VAs use a Java implementation of Repast (see Section 2.10.2.2) along with a generic Vector Agent library developed by Moore (2011) to extract and classify 3D roofs from LiDAR DSM datasets. To assess the performance of the proposed approach, the extracted 3D roofs provided by the VA-based approach were compared with corresponding buildings

created manually by a human expert. The quality of the extracted boundaries of 3D roofs by the VA model is compared with the results of a segmentation process (a region growing algorithm using eCognition) and a Laplacian edge detection kernel. To classify the 3D roofs based on their elevation, T_H is set to 0.20 m. In our case, T_L is 52.57m (see *Rule 1* in Section 7.3).

Figure 7.4(a) shows the boundaries of the 3D roofs provided by the Laplacian edge detection kernel. As can be seen from Figure 7.4(a), the Laplacian edge detector can easily detect and extract the edges that appear on the block boundary. However, the Laplacian kernel misses some building edges, while many other detected edges are not actually buildings. Moreover, the edge map shows that the Laplacian kernel lacks the necessary ability to deal with the noise caused by objects, such as windows and chimneys (shown by yellow arrows in Figure 7.4(a)). Figure 7.4(b) displays the result of a segmentation process. To get the most realistic image objects, segmented objects with small sizes are applied (Figure 7.4(b)). As can be seen from Figure 7.4(b), the segmented objects can readily determine the boundary of the block. However, Figure 7.4(b) shows that the segmented objects cannot accurately address the boundaries between 3D roofs (see yellow arrows in Figure 7.4(b)).

Figure 7.4(c) shows the extracted 3D roofs based on the VA-based approach. Figure 7.4(c) shows that the edges that appear on the block boundary overlap with their corresponding segmented objects and the Laplacian edge detection kernel. A visual assessment of the extracted roofs by the VA-based approach shows that the VA roofs are very close to the real structure of the buildings, while the VA model does not impose any constraints on building shape. Experimental results also indicate that the dynamic structure of VAs allows them to deal with noise caused by features such as windows or chimneys, except when these objects lie on the boundaries between two roofs (see yellow arrows in Figure 7.4(c)). In these cases, the boundaries between VAs may not be delineated accurately due to the noise points on the walls and features (see yellow arrows in Figure 7.3(a)).

Figure 7.4(d) displays the roof VAs projected on a classification map provided based on threshold T_L (e.g. 52.57m). The blue arrows show the areas ignored by the VAs; they have been ignored because these regions are heterogeneous and small (Figure 7.4(d)). Thus, VAs cannot satisfy the initialisation rules (see *Rule 2* in Section 7.3) or growing rule (see Equation 7.1) in these areas to extract 3D roofs.

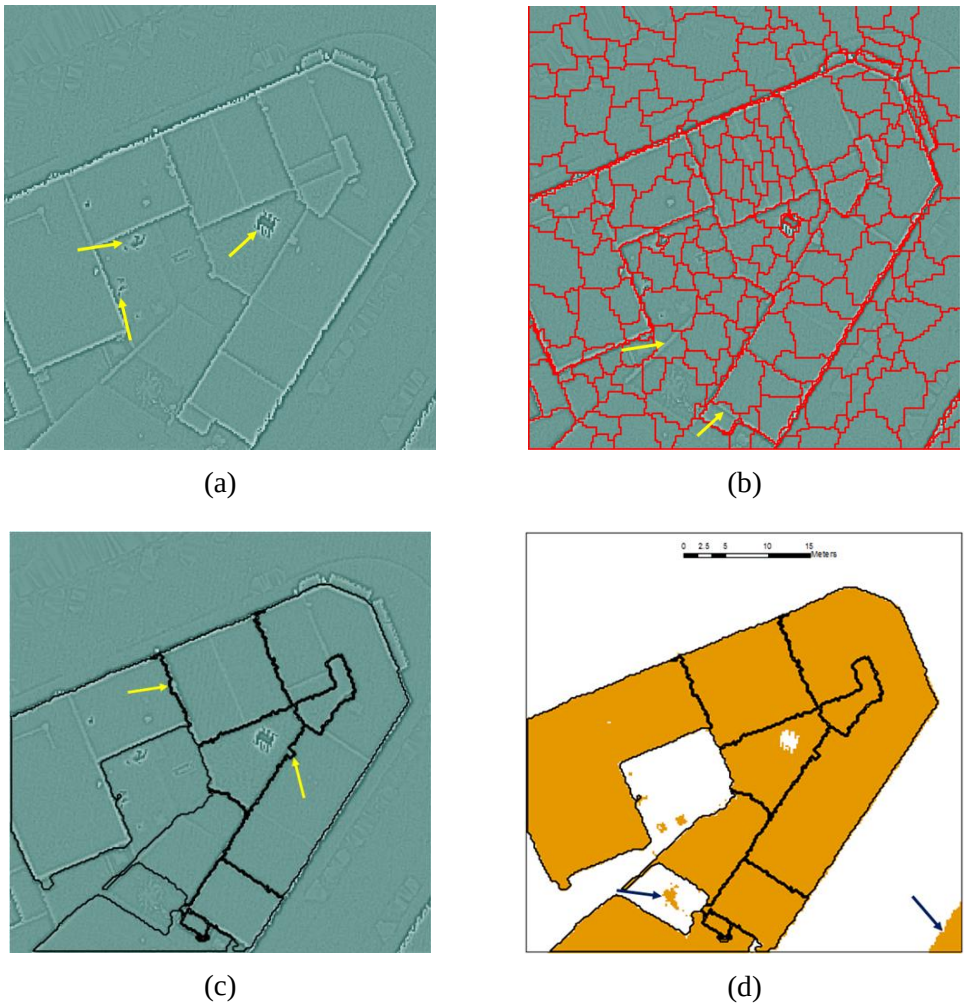


Figure 7.4. (a) An image based on the Laplacian edge detection kernel. (b) The segmented image provided by eCognition software, based on scale=8, shape=0.1 and compactness=0.5. (c) 3D roof VA. (d) a classified image based on T_L =52.57. The brown areas show the pixels of the image in Figure 7.3(b) which have an elevation of more than 52.57m. The black polygons are the 3D roof VAs superimposed on the classified image.

The VA roofs were evaluated by comparing them with reference building models processed manually by a human interpreter. To make a quantitative assessment of extracted roofs, the completeness or shape accuracy of the detected buildings (Table 7.1) is measured by the following metric:

$$\text{Shape}_{accuracy} = 1 - \frac{|A-B|}{A}, \tag{7.7}$$

where A is the area of the roof which is extracted based on human interpretation, and B is the area extracted by the VAs.

Table 7.1. Performance evaluation of the 3D roof VAs based on the shape accuracy index.

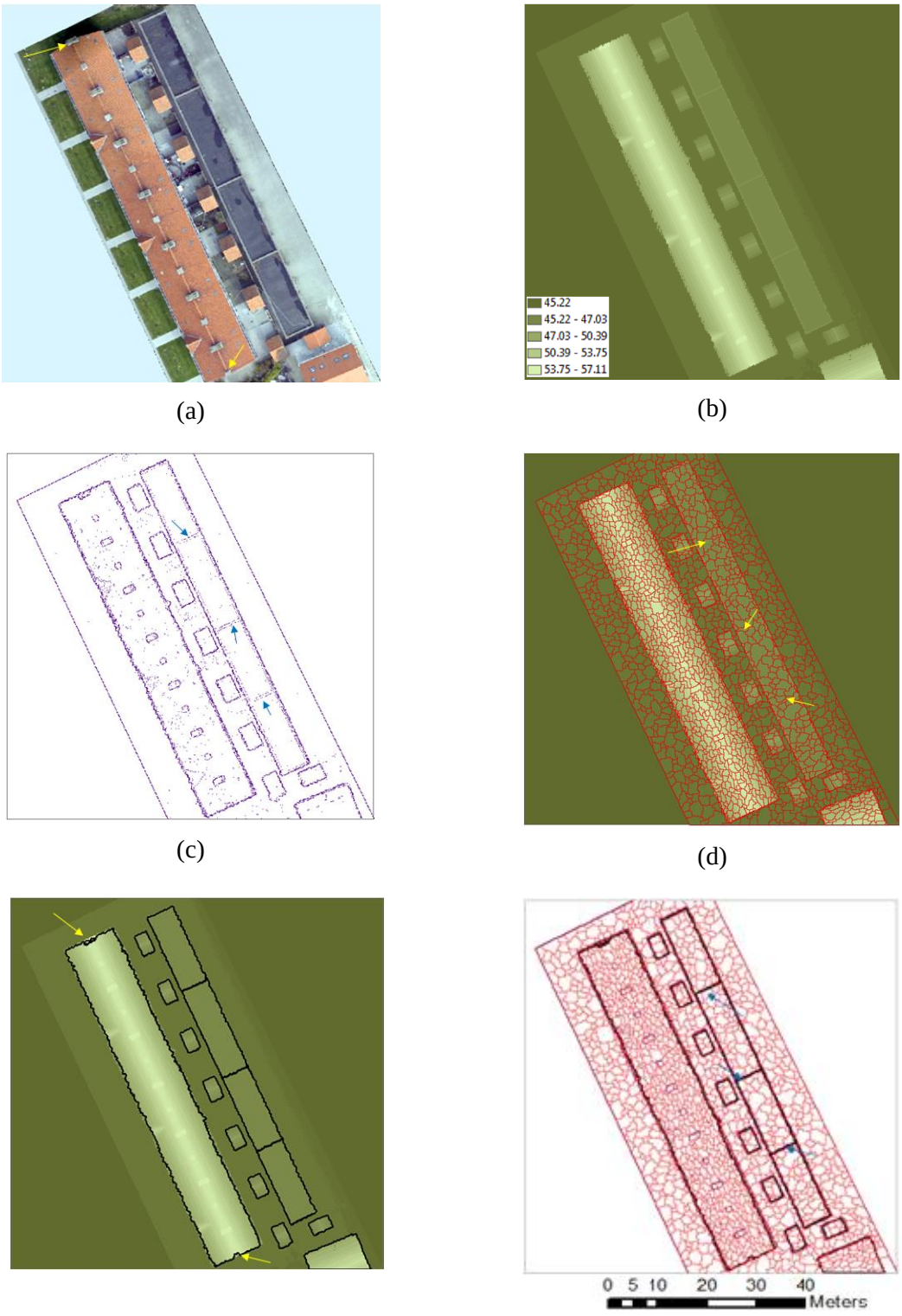
Statistics	VA-based	Segmentation-based
Maximum (%)	99	99
Minimum (%)	81	75
Mean (%)	92	91
Standard deviation (%)	9.3	9.0
Number of buildings	9	9

The experiments show that the proposed approach can reach an average shape accuracy of 92%. The standard deviation value 9.3% and mean value 92% also indicate that the most 3D roof VAs are close to the true 3D roofs manually extracted by a human operator. Although the results of the VA model and the segmentation-based approach are similar, the boundaries of the 3D roofs are manually extracted by merging image objects in the segmentation-based approach.

7.4.2. Dataset 2

In the second experiment, we used a LiDAR DSM dataset with a size of 751×751 pixel (10cm pixel size, Figure 7.5(b)) from an urban area in Zeebrugge, Belgium to test the VA model (GRSS, 2015). Here, T_L is 48.02m (see *Rule 1* in Section 7.3). As can be seen from Figure 7.5(a), the test area contains various buildings of complex shapes. Similar to the previous example, the LiDAR DSM shows the jagged edges in which there are features such as chimneys and windows (Figure 7.5(b)).

In this experiment, VAs use different values of T_H to classify 3D roofs based on different elevations. VAs first model 3D roofs with a threshold T_H of 0.05m. After the construction step, new VAs are generated with a threshold T_H of 0.20m. The process continues until all 3D roofs are identified from raster datasets (Figure 7.5(e)). Figures 7.5(c) and 7.5(d) display the maps produced by the Laplacian edge detection and region-growing segmentation process, respectively. The results in Figure 7.5(c) show that the Laplacian kernel correctly identifies the borders of 3D roofs from LiDAR DSM. However, the kernel misses some other edges (see blue arrows in Figure 7.5(c)). Moreover, the Laplacian edge operator, like other conventional edge detection operators, is sensitive to noise.



