Representation of and reasoning with vagueness in spatial information

A system for handling vague objects

Areti Dilo

Promotor:

Prof. Dr. Ir. Alfred Stein
Professor in the department of Earth Observation Science
International Institute for Geo-Information Science and Earth Observation,
and
Mathematical and Statistical Methods, Wageningen University

Co-Promotor:

Dr. Ir. Rolf A. de By
Associate Professor in the department of Geo-information Processing
International Institute for Geo-Information Science and Earth Observation

Examining Committee:

Prof. Dr. Ir. Peter van Oosterom, Delft University of Technology, the Netherlands
Prof. Dr. Ir. Johan Grasman, Wageningen University, the Netherlands
Prof. Dr. Andrew Frank, Vienna University of Technology, Austria
Prof. Dr. Ing. Manfred Ehlers, University of Osnabrück, Germany

This research is carried out within the C.T. de Wit Graduate School for Production Ecology and Resource Conservation (PE&RC) in Wageningen University, the Netherlands.

Representation of and reasoning with vagueness in spatial information

A system for handling vague objects

Areti Dilo

THESIS

to fulfil the requirements for the degree of Doctor
on the authority of the Rector Magnificus of Wageningen University
Prof. Dr. M.J. Kropff
to be publicly defended on Thursday 15 June 2006 at 15:00 hrs
in the auditorium at ITC, Enschede, The Netherlands

Representation of and reasoning with vagueness in spatial information:
A system for handling vague objects

ISBN: 90-8504-461-8
International Institute for Geo-information Science & Earth Observation (ITC)
Enschede, the Netherlands
ITC Dissertation Number: 135

To my parents and my sister
Milo, Irini, and Elida

Abstract

Vagueness is often present in spatial information. It arises in the presence of borderline cases. It is difficult to decide, for example, whether the land cover at a location is forest or grassland. The transition from one class to another is gradual. Their boundary cannot be represented by a crisp line. Fuzzy sets allow the representation of gradual transitions.

Geographic information systems (GIS) and spatial databases offer functionality for storing and analysing spatial information. Current GIS's or spatial databases can only handle objects with crisp boundaries. The objective of this research is to provide a system of types and operators that can handle vague spatial objects.

First, we formally define vague types and operators using mathematical notions, to assure definitions without ambiguity. The data types that we propose are a set of simple types, a set of general types, and vague partitions. The simple types represent identifiable objects of a simple structure, i.e., not divisible into components. We distinguish between vague points, vague lines, and vague regions. The general types represent classes of simple type objects. They are vague multipoint, vague multiline, and vague multiregion. General types assure closure under set operators. Simple and general types are defined as fuzzy sets in $\mathbb{R}^2$ satisfying specific properties that are expressed in terms of topological notions. These properties assure that set membership values change mostly gradually, allowing stepwise jumps. The type vague partition is a collection of vague multiregions that might intersect each other only at their transition boundaries. All the vague types that we propose include crisp objects as special cases.

We propose a set of operators to reason with vague objects. Operators are defined on general types and vague partitions. They are divided into three groups: operators returning spatial types, spatial relations, and metric operators. The first group consists of regularized fuzzy set operators: union, intersection, and difference; two operators from topology: boundary and frontier; and two operators on vague partitions: overlay and fusion. Spatial relations for vague objects follow the ISO standard model for spatial relations. They extend the true/false values of the ISO relations to the $[0, 1]$ interval, which means the truth of a relation is a matter of degree. The proposed relations are disjoint, touches, crosses, overlaps, within, and equal. They have the property that only one relation is certain at a time. The metric operators that we provide are distance, length, area, diameter, and perimeter.

We propose two different measures, alpha operators and average operators. An alpha operator on vague objects returns for every α in $(0,1]$ the value of the analogous crisp operator applied to the α-cuts of the objects. An average operator produces an average over the returned values of the corresponding alpha operator for all α values. We provide two other measures on vague objects, centroid and vagueness degree. All the operators are equivalent to their crisp analogues when applied to crisp objects.

Then, we implement the proposed types and a subset of the operators in GRASS, an open source GIS software package, using its functionality for vector data formats. Points, lines, and triangulations are used to store vague points, vague lines, and vague regions, respectively. Classes of simple vague objects are stored in layers. A vague partition is stored as a theme of vague region layers, which form a soft classification of space. Classes of a theme are bound together via relations stored in database tables. We offer several modules to create layers of vague objects from input data points, to visualize vague objects and display information about them, and modules that perform union, intersection, and difference of vague object layers.

The vague types and operators allow a representation of vagueness in static spatial information, and reasoning in such a setting. In the final part of the thesis we consider the possibilities for extending the system of types and operators to handle dynamic vague information. A dynamic vague type is defined from each (static) vague type as a function from the time domain to that static vague type. Consequently, the set of operators is extended towards dynamic operators. A dynamic operator applied on dynamic vague objects returns for any moment of their existence the result of the corresponding static operator on the states of the objects. Functionality offered by spatiotemporal databases can be used for the storage and manipulation of dynamic vague objects. Dynamic objects are stored in these databases as a collection of states at discrete moments of time. We propose linear functions to calculate the state of a vague object between two consecutive moments of stored object states.

We use real applications to illustrate the main concepts of the research. Illustrations throughout the thesis are created from data on heavy metal concentration in the sediments of the Maas river in Belgium. We employ our operators to monitor land cover change in Shimla district in India, and beach erosion in Ameland island in the Netherlands.

Samenvatting

Ruimtelijke informatie heeft vaak een vaag karakter. Vaagheid treedt bijvoorbeeld op in grensgevallen. Zo is het soms moeilijk om te besluiten of de landbedekking op een bepaalde locatie 'bos' is of 'grasland', omdat de overgang van de ene naar de andere klasse geleidelijk is. Een scherpe lijn vormt dan geen realistische grens. Onduidelijke ('fuzzy') verzamelingen staan een representatie toe van geleidelijke overgangen.

Geografische informatie systemen (GIS) en ruimtelijke gegevensbestanden bieden functionaliteit voor het opslaan en het analyseren van ruimtelijke informatie. De huidige generaties GIS en ruimtelijke gegevensbestanden kunnen echter alleen omgaan met objecten met scherpe grenzen. Het doel van dit onderzoek is het opzetten van een systeem van types en operatoren die kunnen omgaan met vage ruimtelijke objecten.

We zullen eerst zowel vage types als operatoren hierop formeel definiëren. Hierbij maken we gebruik van wiskundige begrippen om dubbelzinnigheid te vermijden. De gegevenstypen die we voorstellen om vage informatie te representeren kunnen we verdelen in een verzameling simpele types, algemene types en vage partities. De simpele types representeren identificeerbare objecten van een eenvoudige structuur, d.w.z. zodanig dat ze niet in deelsystemen zijn onder te verdelen. We onderscheiden vage punten, vage lijnen en vage gebieden. De algemene types betreffen klassen van simpele objecten. Het zijn een vaag multipunt, een vage multilijn en een vaag multigebied. Algemene typen zijn gesloten onder verzamelingoperatoren. Simpele en algemene types zijn gedefinieerd als vage verzameling in $\mathbb{R}^2$ met lidmaatschapfuncties die aan een aantal eigenschappen voldoen; deze eigenschappen zijn geformuleerd in termen van topologische begrippen. De lidmaatschapfuncties zijn veelal continu, met uitzondering van stapsgewijze toenames van lidmaatschapwaarden. Het type 'vage partitie' betreft een verzameling vage multigebieden die elkaar enkel snijden in de overgangsgebieden. Alle gedefinieerde vage types hebben een equivalent hard object als een speciaal geval.

Vervolgens stellen we een verzameling operatoren voor die geschikt zijn voor redeneringen met vage objecten. Deze operatoren zijn gedefinieerd op algemene types of op vage partities. Ze worden in drie grepen verdeeld: operatoren die resulteren in een ruimtelijk type, die een ruimtelijke relatie vaststellen en die een metrische relaties vaststellen. De eerste groep beslaat drie operatoren die geregulariseerd zijn voor vage verzamelingen (vereniging,

doorsnede, verschil), twee operatoren uit de topologie (grens en front) en twee operatoren voor vage partities (bedekking en versmelting). Ruimtelijke relaties voor vage objecten volgen het ISO standaardmodel voor ruimtelijke relaties. Ze veralgemeniseren de Waar/Onwaar uitkomsten van de ISO relaties naar waarden in het $[0,1]$ interval, waarmee de waarheid van een relatie een gradueel karakter krijgt. De voorgestelde relaties zijn 'disjunct', 'raakt', 'kruist', 'valt samen', 'binnen' en 'gelijk aan'. Ze hebben de eigenschap dat als één relatie Waar is, dat dan alle andere relaties Onwaar zijn. De metrische operatoren die we voorstellen zijn 'afstand', 'lengte', 'oppervlak', 'diameter' en 'omtrek'. We stellen twee maten voor: α-operatoren en gemiddelde-operatoren. Een α-operator voor een vaag object levert als uitkomst een waarde voor iedere a in het $(0,1]$ interval op en is analoog aan de scherpe operator op α-sneden van objecten. Een gemiddelde-operator levert een gemiddelde op over de uitkomsten van de corresponderende α-operatoren voor alle waarden van α. Voor vage objecten definiëren we twee operatoren: de centroide en de mate van vaagheid. Deze operatoren zijn equivalent aan hun scherpe analogieën bij harde objecten.

De voorgestelde typen en een deelverzameling van de operatoren erop hebben we vervolgens geïmplementeerd in GRASS, een GIS pakket dat vrijelijk beschikbaar is. Hierbij hebben we gebruik gemaakt van de functionaliteit voor vectoren als gegevensstructuur. We hebben punten, lijnen en triangulaties gebruikt om respectievelijk vage punten, vage lijnen en vage gebieden op te slaan. Klassen van simpele vage objecten van de typen vaag multipunt, vage multilijn en vaag multigebied worden opgeslagen in aparte lagen. Een vage partitie is opgeslagen als een thema van vage gebiedsklassen, die gezamenlijk een zachte classificatie van de ruimte vormen. Thematische klassen worden gebundeld door middel van relaties die zijn opgeslagen in tabellen van gegevensbestanden. We presenteren verschillende modules om a) lagen van vage objecten te genereren uit puntobjecten, b) vage objecten te visualiseren en informatie erover weer te geven, en c) een vereniging, doorsnede en verschil van lagen met vage objecten uit te voeren. De vage typen en operatoren laten zowel een representatie van vaagheid in ruimtelijke informatie toe als een redeneersysteem hiermee.

In het laatste onderdeel van het proefschrift beschouwen we de mogelijkheden om het systeem uit te breiden naar typen en operatoren voor dynamische informatie. Een dynamisch vaag type is gedefinieerd op basis van de individuele (statische) vage types, en wel als een functie vanuit het tijdsdomein naar de verzameling van vage objecten van dat type. Dit leidt dan tot een uitbreiding van de verzameling operatoren naar de verzameling van dynamische operatoren. Een dynamische operator toegepast op dynamische vage objecten resulteert in de uitkomst van de corresponderende statische operator op de toestanden van de objecten op ieder moment van hun bestaan. De functionaliteit van ruimtelijk-temporele gegevensbestanden kan dan gebruikt worden voor de opslag en de bewerking van dynamische vage objecten. Dynamische objecten slaan we in zulke databases op als een verzameling toestanden op discrete momenten in de tijd. We stellen lineaire functies voor om de toestand van een vaag object tussen twee opeenvolgende momenten van de opgeslagen toestanden van een object te berekenen.

Er is gebruik gemaakt van reële data om de belangrijkste concepten of het onderzoek te illustreren. Afbeeldingen in dit proefschrift maken gebruik

van gegevens over de concentraties zware metalen in de sedimenten van het Belgische deel van de rivier de Maas. De ontwikkelde operaties zijn toegepast bij het monitoren van verandering in landgebruik in het Shimla district in India en stranderosie op Ameland in Nederland.

Përmbledhje

Informacioni hapësinor është shpesh i vagët, gjë që duket në ekzistencën e zonave kufitare. Vështirësi hasen kur, për shembull, duhet të përcaktojmë llojin e mbulimit të tokës (land cover) në një zonë të caktuar, nëse ajo është një zonë pyjore apo shkure. Kjo vështirësi vjen nga fakti se kalimi nga një zonë në tjetrën është gradual dhe mungon një vijë qartësisht e dallueshme që të mund të përfaqësojë kufijtë ndarës ndërmjet tyre. Bashkësitë 'fuzzy' lejojnë përfaqësimin e zonave tranzitore me kalime graduale.

Për ruajtjen dhe analizimin e informacionit hapësinor (të kompjuterizuar) përdoren funksionet e ofruara nga Sistemet e Informacionit Gjeografik (SIGJ) dhe bazat e të dhënave hapësinore. Në stadin aktual këto teknologji mund të punojnë vetëm me objekte hapësinore në me kufij ndarës të qartë. Nisur nga ky status, qëllimi i këtij studimi shkencor është që të dizenjojë një sistem tipesh te dhënash dhe operatoresh, i cili do të mund të punojë edhe me objekte hapësinore të vagëta.

Së pari, tipet dhe operatorët për objektet e vagëta janë përkufizuar formalisht mbi bazën e nocioneve matematikore, për të përjashtuar ambiguitetin. Tipet e të dhënave që propozohen në këtë studim janë grupuar në:'tipe të thjeshta', 'tipe të përgjithshëm' dhe 'ndarje të vagëta'. Grupi i parë përfaqëson objekte të identifikueshëm me një strukturë të thjeshtë, domethënë objekte të cilat nuk mund të ndahen në komponentë përbërës. Objektet që i takojnë këtij grupi janë pika të vagëta, vija të vagëta dhe rajone të vagëta. Grupi i tipeve të përgjithshëm përfaqëson klasa objektesh e thjeshtë (të grupit të parë). Objekte të këtij grupi janë multipika të vagëta, mulitilinja të vagëta dhe multirajone të vagëta . Tipet e përgjithshëm sigurojnë mbylljen ndaj operatorëve të bashkësive, domethënë rezultati i këtyre operatorëve është një objekt i një tipi të përgjithshëm. Tipet e thjeshtë dhe të përgjithshëm janë përcaktuar si 'fuzzy sets' në $\mathbb{R}^2$ me funksione antarësie që plotësojnë veti specifike të shprehura në nocione topologjie. Këto veti sigurojnë që funksionet të jenë pothuaj kudo të vazhdueshëm duke lejuar ndryshime të menjëhershme ne vlerat e antarësimit. Ndarjet e vagëta janë një koleksion multirajonesh të vagëta të cilat mund të priten ndërmjet tyre vetëm në zonat kufitare. Të gjithë tipet e vagëta të lartpërmendura përfshijnë objektet me kufi të qartë (krisp) si raste të veçanta.

Studimi propozon një bashkësi operatorësh për të punuar me objektet e vagëta. Operatorët janë ndarë në tri grupe: operatorë që kthejnë tipe hapësinorë, relacionet hapësinore dhe operatorë metrikë. Grupi i parë përbëhet

nga operatorët e bashkësive 'fuzzy': bashkimi, prerja dhe diferenca; dy operatorë nga topologjia: zona ndërkufitare dhe kufiri frontal; dhe dy operatorë për ndarjet e vagëta: mbivendosja dhe shkrirja. Relacionet hapësinore për objektet e vagët bazohen në modelin e standarteve ISO për relacionet hapësinore. Ato e zgjerojnë gamën e vlerave 'true'/'false' të relacioneve të përcaktuara nga ISO në intervalin [0, 1], që do të thotë se vërtetësia e një relacioni është çështje shkallëzimi. Relacionet e propozuara janë: të ndarë, prek, kalon, pritet, brenda, të njejtë. Këto relacione kanë vetinë që vetëm një prej tyre mund të jetë i sigurtë. Operatorët metrikë që përcaktohen në këtë studim për objektet e vagëta janë: distanca, gjatësia, sipërfaqja dhe perimetri. Për secilin prej tyre propozohen dy masa të ndryshme: operatorët α dhe operatorët e mesëm. Një operator α në objekte të vagëta kthen për çdo α në $(0, 1]$ vlerën e operatorit analog krisp të aplikuar në α-prerjet e objekteve. Një operator i mesëm krijon një mesatare mbi vlerat e kthyera nga α operatori korrespondues. Studimi ofron gjithashtu edhe dy masa të tjera për objektet e vagëta që janë shkalla e paqartësisë dhe e qendra e rëndesës. Të gjithë operatorët janë equivalentë me analogët e tyre krisp kur aplikohen në objektet krisp.

Tipet e propozuar dhe disa prej operatorëve janë implementuar më pas në GRASS, një paketë programi SIGJ, duke përdorur funksionet që ajo ofron për të dhënat në format vektorial. Pikat, linjat dhe triangulacionet janë përdorur për të ruajtur respektivisht informacionin mbi pikat, linjat dhe rajonet e vagëta. Klasat e objekteve të thjeshta janë ruajtur në shtresa të veçanta informacioni. Një ndarje e vagët është ruajtur si një tematikë klasash me rajone të vagëta, dhe formon një klasifikim 'të butë' të hapësirës. Klasat e një tematike janë të lidhura së bashku nëpërmjet lidhjeve të ruajtura në tabelat e bazës së të dhënave. Studimi ofron disa module për të krijuar shtresa objektesh të vagët nga të dhënat fillestare në format pike, një modul për të vizualizuar objektet e vagët dhe informacionin e tyre shoqërues, si dhe disa module të cilët kryejnë bashkimin, prerjen dhe differencen e shtresave që përmbajnë objekte të vagët.

Tipet dhe operatorët e vagët të lartpërmendur lejojnë përfaqësimin e informacionit hapësinor statik, si dhe mundësojnë analizen e tij. Në pjesën e fundit të studimit janë shqyrtuar mundësitë për zgjerimin e sistemit të tipeve dhe operatorëve edhe për trajtimin e informacionit të vagët dinamik. Një tip dinamik i vagët është përcaktuar prej çdo tipi statik të vagët si një funksion nga domeni kohor për në bashkësinë e objekteve të vagëta që i përkasin atij tipi statik. Në vazhdim, bashkësia e operatorëve statikë është zgjeruar në drejtim të operatorëve dinamikë. Një operator dinamik i aplikuar në objektet e vagëta dinamike kthen per çdo moment të egzistencës së tyre rezultatin e operatorit statik korrespondues ne gjendjet e objekteve në atë moment. Funksionet e ofruara nga bazat e të dhënave hapësinore-kohore mund të përdoren për ruajtjen dhe manipulimin e objekteve dinamikë të vagët. Objektet dinamike janë ruajtur në këto baza të dhenash si nje koleksion gjendjesh në një moment të caktuar kohor. Studimi propozon funksionet lineare për të llogaritur gjendjen e objektit të vagët ndërmjet dy momenteve të njëpasnjëshme të objekteve me të cilët po punohet.

Acknowledgements

In the four (and a quarter) years of my PhD, many people were helping with different things at different times. I will mention in this acknowledgement only people that were directly related to the PhD work.

The first to thank are my promotor, prof. dr. Alfred Stein, and co-promotor, dr. Rolf de By. At times, I didn't have the same view of things as Alfred, but at the end we would always reach to a good agreement. A special thanks to him for the willingness and acceptance. The thanks to Rolf for the detailed comments and the scrutiny when going through my writings.

This PhD was part of the REV!GIS project. Thanks to Nic Wilson and Robert Jeansoulin for sharing their ideas about my work. Also, to all junior and senior REV!GIS people for the pleasant work meetings, and the enjoyable after work evenings.

The implementation part of this work was started from Pawalai Kraipeerapun during her MSc thesis at ITC. The work was completed in its major part by Pieter Bos and Pawalai a year later. A thanks to Pieter for the agility and the confidence during the work. Thanks to Pawalai for her careful thinking and her wish to help. In the supervision I was helped by Wim Bakker. Thanks to Wim for the willingness to help.

The first talks about the 'which direction?!' and 'are these results to be discussed seriously?' were with Nirvana Meratnia, who knows well how to listen and encourage. The most frequent discussions were with Daniel van de Vlag, with whom I shared the office for most of the PhD time. Thanks to Daniel for providing the data, and the talks about the data and the application. Also, for the most directly indirectly related help, solving all the administration issues with the Dutch system. Blanca Perez was doing the scripting in Visual Basic when I needed some specific processing in ArcGIS. Thanks to Blanca also for preparing the full set of data that I needed at the last stage. I had talks with a few people about the meaning and use of my research in real applications. Thanks to David Rossiter, Uday Nidumolu, and Robert Strobl for their time.