(a) (b) (c) (d) (e) (f)

Figure 7.5. (a) A subset RGB colour image from an urban area in Zeebrugge, Belgium. (b) A subset of LiDAR DSM datasets. (c) Segmented image provided by eCognition software, based on scale=5, shape=0.15 and compactness=0.5. (d) Image based on the Laplacian edge detection kernel. (e) 3D roof VA. (f) VA map projected on the segmented image.

In Figure 7.5(c), some edges are falsely identified, especially in areas with features such as chimneys or windows. In contrast, Figure 7.5(e) shows that the VA model can readily reduce the noise originating from these features. For situations where features (e.g. chimneys) are nearby the block boundaries, 3D roofs may not be determined correctly (e.g. see yellow arrows in Figure 7.5(e)). Figure 7.5(d) shows the high capability of the segmentation-based approach to identify the boundaries of blocks and individual buildings. However, the results in Figures 7.5(d) and 7.5(f) indicate that image objects lack the ability to address the boundaries between 3D roofs (see the blue and yellow arrows).

As discussed in Chapter 2 (see Section 2.3.1), image objects are formulated based on a predefined set of parameters, such as colour and scale. When there is poor contrast between the borders of the 3D roofs (Figures 7.5(b), 7.5 (d)), the algorithm cannot accurately identify the boundaries between 3D roofs from LiDAR DSM. In contrast, Figure 7.5(e) shows the high potential of the VA model to extract 3D buildings from raster datasets based on different elevations.

Similar to the first experiment, we use the completeness index to make a quantitative assessment of the extracted roofs (Table 7.2).

Table 7.2. Performance evaluation of the 3D roof VAs based on the shape accuracy index.

Statistics	VA-based	Segmentation-based
Maximum (%)	98	99
Minimum (%)	84	84
Mean (%)	91	91
Standard deviation (%)	3.5	5.8
Number of buildings	14	14

The results in Table 7.2 (e.g. average shape accuracy of 91%) indicate the enhanced capabilities of the VA to model 3D roofs, showing that the VA-based approach needs minimum human intervention to extract and classify roofs from raster datasets.

7.5. Conclusions

To identify 3D roofs from LiDAR datasets, automatic building extraction methods usually use a two-stage process of image segmentation and reconstruction. In the first step a segmentation process is used to create planar roof primitives, which are then merged to

reconstruct 3D roofs in the second step. In this case, planar roof primitives remain unchanged once they are created. To address this issue, building extraction methods utilise algorithms that allow buildings to change their geometry during the construction step, such as proposed by Hofmann et al. (2015). However, these algorithms still use a segmentation process to build primitive processing units, namely planar roofs.

In this chapter, we presented a dynamic and automated processing unit, namely VAs, to extract different shapes and structures of 3D roofs from a LiDAR dataset. In contrast to conventional methods based on a specified set of parameters (e.g. scale) or predefined shapes, the VA-based approach identifies, extracts and classifies 3D roofs with minimum human intervention.

To reach this aim, we defined the elements of VAs, including geometry, geometry rules and methods, state, transition rules, neighbourhood and neighbourhood rules. These components were formulated on the characteristics of 3D roofs and input data including LiDAR DSM. The geometry of VAs was defined based on a directional planar graph. To create an irregular dynamic geometry, VAs applied a set of geometric operators such as growing and merging/killing. State and transition rules were defined and formulated in terms of a region growing algorithm based on a standard deviation value, Laplacian edge detection function and a low pass filter kernel. Neighbourhood and neighbourhood rules were defined on the Euclidian distance concept. This structure enables a dynamic geometry for the VAs to address various buildings of complex shapes without setting user-defined parameters (e.g. scale) or using any constraint on building shape. The results indicate the high potential of the VA model to address the limitations of the conventional data-driven approaches.

In our example, VAs are only applied to extract 3D roofs and we assume that buildings are the only available elevated objects. To separate elevated objects from different classes (e.g. trees and buildings), we need to define and formulate new transition rules based on the characteristics of objects in the real world. In the future, we will explore the possibility of developing VAs that can extract and classify elevated objects from different classes. The use of a regularisation algorithm to improve the accuracy of the 3D roof VAs would also be an interesting area to study.

Chapter Eight

Conclusion and future work

Abstract

This chapter begins with a brief conclusion, then goes on to review two fundamental processing units, namely pixels and image objects, for image classification. After describing the proposed VA model, we will discuss the research and present the results by reviewing the applications of the VA model for pixel-based classification, the feature extraction process and object-based image classification. In the last section, we propose various directions for future research in model computation and application areas.

8.1. Purpose

The aim of this thesis was to develop a new automated processing unit to directly address geo-objects in image space. The proposed processing unit uses the VA model to find and extract geo-objects directly from raster datasets (e.g. high resolution satellite images, hyperspectral imagery, and LiDAR DSM).

8.2. The need for a new processing unit for image classification

In order to develop the newly proposed method, it was crucial to identify and understand current methods of modelling geo-objects in image space. Preliminary research in this area concluded that both pixel-based and object-based methods achieve this goal by using a sequential process of segmentation and classification.

In pixel-based approaches, pixels are first labelled via a clustering algorithm (e.g. ML) in the feature space. The labelled pixels are then used in a segmentation process to determine the geometry of geo-objects in the image space. When this method is used to extract geo-objects from raster datasets in a pixel-based approach, two implicit assumptions are made. Firstly, each geo-object should have a unique spectral behaviour in a real-world environment that can be modelled in feature space. Secondly, geo-objects should have a homogenous spectral behaviour in image space. If any of these assumptions are violated, it will result in

a poor spatial model. Although the use of spatial and spectral information has been proposed in several pixel-based approaches (see Chapter 4 and Chapter 5), these approaches lack the ability to directly address geo-objects in image space.

In an object-based approach, image objects are first created in image space. The thematic meaning of these image objects is only determined in the subsequent classification step. Therefore object-based methods cannot address both the geometry and theme of geo-objects simultaneously in image space. Although several studies have addressed this issue using a cycle of segmentation and classification, the geometry of geo-objects remains static for two reasons: Firstly, it is defined according to a set of predefined parameters (see Section 2.3). Secondly, objects cannot change their geometry during the classification step (see Section 1.2.2 and Section 2.3). In this context, we implicitly assume that geo-objects have predictable behaviours in image space.

While reviewing different approaches for spatial modelling, the introduction of vector agents in a geosimulation domain has recently been conceived as a new way of representing geo-objects in image space. This is the basis on which the major objectives of this research were formed (see Chapter 1). These objectives guided the work described in this thesis, the research questions, and the resulting conclusion drawn in this chapter.

8.2.1. The development of the vector agent model

The first objective of this thesis is to develop a generic structure based on s that can address real-world objects in image space. In order to construct the geometry of VAs, we applied a planar graph formulated on the winged-edge data structure. To enable a dynamic geometry for VAs, a set of individual geometric operators were defined and implemented. These operators, e.g. *Vertex displacement*, *Converging vertex displacement*, *Half-edge joining* and *Edge remove*, allowed the VAs to geometrically evolve in the simulation space. To interact with each other, VAs used a set of interaction geometric methods, including *growing*, *merging*, *shrinking* and *splitting*.

A sub-objective in this area was to develop VAs that could find and update their classes and attributes in the simulation space. This was central to developing the model in order to suit different geo-objects in image space. Depending on the application of VAs and available raster datasets, VAs used different strategies, such as the ML classifier or Laplacian edge detector, to implement transition rules.

Additionally, the generic design of VAs enabled them to interact with each other within the simulation space. This objective was fulfilled through the use of neighbourhood rules. Here, neighbourhood rules are explicitly defined on Euclidean distance, and the neighbours can be implemented over vector space without distance limitations.

The integration of these components was one of the challenges raised in this thesis. This integration was achieved through the flexible agent architecture combined with its components: agent's sensor, agent's state, and agent's effectors. The model integrates the agent and its environment within the framework of these components. Each component was implemented with different classes and sub-classes that facilitated the agent's perception of the environment. The agent evaluates its available options in the environment according to its rules and strategies in order to achieve the desired goal. The chosen option is then executed via the agent's effectors.

8.2.2. Vector agent for pixel-based approaches

The second objective of this thesis is to validate this model in comparison with per-pixel classifiers in terms of classification accuracy achievable. As mentioned earlier, several studies have already proposed the use of spatial information to improve the accuracy of pixel-based classification maps. Vector agents have the potential to allow pixel-based approaches to integrate the spatial and spectral spaces for generating reliable training objects. In the unsupervised method, VAs were used to create a set of training objects in a self-training fashion. An ML classifier, trained on the VA-generated samples, was employed to classify the remaining pixels. The method was successfully tested on a high spatial resolution satellite image.

In the supervised approach, VAs were applied to extract training objects from hyperspectral datasets. Here, a similarity metric, formulated according to a nearest-neighbour algorithm, was applied to select the most informative VA samples for learning the SVM algorithm. The simulation yielded satisfactory results from the VA model for semisupervised classification.

8.2.3. Vector agents for GEOBIA approaches

The third objective of this thesis is to assess the capabilities of the VA model in addressing the main limitations of the GEOBIA approach. This was done by utilising VAs to extract geo-objects from a high spatial resolution satellite image and LiDAR DSM dataset. In this case, the VA model used different types of transition rules to address different geo-objects in image space. For the building geo-object, the VA model used the transition rules that were

formulated on the region growing and edge detection operators. To address the meadow and shadow geo-objects in the satellite image, the VA model employed the SVM classifier. The results of the VA model demonstrate its ability to understand and simulate the complexity of a real-world environment.

8.2.4. Vector agents and real-world objects

The fourth objective of this thesis is to explore the proposed VA model and its ability to extract and identify real-world objects from raster datasets in a specific area. This has been achieved via the use of VAs in modelling 3D roofs without the need of labelled training samples. In this instance, transition rules were defined by the characteristics of geo-objects in real-world objects. The model integrates different algorithms, such as Laplacian edge detection and region growing, to find, extract and classify 3D roofs from LiDAR DSM datasets. The extracted 3D roofs exhibit a high degree of similarity with corresponding 3D roofs in the real world.

From the above discussion, it is proposed that an intelligent processing unit should be capable of the following:

1) *Define its own location in space*: The application of VAs in the preceding chapters shows that VAs can find their locations in the simulated space under different schemas.
2) *Represent any discrete geographic phenomena entity through an irregular/regular vector data structure*: The geometric component of the VAs, defined on a directed planar graph, allows them to store any shapes in image space.
3) *Construct and change its geometry*: The geometric methods and rules enable VAs to construct and change their geometry.
4) *Update internal structure in the simulation space:* VAs use the transition rules to find and update their attributes during the simulation process.
5) *Perceive image and feature space*: The structure of the VA model allows it to perceive and act on the image and feature space simultaneously.
6) *Interact with other geo-objects in image space:* VAs can geometrically interact with other VAs and with their environment. They can also affect each other's state through the neighbourhood rules.

7) *Consider the characteristics of a complex system during the simulation process:* The mechanism and structure of the VA model allow the VAs to repeat their rules for producing unique spatial patterns, namely geo-objects, in image space and spectral patterns in feature space.
8) *Use the evidence derived from an image combined with external knowledge during decision-making:* This was achieved when using the VAs to model 3D roofs based on real-world descriptive facts of those roofs.
9) *Integrate and use data from multiple sources:* This can be justified through the application of VAs for the 3D roof extraction together with the GEOBIA method.
10) *Learn from experience:* The dynamic structure of VAs provides an ideal tool for learning from their experiences. In this case however, VAs were regarded as goal-oriented agents with a predefined set of rules. Learning capabilities are considered a potential future implementation.

As such, an intelligent processing unit distinguishes itself from the pixels and image objects covered in the previous chapters, in that a geo-object can autonomously construct its geometry, find its classes, and interact with other geo-objects and its environment. Such a processing unit could be termed as an "***intelligent processing unit***" because this structure gives the VA model intelligent properties in order to detect and extract geo-objects from image space just as a human interpreter does.

8.3. Limitations and Future Work

Through the development of the VA model for image analysis, the following limitations can be addressed in the context of the VA model and VA model applications.

8.3.1. VA model

VAs use a set of predefined rules to accomplish their goals. In this way, they lack the ability to learn from their experiences. When endowed with learning ability in a specific domain, VAs can find a solution that may depend on specific knowledge during the simulation process. For example, after a number of sequences, land and sea VAs can learn to find the best growth direction in order to extract coastlines from an image. In this case, instead of classifying the entire image to extract coastline, the VAs only classify the areas that the coastlines may exist. This improves the performance of the VA model by saving memory and simulation time. Learning can also increase the level of automation of the VAs. For

example, in pixel-based approaches, VAs can use their experiences to determine β, the parameter that determines the confidence level of the extracted training objects. A brief illustration of this concept was presented in Chapter 4 where the VAs automatically changes the threshold β to extract training samples.

To define and formulate the geometric methods, we used a four-neighbour neighbouring system (see Figure 3.3(a)). With the implementation of geometric methods, we demonstrated that the VA can construct its own geometry in image space. For example, in Chapter 6 and Chapter 7, we employed the VA to extract the buildings or roads from the raster datasets. The results for the VA model in extracting geo-objects from raster datasets were promising. It would be interesting further research to test other geometric algorithms, such as using an eight-neighbour, which also includes the neighbours at the corners.

All the experiments were carried out on a computer with an Intel CPU running at 3.40 GHz with 16 GB of memory. The framework of each method was developed in the Repast environment. The processing times for extracting training samples for unsupervised and semi-supervised classification were 214 and 143 seconds, respectively. The roof extraction process took 330 seconds. To extract geo-objects in the context of the GEOBIA approach, the VA model spent 432 seconds to classify the entire image. The elapsed times were within the acceptable limit based on the size of the raster datasets used (average size of 200 pixels). However, the computational cost may increase as the number of vertices within a VA increases. This is due to the stochastic nature of the VA model for certain processes. For example, the VA model uses a first-order neighbouring system to create a new vertex based on one of four main directions (see Section 3.2.3.1). In this case, each vertex of a VA is checked four times according to the transition and geometric rules at each iteration. This can increase the processing time when the number of vertices within a VA is increased. To solve this issue, after a number of specific sequences, VAs can be made to learn which direction to grow to achieve its desired goal based on the condition of its environment. It is worth mentioning that for all examples within this thesis the processing times only include machine running time. A realistic assessment can be computed based on the time spent on codifying transition rules (e.g. Section 6.4.1) or neighbourhood rules, and machine running time. Another issue raised from the stochastic nature of the VA model is the robustness of the VA model. To initialise in the vector space, the VAs in this thesis use a random scheme. Hence, the results may vary slightly for each run within an acceptable limit. Further research might

consider the possibility of developing an intelligent initialisation process that can make the VA model more robust.

8.3.2. VA model application

So far, we have successfully tested the VA in extracting geo-objects (e.g. training samples or 3D roofs) from high resolution multispectral images, hyperspectral images and DSM datasets. The use of the VA for other types of remotely sensed data (e.g. radar) is also a potential area for further research.

In the previous chapters, we used the VA model in the context of image classification processes, such as pixel-based or object-based image classification, to address geo-objects. The application of the VA model in identifying specific targets (e.g. road, lake or coastline) from raster datasets can also be an interesting area for further research.

In the area of software agents, ontologies have a vital role in explicitly describing geo-objects for respective domains and applications in Geographic Information Systems (GIS). From the point of view of the VA model, the conceptual framework in Figure 8.1 can be applied to clarify and model the relationships between the components of VAs in order to address geo-objects. The aim of this framework is to impose a rational and internally consistent description on simulating geographic phenomena. This framework also allows the VA model to describe the transforming process of data from a low level image representation (e.g. pixel) to high level object representations (e.g. geo-object) (Gahegan and Flack, 1999). The construction and implementation of that model is intended to help us in understanding the nature of what it describes.

In this framework, the geo-object's state, namely **L**, **S**, **N** components of the VA model, is expressed according to $\mathbf{M_L}$, $\mathbf{T_S}$, $\mathbf{R_N}$, ($\mathbf{M_L}$, $\mathbf{T_S}$), ($\mathbf{M_L}$, $\mathbf{R_N}$), ($\mathbf{T_S}$, $\mathbf{R_N}$) or ($\mathbf{M_L}$, $\mathbf{T_S}$, $\mathbf{R_N}$) rules. The following examples describe the effect of $\mathbf{M_L}$ methods on (**L**), (**S**), (**L**, **N**) and (**L**, **S**, **N**), as implemented in this thesis, respectively:

1) **(L)**: In the proposed geometry, the building fractions could not be more than $\sqrt{2} \times r$ (Chapter 3, Section 3.2.3);
2) **(S)**: The growing method can change the class of 3D roofs. For example, in Figure 7.4 (d) and Figure 7.5(e), 3D roofs are classified based on different elevations.
3) (**L**, **S**): The growing $\mathbf{M_L}$ method can change the geometry of a pond object and convert it into a lake object (see Figure 6.10).

4) (**L**, **N**): A shrinking/splitting method changes the geometry of a VA and its neighbours (see Figure 3.8).
5) (**L**, **S**, **N**): A merging/killing method moves a VA to an unknown coordinate system, changes the state of the VA to an unknown object, and removes the neighbours of the VA (see Figure 3.6 or Figure 7.1).

The study of other combinations within this framework could be an interesting area for further research. For example, a land geo-object can be considered an island geo-object if it lies within a waterbody. This can be implemented with a vector network (e.g. Delaunay triangular network, Moore (2011)) that includes links between VAs. In this case, the VA model can use **($\mathbf{T_S}$, $\mathbf{R_N}$)** rules to change a land into island object.

Another possibility for research in this area would be the study of modelling dependent objects, such as elevated features (e.g. forest) and their associated shadows, or sand and sea. For example, pairs of VAs can be seeded and evolve together. Such crude reasoning is capable of implementing association rules as part of $\mathbf{R_N}$ that can solve conflicts between objects exhibiting similar spectral characteristics (e.g. shadow and water). In this context, the VA model can also be applied for modelling the behaviour of an object (e.g. shadow) affected by another object or external phenomena (e.g. sun). In this case, the VAs can use ($\mathbf{M_L}$, $\mathbf{T_S}$, $\mathbf{R_N}$) rules to model the effect of the sun on the shadow according to **L**, the geometry of shadow objects.

The full scope of geometry, states, neighbourhood rules, and different combinations thereof, can achieve the main objective of image classification: ***to automatically categorise all pixels of an image into land cover classes or themes.***

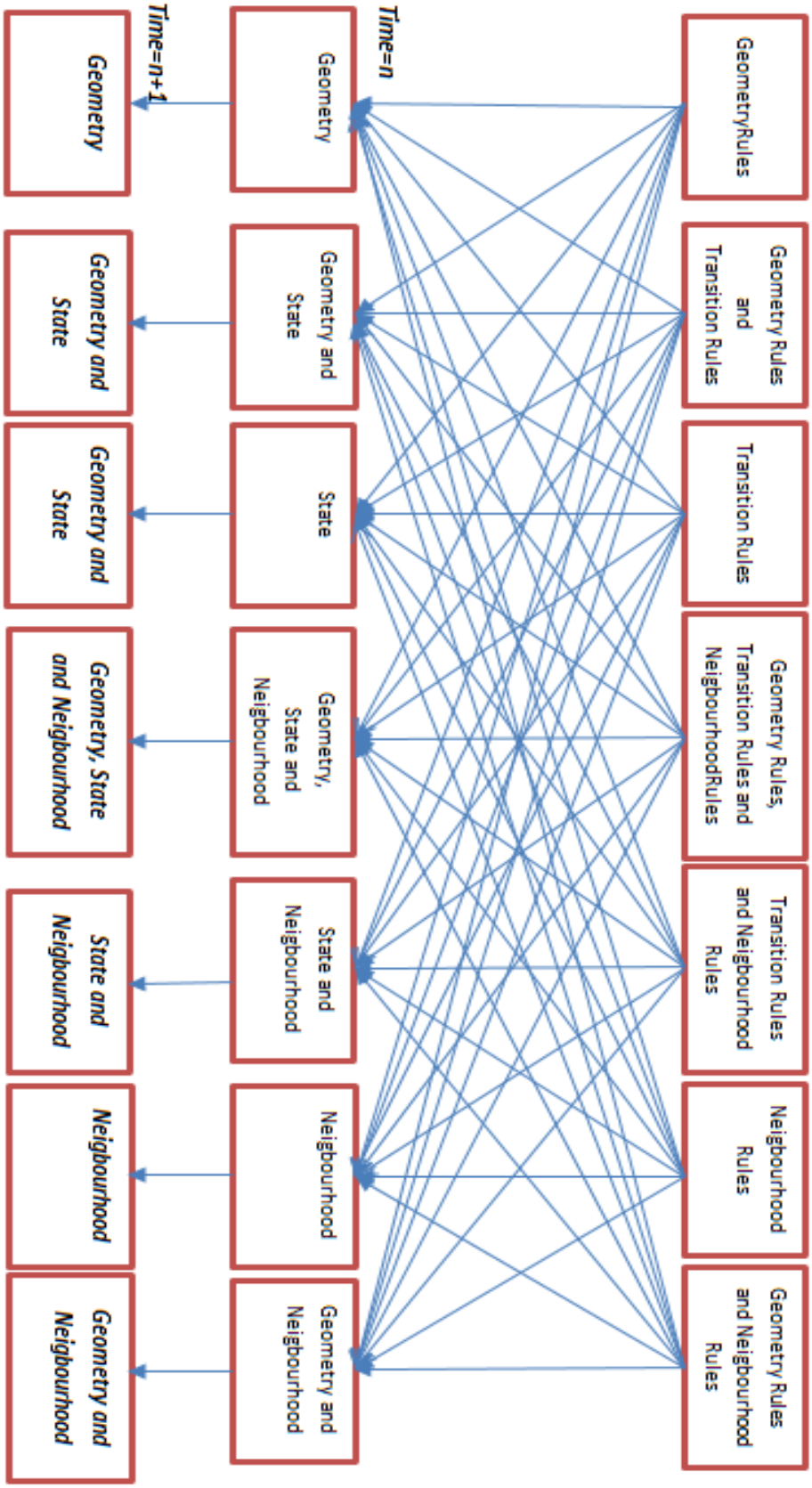


Figure 8.1. The components of each VA including state, transition rules, geometry, geometry rules, neighbourhood and neighbourhood rules, and their evolution through iteration.

References

Abdul-Rahman, A. and Pilouk, M. (2007), *Spatial data modelling for 3D GIS*, Springer Science & Business Media.

Alajlan, N.; Ammour, N.; Bazi, Y. and Hichri, H. (2011), A cluster ensemble method for robust unsupervised classification of VHR remote sensing images, *in* 'Geoscience and Remote Sensing Symposium (IGARSS), 2011 IEEE International', pp. 2896-2899.

Altaweel, M.; Collier, N.; Howe, T.; Najlis, R.; North, M.; Parker, M.; Tatara, E. and Vos, J. (2006), 'Repast: Recursive porus agent simulation toolkit', &lt.

Arvor, D.; Durieux, L.; Andrés, S. and Laporte, M.-A. (2013), 'Advances in geographic object-based image analysis with ontologies: a review of main contributions and limitations from a remote sensing perspective', *ISPRS Journal of Photogrammetry and Remote Sensing* **82**, 125-137.