Following somehow a chronological order, the last people to thank are those who helped in the preparation of this book. I got the (LaTeX) template from Javier Morales, together with explanations how to use and change it where needed. Thanks to Javier for the previous, and for answering later the unpredicted questions. Thanks to Arben Brahimaj for designing the cover of the book, to Enkela Begu for translating the abstract into Albanian, and to Etleva Llagami for the translation of the specific terms. My colleagues

at the new job helped with few other things. Thanks to Friso Penninga for making the last additions to the Dutch abstract, to Wilko Quak and Martijn Meijers for the last minute layout improvements of the book.

The last words of thanks are certainly for the friends. No name mentioning, it would be a long list. I have been in ITC for a long time, and it is a great place for making friends. Without the friends it would have been much harder. Perhaps impossible?!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Spatial information is important to earth scientists, like geologists, soil scientists, ecologists, land use specialists. Also, the work of other scientists, specialists, and organizations relies on geographic information. Forestry departments, civil engineers, urban planners, cadastral agencies, utility companies, epidemiologists, police departments, they all use spatial data to describe and understand the processes that they are dealing with. They take decisions based on the analysis of spatial data. The validity of these decisions depends on the quality of the data, which makes it an important aspect of (spatial) data.

To allow an effective use of the data it is necessary to know its quality. Several factors affect quality, and may cause imperfections in the data. There exist real world entities that cannot be delineated precisely, e.g., 'forest' or 'medium shrubs'. Limitations of the measuring instruments are reflected in the quality of the acquired data. The error in the input data propagates by means of computations that are performed to obtain a required result. Quality degradation can also arise from inappropriate data representation models.

Deficiencies in data quality lead to different kinds of imperfection. Attempts to deal with imperfection [69, 89, 90] often propose general taxonomies relating different kinds and causes of imperfection, e.g., incompleteness, imprecision, inconsistency. The importance of such taxonomies is not so much that they accurately characterize the nature of imperfection, but more that they allow distinctions to be made between different kinds of imperfection [68]. This has led to the development of different formalisms, each intended to capture a particular nuance of imperfection.

According to Smets [88, 89], the main aspects of imperfect data are *imprecision*, *inconsistency*, and *uncertainty*. Imprecision and inconsistency are properties of the data: either more than one world or no world is compatible with the available information, respectively. If, on top of imprecision, we attach weights to the worlds to express our opinion about which might be the actual world, then we are confronted with uncertainty. Uncertainty concerns the state of our knowledge; it is a property of the relation between the available information and our knowl-

edge about the world. Each of these major categories reveals different nuances of imperfection. For example, incompleteness and vagueness are particular kinds of imprecision.

Worboys [112] distinguishes different kinds of imperfection based on factors causing deficiencies in spatial data quality. Imperfection may be *inaccuracy* and *error*: deviation from true values, *incompleteness*: lack of relevant information, *inconsistency*: conflicts arising from the information, *imprecision*: limitation on the granularity or resolution at which the observation is made or the information is represented, and *vagueness*: imprecision in concepts used to describe the information [112].

It is a complex problem to consider together all factors that affect data quality. Different kinds of imperfection are treated separately by different theories. In this thesis, we deal only with vagueness. It is a kind of imperfection that is often present in spatial data. We consider vagueness to be a special kind of imprecision.

1.1 Problem statement

Vagueness is a type of imperfection arising in the presence of borderline cases [92]. A concept is vague if locations exist that cannot be classified unambiguously either to the concept or to its complement. When mapping vegetation for example, it may be difficult to decide whether a certain location belongs to one vegetation class or to another. The transition from one class to another may be gradual, as between forest and grassland in African rangelands. Also, geomorphological units [14, 63], soil types [27, 66], land cover classes [53], and forest types [11], generally exhibit transition zones instead of sharp boundaries. Other examples include soil pollution classes in environmental applications [54], or hydrological studies [9] when spatial objects have to be delineated that cannot be sharply defined.

Several theories have been proposed to handle vagueness, of which fuzzy theory [113, 115] is the most often used. This is also true in the spatial domain. Attention has been given to vagueness, on the one hand by considering spatial objects to be crisp, and reasoning being vague [104, 105]. On the other hand, models have been proposed for objects that are vague [13, 22, 41, 79, 80, 97, 116]. There is also work done for the extraction of fuzzy objects from satellite imagery [64], and their visualization [55].

Some image processing software allow the extraction of fuzzy objects from imagery, or offer tools to do fuzzy reasoning over a continuous space [74]. No functionality exists, however, for handling vague objects in current geographical information systems (GIS), or in spatial database systems. These systems focus on crisp spatial objects. Current GIS's or spatial databases would therefore ben-

efit from an extension with data types for vague spatial objects, and operators for their manipulation.

Theoretical models have been proposed earlier, mainly dedicated to vague regions and their topological relations. These models can be grouped into two approaches. One approach [13, 41, 79] considers vague regions to have a homogeneous two-dimensional boundary instead of a one-dimensional boundary. Locations in the broad boundary all have the same degree of membership to the region. Models of this approach do not provide means to handle gradual transition. The other approach [80, 97, 116] takes gradual changes into consideration and employs fuzzy set theory for modelling purposes. From this group, Schneider's work [80, 82, 83] is the most complete, covering object types and basic operators for a spatial system [46].

1.2 Objectives of the research

The aim of this research is to provide a system of types and operators that allow representation of vagueness in spatial information, and reasoning in its presence. Fuzzy theory is our choice for dealing with vagueness. We want our system to be able to represent and analyse vague spatial phenomena at a certain moment of time. This is the main focus of the research. We then see how this system can be extended, so that it can handle the dynamics of vague phenomena.

The main objectives of the research are:

- to provide a formal system for vague spatial types and operators,
- to implement the vague types and operators using capabilities of existing GIS software,
- to extend the vague types and operators in the time domain, so that they can handle dynamic spatial phenomena.

Before constructing an information system, a precise notion must exist of what is to be built, and one must specify how the system is expected to function. Mathematical specifications are useful in this respect. They precisely describe the required properties of an information system, without unduly constraining how these properties are achieved in its implementation. This can be done by means of mathematical data types and functions for modelling the data and functionality of a system. These types and functions are then not oriented towards computer representation, but obey a rich collection of mathematical laws. As such, one can reason effectively about the way a specified system will behave.

This is the reasoning behind our first objective. It assures correctness of types and operators that we propose for handling vague spatial information. To assure

completeness of our system, we check the list of operators against the ISO standard for spatial data, SQL/MM spatial [1], and an abstract spatial database model, the ROSE algebra [50]. We define the vague spatial types and operators in such a way that they include as special cases the proposed types and operators [23, 50] for crisp objects.

1.3 Structure of the thesis

The results of the research are presented in the chapters composing this thesis. Chapter 2 sets the background for this work. It explains what is *vagueness*, and lists the main theories proposed to handle it, giving more attention to the *fuzzy theory*. It describes the main components of spatial and temporal database systems, focusing more on *types and operators* elements of a spatial data model by taking two examples of such models, the SQL/MM spatial part and the ROSE algebra. It also summarizes the mathematical concepts from general *topology*, fuzzy sets, and *fuzzy topology* that we need for our formal definitions of types and operators. Chapter 3 provides for *vague spatial types*, which are divided into simple and general types representing simple objects and classes of objects, respectively. It also provides for *operators returning spatial types*, e.g., union, intersection. Chapter 4 provides for *spatial relations* between vague objects, which are built upon the intuition of SQL/MM spatial relations between crisp objects. Chapter 5 provides *metric operators* for vague objects, e.g., length, area, distance. We implement the vague types and set operators using existing functionality in GRASS, an open source GIS software package. Chapter 6 is dedicated to this implementation. Chapter 7 explains how the system can be extended with *temporal types and operators*. It then performs the analysis of two (dynamic) spatial applications using operators of the system. Chapter 8 concludes the work of this thesis.

We use different fonts to distinguish between important notions. LucidaCasual font is used for the names of crisp types and operators. *LucidaCasual-Italic* font is used for names of vague types and operators. *LucidaCalligraphy-Italic* is used for dynamic vague types and operators. LucidaFax is used for GRASS data structures, and `LucidaSans-Typewriter` is used for GRASS modules, directories, and files.

Chapter 2

Setting the scene

This chapter describes the theories that constitute the foundation for this thesis work. The discussion is organized in four parts. Section 2.1 explains the concept of vagueness, and summarizes different theories proposed to handle it, putting more attention to advantages and disadvantages of fuzzy theory that we choose for handling vagueness. As the main objective of this work is the design of a spatial system that can handle vagueness, it is important to know the basic components of a spatial system. Section 2.3 describes shortly the main components of a spatial database system, and focuses on spatial types and operators. It summarizes SQL/MM Spatial, the ISO standard for spatial systems, as well as the ROSE algebra, the theoretical model of an implemented spatial database system. At the end, it describes the main concepts of a temporal database system. Formal definitions we provide for vague spatial types and operators are based on notions from general and fuzzy topology. Notions of general topology are also needed for the description of spatial systems; they are presented in Section 2.2. Concepts from fuzzy sets and fuzzy topology are presented in Section 2.4.

2.1 Vagueness and theories to handle it

Vagueness is at the heart of problems known as the Sorites paradox. The name comes from the old Greek $\sigma\omega\rho o\varsigma$ (heap), and the problem is originally known as *The Heap* puzzle [44, 57]. Is one grain of wheat a heap? The answer is 'No'. Are two grains of wheat a heap? No. ... One must admit the presence of a heap sooner or later, so where should the line be drawn? "Nowadays the Sorites paradox identifies a class of paradoxical arguments, also known as little-by-little arguments, caused by the indeterminacy surrounding the limits of the application of the predicates involved" [57].

A current dispute on vagueness concerns its ontological or linguistic nature [102]. Vagueness may either be an inherent property of the phenomenon described by a vague term, or our language is such that it vaguely describes phenomena. Several

theories have been proposed to handle vagueness: epistemic theory [91, 110], supervaluationism [31, 43], rough sets [71], and many-valued logics [99, 115]. The best known of the later are three-valued logic and fuzzy logic. Epistemic theory and supervaluationism suggest that vagueness is linguistic, whereas many-valued logics suggest that vagueness is ontological [57, 102].

The epistemic theory considers vagueness to be a kind of ignorance [91, 110]. The indeterminacy associated to a vague term comes from our inability to determine its exact reference (extension) [102]. The vagueness of the term 'heap' is a matter of ignorance — there exists a cut-off value n which defines the boundary between heap and not heap, except that we do not know it.

The basic idea underlying supervaluationism is that a vague term is one that admits various alternative 'sharpenings' [102]. Supervaluationism says that the truth value of a statement involving vague terms is a function of its truth values under the various admissible sharpenings of those terms [102]. If the statement is true under all such sharpenings, then it is super-true. If it is false under all the admissible sharpenings it is super-false. If the statement comes true under some sharpenings and false under some others, then the statement suffers a truth value gap.

Rough sets have the same view on vagueness as supervaluationism. A rough set can be thought of as a collection of classical sets, each approximately being the set that we want. It is represented by a pair of classical sets called the lower and upper approximation. The lower approximation consists of all elements that certainly belong to the set, the upper approximation consists of all elements that possibly belong to the set [70].

Many-valued logic is a term that designates a group of theories proposed to handle vagueness. A first proposal was three-valued logic, with a truth set equal to {1 (true), 1/2 (indeterminate), 0 (false)}. A general concern with a three-valued logic approach to vagueness is its tripartite division of statements. There is no more ground for supposing the existence of a sharp boundary between true and indeterminate statements, or between indeterminate and false statements, than there was for supposing a sharp boundary to exist between true and false statements [57]. No finite number of divisions seems adequate.

Infinite-valued logic or fuzzy logic overcomes this limitation. It provides a wide range of degrees between false and true values. The truth of a statement is a matter of degree. The truth values are from a totally ordered infinite set, i.e., there is an order between any two elements of the set. Other problems, however, exist with the fuzzy approach to vagueness: the assumption of a totally ordered set of truth-values is problematic. How does the degree to which a certain location belongs to a forest compare to the degree to which a certain borderline rock is part of Mount Everest? Also, a point must exist if one moves from full fledged truth to partial truth, or from partial truth to full fledged falsehood [102]. These problems are carried over in our model for spatial vagueness.

2.2 Basic concepts from general topology

The description of object types and operators of a spatial system needs concepts like single-component, boundary, adjacent regions. General topology offers formal descriptions (i.e., definitions) of these concepts. This section presents the topology concepts that are used for the definition of crisp spatial objects and their operators, as well as concepts needed for our definitions of vague types. The concepts that are directly used for the definitions of crisp or vague objects are emphasized in the text. The treatise is not restricted to only these concepts, as their explanation requires other topology notions. To help the intuition for each concept provided, we give the definition and important properties. The latest are sometimes equivalent definitions of a concept.

A general topology can be defined for any set X, e.g., real numbers $X = \mathbb{R}$, or functions in reals $X = \{f : \mathbb{R} \to \mathbb{R}\}$. Because we work with $\mathbb{R}$, $\mathbb{R}^2$, and $\mathbb{R}^3$, we give the definitions in $\mathbb{R}^n$ space. Most of the definitions are indeed valid for any space X, only few are specific to real spaces. We assume that basic notions from classical set theory are known. We start with metric spaces in $\mathbb{R}^n$, as the topology we work with is a metric topology. The basic topology notions, open and closed sets, are first defined in a metric space, because they are easily understandable in such a setting, and remain the same for the topology we use. Topology operates on a more abstract level than a metric space, and it offers a richer set of concepts together with their interrelations. Some of the topology concepts, not being dealt with in a metric space, are needed for our definitions. Therefore, we provide the full set of (needed) concepts in a topology setting, including the open and closed sets.

A *metric* on $\mathbb{R}^n$ is a function $\rho : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^+$, such that for any $p, q, r \in \mathbb{R}^n$

1. $\rho(p, q) = 0$ if and only if $p = q$,
2. $\rho(p, q) = \rho(q, p)$ (symmetry),
3. $\rho(p, q) + \rho(q, r) \geq \rho(p, r)$ (triangle inequality).

The pair $(\mathbb{R}^n, \rho)$ is called a metric space. The function $\mathsf{d}_n : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^+$, defined as $\mathsf{d}_n((x_1, \ldots x_n), (y_1, \ldots y_n)) = \sqrt{(x_1 - y_1)^2 + \ldots + (x_n - y_n)^2}$ is a metric on $\mathbb{R}^n$ [109]. It is the Euclidean distance, and is called the usual metric on $\mathbb{R}^n$.

Let $(\mathbb{R}^n, \rho)$ be a metric space, p a point of $\mathbb{R}^n$, and ϵ a positive real number. The set $U_\rho(p, \epsilon) = \{q \in \mathbb{R}^n \mid \rho(p, q) < \epsilon\}$ is called the ϵ-disk about p. A set G is open in $(\mathbb{R}^n, \rho)$ if for each $p \in G$ there is an ϵ-disk about p contained in G. A set F is closed if it is the complement of an open set [109]. Figure 2.1(a)–(c) illustrates an ϵ-disk about p, an open set G, and a closed set F in $(\mathbb{R}^2, \mathsf{d}_2)$, respectively. Many sets are neither open nor closed, as, for example, the sets of Figures 2.2(a), 2.3(a), 2.4(a) and 2.4(b).

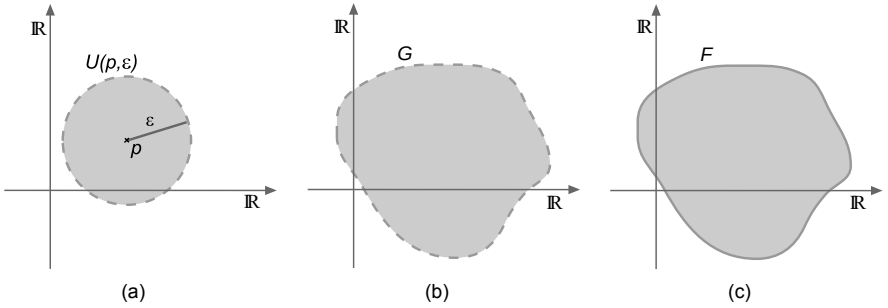

Figure 2.1: Open and closed sets in $(\mathbb{R}^2, \mathrm{d}_2)$: (a) an ϵ-disk about p, (b) an open set G, (c) a closed set F. A dashed line symbolizes that points on the line are not elements of the set enclosed by the line.

The open sets in a metric space $(\mathbb{R}^n, \rho)$ have the following properties: the empty set $\emptyset$ and $\mathbb{R}^n$ are both open, any finite intersection of open sets is an open set, and any union of open sets is an open set [109]. These properties are used to generalize the notions of open and closed sets without employing a metric. A topology T for $\mathbb{R}^n$ is a collection of subsets of $\mathbb{R}^n$, such that (i) $\emptyset, \mathbb{R}^n \in T$, (ii) if $U, V \in T$, then $U \cap V \in T$, (iii) if for an index set J, possibly infinite, a family $\left\{U_j\right\}_{j\in J} \subseteq T$, then $\bigcup_{j\in J} U_j \in T$ [109]. The pair $(\mathbb{R}^n, T)$ is called a topological space. The elements of T are called open sets. A set F is said to be closed (in the topology T) if its complement F^C is an open set. Intersection of closed sets is a closed set, and a finite union of closed sets is also a closed set.

Different topologies can be built for $\mathbb{R}^n$. The trivial topology, containing only the empty set $\emptyset$ and the whole $\mathbb{R}^n$, is an example topology for $\mathbb{R}^n$. Another example is the discrete topology, which is the family of all subsets of $\mathbb{R}^n$. Any subset of $\mathbb{R}^n$ is an open set for the discrete topology. If $(\mathbb{R}^n, \rho)$ is a metric space, its open sets form a topology for $\mathbb{R}^n$. Such a topology is called a metric topology. The metric topology formed by the usual metric is called the *usual topology* on $\mathbb{R}^n$.

A topology is constructed from a base. A base of a topology T is a subfamily $B \subset T$ such that each member of T is the union of members of B. For example, the family of ϵ-disks $\{U_{\mathrm{d}_n}(p, \epsilon) \mid p \in \mathbb{R}^n, \epsilon \in \mathbb{R}^+\}$ is a base for the usual topology in $\mathbb{R}^n$. The family of open intervals $\{(a, b) \mid a, b \in \mathbb{R}\}$ is the base for the usual topology for $\mathbb{R}$. Another topology is constructed for $\mathbb{R}$ from a base consisting of half-open intervals $[a, b) = \{x \in \mathbb{R} \mid a \leq x < b\}$. Elements of this topology are both open and closed, called clopen.

We denote by T_n the usual topology for $\mathbb{R}^n$. The usual topologies for $\mathbb{R}$ and $\mathbb{R}^2$ are the topologies employed for the definitions of spatial objects. From this point onward we refer to the usual topology for $\mathbb{R}^n$, though definitions are independent of the chosen topology. Most of the illustrations in this section are for sets in $\mathbb{R}^2$ with the usual topology T_2, except for the continuity illustration, which is done for functions in $\mathbb{R}$ with the usual topology T_1.

A subset of $\mathbb{R}^n$ is also said to be a subset of the topological space $(\mathbb{R}^n, T_n)$. A subset U_p of $(\mathbb{R}^n, T_n)$ is called neighbourhood of a point p if U_p contains an open set to which p belongs [59]. A subset of a topological space is open if and only if (iff) it contains a neighbourhood of each of its points. A point p of a subset A of $(\mathbb{R}^n, T_n)$ is called an interior point of A if A is a neighbourhood of p. The point p in Figure 2.2(a) and its neighbourhood U_p are inside the set A, thus p is an interior point of A. The set A° of all interior points of A is called the *interior* of A [59]. The interior of A is equal to $A^\circ = \bigcup \{G \subseteq \mathbb{R}^n \mid G \in T_n \wedge G \subseteq A\}$. It is the largest open set contained in A. A set A is open iff it is equal to its interior, i.e., $A = A^\circ$. Figure 2.2(b) shows the interior A° of the set A of Figure 2.2(a).

The closure $\overline{A}$ of a subset A in $(\mathbb{R}^n, T_n)$ is the intersection of all closed sets containing A: $\overline{A} = \bigcap \{F \subseteq \mathbb{R}^n \mid F^C \in T_n \wedge A \subseteq F\}$. It is the smallest closed set containing A. A point q is called an accumulation point of a subset A of $(\mathbb{R}^n, T_n)$ if every neighbourhood of q contains points of A other than q. The points q and r in Figure 2.2(a) are accumulation points of the set A: any neighbourhood, U_q and U_r, contains points of A different from q and r, respectively. The point q is an element of the set A, the point r is not in A. The closure of A is the union of A with the set of its accumulation points [59]. A set A is closed iff it is equal to its closure, $A = \overline{A}$. Figure 2.2(c) shows the closure $\overline{A}$ of the set A of Figure 2.2(a). A sequence in $A \subset \mathbb{R}^n$ is a (partial) function from the natural numbers $\mathbb{N}$ to A, and is denoted by $(p_i)_{i \in \mathbb{N}}$ or simply (p_i). The sequence (p_i) converges to $p \in \mathbb{R}^n$ if for each neighbourhood $U_p \subset \mathbb{R}^n$ of p there is $m_U \in \mathbb{N}$ such that, $k \geq m_U$ implies $p_k \in U_p$ [5]. A point p belongs to the closure of A iff there is a sequence (p_i) in A converging to p.

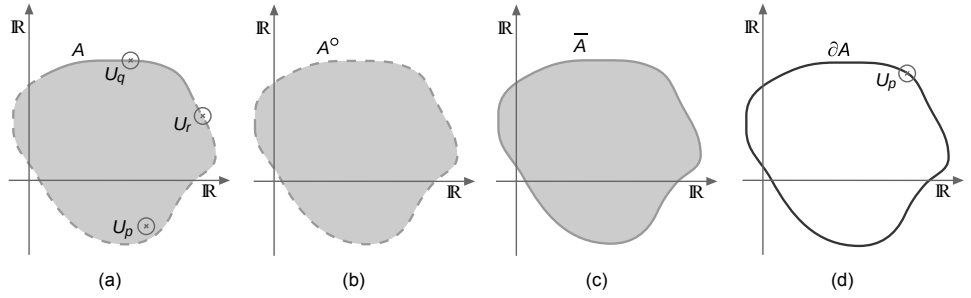

Figure 2.2: Interior, closure, and boundary in $(\mathbb{R}^2, T_2)$: (a) set A in $\mathbb{R}^2$, (b) its interior A°, (c) its closure $\overline{A}$, (d) its boundary ∂A.

The *boundary* ∂A of a subset A in $(\mathbb{R}^n, T)$ is the set of all points that are interior neither to A nor to its complement. Equivalently, p is a point of the boundary of A iff each neighbourhood of p intersects both A and its complement A^C [59]. Figure 2.2(d) shows the boundary ∂A of the set A of Figure 2.2(a). The neighbourhood U_p of the point p in the boundary of A intersects both, the set A and its complement. The boundary of A is $\partial A = \overline{A} \cap \overline{A^C} = \overline{A} - A^\circ$. The boundary of a set A is closed, and it is equal to the boundary of its complement A^C.

A set A is *regularly closed* iff it is equal to the closure of its interior, i.e., $A = \overline{A^\circ}$. The set A of Figure 2.3(a) is not regular closed, while the set of Figure 2.3(d) is regular closed. The set A has punctures, i.e., removed points, it has a cut, i.e., a removed curve, and a dangle curve. The closure $\overline{A}$ shown in Figure 2.3(b) removes the cut and the punctures. Any neighbourhood U_p of a point p in the cut has points from the set A, and is therefore an accumulation point, which means it is in the closure $\overline{A}$. The same is true for any point that is a puncture. Any neighbourhood U_q of a point q in the dangle curve has points from the complement of A. That means there is no neighbourhood U_q that is completely in A, therefore q is not an interior point of A. Figure 2.3(c) shows the interior A° of the set of Figure 2.3(a). The closure of the interior of A is shown in Figure 2.3(d).

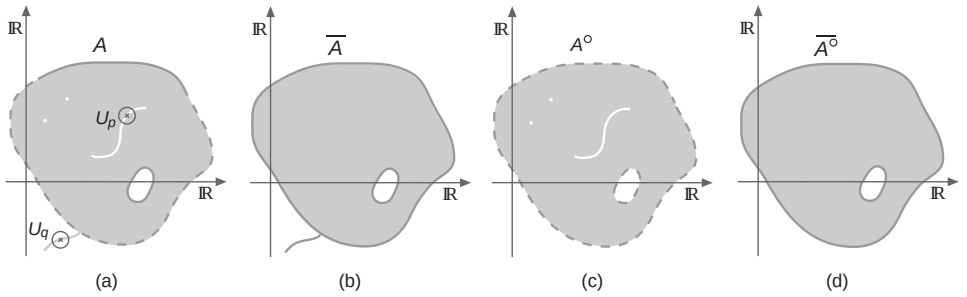

Figure 2.3: Regular closure for $(\mathbb{R}^2, T_2)$: (a) a set A in $\mathbb{R}^2$, (b) its closure $\overline{A}$, (c) its interior A°, (d) the closure of the interior $\overline{A^\circ}$.

Let A be a subset of $(\mathbb{R}^n, T_n)$. The family T_n^A of all intersections of members of T_n with A forms a topology, which is called the relative topology of T_n to A. All the concepts defined for a topological space in $\mathbb{R}^n$ can be transferred to any subset of it, associated with the relative topology. That is to say we can define a concept in $(\mathbb{R}^n, T_n)$ or in a subset of it associated with the relative topology.

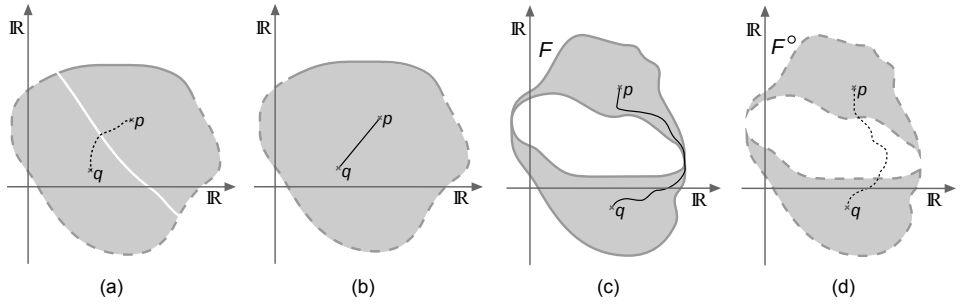

Figure 2.4: Connected and disconnected sets for $(\mathbb{R}^2, T_2)$: (a) a disconnected set (b) a connected set, (c) a connected closed set F (d) the interior of F is disconnected.

Two subsets A and B of $(\mathbb{R}^n, T_n)$ are separated iff $A \cap \overline{B}$ and $\overline{A} \cap B$ are both

empty. A topological space is *connected* iff it is not the union of two non-empty separated subsets [59]. An equivalent definition for connection is: a set A is connected if for any two points in the set there is a path connecting the two and lying completely in the set A. The set of Figure 2.4(a) is disconnected: there is no path connecting the two points p and q that lies completely in the set. For any two points of the set of Figure 2.4(b) there is a path connecting them and lying inside the set. The set of Figure 2.4(b) is connected, as it is also the set F of Figure 2.4(c). The path connecting the two points p and q of the set F lies partially inside the set and partially in the boundary of F that is part of F because the set is closed. The interior of F shown in Figure 2.4(d) is not connected because there is no path connecting the points p and q, which lies completely in F°. A *component* of a topological space is a maximal connected subset, i.e., a connected subset that is not properly contained in any other connected subset. The set of Figure 2.4(d) has two components, one containing the point p and the other containing the point q. A subset A of $(\mathbb{R}^n, T_n)$ is *bounded* iff there is an ϵ-disk about the origin O that contains the set: $A \subseteq U(O, \epsilon)$.

Let X and Y be subsets of $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively, (X, T_n^X) and (Y, T_m^Y) be the relative topologies for X and Y. A function f from X to Y is said to be *continuous* at $x_0 \in X$ if for each neighbourhood V of $f(x_0)$ in Y, there is a neighbourhood U of x_0 in X such that $f(U) \subseteq V$ [109]. The function f is *continuous on* X if f is continuous at every $x \in X$. Figure 2.5(a) shows the graph of a continuous function $f : \mathbb{R} \to \mathbb{R}$. A continuous function preserves proximity, i.e., close points in the domain are mapped to close points in the range. A *homeomorphism*, or topological transformation, is a continuous and bijective function h from the topological space (X, T_n^X) to the topological space (Y, T_m^Y), such that its inverse function f^{-1} is also continuous. The two spaces, (X, T_n^X) and (Y, T_m^Y), are said to be homeomorphic or topologically equivalent [59]. A homeomorphism preserves properties of sets: if X is open, closed, connected, or bounded, the image Y by the homoeomorphism h has the same properties. Properties preserved by a homeomorphism are called topological invariants.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is *lower semi-continuous* at p with respect to the usual topologies in $\mathbb{R}^n$ and $\mathbb{R}$, iff for all $c < f(p)$ a neighbourhood U_p of p exists such that for all $q \in U_p$, $c < f(q)$. The function f is lower semi-continuous in $\mathbb{R}^n$ iff it is lower semi-continuous at every $p \in \mathbb{R}^n$ [58]. Figure 2.5(b) shows the graph of a lower semi-continuous function $g : \mathbb{R} \to \mathbb{R}$. The function is almost everywhere continuous, except for three points x_0, 0, and x_1 where it is lower semi-continuous.

Correspondingly, a function $f : \mathbb{R}^n \to \mathbb{R}$ is *upper semi-continuous* at p with respect to the usual topologies in $\mathbb{R}^n$ and $\mathbb{R}$, iff for all $c > f(p)$ a neighbourhood U_p of p exists such that for all $q \in U_p$, $c > f(q)$. A function f is upper semi-continuous in $\mathbb{R}^n$ iff it is upper semi-continuous at every p [58]. Figure 2.5(c) shows an upper semi-continuous that is the characteristic function of the closed interval $[a, b]$. The function $\chi_{[a,b]}$ is continuous in $\mathbb{R}$ except for the points a and b where it is upper semi-continuous. The characteristic function of a closed

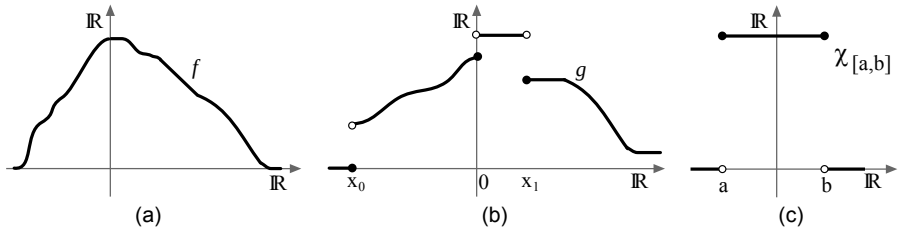

Figure 2.5: Graphs of continuous and semi-continuous functions for the usual topology in $\mathbb{R}$: (a) a continuous function, (b) a lower semi-continuous function, (c) an upper semi-continuous function that is the characteristic function of an interval $[a, b]$. Empty and filled circles are used to show the value of a function at discontinuity points; the full circle denotes the value of the function.

set in $(\mathbb{R}^n, T_n)$ is an upper semi-continuous function, whereas the characteristic function of an open set is a lower semi-continuous function. A function $f : \mathbb{R}^n \to \mathbb{R}$ is continuous at a point p iff it is both upper and lower semi-continuous at p.

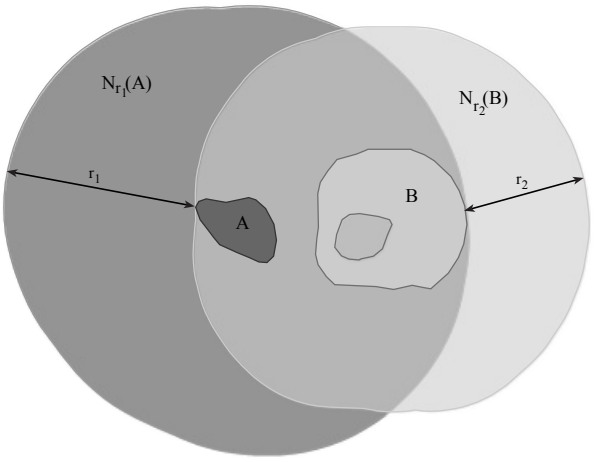

Figure 2.6: The smallest neighbourhood $N_{r_1}(A)$ containing the set B, and the smallest neighbourhood $N_{r_2}(B)$ containing A.

Hausdorff devised a metric between subsets of a metric space, known as the *Hausdorff distance*. Two sets are within Hausdorff distance r from each other if any point of one set is within distance r from some point of the other set, and viceversa. Let A be a subset of the metric space $(\mathbb{R}^n, \mathsf{d}_n)$, and $r > 0$. A neighbourhood of A with radius r is defined as $N_r(A) = \{p \in \mathbb{R}^n \mid \exists q \in A, \mathsf{d}_n(p, q) < r\}$. The Hausdorff distance between two sets A and B in $\mathbb{R}^n$ is $\mathsf{HD}(A, B) = \inf\{r \in \mathbb{R}^+ \mid A \subset N_r(B) \text{ and } B \subset N_r(A)\}$ [33]. Figure 2.6 shows two sets A and and B in $\mathbb{R}^2$, together with their neighbourhoods $N_{r_1}(A)$ and $N_{r_2}(B)$. The neighbourhood $N_{r_1}(A)$ is a buffer around A with the smallest distance r_1 such that any point of B is contained in it. $N_{r_2}(B)$ is built similarly to contain any point of A. The Hausdorff distance between A and B is the maximum of radiuses r_1 and r_2. The

Hausdorff distance HD is a metric for the set of bounded and closed subsets of $\mathbb{R}^n$ [33].

2.3 Spatial and spatiotemporal database systems

A database is a large, computerized collection of data, whereas a database management system (DBMS) is a software package that allows a user to set up, use, and maintain a database [26]. A DBMS offers a data modelling language, indexing and join methods, authorization, integrity constraints, concurrency, and transactions. Important elements of a data modelling language are a collection of types together with their operators (functions). They are used by the data manipulation and query language, which is offered to applications for the storage and analysis of their data. Indices are used by the query optimizer for efficient performance of queries.

Spatial and temporal database systems are both extensions of standard DBMS's. A spatial DBMS is extended with data structures and algorithms for computation over spatial types, together with spatial indexing techniques, and extension of the optimizer for mapping from the query language to the spatial components [51]. A temporal DBMS extends the data modelling language with temporal concepts that are inserted deeply in the data model, in order to assure efficient query performance. Spatiotemporal databases are a combination of structures and techniques from spatial databases and temporal databases.

This thesis works only with data types and operators, therefore this section puts attention to only these elements of a data model. The ROSE algebra [50], a theoretical model of an implemented spatial system [49], and the ISO standard for spatial data [1] were compared and used to compile the list of types and operators for vague objects. Section 2.3.1 discusses the types and operators offered by the ROSE algebra, and Section 2.3.2 discusses the ISO SQL/MM spatial standard for types and operators. Section 2.3.3 discusses the basic notions of a temporal database system.

2.3.1 Spatial systems: ROSE algebra

The ROSE algebra is an abstract spatial model for a DBMS with additional capabilities for the representation and management of spatial data [50]. It consists of formal definitions of spatial data types and their operators fulfilling the following criteria:

(i) closure property, i.e., the return type of any operator is one of the system's data types,

(ii) rigour, i.e., unambiguous definitions,

(iii) types are defined in terms of finite representations available on computers,

(iv) geometric consistency for related objects, e.g., adjacent regions have common boundary.

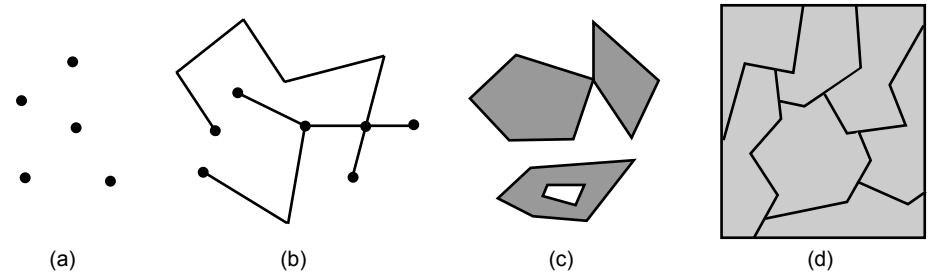

Figure 2.7: ROSE algebra spatial object types: (a) MultiPoint, (b)MultiLinearString, (c) MultiPolygon, (d) Partition.

Data types of the ROSE algebra are a collection of standard data types: boolean, string, integer, real, as well as spatial types: points, lines, regions, together with two super-types *EXT* and *GEO*. For easy comparison with ISO standard types we rename the ROSE spatial types to MultiPoint, MultiLinearString, MultiPolygon, Extension, and Geometry, respectively. A value of the data type MultiPoint represents a set of disjoint points, illustrated in Figure 2.7(a). A value of the type MultiLinearString represents a set of piecewise linear curves forming a planar graph, illustrated in Figure 2.7(b). A value of the type MultiPolygon represents a set of disjoint regions possibly with holes (Figure 2.7(c)). Standard and spatial types are the basic data types, and are the leaves of the tree in the complete type hierarchy [40] given in Figure 2.8. The non-leaf nodes are super-types used by the operators. A spatial object is a value of type Geometry. Another type, very important for classifications of space, is the type Partition. A partition is a collection of MultiPolygon objects that form a disjoint subdivision of space, illustrated in Figure 2.7(d).

The operators of the ROSE algebra are divided into four groups: (i) spatial predicates, (ii) operators returning spatial types, (iii) operators returning numbers, and (iv) operators on collections of spatial objects. In the next paragraphs we use the terms multipoint, multiline, and multipolygon, for an object of type MultiPoint, MultiLinearString, and MultiPolygon, respectively.

Spatial predicates

Spatial predicates compare two spatial objects with respect to their topological relation and return a boolean value. The predicates Disjoint, Equal, and

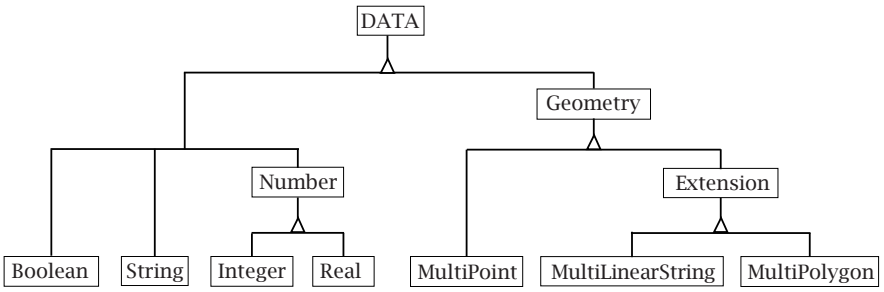

Figure 2.8: Hierarchy of types in the ROSE algebra. A triangle shows sub-type relation.

Not_Equal between two spatial objects return true if the point sets of the objects are disjoint, equal, and not equal, respectively. The predicate Inside between a Geometry object and a multipolygon is true if the Geometry object is a subset of the multipolygon. Area_disjoint and Edge_disjoint between two multipolygons occur when object interiors are disjoint, but they have common boundary lines or points, respectively. Edge_inside and Vertex_inside between two multipolygons occur when they are in an Inside relation and their boundaries have in common at most a non empty finite set of points, or nothing, respectively. Intersects between two Extension type objects is true if two multilines intersect at points; a multiline is partially inside the multipolygon; and two multipolygons share a region part. Meets between two Extension objects is true if two multilines have common end nodes; a multiline has common isolated points with the boundary of a multipolygon; and boundaries of two multipolygons have common isolated points. Adjacent between two multipolygons occurs when their interiors are disjoint, but they share part of the boundary, i.e., a line in common. Encloses between two multipolygons occurs when the second object is inside a hole of the first one. On_border_of between a multipoint and an Extension type objects occurs if the multipoint is an end node of a multiline, or on the boundary of a multipolygon. Border_in_common between two Extension objects occurs when boundaries of multipolygons and/or multilines share a common part. The topological relations corresponding to these operators are not mutually exclusive. Figure 2.11 shows their interrelations, as well as the correspondence with the SQL/MM spatial relations.