Awrangjeb, M.; Lu, G. and Fraser, C. (2014), 'Automatic building extraction from LIDAR data covering complex urban scenes', *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* **40**(3), 25.

Baatz, M.; Benz, U.; Dehghani, S.; Heynen, M.; Нцltje, A.; Hofmann, P.; Lingenfelder, I.; Mimler, M.; Sohlbach, M.; Weber, M. and others (2001), 'eCognition user guide', *Definiens Imaging GmbH*.

Baatz, M.; Hoffmann, C. and Willhauck, G. (2008), Progressing from object-based to object-oriented image analysis 'Object-Based Image Analysis', Springer, pp. 29-42.

Baatz, M. and Schдpe, A. (2000), 'Multiresolution segmentation: an optimization approach for high quality multi-scale image segmentation', *Angewandte Geographische Information sverarbeitung XII* **58**, Herbert Wichmann Verlag: Berlin, Germany, 12-23.

Banerjee, B.; Bovolo, F.; Bhattacharya, A.; Bruzzone, L.; Chaudhuri, S. and Mohan, B. K. (2015), 'A New Self-Training-Based Unsupervised Satellite Image Classification Technique Using Cluster Ensemble Strategy', *Geoscience and Remote Sensing Letters, IEEE* **12**(4), 741--745.

Batty, M. (2007), *Cities and complexity: understanding cities with cellular automata, agent-based models, and fractals*, The MIT press.

Baumgart, B. G. (1975), A polyhedron representation for computer vision, *in* 'Proceedings of the May 19-22, 1975, national computer conference and exposition', pp. 589-596.

Baumgardner, M. F.; Biehl, L. L. and Landgrebe, D. A. (2015), '220 Band AVIRIS Hyperspectral Image Data Set: June 12, 1992 Indian Pine Test Site 3'.

Benenson, I. and Torrens, P. M. (2004a), 'Geosimulation: object-based modeling of urban phenomena', *Computers, Environment and Urban Systems* **28**(1), 1-8.

Benenson, I. and Torrens, P. M. (2004b), *Geosimulation: Automata-based modeling of urban phenomena*, John Wiley & Sons.

Benz, U. C.; Hofmann, P.; Willhauck, G.; Lingenfelder, I. and Heynen, M. (2004), 'Multi-resolution, object-oriented fuzzy analysis of remote sensing data for GIS-ready information', *ISPRS Journal of photogrammetry and remote sensing* **58**(3), 239-258.

Blaha, M. and Premerlani, W. (1997), *Object-oriented modeling and design for database applications*, Prentice-Hall, Inc.

Blaschke, T. (2010), 'Object based image analysis for remote sensing', *ISPRS journal of photogrammetry and remote sensing* **65**(1), 2-16.

Blaschke, T.; Hay, G. J.; Kelly, M.; Lang, S.; Hofmann, P.; Addink, E.; Feitosa, R. Q.; van der Meer, F.; van der Werff, H.; van Coillie, F. and others (2014), 'Geographic object-based image analysis–towards a new paradigm', *ISPRS Journal of Photogrammetry and Remote Sensing* **87**, 180-191.

Blaschke, T.; Lang, S.; Lorup, E.; Strobl, J. and Zeil, P. (2000), 'Object-oriented image processing in an integrated GIS/remote sensing environment and perspectives for environmental applications', *Environmental information for planning, politics and the public* **2**, 555--570.

Bonabeau, E. (2002), 'Agent-based modeling: Methods and techniques for simulating human systems', *Proceedings of the National Academy of Sciences* **99**(suppl 3), 7280-7287.

Borna, K.; Moore, A. and Sirguey, P. (2014), 'Towards a vector agent modelling approach for remote sensing image classification', *Journal of Spatial Science* **59**(2), 283-296.

Brenner, C. (2005), 'Building reconstruction from images and laser scanning', *International Journal of Applied Earth Observation and Geoinformation* **6**(3), 187-198.

Brustoloni, J. C. (1991), *Autonomous agents: characterization and requirements*, Citeseer.

Bruzzone, L.; Chi, M. and Marconcini, M. (2006), 'A novel transductive SVM for semisupervised classification of remote-sensing images', *Geoscience and Remote Sensing, IEEE Transactions on* **44**(11), 3363-3373.

Bruzzone, L. and Persello, C. (2009), 'A novel context-sensitive semisupervised SVM classifier robust to mislabelled training samples', *Geoscience and Remote Sensing, IEEE Transactions on* **47**(7), 2142-2154.

Camps-Valls, G.; Marsheva, T. V. B. and Zhou, D. (2007), 'Semi-supervised graph-based hyperspectral image classification', *Geoscience and Remote Sensing, IEEE Transactions on* **45**(10), 3044-3054.

Castilla, G. and Hay, G. (2008), Image objects and geographic objects 'Object-based image analysis', Springer, pp. 91-110.

Castilla G (2003), 'Object-oriented Analysis of Remote Sensing Images for Land Cover Mapping: Conceptual Foundations and a Segmentation Method to Derive a Baseline Partition for Classification', PhD Thesis, Polytechnic University of Madrid, URL: http://oa.upm.es/133/01/07200302_castilla_castellano.pdf

Chang, C.-C. and Lin, C.-J. (2011), 'LIBSVM: a library for support vector machines', *ACM Transactions on Intelligent Systems and Technology (TIST)* **2**(3), 27.

Chapelle, O.; Sindhwani, V. and Keerthi, S. S. (2008), 'Optimization techniques for semi-supervised support vector machines', *The Journal of Machine Learning Research* **9**, 203-233.

Chi, M.; Feng, R. and Bruzzone, L. (2008), 'Classification of hyperspectral remote-sensing data with primal SVM for small-sized training dataset problem', *Advances in space research* **41**(11), 1793-1799.

Comić, L. and De Floriani, L. (2012), Modeling and Manipulating Cell Complexes in Two, Three and Higher Dimensions 'Digital Geometry Algorithms', Springer, pp. 109-144.

Colonna, A.; di Stephano, V.; Lombardo, S.; Papini, L. and Rabino, A. (1998), Learning

cellular automata: modelling urban modelling, *in* '3rd Int. Conf. on GeoComputation', pp. 17-19.

Couclelis, H. (1992), People manipulate objects (but cultivate fields): beyond the raster-vector debate in GIS 'Theories and methods of spatio-temporal reasoning in geographic space', Springer, pp. 65-77.

Cova, T. J. and Goodchild, M. F. (2002), 'Extending geographical representation to include fields of spatial objects', *International Journal of Geographical Information Science* **16**(6), 509-532.

Crooks, A. T. and Castle, C. J. (2012), The integration of agent-based modelling and geographical information for geospatial simulation 'Agent-based models of geographical systems', Springer, pp. 219-251.

Crooks, A. T. and Hailegiorgis, A. B. (2014), 'An agent-based modeling approach applied to the spread of cholera', *Environmental Modelling & Software* **62**, 164-177.

Crooks, A. T. and Heppenstall, A. J. (2012), Introduction to agent-based modelling 'Agent-based models of geographical systems', Springer, pp. 85-105.

David, B.; Herrewegen, M. and Salge, F. (1996), *Conceptual models for geometry and quality of geographic information*, Taylor & Francis.

Dópido, I.; Li, J.; Marpu, P. R.; Plaza, A.; Bioucas Dias, J. M. and Benediktsson, J. A. (2013), 'Semisupervised self-learning for hyperspectral image classification', *Geoscience and Remote Sensing, IEEE Transactions on* **51**(7), 4032-4044.

Dópido, I.; Villa, A.; Plaza, A. and Gamba, P. (2012), 'A quantitative and comparative assessment of unmixing-based feature extraction techniques for hyperspectral image classification', *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of* **5**(2), 421-435.

Drăguţ, L.; Tiede, D., and Levick, S. R. (2010), 'ESP: a tool to estimate scale parameter for multiresolution image segmentation of remotely sensed data', *International Journal of Geographical Information Science*, 24(6), 859-871.

Duda, R. O.; Hart, P. E. and Stork, D. G. (2012), *Pattern classification*, John Wiley & Sons.

Egenhofer, M. J. (1989), A formal definition of binary topological relationships 'Foundations of data organization and algorithms', Springer, pp. 457-472.

Epstein, J. M. (1999), 'Agent-based computational models and generative social science', *Complexity* **4**(5), 41-60.

Espinola, M.; Piedra, J. A.; Ayala, R.; Iribarne, L.; Leguizamón, S. and Menenti, M. (2012), ACA multiagent system for satellite image classification' Trends in Practical Applications of Agents and Multiagent Systems', Springer, pp. 93-100.

Feitosa, R. Q.; Costa, G. A. O. P.; Cazes, T. B., and Feijo, B., (2006), 'A genetic approach for the automatic adaptation of segmentation parameters', *In Proceedings of the First International Conference on Object-Based Image Analysis*, Salzburg, Austria (Vol. 45).

Filin, S. and Pfeifer, N. (2006), 'Segmentation of airborne laser scanning data using a slope adaptive neighborhood', *ISPRS journal of Photogrammetry and Remote Sensing* **60**(2), 71-80.

Firesmith, D. G.; Eykholt, E. M. and Siegel, J. (1995), *Dictionary of object technology: the definitive desk reference*, Cambridge University Press.

Foody, G. M.; Campbell, N.; Trodd, N. and Wood, T. (1992), 'Derivation and applications of probabilistic measures of class membership from the maximum-likelihood classification', *Photogrammetric Engineering and Remote Sensing* **58**(9), 1335-1341.

Foody, G. M. and Mathur, A. (2004), 'A relative evaluation of multiclass image classification by support vector machines', *Geoscience and Remote Sensing, IEEE Transactions on* **42**(6), 1335-1343.

Franklin, S. and Graesser, A. (1996), Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents 'Intelligent agents III agent theories, architectures, and languages', Springer, , pp. 21-35.

Gao, J. (2008), *Digital analysis of remotely sensed imagery*, McGraw-Hill Professional.

Gahegan, M., & Flack, J. (1999), 'The integration of scene understanding within a geographic information system: a prototype approach for agricultural applications', *Transactions in GIS*, 3(1), 31-49.

Ghaffarian, S. and Ghaffarian, S. (2014), 'Automatic histogram-based fuzzy C-means clustering for remote sensing imagery', *ISPRS Journal of Photogrammetry and Remote Sensing* **97**, 46-57.

Gimblett, H. R. (2002), 'Integrating geographic information systems and agent-based technologies for modeling and simulating social and ecological phenomena', *Integrating geographic information systems and agent-based modeling techniques for simulating social and ecological processes*, 1-20.

Gitas, I. Z.; Mitri, G. H. and Ventura, G. (2004), 'Object-based image classification for burned area mapping of Creus Cape, Spain, using NOAA-AVHRR imagery', *Remote Sensing of Environment* **92**(3), 409-413.

Goldstein, N. C. (2004), 'Brains versus brawn—comparative strategies for the calibration of a cellular automata-based urban growth model', *GeoDynamics*, 249-272.

Goodchild, M. F.; Yuan, M. and Cova, T. J. (2007), 'Towards a general theory of geographic representation in GIS', *International journal of geographical information science* **21**(3), 239-260.

Goodrich, M. and Ramaiyer, K. (1998), 'Geometric data structures', *Handbook of Computational Geometry, page*.

Gruber, T. R. (1995), 'Toward principles for the design of ontologies used for knowledge sharing?', *International journal of human-computer studies* **43**(5), 907-928.

Hammam, Y. (2008), 'Geographical vector agents'.

Hammam, Y.; Moore, A. and Whigham, P. (2007), 'The dynamic geometry of geographical vector agents', *Computers, Environment and Urban Systems* **31**(5), 502-519.

Hay, G. J. and Castilla, G. (2008), Geographic Object-Based Image Analysis (GEOBIA): A new name for a new discipline 'Object-based image analysis', Springer, pp. 75-89.