Operators returning spatial types

The second group consists of operators returning spatial values as results. The operators Plus, Minus between any two spatial objects, and Intersection between two spatial objects of the same type (e.g., between two multipolygons) assure the closure property of the ROSE algebra. They are regularized set operators, union, difference, and intersection, respectively. The Common_border operator between two Extension objects returns the common (boundary) multiline.

The Vertices operator on an Extension object returns multipoint that are vertex points of a multiline, and vertex points of the boundary of a multipolygon. The Contour operator on a multipolygon returns its boundary that is a multiline. The Interior operator is applied on a multiline and returns the multipolygon enclosed by the multiline.

Operators returning numbers

Operators of the third group return numbers. The No_of_components operator on a spatial object returns the number of its components, which is an integer value. The Dist operator calculates the (minimal) distance between any two spatial objects, returning a real value. The Diameter of a spatial object is calculated as the largest distance between any of its locations. The Length operator calculates the total length of segments of a multiline. The Area operator computes the sum of the areas of the components of a multipolygon. The Perimeter operator calculates the sum of lengths of the boundary lines of a multipolygon.

Operators on collections of spatial objects

Operators of the last group take collections of spatial objects as operands; some of them create new collections as a result. The Sum operator aggregates over the values of some spatial attribute of an object set, and computes the geometric union of all these values. The Closest operator returns that object from a collection that is the nearest to some reference object. For every composite object, the Decompose operator gives the set of its connected components. The Overlay operator superimposes two partitions of the plane. The result is a new partition with regions obtained from the intersections of a region of the first partition with a region of the second partition. The Fusion operator dissolves a partition (by merging regions) based on the equality of some attribute value of regions.

2.3.2 Spatial systems: ISO SQL/MM spatial

ISO/IEC 13249-3 SQL/MM Part 3: Spatial [1] is the international standard that defines how to store, retrieve and process spatial data using SQL. It defines how spatial data is to be represented as values, and which functions are available to convert, compare, and process this data in various ways [95]. The standard was originally derived from the OpenGIS Simple Feature Specification for SQL [23] by the OpenGIS Consortium (now Open GeoSpatial, OGC). The Simple Feature Specification defines a geometry model consisting of a class hierarchy shown in Figure 2.9. The geometry model is an abstract model used to define relation-

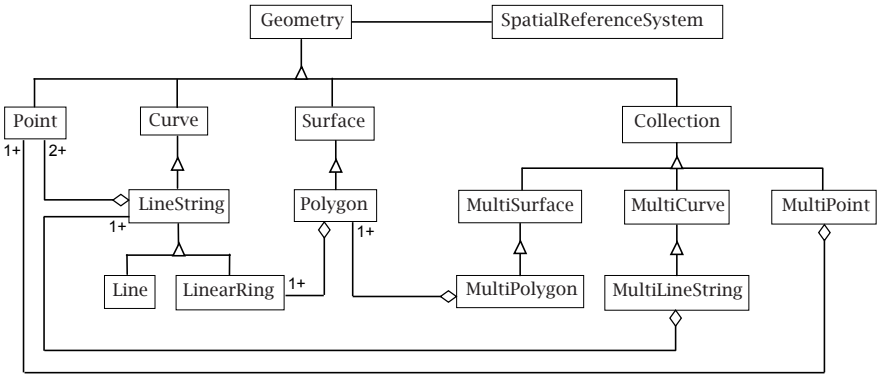

Figure 2.9: OGC *Geometry* class hierarchy. Diamonds show aggregation, triangles show sub-class.

ships between classes and inheritance rules for methods (functions) working on instances of classes and subclasses.

The type hierarchy defined in the SQL/MM Spatial standard is adapted from the OGC class hierarchy. Figure 2.10 shows the SQL/MM type hierarchy; the shaded types are not instantiable. SQL/MM standard uses the prefix ST_ for all types and functions. The types ST_Point, ST_Curve, ST_Surface, and their subtypes represent single component objects. Type ST_GeomCollection and its subtypes represent multi component objects. ST_Geometry represents a spatial object of any type.

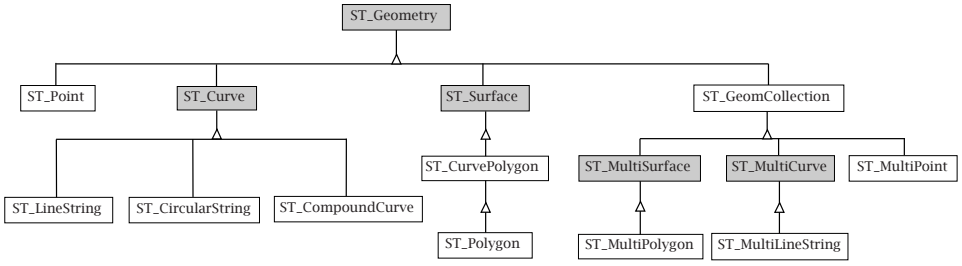

Figure 2.10: SQL/MM Spatial type hierarchy.

The SQL/MM functions (methods) are basically the same functions proposed by OGC [23]. They can be grouped into one of the following categories [95]: (i) conversion between geometries and external data formats, (ii) retrieval of properties or measures from a geometry, (iii) comparison of two geometries with respect to their spatial relation, (iv) generation of new geometries from others. There are a few more functions not falling in any of the above categories, e.g., ST_Distance between two ST_Geometry values. Functions of the first group allow conversions between three different external data formats defined by the SQL/MM standard.

Operators returning properties or measures of spatial objects

Functions of the second group ST_Dimension and ST_GeometryType return the dimension and type of an ST_Geometry object, respectively. ST_IsEmpty function tests if an ST_Geometry object is an empty set. The function ST_IsClosed tests if an ST_Curve object is a closed line. Functions returning measures of ST_Geometry values are ST_Length on ST_Curve or ST_MultiCurve, ST_Area on ST_Surface or ST_MultiSurface, ST_Perimeter on ST_Surface or ST_MultiSurface. The function ST_NumGeometries on ST_GeomCollection returns the number of simple objects in the collection.

Spatial relations

Spatial relations are identified using the 9-intersection model [39] extended by the dimension of the intersections, a model called DE-9IM [23]. The generic ST_Relate method defined for this model enables a large number of spatial relations to be tested. However, it is a low level building block and does not have a corresponding natural language equivalent [1]. For this reason, commonly used spatial relations have been specified: ST_Disjoint, ST_Touches, ST_Crosses, ST_Overlaps, ST_Within, ST_Equals, ST_Intersects, and ST_Contains. The last two relations can be expressed in terms of others: the relation ST_Intersects is the negation of ST_Disjoint, and the relation ST_Contains is the same with ST_Within. The first six relations are mutually exclusive, and provide a coarse but complete covering of all the topological cases. They are binary operators applied to geometries of either the same or different dimension, and return a truth value. Each of the relations can be expressed in terms of a corresponding DE-9IM pattern. Their definitions and illustrations are provided in Section 4.1.1 because we follow this model to define vague spatial relations.

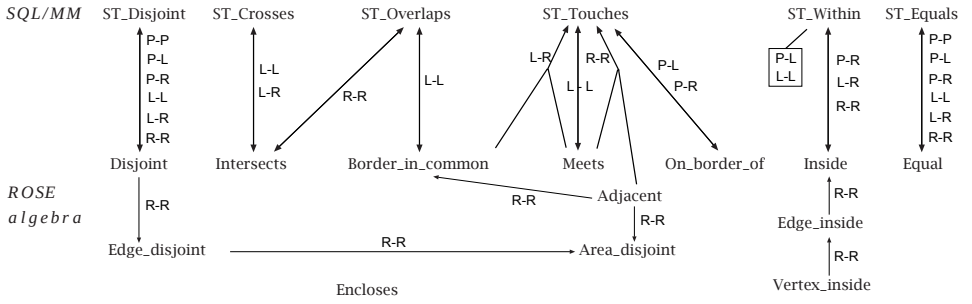

Figure 2.11: The correspondence between SQL/MM and ROSE algebra spatial relations. Labels P, L, R, denote multipoint, multiline, and multiregion, respectively. A P-P label denotes a multipoint–multipoint relation. The other labels are to by interpreted similarly. A double arrow shows full correspondence; a one-side arrow shows inclusion.

Figure 2.11 shows the correspondence between SQL/MM spatial relations and

ROSE spatial predicates, together with interrelations of ROSE predicates. For example, relations Equaland ST_Equals are the same; relations Adjacent and Meets between regions are covered by ST_Touches; the relation ST_Within between points and lines and between lines is not expressed by any ROSE algebra relation; the Encloses relation is not expressed by any of the six spatial SQL/MM relations.

Operators returning spatial types

Operators of the last group return spatial types. ST_Union, ST_Intersection, ST_Difference, and ST_SymDifference return a new ST_Geometry object that is the point set union, intersection, difference, and symmetric difference of two ST_Geometry objects, respectively. ST_Boundary returns the closure of the combinatorial boundary[1] of an ST_Geometry value. ST_StartPoint and ST_EndPoint return the start and end node of an ST_Curve object, respectively. The operator ST_Centroid returns an ST_Point object that is the mathematical centroid of an ST_Surface or ST_MultiSurface object. ST_PointOnSurface returns an ST_Point object that is guaranteed to be on the ST_Surface or ST_MultiSurface object. ST_Envelope and ST_ConvexHull on ST_Geometry return the bounding box and the convex hull of the ST_Geometry object, respectively. ST_Buffer returns an ST_Geometry object that represents all points whose distance from a given ST_Geometry object is less than or equal to a specified distance.

2.3.3 Temporal database systems

The databases managed by standard DBMS normally describe the current state of the world. A standard DBMS offers data types like date and time that can be used from the attributes. If an application needs to keep track of the history of changes, it has to manage time itself by adding it explicitly as attribute(s), and performing the right kind of computation in the queries [51]. When a join is done between two tables extended by time attributes, explicit conditions should be added to the query to assure concurrency in the life time of joined tuples. This results soon in quite complicated queries, and long execution times. A temporal DBMS takes care that such conditions are checked automatically, and there is no need to include them in the query. The objective of a temporal DBMS is the integration of temporal concepts deeply into its data model and query language, to achieve efficient execution of queries.

The basic concepts of a temporal DBMS are the time domain and the time dimensions [51]. Time can be seen as discrete or continuous. While time is perceived as continuous, for practical reasons temporal databases work with discrete time. We are generally interested in certain instants or periods of time.

[1] The combinatorial boundary is explained in Section 4.1.1.

The data types instant, period, and periods are used from temporal databases. The type periods is a set of disjoint periods. Two important time dimensions are *valid time* and *transaction time*. The valid time refers to the real world time instant when a change occurs, or the period during which a fact is valid. The transaction time refers to the time when the change is reflected in the database, or the period during which the database is in a particular state.

Standard databases are called snapshot databases, those dealing with valid time are called historical databases, while temporal databases are those offering any kind of time support. Relational and object-oriented data models are extended with the temporal concepts. Time is added at the tuple (object) level, or at the attribute level. Several temporal models have been proposed (see [51] for an overview), some storing change at the instant it occurs, others storing the period during which a fact or a database state exists. For example, a temporal model working with valid time at the tuple level may add a new tuple for each change, timestamping it with the instant it became valid, or timestamping every tuple with the period they are valid.

2.4 Concepts from fuzzy theory

The term fuzzy theory is best depicted as a triad of branches consisting of fuzzy logic, fuzzy sets, and fuzzy mathematics [65]. Fuzzy logic is the logic of fuzzy propositions. Fuzzy propositions are sentences, of which the truth value is not just 'true' or 'false,' but a matter of degree. As a logic, it studies the notion of consequence [52], dealing with sets of propositions related by connectives like conjunction, disjunction, negation, and implication, and using inference rules for drawing conclusions. The theory of fuzzy sets is a theory of classes with unsharp boundaries [52]. It is a generalization of the classical set theory. Fuzzy mathematics refers to the extension of different branches of mathematics to fuzzy sets. One of these branches, fuzzy topology, is used in this thesis for describing the system of vague types and operators. This section describes basic concepts from fuzzy sets, followed by fuzzy topology concepts. The definitions are given for real spaces.

2.4.1 Fuzzy sets

For a classical set A on $\mathbb{R}^n$ we can crisply answer the question 'does $p \in \mathbb{R}^n$ belong to A?' In contrast, for a fuzzy set the membership is not a 'yes/no' matter, but a matter of degree. A fuzzy set μ associates a membership value with any element $p \in \mathbb{R}^n$, which quantifies to what degree p belongs to μ. We postulate membership values to fall into the unit interval $[0, 1]$ denoted by I. A *fuzzy set* μ in $\mathbb{R}^n$ is then a (total) function $\mu : \mathbb{R}^n \to I$. This function is also called the membership function of the fuzzy set. A classical set A can be represented

by its characteristic function $\chi_A : \mathbb{R}^n \rightarrow \{0,1\}$. A characteristic function χ is called a *crisp set*. The set of all fuzzy sets in $\mathbb{R}^n$ is denoted by $\mathcal{F}(\mathbb{R}^n)$.

An α*-cut* of a fuzzy set μ on $\mathbb{R}^n$ is the set of elements of $\mathbb{R}^n$ with membership value greater than or equal to α: $\mu_\alpha = \{p \in \mathbb{R}^n \mid \mu(p) \geq \alpha\}$. Similarly, a strict α-cut of μ is defined as: $\mu_{\overline{\alpha}} = \{p \in \mathbb{R}^n \mid \mu(p) > \alpha\}$ [61]. Both cuts are classical sets. For $\alpha < \beta$ it is true that $\mu_\beta \subseteq \mu_\alpha$. The same holds for strict α-cuts. The strict 0-cut of a fuzzy set $\mu_{\overline{0}} = \{p \in \mathbb{R}^n \mid \mu(p) > 0\}$ is called its *support set*, and is denoted by $supp(\mu)$. The 1-cut of a fuzzy set $\mu_1 = \{p \in \mathbb{R}^n \mid \mu(p) = 1\}$ is called the *core* of μ. A fuzzy set is convex if all its α-cuts are convex.

Two fuzzy sets μ and ν are disjoint if their support sets are disjoint, i.e., $supp(\mu) \cap supp(\nu) = \emptyset$. A fuzzy set μ is a *subset of* a fuzzy set ν, denoted by $\mu \sqsubseteq \nu$, iff $\forall p \in \mathbb{R}^n, \mu(p) \leq \nu(p)$. The general union and intersection between fuzzy sets are defined using t-conorm and t-norm, respectively. Both, a t-conorm and a t-norm are functions $t : [0,1] \times [0,1] \rightarrow [0,1]$ commutative, associative, monotonic increasing and decreasing, respectively, satisfying a so-called boundary condition for values 1 and 0, respectively. Continuity is another condition that is often added to a t-conorm and a t-norm function. (See [61] for commonly used intersection and union operators.) We work with the standard operators, of which the definitions follow. The *union* of two fuzzy sets μ and ν is $\mu \sqcup \nu = \{(p, \max\{\mu(p), \nu(p)\}) \mid p \in \mathbb{R}^n\}$. It is the smallest fuzzy set containing both of them. The *intersection* of two fuzzy sets μ and ν is $\mu \sqcap \nu = \{(p, \min\{\mu(p), \nu(p)\}) \mid p \in \mathbb{R}^n\}$. It is the biggest fuzzy set contained in both of them. The *complement* of a fuzzy set μ is $1^{\mathbb{R}^n} - \mu = \{(p, 1 - \mu(p)) \mid p \in \mathbb{R}^n\}$.

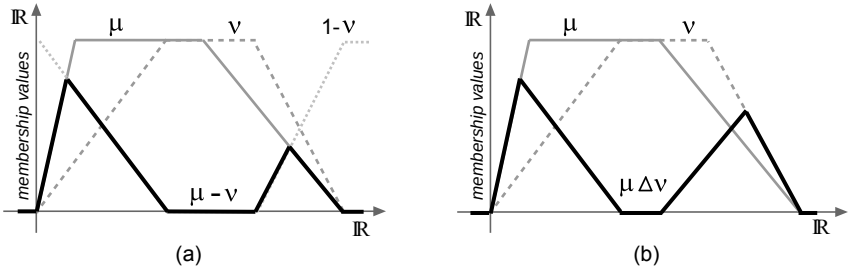

Figure 2.12: (a) difference and (b) symmetric difference on fuzzy sets in $\mathbb{R}$.

The *difference* of two fuzzy sets μ and ν is the intersection of the first with the complement of the second: $\mu - \nu = \mu \sqcap (1^{\mathbb{R}^n} - \nu)$. The *symmetric difference* between two fuzzy sets μ and ν is defined from the difference as $\mu \triangle \nu = (\mu - \nu) \sqcup (\nu - \mu)$. Figures 2.12(a) and 2.12(b) illustrate the difference and symmetric difference between two fuzzy sets in $\mathbb{R}$, respectively. The *bounded difference* between two fuzzy sets μ and ν is defined by: $\forall p, \mu \nabla \nu(p) = \max\{0, \mu(p) - \nu(p)\}$. The *absolute difference* between two fuzzy sets μ and ν is defined by: $\forall p, \mu|-|\nu(p) = |\mu(p) - \nu(p)|$. Figure 2.13(a) illustrates the bounded difference between two fuzzy sets in $\mathbb{R}$, and Figure 2.13(b) illustrates the absolute difference. The fuzzy

difference and the bounded difference are the analogues of the difference operator on classical sets; both operators give the same result as the set difference when applied to crisp sets. The fuzzy symmetric difference and the absolute difference are analogous to the symmetric difference operator on classical sets; they give the same result as their crisp analogue when applied to crisp sets.

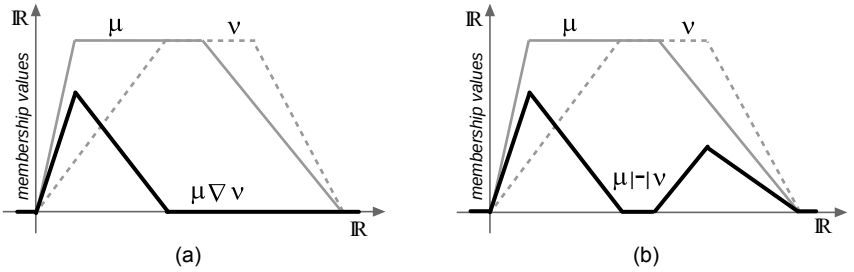

Figure 2.13: (a) bounded difference and (b) absolute difference on fuzzy sets in ℝ.

Let X and Y be subsets of $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. A function $f : X \to Y$ induces two functions $\tilde{f} : \mathcal{F}(X) \to \mathcal{F}(Y)$ and $\tilde{f}_i : \mathcal{F}(Y) \to \mathcal{F}(X)$ that produce an image of a fuzzy set in X as a fuzzy set in Y, and an inverse image of a fuzzy set in Y as a fuzzy set in X, respectively. This is known as the extension principle [61]. The image of a fuzzy set $\mu \in \mathcal{F}(X)$ is $\nu = \tilde{f}(\mu) \in \mathcal{F}(Y)$ such that

$$\forall y \in Y, \nu(y) = \begin{cases} \sup\{\mu(x) \mid x \in X, f(x) = y\} & \exists x \in X, f(x) = y \\ 0 & \text{otherwise} \end{cases}$$

The inverse image of a fuzzy set $\nu \in \mathcal{F}(Y)$ is $\mu = \tilde{f}_i(\nu)$ such that $\forall x \in X, \mu(x) = \nu(f(x))$.

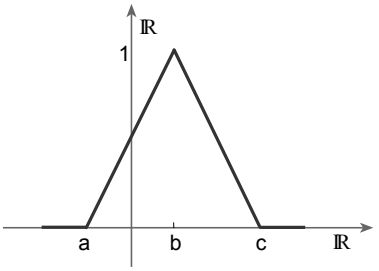

Figure 2.14: Graph of a fuzzy number.

A fuzzy number is a fuzzy set μ in ℝ. Figure 2.14 shows the graph of a fuzzy number that has a piecewise linear function. There are several definitions for a fuzzy number [12, 30, 61, 96] which differ very little from each other. The definition given here originates from Dubois and Prade [30]. A fuzzy number is a function $\mu : \mathbb{R} \to \mathbb{R}$ satisfying the conditions:

(i) there exists only one $x_0 \in \mathbb{R}$ such that $\mu(x_0) = 1$,

(ii) μ is convex, i.e., for any $a \in [0,1]$ and $x, y \in \mathbb{R}$ it is true that $\mu(a \cdot x + (1 - a) \cdot y) \geq \min\{\mu(x), \mu(y)\}$,

(iii) μ is upper semi-continuous,

(iv) $supp(\mu)$ is bounded.

2.4.2 Fuzzy topology

This section provides various definitions of basic notions from fuzzy topology. It explains what is a fuzzy topology, open and closed fuzzy sets, interior and closure, two definitions of boundary, connected and bounded fuzzy sets. Most of these fuzzy topology notions are described by one of the equivalent definitions of the corresponding notion in general topology, put in a fuzzy setting. We give two examples of fuzzy topologies for real spaces, a crisp topology and an induced fuzzy topology. Our definitions of vague objects use the induced fuzzy topologies from the usual topologies for real spaces. The above mentioned notions are explained for the induced fuzzy topologies, and some of them are illustrated for fuzzy sets in $\mathbb{R}$.

Let J be a possibly infinite index set. A family of fuzzy sets $\{\mu_j\}_{j\in J}$ is a subset of $\mathcal{F}(\mathbb{R}^n)$. The union of a family $\bigsqcup \left\{\mu_j\right\}_{j\in J}$ is $\left\{(x, \sup\{\mu_j(x)\}_{j\in J}) \mid x \in \mathbb{R}^n\right\}$. The intersection of a family of fuzzy sets is $\bigsqcap \left\{\mu_j\right\}_{j\in J} = \left\{(x, \inf\{\mu_j(x)\}_{j\in J}) \mid x \in \mathbb{R}^n\right\}$. The two operators are called generalized fuzzy union, and generalized fuzzy intersection, respectively. The crisp set corresponding to the whole $\mathbb{R}^n$ is denoted by $1^{\mathbb{R}^n}$, and the empty set is denoted by $0^{\mathbb{R}^n}$. A *fuzzy topology* for $\mathbb{R}^n$ is a collection $\mathcal{T} \subseteq \mathcal{F}(\mathbb{R}^n)$, such that: (i) $0^{\mathbb{R}^n}, 1^{\mathbb{R}^n} \in \mathcal{T}$, (ii) if $\mu, \nu \in \mathcal{T}$, then $\mu \sqcap \nu \in \mathcal{T}$, and (iii) if $\{\mu_j\}_{j\in J} \subseteq \mathcal{T}$, then $\bigsqcup \left\{\mu_j\right\}_{j\in J} \in \mathcal{T}$ [15]. The pair $(\mathbb{R}^n, \mathcal{T})$ is called a fuzzy topological space. The elements of $\mathcal{T}$ are the open fuzzy sets in $(\mathbb{R}^n, \mathcal{T})$. The complement of an open set is a closed fuzzy set. The closed fuzzy sets for the topology $\mathcal{T}$ are the elements of the collection $\{1^{\mathbb{R}^n} - \mu \mid \mu \in \mathcal{T}\}$.

The collection C_n of crisp sets from $\mathbb{R}^n$ which (corresponding classical sets) are open in the usual topology, forms a fuzzy topology for $\mathbb{R}^n$. Closed fuzzy sets for this topology are the closed crisp sets in the usual topology for $\mathbb{R}^n$. We call the fuzzy topology C_n the crisp topology built from the usual topology for $\mathbb{R}^n$.

A general topology can give rise to a fuzzy topology, which is then called an induced fuzzy topology. A set $F \subset \mathbb{R}^n$ is closed in the usual topology T_n if for any sequence $(p_n) \subset F$ converging to a point p, p is contained in F. The containment relation for a fuzzy set is translated into comparison of membership values. Hence, a fuzzy set μ would be closed if for any sequence (p_n) converging to a point p, p is at least as much in μ as the p_n's ultimately are, i.e.,

$\mu(p) \geq \limsup_{n\in\mathbb{N}} \mu(p_n)$ [108]. This is the condition that the function $\mu : \mathbb{R}^n \to [0,1]$ is upper semi-continuous. If a function μ is upper semi-continuous, then its complement $1^{\mathbb{R}^n} - \mu$ is lower semi-continuous. The collection $\mathcal{T}_n$ of lower semi-continuous functions in $(\mathbb{R}^n, T_n)$ forms a fuzzy topology, which is called the *induced fuzzy topology* from the usual topology for $\mathbb{R}^n$ [108]. The fuzzy sets with lower semi-continuous membership function are the open sets in $\mathcal{T}_n$, and those with upper semi-continuous membership function are the closed sets. Open and closed fuzzy sets for $\mathcal{T}_n$ can be expressed in terms of open and closed α-cuts, using properties of semi-continuous functions. A function $f : \mathbb{R}^n \to \mathbb{R}$ is lower semi-continuous iff, for each $r \in \mathbb{R}$, $\{x \in \mathbb{R}^n \mid f(x) > r\}$ is open. Correspondingly, f is upper semi-continuous iff $\{x \in \mathbb{R}^n \mid f(x) \geq r\}$ is closed. A fuzzy set μ in $(\mathbb{R}^n, \mathcal{T}_n)$ is open if all its strict α-cuts are open for T_n. It is closed for $(\mathbb{R}^n, \mathcal{T}_n)$ if all its α-cuts are closed for T_n [108]. A fuzzy set that has a continuous function is open and closed (clopen) fuzzy set, e.g., the fuzzy set in $\mathbb{R}$ which graph is shown in Figure 2.5(a).

A *fuzzy point* in $(\mathbb{R}^n, \mathcal{T}_n)$ is a fuzzy set that has a positive value $\lambda > 0$ in just one point, say $p \in \mathbb{R}^n$ [67]:

$$\forall q \in \mathbb{R}^n, \mu(q) = \begin{cases} \lambda & q = p, \\ 0 & q \neq p. \end{cases}$$

A fuzzy point is denoted by p_λ, where p is the unique location with a positive membership, and λ is the membership value at p.

A fuzzy point p_λ in $(\mathbb{R}^n, \mathcal{T}_n)$ belongs to a fuzzy set μ, denoted by $p_\lambda \in \mu$, iff $\lambda \leq \mu(p)$ [67]. A fuzzy set μ in $\mathbb{R}^n$ is a neighbourhood of a fuzzy point p_λ iff there exists $\nu \in \mathcal{T}_n$ such that $p_\lambda \in \nu \sqsubseteq \mu$ [67]. A family $\mathcal{B} \subseteq \mathcal{T}_n$ is a base for $\mathcal{T}_n$ iff for each element μ of $\mathcal{T}_n$ there exists $\mathcal{B}_\mu \subseteq \mathcal{B}$ such that $\mu = \bigsqcup \{\nu \mid \nu \in \mathcal{B}_\mu\}$ [67]. A lower semi-continuous function can be taken as the superior of some continuous functions. The family $\mathcal{B} = \{\mu \mid \mu : \mathbb{R}^n \to [0,1] \text{ is continuous}\}$ forms a base for $\mathcal{T}_n$ [67].

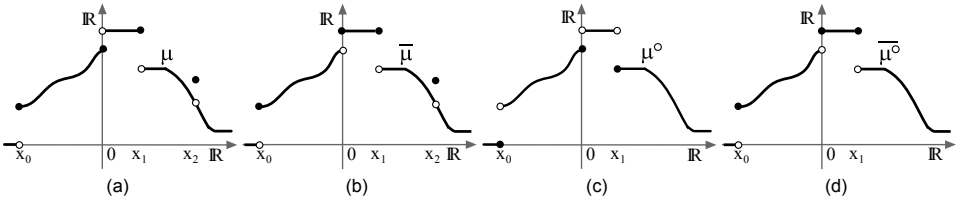

Figure 2.15: Interior, closure, and regular closure of fuzzy sets for $(\mathbb{R}, \mathcal{T}_1)$: (a) a fuzzy set μ, (b) its interior μ°, (c) its closure $\overline{\mu}$, (d) the closure of the interior of μ.

The *interior* of a fuzzy set μ is the union of all open sets contained in μ: $\mu^\circ = \bigsqcup \{\nu \in \mathcal{T} \mid \nu \sqsubseteq \mu\}$. It is the biggest open set contained in μ. A fuzzy point x_λ belongs to the interior μ° iff x_λ has a neighbourhood contained in μ [67]. The *closure* of a fuzzy set μ is the intersection of all closed sets containing μ: $\overline{\mu} =$

$\bigsqcap\{1-\nu \in \mathcal{T} \mid \mu \sqsubseteq \nu\}$. It is the smallest closed set containing μ. Figure 2.15(a) illustrates a fuzzy set μ in $\mathbb{R}$ that is neither open nor closed. Figure 2.15(c) illustrates its interior μ°. The interior changes the value of the function at the discontinuity points by putting it to the lower value, if the continuity is not to the lower value. The value of μ at points x_0, x_1, and x_2 is put to the lower value, while at point $x = 0$ it is already continuous to the lower value. Figure 2.15(b) illustrates the closure of μ. The closure changes the value of the function at discontinuity points putting it at the highest value. The closure of μ has put the value at $\mu(0)$ to the highest value. A fuzzy set μ is *regular closed* iff it is equal to the closure of its interior: $\mu = \overline{\mu^\circ}$. The fuzzy set of Figure 2.15(d) illustrates the regular closure of the fuzzy set of Figure 2.15(a).

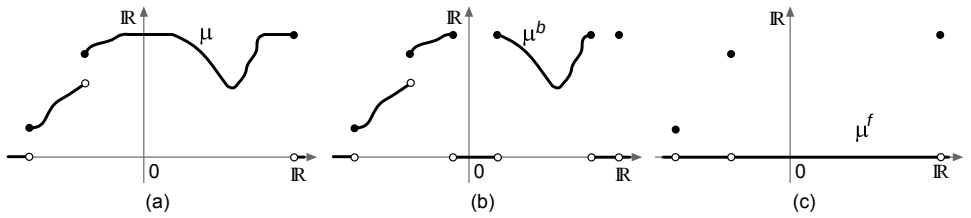

Figure 2.16: Boundary and frontier of fuzzy sets for $(\mathbb{R}, \mathcal{T}_1)$: (a) a regular closed fuzzy set μ, (b) its boundary μ^b, (c) its frontier μ^f.

The notion of boundary in general topology satisfies several properties, e.g., the boundary of any set is closed, the closure of a set is the union of its interior and its boundary, the boundary of a set is equal to the boundary of its complement. None of the many equivalent definitions of boundary in a general topology can be translated into a fuzzy topological setting in such a way that all the properties will hold [106]. Warren [106] and Cuchillo-Ibáñez and Tarrés [24] give two different definitions for the boundary of a fuzzy set. We call the first definition a fuzzy boundary, and the second a fuzzy frontier, keeping to the naming provided in their papers. The *fuzzy boundary* μ^b of a fuzzy set μ is the intersection of all closed sets ν in $\mathcal{F}(\mathbb{R}^n)$ such that $\nu(x) \geq \overline{\mu}(x)$ at all $x \in \mathbb{R}^n$ for which $\overline{\mu} \sqcap \overline{1-\mu}(x) > 0$ [106]. This definition of the boundary satisfies the first two properties mentioned above, but not the third: the boundary of a fuzzy set is different from the boundary of its complement. The fuzzy boundary of a set μ in $(\mathbb{R}^n, \mathcal{T}_n)$ is equal to μ at the uncertain part $\{p \in \mathbb{R}^n \mid 0 < \mu(p) < 1\}$, has value 1 at the (crisp) boundary of the core, and value 0 everywhere else. Figure 2.16(b) shows the graph of the fuzzy boundary of the fuzzy set of Figure 2.16(a). The *fuzzy frontier* μ^f of μ is the intersection of all closed sets ν in $\mathcal{F}(\mathbb{R}^n)$ such that $\nu(x) \geq \overline{\mu}(x)$ at all $x \in X$ for which $\overline{\mu}(x) > \mu^\circ(x)$ [24]. This definition satisfies all properties of the Warren boundary, and furthermore the property that the boundary of a clopen fuzzy set is empty $0^{\mathbb{R}^n}$. The frontier μ^f of a fuzzy set μ in $(\mathbb{R}^n, \mathcal{T}_n)$ has a positive value only at the discontinuity points of μ. It is an empty fuzzy set if μ is continuous. Figure 2.16(c) shows the graph of the fuzzy frontier of the fuzzy set of Figure 2.16(a).

Let μ be a fuzzy set in $(\mathbb{R}^n, \mathcal{T}_n)$, and $X \subset \mathbb{R}^n$. The fuzzy set $\mu^{|X}$ on $\mathbb{R}^n$ that has the same membership value as μ for all $x \in X$ and value 0 for all $x \in X^C$ is a fuzzy set on X. We call the set $\mu^{|X}$ on $\mathbb{R}^n$ the restriction of μ on X. The family $\mathcal{T}_n^X = \left\{\mu^{|X} \mid \mu \in \mathcal{T}_n\right\}$ is a fuzzy topology in X and is called the relative fuzzy topology for X. The fuzzy topological space $(X, \mathcal{T}_n^X)$ is a subspace of $(\mathbb{R}^n, \mathcal{T}_n)$ [67].

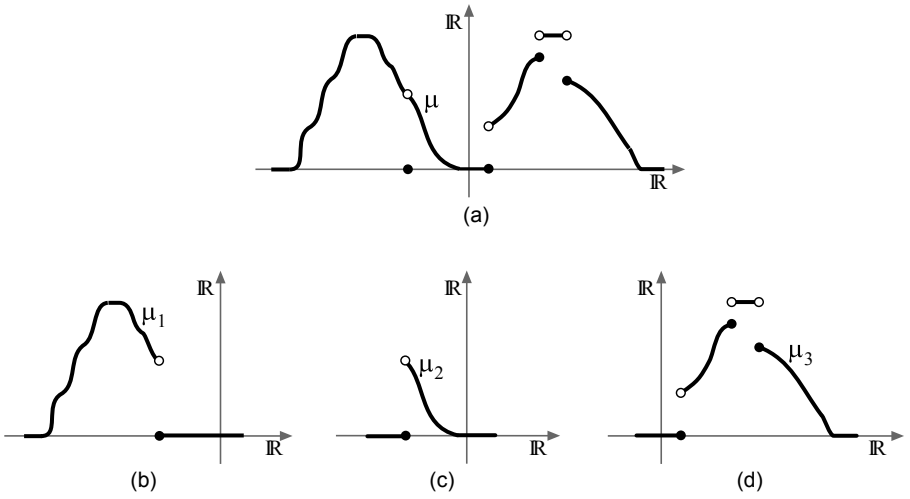

Figure 2.17: Disconnected and connected fuzzy sets for $(\mathbb{R}, \mathcal{T}_1)$: (a) a disconnected open fuzzy set μ, (b)–(d) connected fuzzy sets that are the components of μ.

A fuzzy set μ is bounded if every α-cut μ_α is bounded [67]. The set μ is bounded in $(\mathbb{R}^n, \mathcal{T}_n)$ if its support set is bounded. The fuzzy set μ of Figure 2.16(a) is bounded, and regular closed. A fuzzy set μ is disconnected if there are closed sets γ and δ in the subspace $supp(\mu)$ associated with the relative fuzzy topology, such that $\mu \sqcap \gamma \neq 0^{\mathbb{R}^n}$, $\mu \sqcap \delta \neq 0^{\mathbb{R}^n}$, $\gamma \sqcap \delta = 0^{\mathbb{R}^n}$, and $\mu \sqsubseteq \gamma \sqcup \delta$ [67]. A fuzzy set μ is *connected* if it is not disconnected. We adopt the connectedness notion proposed by Pu and Liu [67]. Weiss [108] proposes a more strict notion for connectedness: a fuzzy set is connected if all its α-cuts are connected. A fuzzy set μ is Pu-Liu connected for $(\mathbb{R}^n, \mathcal{T}_n)$ if it has a connected support set. Let μ be a fuzzy set in $\mathbb{R}^n$. The maximal connected fuzzy set contained in μ is called a *component* of μ. The fuzzy set μ of Figure 2.16(a) is connected, whereas the set μ of Figure 2.17(a) is disconnected. Figure 2.17(b)–(d) are the components of the fuzzy set of Figure 2.17(a).

2.5 Following up

The two spatial models presented in Section 2.3 are used to compile the list of types and operators we propose for vague objects. The separation of operators into groups, reflected in the structure (chapters) of the thesis, follows mainly the grouping proposed by the ROSE algebra. We join the operators returning spatial objects with operators on collection of spatial objects, from which we define only two operators on partitions. All the spatial types, including partitions, and operators returning these spatial types are presented together in Chapter 3. Spatial predicates, which we call spatial relations, and present in Chapter 4, follow the model of topological relations proposed by the SQL/MM spatial. Operators returning numbers are, except for the number of components, metric operators that are presented in Chapter 5.

The description of crisp types and our definitions of vague types use the concept of regular closure, connectedness, boundedness, from the general and fuzzy topology, respectively. The usual topologies T_1 and T_2 are used for definitions of crisp types. We use the induced fuzzy topologies from usual topologies, $\mathcal{T}_1$ and $\mathcal{T}_2$, for the definition of vague types.

Chapter 3

Vague spatial types and operators returning spatial types

This chapter provides formal definitions of vague spatial types and of operators returning these types. We distinguish between simple types and general types. A simple type represents an identifiable object with the simplest structure, i.e., non divisible into components. The simple types are *VPoint*, *VLine*, and *VRegion*, representing a vague point, a vague line, and a vague region, respectively. A general type represents a class of simple objects. The general types are *VMPoint*, *VMLine*, and *VMRegion*, representing a vague multipoint, a vague multiline, and a vague multiregion, respectively. The operators are regularized set operators: union, intersection, and difference, together with two operators from topology: frontier and boundary. The general types assure closure for the set operators, and the frontier operator. The return type of a boundary operator is none of the above types. To cover for this, we propose two other types, *VExt* and *VLDim*. A *VExt* object is a collection of vague lines and vague regions, a *VLDim* object is a collection of vague points and vague lines. To represent a soft classification of space we propose a type *VPartition*. Figure 3.1 shows the vague spatial types and their relations, subclass and aggregation. *SVSpatial* represents a simple object of any of the simple types *VPoint*, *VLine*, or *VRegion*. *GVSpatial* represents an object of any of the general types *VMPoint*, *VMLine*, or *VMRegion*. *VSpatial* represents a (vague) spatial object of any type.

A simple or a general type represents an object which essential property is expressed in vague terms. Such an object cannot be characterized only by the set of locations that form its extent. Any location is associated with a degree of membership to the object extent. An object is thus characterized by a function that determines the membership degree at each location. The vagueness present at these objects is thematic, and not locational. The location of the objects is assumed to be known. Though, this is not always the case for vague regions, for which the locational vagueness is often the result of thematic vagueness, e.g., regions formed by a classification over space.

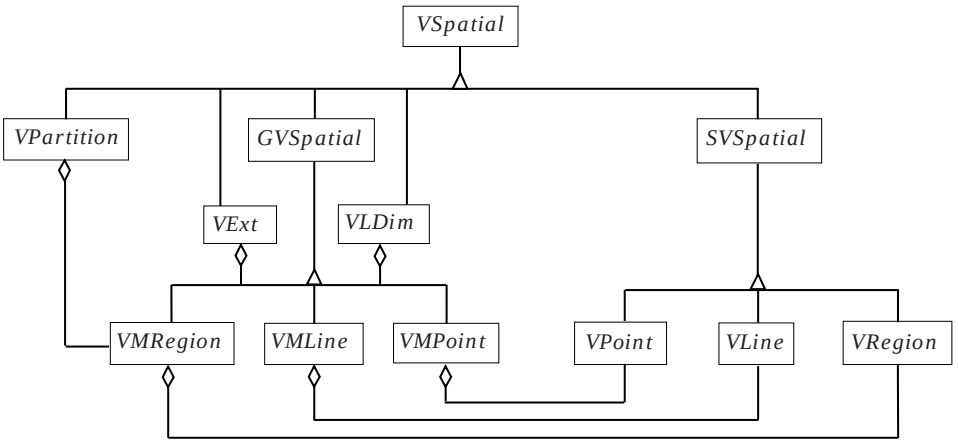

Figure 3.1: The hierarchy of vague spatial types.

We expect vague objects to have mostly gradual transitions of membership values. An abrupt change of membership values may also happen at some locations. Membership values range between 0 and 1, normally covering the whole range $[0,1]$. There are however applications that might need a finite set of memberships. It is desirable to define the vague types such that they include crisp objects as special cases. Semi-continuous functions satisfy all the above, being mostly continuous functions that allow jumps. All simple and general types are defined as fuzzy sets in $\mathbb{R}^2$ that satisfy some well-defined properties. To express these properties we use the fuzzy topologies $\mathcal{T}_1$ and $\mathcal{T}_2$ of semi-continuous functions in $\mathbb{R}$ and $\mathbb{R}^2$, respectively.