Hay, G. J.; Castilla, G.; Wulder, M. A. and Ruiz, J. R. (2005), 'An automated object-based approach for the multiscale image segmentation of forest scenes', *International Journal of Applied Earth Observation and Geoinformation* **7**(4), 339-359.

Hofmann, P.; Lettmayer, P.; Blaschke, T.; Belgiu, M.; Wegenkittl, S.; Graf, R.; Lampoltshammer, T. J. and Andrejchenko, V. (2015), 'Towards a framework for agent-based image analysis of remote-sensing data', *International Journal of Image and Data*

Fusion **6**(2), 115-137.

Howe, T.; Collier, N.; North, M.; Parker, M.; Vos, J. and others (2006), Containing agents: Contexts, projections, and agents, *in* 'Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects. Argonne, Illinois, USA: Argonne National Laboratory'.

Hughes, G. P. (1968), 'On the mean accuracy of statistical pattern recognizers', *Information Theory, IEEE Transactions on* **14**(1), 55-63.

Hung, C.-C.; Kulkarni, S. and Kuo, B.-C. (2011), 'A new weighted fuzzy C-means clustering algorithm for remotely sensed image classification', *Selected Topics in Signal Processing, IEEE Journal of* **5**(3), 543-553.

Jackson, J.; Forest, B. and Sengupta, R. (2008), 'Agent-Based Simulation of Urban Residential Dynamics and Land Rent Change in a Gentrifying Area of Boston', *Transactions in GIS* **12**(4), 475-491.

Jensen, J. R. (1986), 'Introductory digital image processing: a remote sensing perspective', Technical report, Univ. of South Carolina, Columbus.

Kainz, W. (2004), 'The Mathematics of GIS', ESRI Press (to appear).

Kay, A. (1984), 'Computer Software', *Scientific American* **251**(3), 53-59.

Kearns, M.; Mansour, Y. and Ng, A. Y. (1998), An information-theoretic analysis of hard and soft assignment methods for clustering ‘Learning in graphical models', Springer, pp. 495-520.

Kim, K. and Shan, J. (2011), 'Building roof modeling from airborne laser scanning data based on level set approach', *ISPRS Journal of Photogrammetry and Remote Sensing* **66**(4), 484-497.

Kim, M.; Madden, M. and Warner, T. (2008), Estimation of optimal image object size for the segmentation of forest stands with multispectral IKONOS imagery ‘Object-based image analysis', Springer, pp. 291-307.

Koltunov, A.; Crouvi, O. and Ben-Dor, E. (2006), 'Geomorphologic mapping from hyperspectral data, using Gaussian mixtures and lower confidence bounds', *International Journal of Remote Sensing* **27**(20), 4545-4566.

Krithara, A.; Amini, M. R.; Goutte, C. and Renders, J.-M. (2011), 'Learning aspect models with partially labelled data', *Pattern Recognition Letters* **32**(2), 297-304.

Kwak, E. and Habib, A. (2014), 'Automatic representation and reconstruction of DBM from LiDAR data using Recursive Minimum Bounding Rectangle', *ISPRS Journal of Photogrammetry and Remote Sensing* **93**, 171-191.

Lafarge, F.; Descombes, X.; Zerubia, J. and Pierrot-Deseilligny, M. (2010), 'Structural approach for building reconstruction from a single DSM', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **32**(1), 135-147.

Langran, G. and Chrisman, N. R. (1988), 'A framework for temporal geographic information', *Cartographica: The International Journal for Geographic Information and Geovisualization* **25**(3), 1-14.

Lari, Z. and Habib, A. (2014), 'An adaptive approach for the segmentation and extraction of planar and linear/cylindrical features from laser scanning data', *ISPRS Journal of Photogrammetry and Remote Sensing* **93**, 192-212.

Li, N.; Huo, H.; Zhao, Y.-m.; Chen, X. and Fang, T. (2013), 'A spatial clustering method

with edge weighting for image segmentation', *Geoscience and Remote Sensing Letters, IEEE* **10**(5), 1124--1128.

Li, X. and Yeh, A. G.-O. (2001), 'Calibration of cellular automata by using neural networks for the simulation of complex urban systems', *Environment and Planning A* **33**(8), 1445-1462.

Li, Z.; Wu, X. M.; Chang, S. F. (2012, 'Segmentation using superpixels: A bipartite graph partitioning approach'. *In Computer Vision and Pattern Recognition (CVPR), IEEE Conference*, pp. 789-796.

Ligtenberg, A.; Bregt, A. K. and Van Lammeren, R. (2001), 'Multi-actor-based land use modelling: spatial planning using agents', *Landscape and urban planning* **56**(1), 21-33.

Lim, K.; Deadman, P. J.; Moran, E.; Brondizio, E. and McCracken, S. (2002), 'Agent-based simulations of household decision making and land use change near Altamira, Brazil', *Integrating Geographic Information Systems and Agent-Based Modeling: Techniques for Simulating Social and Ecological Processes*, 277-310.

Liu, X.-y.; Liao, Z.-w.; Wang, Z.-s. and Chen, W.-f. (2006), Gaussian Mixture Models Clustering using Markov Random Field for Multispectral Remote Sensing Images, *in* 'Machine Learning and Cybernetics, 2006 International Conference on', pp. 4155-4159.

Lu, D. and Weng, Q. (2007), 'A survey of image classification methods and techniques for improving classification performance', *International journal of Remote sensing* **28**(5), 823-870.

Ma, R. (2005), 'DEM generation and building detection from lidar data', *Photogrammetric Engineering and Remote Sensing* **71**(7), 847-854.

Maas, H.-G. and Vosselman, G. (1999), 'Two algorithms for extracting building models from raw laser altimetry data', *ISPRS Journal of photogrammetry and remote sensing* **54**(2), 153-163.

Macal, C. M. and North, M. J. (2009), Agent-based modeling and simulation, *in* 'Winter simulation conference', pp. 86-98.

Mahmoudi, F. T.; Samadzadegan, F. and Reinartz, P. (2013), 'Object oriented image analysis based on multi-agent recognition system', *Computers & Geosciences* **54**, 219-230.

Marceau, D. and Moreno, N. (2008), An object-based cellular automata model to mitigate scale dependency ‘Object-based image analysis', Springer, pp. 43-73.

Marceau, D. J. and Benenson, I. (2011), *Challenges and perspectives in Geosimulation*, Bentham Science Publisher.

Martin, J. and Odell, J. J. (1992), *Object-oriented analysis and design*, Prentice-Hall, Inc..

Mather, P. and Tso, B. (2009), *Classification methods for remotely sensed data*, CRC press.

Mather, P. M. and Koch, M. (2011), *Computer processing of remotely-sensed images: an introduction*, John Wiley and Sons.

Mathieu, R.; Freeman, C. and Aryal, J. (2007), 'Mapping private gardens in urban areas using object-oriented techniques and very high-resolution satellite imagery', *Landscape and Urban Planning* **81**(3), 179-192.

Mathur, A. and Foody, G. M. (2008), 'Crop classification by support vector machine with intelligently selected training data for an operational application', *International Journal of Remote Sensing* **29**(8), 2227-2240.

Melgani, F. and Bruzzone, L. (2004), 'Classification of hyperspectral remote sensing images with support vector machines', *Geoscience and Remote Sensing, IEEE Transactions on* **42**(8), 1778-1790.

Meng, Z. and Yao, G. (2015), Research on the Algorithm of Image Classification Based on Gaussian Mixture Model, *in* 'International Conference on Computer Information Systems and Industrial Applications'.

Molenaar, M. (1998), *An introduction to the theory of spatial object modelling for GIS*, CRC Press.

Moore, A. (2011), *Geographical vector agent based simulation for agricultural land use modelling*, Bentham Science Publisher.

Moreno, N.; Wang, F. and Marceau, D. J. (2010), 'A geographic object-based approach in cellular automata modeling', *Photogrammetric Engineering & Remote Sensing* **76**(2), 183-191.

Moreno, N.; Wang, F. and Marceau, D. J. (2009), 'Implementation of a dynamic neighborhood in a land-use vector-based cellular automata model', *Computers, Environment and Urban Systems* **33**(1), 44-54.

Mountrakis, G., Im, J., and Ogole, C. (2011), 'Support vector machines in remote sensing: A review, ' *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(3), 247-259.

Muller, D. E. and Preparata, F. P. (1978), 'Finding the intersection of two convex polyhedra', *Theoretical Computer Science* **7**(2), 217-236.

Mukhopadhyay, A., and Maulik, U. (2009). 'Unsupervised pixel classification in satellite imagery using multiobjective fuzzy clustering combined with SVM classifier', *IEEE transactions on geoscience and remote sensing*, **47**(4), 1132-1138.Mylonas, S. K.; Stavrakoudis, D. G. and Theocharis, J. B. (2013), 'GeneSIS: A GA-based fuzzy segmentation algorithm for remote sensing images', *Knowledge-Based Systems* **54**, 86-102.

Navulur, K. (2006), *Multispectral image analysis using the object-oriented paradigm*, CRC press.

Okabe, A.; Boots, B.; Sugihara, K. and Chiu, S. N. (2009), *Spatial tessellations: concepts and applications of Voronoi diagrams*, Vol. 501, John Wiley and Sons.

Omran, M. G. and Engelbrecht, A. P. (2006), Self-adaptive differential evolution methods for unsupervised image classification, *in* 'Cybernetics and Intelligent Systems, 2006 IEEE Conference on', pp. 1-6.

O'Sullivan, D.; Millington, J.; Perry, G. and Wainwright, J. (2012), Agent-Based Models–Because They're Worth It? 'Agent-based models of geographical systems', Springer, pp. 109-123.

Peeters, A. and Etzion, Y. (2010), 'Automated recognition of urban objects and their morphological attributes using GIS', *ISPRS Archives* **38**(Part 4), 8-2.

Pelekis, N.; Theodoulidis, B.; Kopanakis, I. and Theodoridis, Y. (2004), 'Literature review of spatio-temporal database models', *The Knowledge Engineering Review* **19**(03), 235-274.

Peuquet, D. J. (1994), 'It's about time: A conceptual framework for the representation of temporal dynamics in geographic information systems', *Annals of the Association of american Geographers* **84**(3), 441-461.

Pullar, D. and Egenhofer, M. (1988), Toward formal definitions of topological relations

among spatial objects, *in* 'Proceedings of the Third International Symposium on Spatial Data Handling', pp. 42.

Raubal, M. (2001), *Agent-based simulation of human wayfinding: A perceptual model for unfamiliar buildings*, na.

Raza, A. (2001), *Object-oriented temporal GIS for urban applications*, University of Twente.

Reynolds, C. W. (1987), Flocks, herds and schools: A distributed behavioural model, *in* 'ACM SIGGRAPH computer graphics', pp. 25-34.

Richards, J. A. and Richards, J. (2006), *Remote sensing digital image analysis*, Vol. 4, Springer.

Rodrigues, A.; Grueau, C.; Raper, J. and Neves, N. (1998), 'Environmental planning using spatial agents', *Innovations in GIS* **5**, 108-118.

Russell, S.; Norvig, P. and Intelligence, A. (1995), 'A modern approach', *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs* **25**, 27.

Sack, J.-R. and Urrutia, J. (1999), *Handbook of computational geometry*, Elsevier.

Samadzadegan, F.; Mahmoudi, F. T. and Schenk, T. (2010), 'An agent-based method for automatic building recognition from lidar data', *Canadian Journal of Remote Sensing* **36**(3), 211-223.

Samet, H. (2006), *Foundations of multidimensional and metric data structures*, Morgan Kaufmann.

Sampath, A. and Shan, J. (2007), 'Building boundary tracing and regularization from airborne LiDAR point clouds', *Photogrammetric Engineering and Remote Sensing* **73**(7), 805-812.

Schiewe, J., Tufte, L. and Ehlers, M. (2001), 'Potential and problems of multi-scale segmentation methods in remote sensing', *GeoBIT/GIS* **6**(01), 34-39.

Schiewe, J. (2002), 'Segmentation of high-resolution remotely sensed data-concepts, applications and problems, ' *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences* 34(4), 380-385.

Schlieder, C. (1996), 'Qualitative shape representation', *Geographic objects with indeterminate boundaries* **2**, 123-140.

Schwalbe, E.; Maas, H.-G. and Seidel, F. (2005), '3D building model generation from airborne laser scanner data using 2D GIS data and orthogonal point cloud projections', *Proceedings of ISPRS WG III/3, III/4* **3**, 12-14.

Semboloni, F. (2000), 'The growth of an urban cluster into a dynamic self-modifying spatial pattern', *Environment and Planning B: Planning and Design* **27**(4), 549-564.

Shahshahani, B. M. and Landgrebe, D. A. (1994), 'The effect of unlabeled samples in reducing the small sample size problem and mitigating the Hughes phenomenon', *Geoscience and Remote Sensing, IEEE Transactions on* **32**(5), 1087-1095.

Shen, Z.; Kawakami, M. and Kawamura, I. (2009), 'Geosimulation model using geographic automata for simulating land-use patterns in urban partitions', *Environment and planning B: planning and design* **36**(5), 802-823.

Shi, W. and Pang, M. Y. C. (2000), 'Development of Voronoi-based cellular automata-an integrated dynamic model for Geographical Information Systems', *International Journal of Geographical Information Science* **14**(5), 455-474.

Sisodia, P. S.; Tiwari, V. and Kumar, A. (2014), A comparative analysis of remote sensing image classification techniques, *in* 'Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on', pp. 1418-1421.

Smith, B. and Mark, D. M. (1998), 'Ontology and geographic kinds'.

Smith, D. C.; Cypher, A. and Spohrer, J. (1994), 'KidSim: programming agents without a programming language', *Communications of the ACM* **37**(7), 54-67.

Song, J.; Wu, J. and Jiang, Y. (2015), 'Extraction and reconstruction of curved surface buildings by contour clustering using airborne LiDAR data', *Optik-International Journal for Light and Electron Optics* **126**(5), 513--521.

Srivastava, P. K.; Han, D.; Rico-Ramirez, M. A.; Bray, M. and Islam, T. (2012), 'Selection of classification techniques for land use/land cover change investigation', *Advances in Space Research* **50**(9), 1250-1265.

Sun, S. and Salvaggio, C. (2013), 'Aerial 3D building detection and modeling from airborne LiDAR point clouds', *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of* **6**(3), 1440-1449.

Takeyama, M. (1997), 'Building spatial models within GIS through Geo-Algebra', *Transactions in GIS* **2**(3), 245-256.

Tan, K.; Hu, J.; Li, J. and Du, P. (2015), 'A novel semi-supervised hyperspectral image classification approach based on spatial neighborhood information and classifier combination', *ISPRS Journal of Photogrammetry and Remote Sensing* **105**, 19-29.

Tan, K.; Li, E.; Du, Q. and Du, P. (2014), 'An efficient semi-supervised classification approach for hyperspectral imagery', *ISPRS Journal of Photogrammetry and Remote Sensing* **97**, 36-45.

Tao, J.; Shu, N.; Wang, Y.; Hu, Q. and Zhang, Y. (2016), 'A study of a Gaussian mixture model for urban land-cover mapping based on VHR remote sensing imagery', *International Journal of Remote Sensing* **37**(1), 1-13.

Tarsha-Kurdi, F.; Landes, T.; Grussenmeyer, P. and Koehl, M. (2007), 'Model-driven and data-driven approaches using LIDAR data: Analysis and comparison', *Int. Arch. Photogrammetry. Remote Sens. Spat. Inf. Sci* **36**, 87-92.

Thomas, N.; Hendrix, C. and Congalton, R. G. (2003), 'A comparison of urban mapping methods using high-resolution digital imagery', *Photogrammetric Engineering & Remote Sensing* **69**(9), 963-972.

Tian, J. and Chen, D.-M. (2007), 'Optimization in multi-scale segmentation of high-resolution satellite images for artificial feature recognition', *International Journal of Remote Sensing* **28**(20), 4625-4644.

Torrens, P. M. (2000), 'How cellular models of urban systems work (1. Theory)'.

Torrens, P. M. and Benenson, I. (2005), 'Geographic automata systems', *International Journal of Geographical Information Science* **19**(4), 385-412.

Tso, B. and Olsen, R. C. (2005), 'Combining spectral and spatial information into hidden

Markov models for unsupervised image classification', *International Journal of Remote Sensing* **26**(10), 2113-2133.

Tyagi, M.; Bovolo, F.; Mehra, A. K.; Chaudhuri, S. and Bruzzone, L. (2008), 'A context-sensitive clustering technique based on graph-cut initialization and expectation-maximization algorithm', *Geoscience and Remote Sensing Letters, IEEE* **5**(1), 21-25.

Vapnik, V. N. and Chervonenkis, A. Y. (2015), On the uniform convergence of relative frequencies of events to their probabilities ‘Measures of Complexity', Springer, pp. 11-30.

Walter, V. (2004), 'Object-based classification of remote sensing data for change detection', *ISPRS Journal of photogrammetry and remote sensing* **58**(3), 225-238.

Wang, L.; Hao, S.; Wang, Y.; Lin, Y. and Wang, Q. (2014), 'Spatial–spectral information-based semisupervised classification algorithm for hyperspectral imagery', *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of* **7**(8), 3577-3585.

White, R. and Engelen, G. (2000), 'High-resolution integrated modelling of the spatial dynamics of urban and regional systems', *Computers, environment and urban systems* **24**(5), 383-400.

Wooldridge, M. (2009), *An introduction to multiagent systems*, John Wiley & Sons.

Wooldridge, M.; Jennings, N. R. and others (1995), 'Intelligent agents: Theory and practice', *Knowledge engineering review* **10**(2), 115-152.

Worboys, M. F. and Duckham, M. (2004), *GIS: a computing perspective*, CRC press.

Wu, F. (1996), 'A linguistic cellular automata simulation approach for sustainable land development in a fast growing region', *Computers, Environment and Urban Systems* **20**(6), 367-387.

Yang, Y. and Huang, S. (2012), 'Image segmentation by fuzzy C-means clustering algorithm with a novel penalty term', *Computing and Informatics* **26**(1), 17-31.

Zanaty, E. and Afifi, A. (2013), 'A new approach for automatic fuzzy clustering applied to magnetic resonance image clustering', *American Journal of Remote Sensing* **1**(2), 38-46.

Zheng, N.; Zhang, H.; Fan, J. and Guan, H. (2014), 'A fuzzy local neighbourhood-attraction-based information c-means clustering algorithm for very high spatial resolution imagery classification', *Remote Sensing Letters* **5**(9), 843-852.

Zhong, Y.; Zhao, B. and Zhang, L. (2014), 'Multiagent object-based classifier for high spatial resolution imagery', *Geoscience and Remote Sensing, IEEE Transactions on* **52**(2), 841-857.

Zhou, Y. and Wang, Y. (2008), 'Extraction of impervious surface areas from high spatial resolution imagery by multiple agent segmentation and classification', *Photogrammetric Engineering & Remote Sensing* **74**(7), 857-868.

Appendix

A- VA model structure in Repast

A.1. Introduction

Image classification approaches usually utilise pixels or image objects as the fundamental processing unit. These conventional classification methods usually depend on a two-stage sequential process of image segmentation and classification (or vice versa) in order to identify geo-objects in raster datasets. However, these methods lack the necessary capabilities of simultaneously extracting geometric and thematic meaning of geo-objects from image space. This issue can be addressed using Vector Agents (VAs).

A.1.1.1. Purpose

This section is designed to describe the main elements of the VAs in order to extract geo-objects from raster datasets in a unified classification process.

A.1.1.2. Data

To describe the VA model, a subset of WorldView-3 image 8 multispectral bands (red, red edge, coastal, blue, green, yellow, near-IR1 and near-IR2) from a rural area near Dunedin, New Zealand is applied (Figure Appendix A.1(a)). The image dataset is a subset with a resolution of 140×90 pixels (1.20m pixel size) of a complex scene containing six land cover classes: bare soil, grass, lake, pond, river and shadow (Figure Appendix A.1(a)). You can find this image named *'ImageData.xls'* (converted into an Excel file) in the source folder.

In our example, the proposed method uses nine labelled pixels for each cluster in order to train the Support Vector Machine (SVM) classifier through the SVM class. The training data can be found in the source folder, named *'TrainData.txt'* in the source folder. The VA model applies the SVM classifier to implement the transition rules. Since the waterbody clusters, namely pond, river and lake, have similar spectral reflectance, the proposed method only uses 36 pixels to train the SVM classifier in the initialization step.

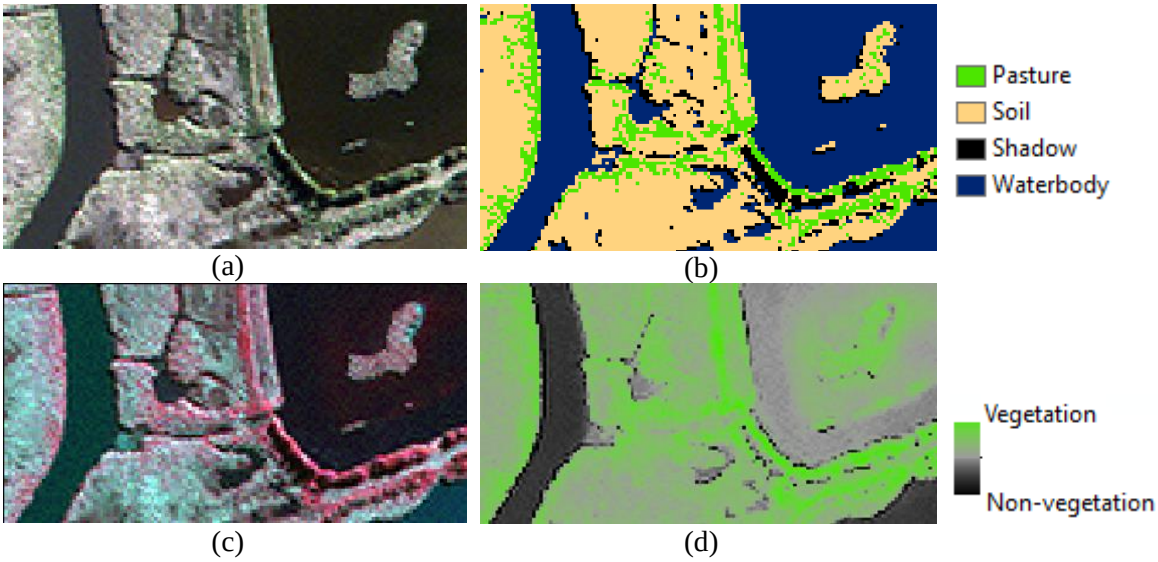
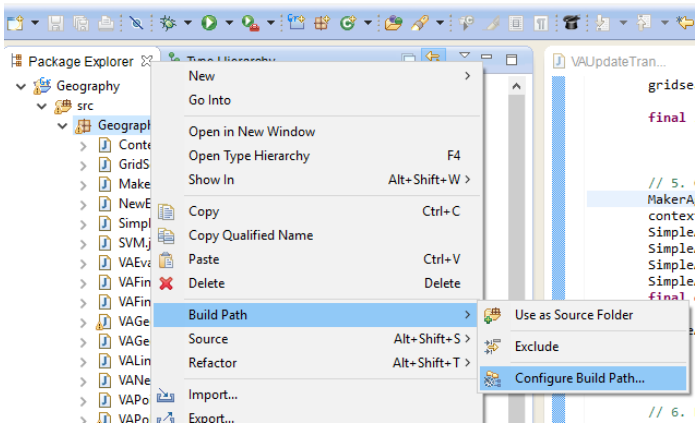


Figure Appendix A- 1. (a) It displays a true colour combination band of WorldView-3multispectral image, Digital Globe Foundation, www.digitalglobefoundation.org (b) The classified image is provided using the SVM classifier based on the initial label samples, (c) a false colour band combination of the given image and (d) it displays the given image based on the Normalized Difference Vegetation Index (NDVI).