The chapter is organized as follows. Section 3.1 summarizes theoretical models proposed to handle spatial vagueness. Section 3.2 presents examples of spatial phenomena that can be described by the object types that we propose. Section 3.3 provides definitions of types and operators for vague point objects. Section 3.4 provides for types and operators of vague line objects. Section 3.5 provides types and operators for vague region objects. Vague partitions and their operators are discussed in Section 3.6. Section 3.7 summarizes the work of this chapter. It also discusses how the vague point and line types could be changed to cover for a locational vagueness. For the illustrations of vague objects in this chapter and the following ones, we use colour saturation to show membership values. Full saturation indicates the highest membership value, lower saturation indicates lower membership. Colour hue is used to distinguish different objects, when it is needed.

3.1 Existing models for spatial vagueness

Most work on vagueness is dedicated to vague regions and topological relations between them. A vague region is a region with a broad boundary, i.e., its boundary is not necessarily a sharp line, but a zone of transition. Current work on vague regions either considers the broad boundary as a homogenous unit [7, 13, 18, 19, 21, 22, 41, 79], or considers gradual transition in the boundary, that is different locations in the broad boundary have different degrees of membership to the region [80, 81, 82, 83, 84, 85, 97, 116, 117]. The next two sections list the main proposals of each group.

3.1.1 Broad boundary regions

The Egg-Yolk model [21, 22] describes a vague region as a pair of crisp regions, one enclosing the other. The inner region, called the yolk, gives the certain part of the vague region. The outer region, called the white, is the broad boundary which delineates limits on the range of vagueness. The white and yolk together form the egg that is the full extent of the vague region. All the regions, the egg, the yolk, and the white, are RCC regions [20, 72, 73], a theory based on mereology [103]. The RCC regions cannot be empty, therefore the model cannot express crisp regions.

The proposal of Clementini and di Felice [18, 19] is basically the same with the egg-yolk model. The definitions are based on general topology. A vague region A consists of two sets A_1, A_2 of $\mathbb{R}^2$ such that $A_1 \subseteq A_2$. The broad boundary ΔA is the closure of their difference: $\Delta A = \overline{A_2 \backslash A_1}$. Each set, A_1 and A_2, is a crisp region, i.e., a bounded, regular closed set in $\mathbb{R}^2$ with connected interior [19]. The inner region A_1 gives the certain part of a vague region, and the broad boundary ΔA delineates limits of the vagueness.

Erwig and Schneider [41] define vague regions on the basis of general topology. A vague region is defined as a pair of disjoint sets: the kernel that is the certain part of the region, and the boundary that is its uncertain part. Kernel is a crisp region, whereas the boundary can be a region or a line, the later allowing a crisp region to be a special case of a vague region. This feature is not supported by the Egg-Yolk and Clementini and di Felice models. The three models can handle both ontological and linguistic spatial vagueness.

Rough sets are used to model vague regions in [7, 79]. A vague region is represented by a pair of RCC regions corresponding to the lower and upper approximation. When the lower and upper approximation are equal, the region is crisp. This model is a generalization of the Egg-Yolk model. Three–valued Łukasiewicz algebras are used as a formal context for vague regions and their operators in [79]. The model of spatial regions can be used to handle any type of vagueness, but the operators on regions assume a linguistic vagueness.

3.1.2 Fuzzy spatial objects

The works of this second group take gradual changes into consideration and employ fuzzy set theory to model spatial objects. Zhan [116, 117] provides mathematical definitions of fuzzy regions and topological relations between them. Fuzzy regions are represented as fuzzy sets in [116]. This allows the existence of irregularities, e.g., isolated points and lines that are not desirable for regions. In [117] Zhan redefines fuzzy regions in terms of fuzzy convexity. This excludes irregularities, but it is unnecessary restrictive.

Schneider's fundamental work provides formal definitions of fuzzy types [80, 85], and definitions of topological and metric operators on fuzzy objects [82, 83, 84]. In [80] Schneider defines fuzzy points, fuzzy lines and fuzzy regions as fuzzy sets in $\mathbb{R}^2$. The definitions of fuzzy regions and their operators are built from a regularization function. This function is expressed as a combination of interior and closure operators, but without specifically indicating the employed topology. The regularization function applied to a fuzzy set gives different fuzzy sets for different fuzzy topologies. The function is thus ambiguous, which in turn leads to ambiguity in the definitions of region types and operators. In the next paragraph we show the ambiguity of the regularization function by an example.

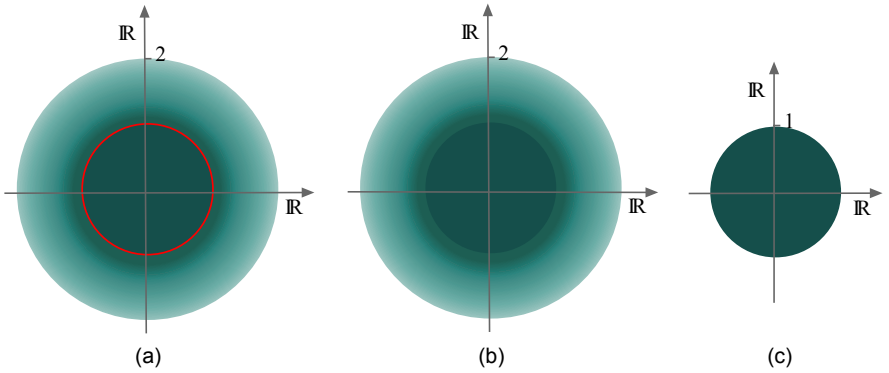

Figure 3.2: Regularization of a fuzzy set for two different fuzzy topologies: (a) a fuzzy set in $\mathbb{R}^2$, (b) its regularization for $\mathcal{T}_2$, (c) its regularization for C_2. Stronger tone indicates higher membership value, lighter tone indicates lower membership.

Let $\psi \in \mathbb{R}^2$ be a fuzzy set defined by

$$\forall p \in \mathbb{R}^2, \psi(p) = \begin{cases} 1 & \mathrm{d}_2(p,O) \leq 1 \\ 2 - \mathrm{d}_2(p,O) & 1 < \mathrm{d}_2(p,O) \leq 2 \\ 0 & \mathrm{d}_2(p,O) > 2 \end{cases}$$

Figure 3.2(a) illustrates the fuzzy set ψ using saturation to show membership values; the boundary of the core is drawn in red. Let us consider two fuzzy topologies, $\mathcal{T}_2$, the induced fuzzy topology from the usual topology T_2, and

C_2, the crisp topology built from T_2. A frontier notion is used in [80] to build the regularization function. The frontier of a fuzzy set is its restriction (see page 28) to the difference of its support set with the support set of its interior: $frontier(\psi) = \psi^{|supp(\psi)\setminus supp(\psi^\circ)}$. The regularization function is then defined as $reg(\psi) = \overline{\psi}^\circ \sqcup (frontier(\psi) \sqcap frontier(\overline{\psi}^\circ))$. Application of the regularization function on ψ for the topology $\mathcal{T}_2$ gives the set itself[1]: $reg_{\mathcal{T}_2}(\psi) = \psi$. Such regularization for the crisp topology gives the crisp closed unit disk[2]: $reg_{C_2}(\psi) = \chi_{\overline{U(O,1)}}$. Regularization yields different results when performed for different topologies. Therefore, introducing it without specifying a topology makes the definitions ambiguous.

In [85] Schneider defines fuzzy objects based on a finite collection of elements from a regular grid, forming a partition of a bounded subspace of $\mathbb{R}^2$. Membership values are assigned to the elements of the grid: points, edges, and cells. Each fuzzy object is built from the grid elements. The model can be directly implemented in raster data format, but it does not address implementation problems like efficiency or indexing. The model eliminates anomalies of calculations on real numbers performed with a finite set of numbers available in computers. It is a justification for the suitability of the raster format to represent fuzzy spatial data, but it is restricted to this representation. Indeed any other computer representation used in curve and surface modelling, such as wavelets or TINs, would be possible as well. We use TIN-like structures in our implementation (Chapter 6).

Tang [97] provides two different definitions for spatial regions using a crisp topology and a general fuzzy topology. The vague regions are defined as fuzzy sets satisfying a list of properties in these topological spaces. The crisp topology he employs for the definitions is C_2. The general fuzzy topology is not specified, which makes the definitions prone to ambiguity, in the same way ambiguity arises in Schneider's definitions [80].

3.2 Examples of spatial vagueness

In ordinary natural language adjectives are commonly attributed to phenomena. This is certainly the case in the cognition of geography, where characteristics of spatial phenomena are expressed in ordinary language terms, which are generally vague.

- When considering densely populated residential centres, we have to iden-

[1] The function ψ is continuous, therefore the fuzzy set is clopen for $\mathcal{T}_2$, and as such it is regular closed. Its *frontier* is empty, as well as the *frontier* of its interior.

[2] The interior of ψ for C_2 is the interior of its core, $U(O,1)$. The regular closure gives the closed disk $\overline{U(O,1)}$. The $frontier(\overline{\psi}^\circ)$ is the restriction of ψ in the unit circle, and is a subset of $frontier(\psi)$.

tify known locations that have different degrees of being densely populated. We know the location precisely, but the density level itself is vague.

- Considering polluted rivers, we know precisely where the river is. Close to the source of pollution the river is certainly polluted, but further away riverine tracks may exist with less severe pollution. Therefore, at different locations along the river, different degrees of pollution exist that change gradually.

- A traffic congestion on a road network relates a characteristic — level of congestion — to a geographic phenomenon — the roads. Part of the road is completely blocked, and hence certainly belongs to the traffic congestion, whereas away from the congestion, the car build-up becomes less severe. A congestion that is dissolving at the end of a rush hour does no longer have such a certain part. This vague characteristic, however, is still spread on the roads.

- Agricultural land suitability in [87] is based on farmers' knowledge on soils, like soil texture, colour, depth and slope. These parameters are linguistic variables taking values, e.g., 'fine', 'moderately fine', and 'coarse' for soil texture. The suitability map is built from a combination of values of these variables according to specific rules. Suitability derived in this way is also a linguistic variable with values 'least suitable', 'moderately suitable', 'suitable', and 'most suitable.' This attribute, suitability, spread over the space, determines objects which locations have membership values from a finite set (of four values).

- It may be arbitrary to consider a particular location as part of a vegetated area, or of a non-vegetated area. Some locations are certainly vegetated, whereas other locations can be considered vegetated or non-vegetated to some degree. There are transition zones where vegetation becomes sparse. In some places the transition is gradual, whereas in other places the change may be abrupt.

The above examples illustrate thematic vagueness, but not (always) locational vagueness. Objects describing such spatial phenomena have a crisp location, but their essential properties can only be expressed in vague terms. The object types proposed in the next few sections can handle a thematic kind of vagueness. For regions, thematic vagueness in most instances also covers locational vagueness, which is the case in the last two examples. There are situations, however, where locational vagueness is independent of thematic vagueness, as for example a forest region from which one way or another we know the shape, but not the precise location.

3.3 Vague point types and operators

We propose two types of vague point objects: a *vague point* representing the simplest identifiable point object, and *a vague multipoint* representing a class of simple point objects. Figures 3.3(a) and 3.3(b) illustrate a vague point and a vague multipoint object, respectively.

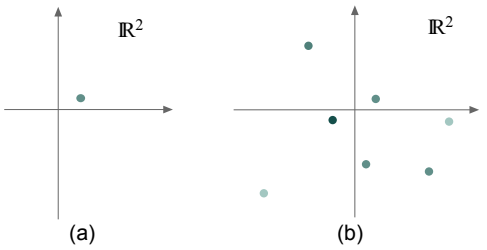

Figure 3.3: Vague point objects: (a) a vague point, (b) a vague multipoint.

A *vague point* is a site with a known location, but with uncertain membership to a phenomenon of interest. It is defined as a fuzzy point p_λ in $\mathcal{T}_2$. The membership value λ represents the degree of belonging of the site $p(x,y)$ to the phenomenon of interest. The set of vague points is

$$\mathit{VPoint} \equiv \left\{\mu \in \mathcal{F}(\mathbb{R}^2) \mid \exists!(x,y) \in \mathbb{R}^2, \mu(x,y) > 0\right\}.$$

The restriction of a fuzzy point μ to its support set is the singleton set $\{((x,y),\lambda)\}$. If the membership value $\mu(x,y)$ is equal to 1, then μ is a crisp point.

A *vague multipoint* is a finite collection of disjoint vague points. It is defined as a fuzzy set in $\mathbb{R}^2$ that has positive membership values in a finite set of locations. The set of vague multipoints is

$$\mathit{VMPoint} \equiv \left\{\mu \in \mathcal{F}(\mathbb{R}^2) \mid \exists\, \{\mu_i\}_{i=1}^{n} \subset \mathit{VPoint}, \mu = \bigsqcup_{i=1}^{n} \mu_i\right\}.$$

A vague point is a special case of a vague multipoint, for $n = 1$. We allow the empty set $0^{\mathbb{R}^2}$ to be a special case of a vague multipoint, having $n = 0$. The restriction of a vague multipoint to its support set is a finite set of triples providing vague point locations and their membership values: $\{((x_1,y_1),\lambda_1),((x_2,y_2),\lambda_2)$ $\ldots,((x_n,y_n),\lambda_n)\}$.

The operators union, intersection, and difference, for vague multipoints are the fuzzy union, fuzzy intersection, and fuzzy difference operators, respectively. The *union* between two vague multipoints is a vague multipoint of which the locations are the union of input point locations. The membership value at each location of the result is the maximum membership of input points at a common

location, and simply the membership of the input point at any other location. The operator *PUnion* is defined as

$$PUnion : VMPoint \times VMPoint \rightarrow VMPoint$$
$$\forall \mu, \nu \in VMPoint,\ PUnion(\mu, \nu) = \mu \sqcup \nu.$$

The *intersection* between two vague multipoints is a vague multipoint of which the locations are the common locations of input points. The membership value at each location of the result is the minimum of memberships of input points at that location. The operator *PIntersection* is defined as

$$PIntersection : VMPoint \times VMPoint \rightarrow VMPoint$$
$$\forall \mu, \nu \in VMPoint,\ PIntersection(\mu, \nu) = \mu \sqcap \nu.$$

The *difference* of two vague multipoints is a vague multipoint of which the locations are those of the first input object. The membership value at a common location is the minimum of the membership of the first object and the complemented membership of the second object. At all other locations it is the membership of the first object. The operator *PDifference* is defined as

$$PDifference : VMPoint \times VMPoint \rightarrow VMPoint$$
$$\forall \mu, \nu \in VMPoint,\ Pdifference(\mu, \nu) = \mu - \nu.$$

Figure 3.4 illustrates two vague multipoints in part (a) and (b), and the results of union, intersection, and difference in Figures 3.4(c)–(e), respectively.

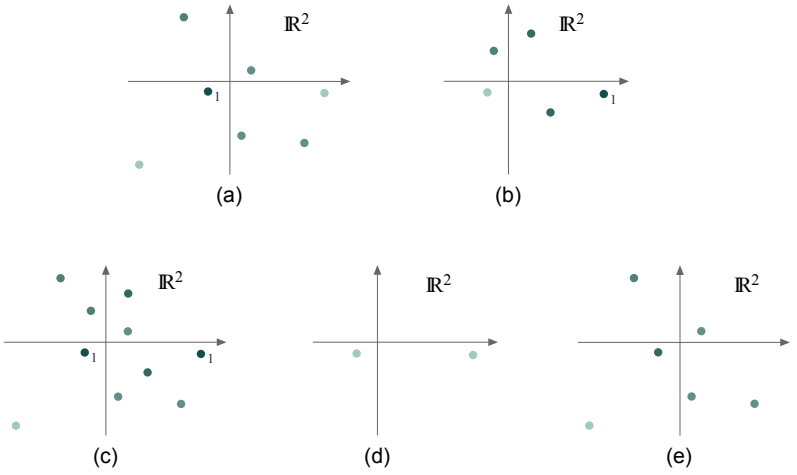

Figure 3.4: Results of vague multipoint operators: (a) and (b) two vague multipoints, (c)–(e) results of their union, intersection, and difference, respectively. Vague points with membership value equal to 1 are labelled with the value 1.

The set of vague multipoints is closed under these operators, i.e., the union, intersection, and difference of two *VMPoint* objects is a *VMPoint* object. It can be

seen that *PUnion*, *PIntersection*, and *PDifference* applied to crisp multipoints are equivalent to the point set union, intersection, and difference, respectively. Hence, they give the corresponding crisp operator when applied to crisp multipoints.

The *boundary* of a vague multipoint μ is its uncertain part, i.e., the locations with membership value smaller than 1. It is constructed from the fuzzy boundary μ^b for the relative topology $\mathcal{T}_2^\mu$. The operator *PBoundary* is defined as

$$PBoundary : VMPoint \rightarrow VMPoint$$
$$\forall \mu \in VMPoint,\ PBoundary(\mu) = \mu^b.$$

The boundary of a vague multipoint μ is its restriction (see page 28) to locations

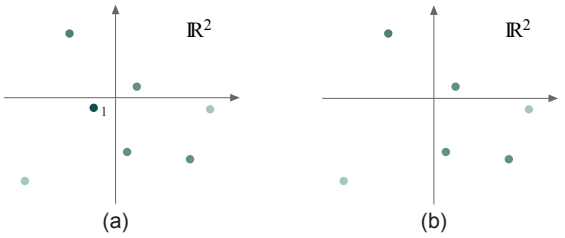

Figure 3.5: Boundary of a vague multipoint: (a) a vague multipoint, and (b) its boundary.

with positive membership smaller than 1: $PBoundary(\mu) = \mu^{|\{(x,y)\in\mathbb{R}^2 | 0<\mu(x,y)<1\}}$.

The *frontier* of vague multipoint μ is its fuzzy frontier μ^f for the relative topology $\mathcal{T}_2^\mu$. The frontier *PFrontier* of a vague multipoint is empty:

$$PFrontier : VMPoint \rightarrow VMPoint$$
$$\forall \mu \in VMPoint,\ PFrontier(\mu) = 0^{\mathbb{R}^2}.$$

The boundary and the frontier of a crisp multipoint are empty. Thus, both operators give the crisp boundary operator when applied to crisp multipoints.

3.4 Vague line types and operators

We propose two types for representing vague objects of a linear nature. A *vague line* represents an identifiable linear object of the simplest structure, and a *vague multiline* represents a collection of vague lines that have the same membership value at their intersection points. Figures 3.6(a) and 3.6(b) illustrate a vague line and vague multiline, respectively. Section 3.4.1 provides description and definitions of vague line types, and Section 3.4.2 provides for vague line operators.

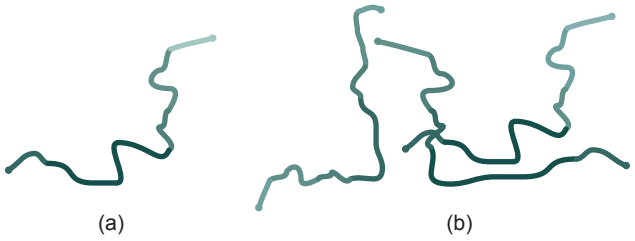

Figure 3.6: Vague line objects: (a) a vague line, (b) a vague multiline.

3.4.1 Vague line types

A *vague line* is a linear feature with known position, but with an uncertain extent, i.e., any point of the line has some certainty degree of belonging to the line. A vague line is a simple curve (i.e., it is contiguous and non self-intersecting) with mostly gradual transitions of membership values between neighbour points on the line. Membership values are positive at every location on the line, except, perhaps, at the end nodes. Stepwise changes of membership values may occur along the line, but isolated discontinuities are not permitted. The functions $\chi_{[0,1]}$ and η of Figure 3.7(a) and 3.7(b) are mostly continuous, both having stepwise changes at points 0 and 1, and at points 0 and x_0, respectively. Both functions have the type of continuity we want for membership functions along vague lines. The function of Figure 3.7(c) has an isolated discontinuity at x_1, and is therefore not a valid membership function for a vague line.