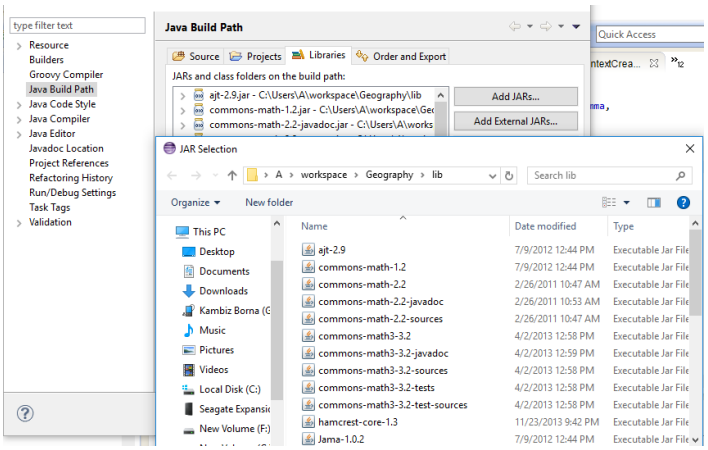
A.1.1.3. User requirements

To run the example, you need Repast Simphony (version 2.1 or later) and the Java JDK (Java development kit) which can be downloaded on the Repast website and Java Standard Edition Downloads Page, respectively. After installing Repast and Java JDK, copy and paste the Geography folder to your root directory (e.g., "C:\").

To run the Geography model, you also need to add a collection of java libraries to your Repast project. You can find these libraries in the Geography folder (e.g., "F:\ Geography\lib"). Figure Appendix A.2 shows how you can add them to the project.



(a)



(b)

Figure Appendix A- 2. (a) and (b) display how you can add the java libraries to the Repast.

Please check that you have set the image file path in the ContextCreator class in your project.

A.2. Classes

In this section, we present the main components of the proposed method that consists of procedure, geometry, state, neighbourhood and utility (see Figure 3.12 for more details).

A.2.1.1. Procedure

The procedure component includes the classes that describe the agents and their environment.

A.2.1.1.1. SimpleAgent

SimpleAgent provides initial information on input data for all agents. This information is defined based on the input data and spatial/ non-spatial information of geo-objects, such as elongation index for the waterbody classes.

- ***Variables***

In our experiment, the SimpleAgent uses the variables in Table Appendix A.1-3 to classify the image:

Table Appendix A- 1. A VecAgent generally uses these variables to control its geometry and state.

Variable name	Variable type	Value	Comment
Hole_Size_VA	int	15	Determines the minimum size for each interior ring in a VA. It is specified based on the number of vertices within each VA.
Number_Elgible_Pixels	int	15	Determines the number of pixels applied by the proposed method to train the SVM classifier. These pixels are selected from the most informative VA-generated samples.
Size_VA	int	15	Used to determine the minimum size of each VA. It is defined based on the number of vertices within each VA. A VA is removed if its size is less than the Size_VA threshold. The variables can independently be specified for each cluster or each geo-objects. For example, in a given image, the forest cluster can be divided into two clusters based on the number of holes.

Table Appendix A- 2. The method uses these variables to classify the waterbody objects.

Variable name	Variable type	Value	Comment
Waterbody_Area_Threshold	int	300	Used to classify the waterbody objects based on their geometric characteristics.
Elongation_Threshold	int	2	

Table Appendix A- 3. Variables which are used by all classes.

Variable name	Variable type	Value	Comment
cont	Context	…	Built in the initialization step by the ContextCreator class.
svmModel	svm_model	…	The ContextCreator class uses the SVM class to train the SVM classifier based on the initial labelled samples (36 pixels).
Image_Data	double [][][]	…	Initialized via the ContextCreator class.
feature	double [][]	…	The ContextCreator computes this variable via the VAUpdateTransitionRules.
trainData	double [][]	…	Initialized by the ContextCreator class.
Grid_SizeX	int	141	Automatically initialized via the ContextCreator.
Grid_SizeY	int	90	Automatically initialized via the ContextCreator.
Num_Band	int	8	Automatically initialized via the ContextCreator.
fac	VAGeometry Factory	…	Constructed by the SimpleAgent class.
Poly_RemovedVA	Polygon	A predefined polygon based on the size of the image	A polygon used to transfer a VA from a georeferenced plane to a non-georeferenced plane. In fact, the method uses this geometry to remove or kill a VA.
Poly_Frame	Polygon	A predefined polygon based on the size of the image	Used to control the location of the VAs within a georeferenced plane.
status	String	growth	The MakerAgent uses this variable to coordinate the behaviour of the VAs in each stage. The proposed method is performed in four main steps: growth, development, construction and production.

❖ ***Method***

i. public void step ()

This method is called by the MakerAgent and VecAgent at every iteration.

A.2.1.1.2. MakerAgent

The proposed method utilises the MakerAgent class to first generate the VAs, placing them into context through the generateVA method. MakerAgent is also responsible for facilitating the coordination between GVAs. To do this, MakerAgent applies the controlVA method.

- ***Variables***

  This class only includes local variables.

- ***Method***

  The proposed method consists of four main steps: *growth*, *development*, *construction* and *production*. In each step, different geometric operators are used to identify geo-objects from the image space (Table Appendix A.4).

Table Appendix A- 4. Different stages of the VA-based image classification.

Step name	geometric operators	SVM model
growth	born, grow, merge and kill	The SVM classifier is only trained based on the initial labelled samples.
development	born, grow, merge and kill	The SVM classifier is trained based on the initial and VA-generated samples.
construction	born, grow, merge and kill	The VA model uses the SVM model trained in the development step.
production	born, grow, merge, shrink, split and kill	The VA model uses the SVM model trained in the construction step.

As can be seen from the above table, different strategies are used to train the SVM classifier at each stage.

In the construction step, a VA captures a pixel if it is an adjacent pixel. VAs utilise the shrinking, shrinking/splitting and shrinking/killing operators only in the production step. Figure *Appendix A*.3 displays the results of each step (see the video clip1 in the source folder for more detail). In this example, VAs do not use shrinking/splitting and shrinking/killing operators in the production step.

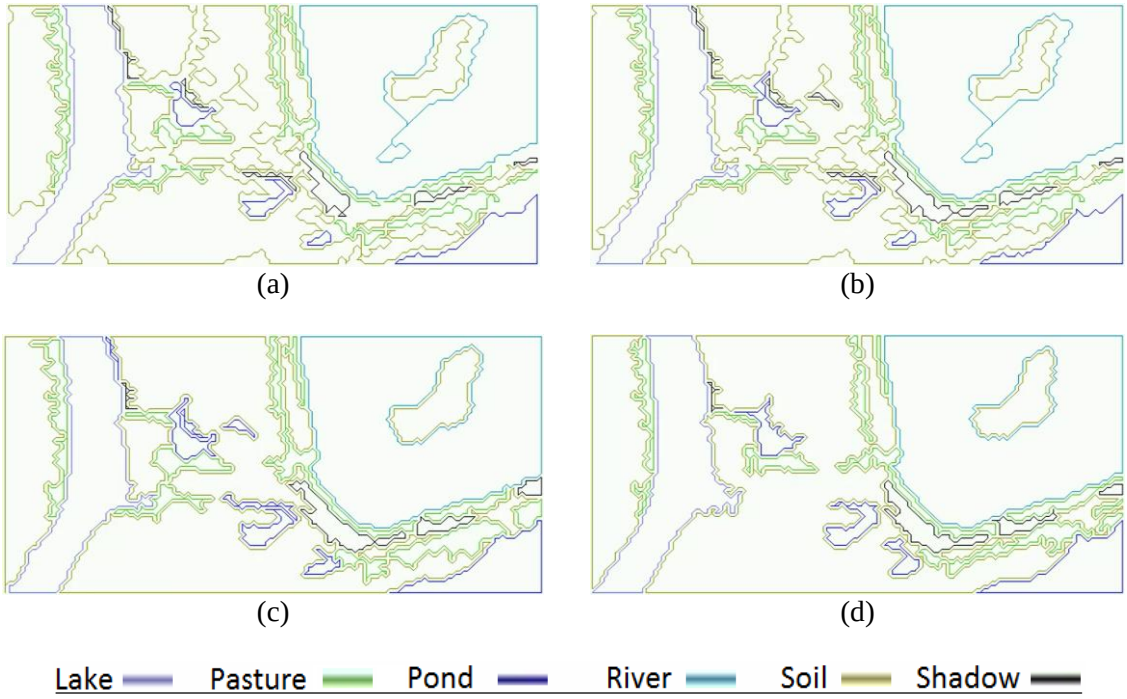


Figure Appendix A- 3. (a) and (b) display the extracted geo-objects after the growth and development steps, (c) displays the results of the construction step when all VAs become passive and (d) shows the results in the production step.

In the case that VAs utilise shrinking/splitting and shrinking/killing operators, the method uses an additional stage, called *edition*, to label remaining pixels in the image space (see the video clip2 in the source folder). In the production step, a VA may be removed if its size is less than a specific threshold.

i. **public void step ()**

The MakerAgent uses the following methods to control the VAs and generate the VAs at every time step.

ii. **public void generateVA ()**

The MakerAgent applies this method to create and place a VA in an appropriate location in the context.

iii. **public void generateVASubProcess ()**

This method creates a VA based on a coordinate checked by the generateVA method.

iv. **public void controlSeedPoint ()**

It controls the class of each seed point based on its coordinate in the image space.

v. **public void controlVA ()**

This method returns true if all VAs become passive in the context. In our example, we use red polygons to show the passive VAs (see the attached video clip).

A.2.1.1.3. VecAgent

The VecAgent class utilises the following variables and methods to change their geometry and states. This is performed through the geometry, state and neighbourhood components.

❖ ***Variables***

Each VA is specified based on the variables in Table Appendix A.5.

Table Appendix A- 5. Variables of each VA.

Variable name	Variable type	Value	Comment
spectralInfo	double [][]	…	Stores the DNs of each pixel that lies within each VA.
classNameInitial	int	…	Determines the class of each VA based only on the spectral information.
classNameFinal	int	…	Determines the class of each VA based on the spectral and geometric information of each VA. In our example, it is used by the VAs to classify the waterbody classes.
ID	int	…	It is a unique value for each VA.
isActive	boolean	true	In the birth event, it is true for each VA.
py	Polygon	…	Stores the geometry of each VA at each time step.
coordList	Vector		Used by the VAs to store coordinates evaluated. In each stage, it is redefined.

❖ ***Method***

i. **public void step ()**

This method is called by each VA in each time step.

ii. **public int getColor ()**

This method returns the colour of a VA.

iii. **public void setColor (final int classNameFinal)**

This method changes the colour of a VA.

iv. **public void update ()**

This method manipulates and updates the geometry and state of each VA at each iteration.

A.2.1.1.4. ContextCreator

ContextCreator is the class for running the proposed method. In our method, there is only one context for all agents.

❖ ***Variables***

This class only includes local variables.

❖ ***Method***

i. public Context<Object> build (final Context<Object> context)

This method provides a vector space in which VAs can change their geometry and state and upon which they can interact with each other and their environment. The method also has some functions that allow it to read and normalize a raster dataset, and solve an SVM problem.

A.2.1.2. Geometry

This component provides the classes that allow a VA to build and change its geometry or affect the geometry of another VA.

A.2.1.2.1. VAGeometryFactory

A VA uses this class to create a new point in every time step.

❖ ***Variables***

This class only includes local variables.