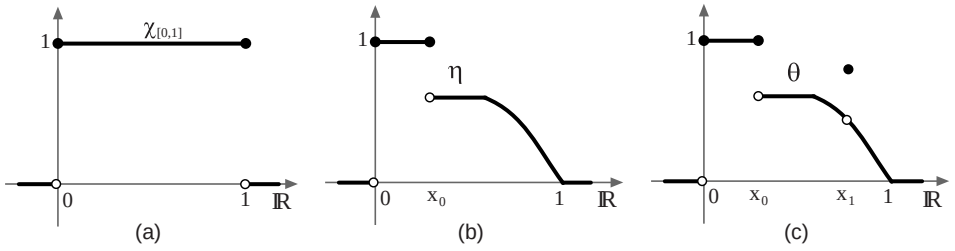

Figure 3.7: Membership functions for vague lines: (a) the characteristic function $\chi_{[0,1]}$ of the closed interval $[0,1]$, (b) a function η having stepwise discontinuity at x_0, (c) a function θ having an isolated discontinuity at x_1 — this is not a valid membership function.

We want the extension of a vague line to be a simple curve that is continuous, non-self intersecting curve, but possibly looped. (We call this crisp line from here onwards). The membership values of the vague line are given by a membership function defined over its extent. We require this function to be almost everywhere continuous, allowing stepwise changes in a finite number of locations.

A crisp line is topologically equivalent to the unit interval $[0,1]$, i.e., there is a

homeomorphism h from $[0,1]$ to the line in $\mathbb{R}^2$. We can build a fuzzy set in $[0,1]$ that satisfies the above mentioned properties for membership functions, then transfer its membership values to the crisp line via the homeomorphism h. Figure 3.8(b) illustrates the construction of a vague line μ from the fuzzy set η of Figure 3.7(b), via the homeomorphism h. The set η is drawn in Figure 3.8(b) using saturation for displaying membership values. The vague line μ is built from the extension principle (described in page 24) as the image $\tilde{h}(\eta)$ of the fuzzy set $\eta \in \mathbb{R}$.

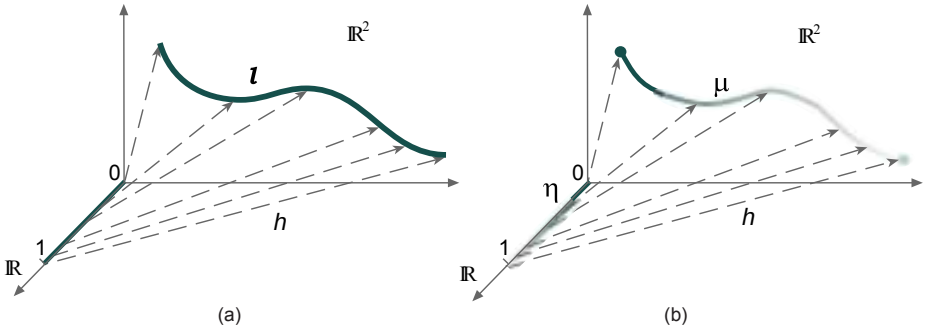

Figure 3.8: Construction of a crisp and vague line from sets in $\mathbb{R}$: (a) crisp line built from the homeomorphism h in $(0,1)$, (b) vague line built by transferring the membership values of a fuzzy set η via the homeomorphism h.

A crisp line is the image of the $[0,1]$ interval by the homeomorphism h: $\{h(t) = (x(t), y(t)) \mid t \in [0,1]\}$. To allow looped lines, the homeomorphism is restricted to $(0,1)$, requiring continuity at the end points 0 and 1 [28]. An upper semi-continuous function satisfies most of the properties we want for the membership function of a vague line, but allows isolated discontinuities, e.g., the function of Figure 3.7(c). The regular closure removes such isolated discontinuities. A vague line can now be built from a regular closed fuzzy set η in $[0,1]$ and the homeomorphism h. To assure that the vague line has a continuous extent, we require the fuzzy set η to have positive membership in $[0,1]$, that is η is connected for $(\mathbb{R}, \mathcal{T}_1)$. A vague line is thus built as the image of a homeomorphism of a regular closed and connected fuzzy set in $[0,1]$. When the vague line is looped, the membership values at both end nodes are equal. The set of vague lines is defined as

$$\begin{aligned} \mathit{VLine} \equiv\ & \{\mu \in \mathcal{F}(\mathbb{R}^2) \mid \exists \eta \in \mathcal{F}([0,1]), \eta = \overline{\eta^\circ}, \text{ and connected,} \\ & \exists h : [0,1] \to \mathbb{R}^2 \text{ homeomorphism in } (0,1), \text{continuous in } \{0,1\}, \\ & \mu = \tilde{h}(\eta) \text{ and } (h(0) = h(1) \Rightarrow \eta(0) = \eta(1))\}. \end{aligned}$$

The homeomorphism h builds the extension of a vague line μ as topologically equivalent with the interval $(0,1)$ that is a 1–dimensional set. The extension of a vague line cannot be a finite set of points. Hence, the type vague line is different from the vague point and vague multipoint types. If the fuzzy set η in $[0,1]$ is a crisp set, then the vague line is a crisp line.

A *vague multiline* is a finite collection of vague lines, of which the extensions intersect only at their end nodes, and the lines have the same membership value at the common end nodes (see Figure 3.6(b)). It is constructed from the union of vague lines from the finite collection. The set of vague lines is

$$\begin{aligned} \mathit{VMLine} \equiv \Big\{ & \mu \in \mathcal{F}(\mathbb{R}^2) \mid \exists \{\mu_i \mid \mu_i = \tilde{h}_i(\eta_i) \in \mathit{VLine}, i \in \{1 \ldots n\}\}, \mu = \bigsqcup_{i=1}^{n} \mu_i, \\ & \forall i, j \in \{1 \ldots n\},\ i \neq j \Rightarrow \mu_i \sqcap \mu_j \sqsubseteq \mu_i^{|\{h_i(0), h_i(1)\}} \sqcup \mu_j^{|\{h_j(0), h_j(1)\}} \Big\}. \end{aligned}$$

The last condition assures that if the vague line components intersect, they do so only at their end nodes. A vague line is a special case of a vague multiline, having $n = 1$. If $n = 0$ the vague multiline is the empty set.

A union type *VLDim* has values that are collections of vague lines and vague points. The type is defined as

$$\mathit{VLDim} \equiv \{\mu \in \mathcal{F}(\mathbb{R}^2) \mid \exists \nu \in \mathit{VMLine}, \exists \upsilon \in \mathit{VMPoint}, \mu = \nu \sqcup \upsilon\}.$$

A *VLDim* object μ can be a vague multiline if the point component υ is empty, and it can be a vague multipoint if its line component ν is empty.

3.4.2 Vague line operators

The union, intersection, and difference operators for vague lines are built from the corresponding fuzzy set operators. The *union* of two vague multilines is a vague multiline produced by the fuzzy set union of the input line objects. The union operator *Lunion* is defined as

$$\begin{gathered} \mathit{LUnion} : \mathit{VMLine} \times \mathit{VMLine} \to \mathit{VMLine} \\ \forall \mu, \nu \in \mathit{VMLine},\ \mathit{LUnion}(\mu, \nu) = \mu \sqcup \nu. \end{gathered}$$

The set of vague multilines is closed under union.

The *intersection* of two vague multilines produces the intersection points of the two line extensions, associated by the membership values at these points calculated from the fuzzy intersection operator. The (point) intersection of the extensions of two vague multilines is produced by the intersection operator for crisp lines. Let μ and ν be two vague multilines. We denote by $EI_{\mu,\nu}$ the (crisp) intersection of their extensions: $EI_{\mu,\nu} = \mathsf{Intersection}(supp(\mu), supp(\nu))$. The intersection operator between vague multilines is defined from the fuzzy restriction to the intersection of their extensions:

$$\begin{gathered} \mathit{LIntersection} : \mathit{VMLine} \times \mathit{VMLine} \to \mathit{VMPoint} \\ \forall \mu, \nu \in \mathit{VMLine},\ \mathit{LIntersection}(\mu, \nu) = (\mu \sqcap \nu)^{|EI_{\mu,\nu}}. \end{gathered}$$

The difference operator between two vague multilines produces a multiline taken from the fuzzy difference of the fuzzy sets. The extension of the result vague

multiline is the (classical set) difference of the extensions of the input vague multilines. Membership values along the extension are calculated from the fuzzy difference. If two vague multilines μ and ν intersect at points, there might be isolated discontinuity at the result of the fuzzy difference. To correct for this, we take the regular closure of fuzzy difference $\overline{(\mu - \nu)^\circ}$ (in the relative topology $\mathcal{T}_2^{\mu-\nu}$). The difference operator is then defined as

$$\begin{gathered} \mathit{LDifference} : \mathit{VMLine} \times \mathit{VMLine} \to \mathit{VMLine} \\ \forall \mu, \nu \in \mathit{VMLine},\ \mathit{LDifference}(\mu, \nu) = \overline{(\mu - \nu)^\circ}. \end{gathered}$$

These three operators give the corresponding crisp operators when applied to crisp multilines.

The *boundary* of a vague multiline is its uncertain part. It is constructed from the union of boundaries of its vague line components. For a vague line μ expressed by a fuzzy set η and a homeomorphism h, the boundary is constructed as the image by h of the fuzzy boundary of η, $\tilde{h}(\eta^b)$. When the vague line μ is crisp, its boundary consists of vague points that are the end nodes of the line. When the membership function along the line is continuous, which means that η is continuous, the boundary of μ consists of vague line components. In general, the boundary of a vague line consists of vague points and vague lines. The boundary of a vague multiline $\mu = \bigsqcup_{i=1}^{n} \mu_i$ is the union of the boundaries of its components μ_i. The boundary operator for vague multilines is defined as

$$\begin{gathered} \mathit{LBoundary} : \mathit{VMLine} \to \mathit{VLDim} \\ \forall \mu \in \mathit{VMLine}, \mu = \bigsqcup \left\{ \mu_i \mid \mu_i = \tilde{h}_i(\eta_i) \in \mathit{VLine}, i \in \{1 \dots n\} \right\}, \\ \mathit{LBoundary}(\mu) = \bigsqcup_{i=1}^{n} \tilde{h}_i(\eta_i^b). \end{gathered}$$

Figure 3.9(a) illustrates a vague multiline. The end nodes of its core are indicated in red. Figure 3.9(b) shows its boundary, which consists of vague lines and vague points. The boundary of a vague multiline μ is the restriction to the set of locations with positive membership smaller than 1, extended by the boundary of the core. The last part consists of the end nodes of the cores of vague line components.

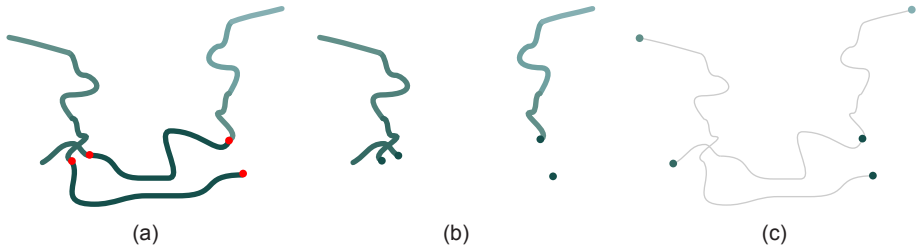

Figure 3.9: Boundary and frontier of a vague multiline: (a) a vague multiline and the boundary of its core in red, (b) its boundary, (c) its frontier (the line extension is drawn in light grey).

The *frontier* of a vague multiline is constructed in a similar way from the fron-

tiers of its vague line components. The frontier of a vague line $\mu = \tilde{h}(\eta)$ is the image of the fuzzy frontier of η, $\tilde{h}(\eta^f)$. The frontier of a vague multiline is a vague multipoint. The operator is defined as

$$\begin{gathered} \mathit{LFrontier} : \mathit{VMLine} \rightarrow \mathit{VMPoint} \\ \forall \mu \in \mathit{VMLine}, \mu = \bigsqcup \left\{ \mu_i \mid \mu_i = \tilde{h}_i(\eta_i) \in \mathit{VLine}, i \in \{1 \ldots n\} \right\}, \\ \mathit{LFrontier}(\mu) = \bigsqcup_{i=1}^{n} \tilde{h}_i(\eta_i^f). \end{gathered}$$

Figure 3.9(c) shows the frontier of the vague multiline of Figure 3.9(a). The frontier of a vague multiline is its restriction to the set of discontinuity locations of the membership function.

The boundary *LBoundary* and the frontier *LFrontier* applied on a crisp multiline produce the set of end nodes of its line components. Thus, both operators give the crisp boundary operator when applied to crisp multilines.

3.5 Vague region types and operators

We distinguish two types of vague region objects, *vague region* representing the simplest identifiable object, and *vague multiregion* representing a class of vague regions. A vague region is a single-component fuzzy set that does not have irregularities: isolated vague points and vague lines, or punctures and cuts, i.e., removed vague points and vague lines, respectively. A vague multiregion is a collection of disjoint vague regions. The fuzzy set of Figure 3.10(a) has a puncture and a cut, both irregularities that are not allowed for a vague region object. Figure 3.10(b) illustrates a vague region, and Figure 3.10(c) illustrates a vague multiregion.

Section 3.5.1 provides definitions for vague regions, vague multiregions and the type vague extent. Section 3.5.2 provides definitions for vague region operators. Illustrations for both sections are produced from data on heavy metal concentration in the sediments of the Maas river in Belgium.

3.5.1 Vague region types

A *vague region* is a broad boundary region, such that points in the broad boundary typically have different positive membership values, which change mostly gradually between neighbour points in the region. The membership values can change abruptly along a line, making a stepwise jump. Abrupt changes only at one location, or membership values along a line changing abruptly from both sides, are not allowed. Figure 3.11 illustrates different vague regions. The vague region of Figure 3.11(a) has a connected core and does not have holes. Its membership function decreases gradually from the boundary of the core to

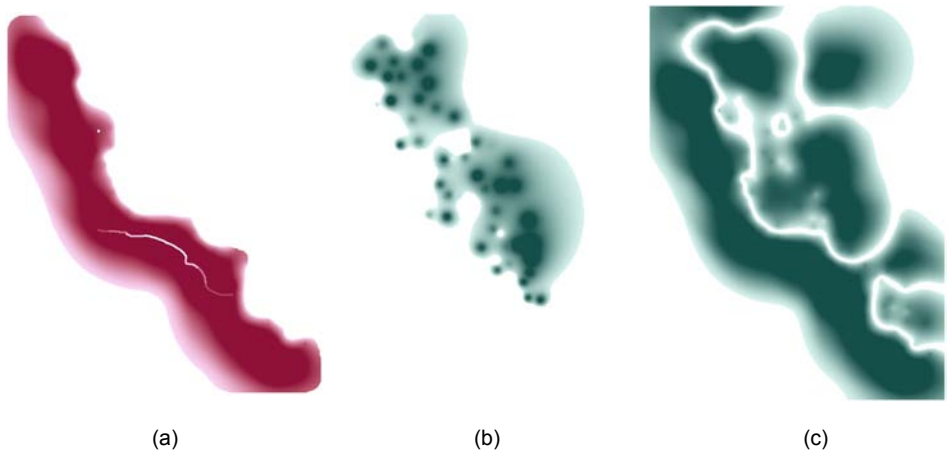

Figure 3.10: Fuzzy sets in $\mathbb{R}^2$: (a) a fuzzy set that is not a vague region, (b) a vague region, (c) a vague multiregion.

the boundary of the support set. Every α-cut of the region is connected, that is, the vague region is Weiss-connected (see page 28). The vague region of Figure 3.11(b) has two cores. It is (Pu-Liu) connected, but not Weiss-connected. The vague region of Figure 3.11(c) has a single-component core, and it contains holes. It is Weiss connected. The vague region of Figure 3.11(d) has several cores and several holes.

Figure 3.11: Vague region objects.

We want the support set of a vague region to be a crisp region, and the membership function to be almost everywhere continuous in the support set, allowing stepwise jumps along linear features. A crisp region is bounded, regular closed, and with connected interior. Figure 2.4(c) illustrates a set that is bounded, regular closed, and connected, but its interior (Figure 2.4(d)) is not connected. Figure 2.3(d) illustrates a set that is bounded, regular closed, and has a connected interior. The set is taken from the regular closure of the set of Figure 2.3(a).

We require the same properties to be satisfied by a vague region in the fuzzy

topology setting: bounded, regular closed, and have a connected interior for $(\mathbb{R}^2, \mathcal{T}_2)$. Bounded and connected interior property of the fuzzy set in $\mathcal{T}_2$ assure that the same properties are satisfied by its support set in T_2. A regular closed fuzzy set $\mu = \overline{\mu^\circ}$ does not have cuts, punctures, isolated lines, or isolated points. The regular closure assures stronger properties than the upper semi-continuity: the discontinuities are stepwise jumps along lines; no isolated discontinuities are allowed, and no discontinuity from both sides of a line occur. It satisfies the membership function requirement. The set of vague regions is then defined as

$$\mathit{VRegion} \equiv \left\{\mu \in \mathcal{F}(\mathbb{R}^2) \mid \mu \text{ is bounded}, \mu = \overline{\mu^\circ}, \mu^\circ \text{ is connected}\right\}.$$

The highest membership value may be less than 1. The regular closure property for $\mathcal{T}_2$ excludes the possibility for a vague line or vague multiline to be a vague region. A crisp region is a specific case of a vague region, when the set μ is a crisp set.

A *vague multiregion* represents a class of vague region objects. It is a multi-component fuzzy set that is bounded and regular closed. Figure 3.10(b) illustrates a vague multiregion with only one component, and Figure 3.10(c) illustrates a multi-component region. The set of vague multiregions is defined as

$$\mathit{VMRegion} \equiv \left\{\mu \in \mathcal{F}(\mathbb{R}^2) \mid \mu \text{ is bounded}, \ \mu = \overline{\mu^\circ}\right\}.$$

A vague region is a special case of a vague multiregion, being a region with a single component. A vague multiregion can also be empty.

A vague extension is a collection of vague multiregions and vague multilines. The set of vague extensions is defined as

$$\mathit{VExt} \quad \equiv \quad \{\mu \in \mathcal{F}(\mathbb{R}^2) \mid \exists \nu \in \mathit{VMRegion}, \exists \upsilon \in \mathit{VMLine}, \ \mu = \nu \sqcup \upsilon\}.$$

A *VExt* object μ can be a vague multiregion if the line component υ is empty, and it can be a vague multiline if its region component ν is empty.

3.5.2 Vague region operators

The union, intersection, and difference operators for vague regions are regularized fuzzy set operators. The type *VMRegion* is closed under these operators, i.e., the union, intersection or difference of two vague multiregions is a vague multiregion. The *union* of two vague multiregions is simply the fuzzy set union. The union of two bounded fuzzy sets is a bounded set. The union of two regular closed fuzzy sets is a regular closed fuzzy set. Therefore, the fuzzy set union of two vague multiregions produces a vague multiregion. The union operator *RUnion* is defined as

$$\begin{gathered}\mathit{RUnion} : \mathit{VMRegion} \times \mathit{VMRegion} \to \mathit{VMRegion} \\ \forall \mu, \nu \in \mathit{VMRegion}, \ \mathit{RUnion}(\mu, \nu) = \mu \sqcup \nu.\end{gathered}$$

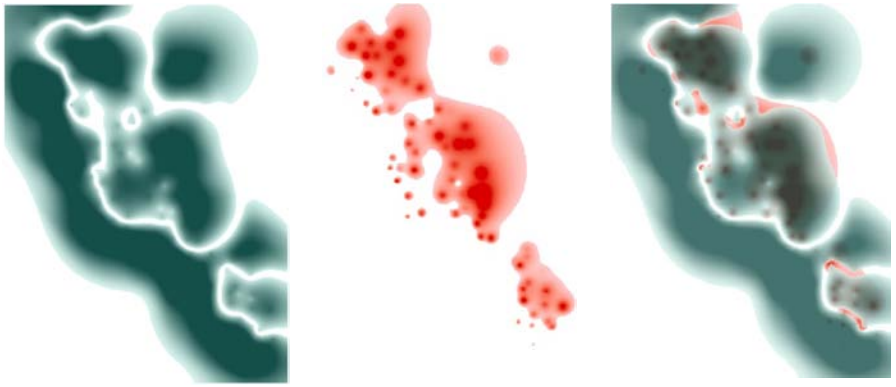

Figure 3.12: Two vague multiregions overlayed: (a) and (b) vague multiregions, (c) overlayed and displayed by using transparency for the top region.