❖ ***Method***

i. double [] [] pointDisplacement (final VecAgent VA)

This method is used to create a new point based on a base vertex on a VA randomly selected. It returns the coordinate of the new point along with the index of the base vertex. It also determines the geometric operator for the VA through the VAEvaluateState class.

ii. public Point [] createCardinalPoints (final Coordinate coord)

The VAGeometry uses this method to create a set of points.

iii. public static Polygon createPolygon (final Coordinate coord)

This method creates a bounding box. The center point of the bounding box is specified by a coordinate.

A.2.1.2.2. VAGeometry

The VAGeometry class provides a set of ***constructive methods*** that allow a VA to geometrically interact with other VAs in the context. The VAGeometry class also includes some ***functional methods*** applied by the method that can test the structure of a VA based on the geometry rules (Chapter 3), edit the geometry of a VA, change order of vertices and to reduce the processing time.

❖ ***Variables***

This class only includes local variables.

❖ ***Constructive methods***

i. **public static Polygon unionTwoPolygonsMain (final Polygon geom1, final Polygon geom2, final double [][] coordNewPt)**

This method returns a newly generated polygon by merging two polygons into each other.

ii. **public static Polygon unionTwoPolygonsSubProcess (Polygon poly1, Polygon poly2, Coordinate coord)**

This method is applied by a VA to merge two polygons into each other.

iii. **public static Polygon [] shrinkSplitMain (final Polygon py, final Coordinate coordNewPt)**

An active VA uses this method to affect the geometry of a passive VA. In this case, the active VA removes a new vertex that lies on the polygon of the passive VA. If the new vertex is a duplicated vertex on the polygon of the passive VA, the active VA splits the polygon of the passive VA into two.

iv. **public static Polygon [] shrinkSplitSubProcess1(final Polygon poly, final Coordinate coordPoint, boolean onInteriorRing)**

It removes a vertex on a polygon or splits a polygon into two polygons.

v. **public static Coordinate [] shrinkSplitSubProcess2(final Coordinate [] coord, final Coordinate coordPoint, boolean onInteriorRing)**

It returns a list of coordinates that can construct a ring.

❖ ***Functional methods***

i. **public static boolean intersectionPolygonPoint (final Polygon poly, final Coordinate coord)**

It returns true if there is intersection between a point and a specific polygon.

ii. **public static boolean testIntersection (final Polygon py, final Coordinate coord, int ID, int ID1)**

The class uses this method to determine intersections between a collection of polygons. It returns true if there is an intersection between two polygons.

iii. **public static boolean intersectionPolygons (final Polygon py1, final Polygon py2)**

If two polygons intersect each other, it returns true. To do this, it also uses the intersectionPolygons1 method.

iv. **public static boolean controlGeometry (final Polygon poly)**

In the case that the distance between two successive vertices on a polygon is more than $r\sqrt{2}$, it returns false. r is defined based on the raster cell size.

v. **public static boolean qulifiedPoint (Point pt, int ID, double classNameInitial)**

The MakerAgent uses this method to check the location of a seed point. It returns false if a candidate point lies within an interior ring of a VA, and the VA has the same class.

vi. **public static Polygon controlInteriorRing (Polygon py)**

This method is applied by a VA to remove a vertex on an interior ring. A candidate vertex is removed if the distance between its successor and predecessor is less than or equal to $r\sqrt{2}$.

vii. **public static Coordinate [] removeDuplicatedPoint (Coordinate [] coord)**

It deletes a duplicate point if it is equal to its successor.

viii. **public static Polygon changePositionStartPoint (Polygon poly)**

VAs use this method to change the start point of a polygon.

ix. **public static Polygon elongationRatio (Polygon py)**

This method is used by the VAs to compute the elongation ratio.

A.2.1.2.3. VAPolygon

VAs use this class to update the py variable.

❖ ***Variables***

The VAPolygon uses the following variables to change the geometry of the VA at each time step.

Table Appendix A- 6. Variables of the VAPolygon.

Variable name	Variable type	Value	Comment
newPlusRandPoint	VAPoint[]	….	It is initialized through the VAGeometryFactory class
py	Polygon	…	It is initialized through the VecAgent class

❖ ***Method***

i. **public void manipulate ()**

This method allows a VA to change its geometry without interaction with other VAs.

ii. **Polygon generateInteriorRing (Polygon py1, Polygon py2, Coordinate [] coord, Coordinate coordinate)**

This method compares two polygons that have intersected at a specific coordinate to generate a new interior ring.

iii. **public double [] [] sort4PointsClockwise (final Coordinate [] coord)**

This method is used to sort a set of points including a new point, a random point and its successor and predecessor. To do this, the method uses the coordinates of these points.

A.2.1.2.4. VALineString

The aim of this class to create a new line constructed formed by two half-edges.

- ***Variables***

  This class only includes local variables.

- ***Method***

i. **public static Polygon halfEdgeJoiningMain (Polygon pyInput, Coordinate coorNewPt)**

This method change the geometry of the VA based on the half-edge rule in a specific coordinate.

ii. **public static Polygon halfEdgeJoiningVertices (Polygon pyInput)**

This method controls and changes the geometry of the VA based on the half-edge rule for all vertices of a polygon.

A.2.1.2.5. VAPoint

The aim of this class is to create a new point based on a base point randomly selected via the VAGeometryFactory.

- ***Variables***

  Table Appendix A.7 shows the variable of the VAPoint.

Table Appendix A- 7. Variable of the VAPoint.

Variable name	Variable type	Value	Comment
pt	Point	….	It is constructed by the VAPoint.

❖ ***Method***

i. **public VAPoint createRandomPointCardinalDirection (final Point point, final int direction)**

The VAGeometryFactory uses this method to create a new point if a base point lies in an exterior ring.

ii. **public VAPoint createRandomPointDiagonalDirection (final Point point, final int direction)**

In the case that a base point lies in an exterior ring, the VAGeometryFactory applies this method to create a new point.

A.2.1.3. State

VAs utilise the State component to find and update transition rules.

A.2.1.3.1. VAFindState

VAs use this class to find their state.

❖ ***Variables***

This class only includes local variables.

❖ ***Method***

i. **public static int findState (final Coordinate coord)**

The MakerAgent uses this method to identify the class of a seed point.

A.2.1.3.2. VAEvaluateState

The aim of this class is to assess the class of a dependent (belonging to a VA) or an independent pixel.

❖ ***Variables***

This class only includes local variables.

❖ ***Method***

i. **public static String [] evaluatePixel (final VecAgent VA, final Coordinate coord)**

The method uses the SVM model to assess the class of a dependent (belonging to a VA) or independent pixel.

ii. **public static void addPixel (final VecAgent VA, final Coordinate coord)**

A VA uses this method to update its spectralInfo variable based on a specific coordinate.

iii. **public static void removePixel (final VecAgent VA, final Coordinate coord)**

This method is used by a VA to delete the DNs of a pixel from its spectralInfo variable based on a specific coordinate.

iv. **public static void updateSpectralinfoVA (final VecAgent VA)**

A VA employs this method to update its spectralInfo variable based on the py variable.

v. **public static void updateSpectralinfoCombinedPolygon (final VecAgent vec1, final VecAgent vec2)**

A VA uses this method to update its spectralInfo variable after merging process.

A.2.1.3.3. VAUpdateTransitionRules

The aim of this class to update the SVM model based on the VA-generated samples. This class is also applied by the method to classify the waterbody objects.

- ***Variables***

This class only includes local variables.

- ***Method***

i. **public static void updateSVM ()**

It is used to update the SVM model based on the VA-generated samples. The method first applies the *featureExtractor*, *selectVA*, *findBestVA functions*, *featureExtractor*, *featureExtractor1* and *randomizeTable* to select the most informative VA samples. Next it randomly selects a number of pixels, specified based on the Number_Elgible_Pixels parameter, to train the SVM classifier.

ii. **public static void classifyWaterBody (VecAgent vecAgent)**

The aim of this method is classify the waterbody VAs based on their geometric information.

A.2.1.4. Neighbourhood

The elements of this component enable the VAs to perceive and interact with each other.

A.2.1.4.1. VAFindAdjacent

This class is applied by a VA uses to find its neighbours.

- ***Variables***

  This class only includes local variables.

- ***Method***

i. public static int findAdjacentObjects (final Coordinate coord, final int id)

A VA uses this method to find its neighbours based on a specific coordinate.

A.2.1.4.2. VANeigbourhoodRules

The aim of this class is to implement neighbourhood rules defined by a user. For example, a pond object is a shadow object if it is next to a tree object.

- ***Variables***

  This class only includes local variables.

- ***Method***

i. public static void implementNeighbouringRules (VecAgent vecAgent)

A shadow VA is converted into a pond VA if it cannot find an elevated feature, such as tree. Here, the pasture VAs are considered as an elevated object.

A.2.1.5. Utility

This component is composed of a set of utility classes applied by the proposed method in order to read a file and solve the SVM model.

A.2.1.5.1. NewExcel

This class is used by the proposed method to read the image converted into an Excel file.

- ***Variables***

  The following variables are applied by the NewExcel Class to read a file.

Table Appendix A- 8. Variables of the NewExcel class.

Variable name	Variable type	Value	Comment
inputFile	String	…	Initialized via the ContextCreator class.
num	int	…	Initialized via the ContextCreator class.
data	double[][]	…	Initialized via the NewExcel class.

❖ ***Method***

i. **public void setInputFile (final String inputFile, final int num)**

This method sets the address of the input datasets.

ii. **public double [] [] read ()**

This method reads the input datasets.

A.2.1.5.2. SVM

This class is used by the VA model to formulate the transition rules.

❖ ***Variables***

This class only includes local variables.

❖ ***Method***

i. **public static svm_model svmTrain (final double [] [] train, final double gamma, final double c)**

This method is applied to train the SVM model.

ii. **public static double [] evaluate (final double [] features)**

The VA model uses this method to assess the class of a candidate pixel.

iii. **public static double [] comapreProbability (final double [] features, final double classname1, final double classname2)**

In the case that two VAs have different classes, an active VA model uses this method to assess a pixel belonging to a passive VA.

A.2.1.5.3. GridSearch

The aim of this class is to find the optimum values for the gamma and C parameters in the RBF kernel applied by the SVM classifier.

❖ ***Variables***

The following variables are applied by the GridSearch.

Table Appendix A- 9. Variables of the GridSearch class.

Variable name	Variable type	Value	Comment
classifier	LibSVM	…	
dataset	Dataset	…	
folds	int	…	
cv	CrossValidation	…	
bestAccuracy	double	…	
bestC	double	…	
bestGamma	double	…	
C	double []	…	
gamma	double []	…	
svmParameters	svm_parameter	…	

❖ ***Method***

i. **public GridSearch (final LibSVM classifier, final Dataset dataset1, final int folds)**

This method provides a grid of parameter values specified based on the C [] and gamma [].

ii. **private void crossValidation (final Integer CIndex, final Integer gammaIndex)**

This method uses the crossValidation algorithm to find the best value for the C and gamma parameters.

B- VA model for GEOBIA image classification

Supporting data for the application of the VA model for the GEOBIA is available in the attached file. In this example, we used a subset of multispectral IKONOS image and LiDAR DSM dataset to extract five different classes, building, meadow, road, shadow and tree, from (Figure 6.1). The framework of this method is developed in Repast using an Intel CPU running at 3.40 GHz with 16 GB of memory. The VA model took 432 seconds to classify the image.

C- VA model for extraction 3D roofs

Supporting data for the application of the VA model for extraction and classification of 3D roofs is available in the attached file. In this example, we used the LiDAR DSM dataset of 251× 251 pixels with 20cm resolution LiDAR DSM (Figure 7.3(b)) which covers an urban area in Zeebrugge, Belgium (GRSS, 2015). The method is implemented using Repast and powered by an Intel CPU running at 3.40 GHz with 16 GB of memory in 330 Seconds.