The *intersection* of two vague multiregions is the regular closure of their fuzzy set intersection. Fuzzy intersection of two bounded fuzzy sets is a bounded fuzzy set. Fuzzy intersection of two regular closed fuzzy sets is not always regular closed. We obtain a vague multiregion by applying the regular closure on the result of the fuzzy intersection of two vague multiregions. The interior of a fuzzy intersection is equal to the fuzzy intersection of the interiors. Therefore, we can define the intersection operator between regions as

$$RInterection : VMRegion \times VMRegion \rightarrow VMRegion$$
$$\forall \mu, \nu \in VMRegion,\ RInterection(\mu, \nu) = \overline{\mu^\circ \sqcap \nu^\circ}.$$

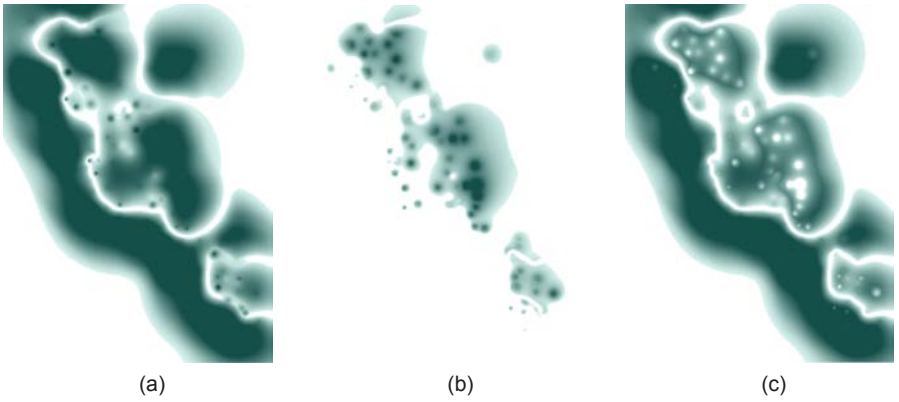

Figure 3.13: Results of operators on vague multiregions of Figure 3.12: (a)–(c) union, intersection, and difference, respectively.

The *difference* operator between two vague multiregions μ and ν is built from the fuzzy difference, which in turn is defined in terms of fuzzy intersection:

$\mu \sqcap (1^{\mathbb{R}^2} - \nu)$. The fuzzy difference between two vague multiregions produces a bounded fuzzy set, but not always a regular closed fuzzy set. Again, we apply the regular closure on the result of the fuzzy operator, in order to get a vague multiregion. The interior of the fuzzy intersection is equal to the intersection of the interiors, and the complement of ν is an open set. We can, therefore, define the difference between vague multiregions as

$$RDifference : VMRegion \times VMRegion \rightarrow VMRegion$$
$$\forall \mu, \nu \in VMRegion,\ RDifference(\mu, \nu) = \overline{\mu^{\circ} \sqcap (1^{\mathbb{R}^2} - \nu)}.$$

The *boundary* of a vague multiregion μ is its uncertain part, and it is constructed from the fuzzy boundary μ^b. The boundary of vague multiregion may consist of vague regions, vague lines, or both. It is a vague extension. The boundary of a crisp region consists only of lines, whereas the boundary of a vague multiregion with continuous membership function is a vague multiregion. Figure 3.14(a) illustrates a vague multiregion with its core boundary drawn in red, and Figure 3.14(b) shows its boundary, which is a vague extension. The boundary operator is defined as:

$$RBoundary : VMRegion \rightarrow VExt$$
$$\forall \mu \in VMRegion,\ RBoundary(\mu) = \mu^b.$$

The boundary of a vague multiregion μ is the restriction of μ to locations with membership values smaller than 1, extended by the boundary of the core: $RBoundary(\mu) = \mu^{|\{p \in \mathbb{R}^2 \mid 0 < \mu(p) < 1\} \cup \partial\mu_1}$

Figure 3.14: Boundary and frontier of a vague multiregion: (a) a vague region with the core boundary in red, (b) its boundary and (c) its frontier.

The *frontier* of a vague multiregion μ is calculated as the fuzzy frontier μ^f. The frontier operator on vague multiregions returns a vague multiline. If μ has discontinuities, the frontier returns all lines of discontinuity. When μ is continuous,

its frontier is empty. The operator *RFrontier* is defined as:

$$\begin{gathered} \mathit{RFrontier} : \mathit{VMRegion} \to \mathit{VMLine} \\ \forall \mu \in \mathit{VMRegion},\ \mathit{RFrontier}(\mu) = \mu^f. \end{gathered}$$

Both operators, the boundary and the frontier, produce the crisp boundary when applied to crisp multiregions.

3.6 Vague partitions and their operators

In practical applications, vague multiregions may originate from a soft classification of space, for example based on remote sensing imagery. Vague multiregions representing different classes may not be disjoint, as we expect transition zones to intersect with each other. A soft classification cannot give a crisp partition of space, but some characteristics of such a partition should be kept to make a meaningful classification in space. A *vague partition* serves this purpose. It is a collection of vague multiregions that may intersect only at their uncertain parts. The core of one region can intersect with the support set of the other region only at their boundaries. The set of vague partitions is defined as

$$\begin{aligned} \mathit{VPartition} \quad \equiv \quad & \{\{\mu_i\}_{i=1}^n \subset \mathit{VMRegion} \mid \forall i, j \in \{1 \ldots n\}, i \neq j \Rightarrow \\ & \mu_i \sqcap \mu_j \sqsubseteq \mathit{RBoundary}(\mu_i) \sqcap \mathit{RBoundary}(\mu_j)\}. \end{aligned}$$

To this definition we might add a condition that any location has a positive membership to at least one vague class: $\forall p, \exists i, \mu_i(p) > 0$.

The operators we define for vague partitions are the overlay and the fusion operator. The *overlay* operator *VPOverlay* superimposes two vague partitions, and creates a new vague partition with vague multiregions obtained from the intersections of a vague multiregion of the first partition with a vague multiregion of the second partition. It is defined as

$$\begin{gathered} \mathit{VOverlay} : \mathit{VPartition} \times \mathit{VPartition} \to \mathit{VPartition} \\ \forall \mathcal{P}_1 = \{\mu_i\}_{i=1}^n, \mathcal{P}_2 = \{\nu_j\}_{j=1}^m \in \mathit{VPartition}, \\ \mathit{VOverlay}(\mathcal{P}_1, \mathcal{P}_2) = \left\{\zeta_{i,j} \mid i \in \{1 \ldots n\}, j \in \{1 \ldots m\}, \zeta_{i,j} = \mathit{RInterection}(\mu_i, \nu_j)\right\}. \end{gathered}$$

It can be shown that the set $\left\{\zeta_{i,j} \mid i \in \{1 \ldots n\}, j \in \{1 \ldots m\}\right\}$ forms a vague partition. The overlay operator combines two vague classifications of space, and creates a new classification that is more refined.

The *fusion* operator dissolves a vague partition by merging vague multiregions based on grouping or equality of some attribute value of regions. The operator assumes an attribute to be associated to vague multiregions of a vague partition. Let us call such a partition an *attribute extended vague partition*, and let us

denote by ADomain the domain of attribute values. The set of such partitions is

$$AVPartition \equiv \left\{ \{(\mu_i, v_i)\}_{i=1}^{n} \subset VMRegion \times \mathsf{ADomain} \mid \{\mu_i\}_{i=1}^{n} \in VPartition \right\}.$$

For simplicity we consider one attribute attached to the vague regions of a partition. The set ADomain can generally be a Cartesian product of domains of several attributes. A grouping of attribute values is a function $g : \mathsf{ADomain} \to \mathsf{ADomain}$. This function is defined on the assumption that the group values are in the same domain ADomain. Such a function g is an element of the power set $\mathbb{P}(\mathsf{ADomain} \times \mathsf{ADomain})$, the collection of subsets of the Cartesian product of ADomain with itself. The fusion operator is then defined as

$$\begin{aligned} &VFusion : AVPartition \times \mathbb{P}(\mathsf{ADomain} \times \mathsf{ADomain}) \to AVPartition \\ &\forall \mathcal{A} = \{(\mu_i, v_i)\}_{i=1}^{n} \in AVPartition,\ \forall g \in \mathbb{P}(\mathsf{ADomain} \times \mathsf{ADomain}), \\ &VFusion(\mathcal{A}) = \Big\{ \{(\zeta_j, w_j)\}_{j=1}^{m} \mid \{w_j\}_{j=1}^{m} = \mathrm{ran}(g), \\ &\qquad \forall j \in \{1 \ldots m\}, \zeta_j = \bigsqcup \{\mu_i \mid g(v_i) = w_j\} \Big\}. \end{aligned}$$

The generalized fuzzy union of vague multiregions has a vague multiregion as its output, therefore ζ_j's are *VMRegion* objects. It can be shown that the set $\left\{\zeta_j\right\}_1^m$ forms a vague partition. The fusion operator allows to generalize a vague partition.

3.7 Discussions and Conclusions

The spatial types introduced in this chapter reflect an ontological view of vagueness. They are adequate to represent objects from spatial phenomena where vagueness is inherent. We assume that vagueness is present in properties of the objects, and not in their location. The proposed vague region types, however, can also cover locational vagueness. To present points and lines exhibiting locational vagueness we could use the vague region types with an additional restriction that the membership value should not reach the value 1. This restriction is based on the assumption that a location with a membership value equal to 1 gives with certainty the point, or a part of the line extension. The real differentiation between the objects, points, lines, and regions, would then be left to their operators, which should hold a different semantic for each object type.

This chapter provided the set of vague spatial types that we use throughout this thesis. The vague types are separated into simple and general ones. The simple types represent atomic objects, i.e., identifiable entities that are not composed of others. A simple object is an element of the set

$$SVSpatial = VPoint \cup VLine \cup VRegion.$$

To such objects we can assign attribute values, which enrich the geometric data, and give it the real importance for use in different applications. The general

types assure closure under operators. They represent classes of simple objects that have the same value on a specific attribute. A class of simple objects is an element of the set

$$GVSpatial = VMPoint \cup VMLine \cup VMRegion.$$

Vague partitions allow for a soft classification of space. The two other spatial types, *VLDim* and *VExt* assure closure under the boundary operators. The set of all vague spatial types is

$$VSpatial = SVSpatial \cup GVSpatial \cup VLDim \cup VExt \cup VPartition.$$

The vague types proposed are such that they include crisp objects as special cases.

The vague operators defined in this chapter are the operators returning spatial types. One group of operators are the regularized fuzzy set operators: union, intersection, and difference. Other operators can be built from those, e.g., symmetric difference. The other group consists of two operators from topology: boundary and frontier. The boundary operator extracts the transition zone, which is the main characteristic of a vague object. The frontier gives the locations where abrupt changes occur on the membership values of a vague object. The last group of operators are the operators on vague partitions. The overlay operator combines two vague partitions to form a new one more refined, and the fusion operator allows for generalization of a vague partition. All the operators are equivalent with their crisp correspondents when applied to crisp objects. The frontier operator preserves the relation between objects of different dimension, which is the property of the crisp boundary operator.

Chapter 4

Spatial relations between vague objects

Spatial relations between objects are an essential information source for reasoning about space. They are mostly binary relations between objects either of the same type or of different types. Examples of such relations are disjoint, overlap, or equal. A few models exists for relations between crisp objects, each proposing a complete set of relations[1] and defining them formally. Most of the relations proposed in different models are intuitively the same, following common understanding and named in natural language terms. The objective of this chapter is to define spatial relations between vague objects, by extending proposed relations between crisp objects (crisp relations hereafter).

This chapter provides mathematical definitions of spatial relations between vague multipoints, vague multilines and vague multiregions. The spatial relations provided take value in the interval $[0,1]$. A value v between 0 and 1 for a relation $R(\mu,\nu)$, means that objects μ and ν are in relation R to the degree v. A value 0 for $R(\mu,\nu)$ means that μ and ν are certainly not in relation R, whereas a value 1 means the two are certainly in R. Spatial relations are defined from membership values of the objects involved, considering extreme values that support a relation or disapprove it. Relations are such that only one relation is certain at a time. They include crisp relations as special cases, meaning, when applied to crisp objects they return the same result as the corresponding crisp relation.

The chapter is organized as follows. Section 4.1 summarizes the main approaches on spatial relations between crisp and vague objects: the 4– and 9–intersection models[37, 39] including SQL/MM [1], the Region Connection Calculus [72, 73] (RCC model), and their extensions to relations between vague objects. Our set of relations between vague objects is the same with the SQL/MM set of spatial relations, and follows the intuition behind them. Therefore, we put more attention to the description of SQL/MM spatial relations. Section 4.2 provides the definitions of spatial relations between vague objects. Section 4.3 describes properties

[1] The term complete here means the set of relations is pairwise disjoint and jointly exhaustive.

of these relations. Section 4.4 discusses another way of defining spatial relations as a kind of average over membership values of the objects involved. Section 4.5 summarizes the chapter.

4.1 Previous work

Topology and mereology are two main approaches to represent space and reason about important characteristics of it. Topology [60] takes points as primitives and builds objects as sets of points having specific properties like being open or closed, compact, connected, and so forth[2]. Mereology (from the old Greek $\mu\varepsilon\rho o\varsigma$, 'part') is the theory of parthood relations — relations of part to whole and relations of part to part within a whole [103]. Mereology takes regions as primitives, and does not consider lower dimension objects, points and lines.

Formal models for representing objects in space and their spatial relations are provided by both approaches. The 4– and 9–intersection models [35, 36, 37, 38, 39] are based on topology, whereas the RCC model [20, 72, 73, 93] is based on mereology. Both models deal with relations between crisp objects. Several models have been proposed subsequently on spatial relations between vague objects, following one of the crisp models. Most of them [7, 18, 19, 21, 22, 79, 94] are dedicated to relations between vague regions, considering them as broad boundary regions. Others [80, 83, 84, 97, 116, 117] employ fuzzy set theory to model spatial objects, considering gradual transitions in the broad boundary. Some of these models extend the set of true–false relations to allow for more configurations [18, 21, 97]. Others keep the same set of relations extending their values to three, including 'maybe' [41, 79] or to the infinite $[0,1]$ interval [83, 84, 116]. The upcoming sections summarize the main models of crisp relations, and relations between vague objects.

4.1.1 Spatial relations between crisp objects

The 9–intersection model identifies the spatial relations between two objects by first partitioning the space into interior, boundary, and exterior for each object, followed by the identification of meaningful configurations from the intersections of any combination of two parts. Each configuration excludes the other, and all cover the whole range of possibilities, making the relations pairwise disjoint and jointly exhaustive. The model built in that way presents the relations as independent from each other. The RCC model on the other side, captures the logical relationship between different spatial relations. It only deals with region objects, ignoring points and lines. It considers a connection relation as the basic relation, and expresses all other relations in terms of connection di-

[2] Section 2.2 gave a short treatment of some basic topological properties.

rectly or through other relations, via logical formulas. It proposes two sets of pairwise disjoint and jointly exhaustive relations. The next section discusses the 9–intersection model together with its derivatives, describing in more detail the SQL/MM relations. The section after that discusses the RCC model, and the correspondence between its relations and the 9–intersection model relations.

The 9–intersection model

Egenhofer and Franzosa [37] proposed a 4–intersection model that identifies binary spatial relations between two regions based on the intersections of their interiors and boundaries. For two regions A and B, the model determines the empty/non-empty value of the intersections between the interior A° or the boundary ∂A of A, and the interior B° or the boundary ∂B of B. From 2^4 combinations only eight configurations are possible, giving in turn eight spatial relations. Figure 4.1 illustrates these relations together with the values of the intersections between interiors and boundaries.

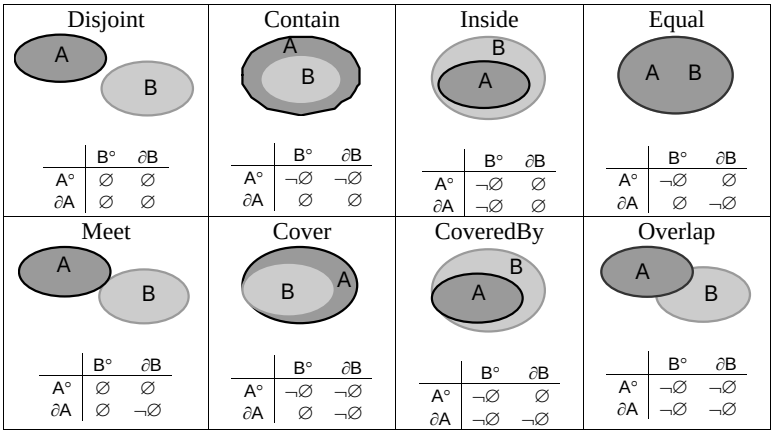

Figure 4.1: Spatial relations between regions identified by the 4–intersection model, and the respective empty/non-empty values of the interior and boundary intersections.

The 4–intersection model was refined in [35, 38] with further topological invariants, e.g., dimension of intersection components, type of components, etc., to account for more detailed spatial relations. In [39], the 4–intersection model was generalized for n-dimensional spaces, forming a 9–intersection model. The objects of space are n-cells, defined from combinatorial topology [2]. Spatial relations between n-cells are defined considering the empty/non-empty intersections of interiors, boundaries, and exteriors of two n-cells. The 9–intersection model gives the same set of eight spatial relations when applied to 2-cells (regions).

The 9–intersection model extended by the dimension of the intersections, DE-9IM, was used to define SQL/MM spatial relations [1]. In what follows, we denote the exterior of a region A by A^-. The derivation of relations between regions A and B can be expressed concisely by the matrix:

$$R(A,B) = \begin{bmatrix} Dim(A^\circ \cap B^\circ) & Dim(A^\circ \cap \partial B) & Dim(A^\circ \cap B^-) \\ Dim(\partial A \cap B^\circ) & Dim(\partial A \cap \partial B) & Dim(\partial A \cap B^-) \\ Dim(A^- \cap B^\circ) & Dim(A^- \cap \partial B) & Dim(A^- \cap B^-) \end{bmatrix}$$

Combinatorial topology is used to define the interior, boundary and exterior of objects. A multipoint is a 0–dimensional object; a multiline is a 1–dimensional object; a multipolygon[3] is a 2-dimensional object. The boundary of a multipoint is the empty set. The boundary of a multiline is a multipoint, consisting of the end nodes (at an odd number) of its line components. The boundary of a multipolygon is a multiline, consisting of the set of linear rings that are the outer boundary and holes boundaries of each polygon component. The interior of an object is the difference between the object and its boundary.

The SQL/MM spatial relations, Disjoint, Touches, Crosses, Overlaps, Within, and Equal, are defined as binary operators on objects of any type. An SQL/MM relation is undefined for a combination of object types for which it is always false. In the description of these relations that we give below, we consider them to be defined for all the types, noting the combination of types for which the relation is always false. We do not always follow the formal definition of a relation as it is given in ISO standard, replacing it with a simpler form that is good enough for our purpose. For the definitions of relations we use A and B to denote objects of any type. For the illustrations we use different grey levels to show different objects. The relations are defined as:

- Disjoint(A, B) is true if *objects do not intersect*: $A \cap B = \emptyset$.
- Touches(A, B) is true if *objects intersect* but *their interiors do not*: $(A \cap B \neq \emptyset) \wedge (A^\circ \cap B^\circ = \emptyset)$. A multipoint–multipoint relation would therefore

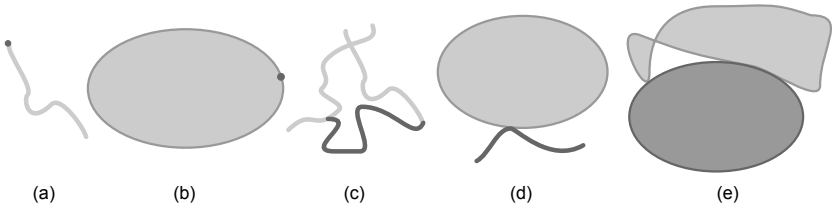

Figure 4.2: The Touches relation between objects of different dimension: (a) a point touching a line, (b) a point touching a region, (c) two touching multilines, (d) a line touching a region, (e) two touching multiregions.

[3] A polygon is a computer representation of a region. For consistency with our labelling we use the term region in the description of the relations.

always be 'false'. Figure 4.2 illustrates the relation for all the other combinations. The Touches relation between multiregions is equivalent to the Meet relation of the 4-intersection model.

- Crosses(A, B) is true if *object interiors intersect in a lower dimension* than the maximal dimension of the objects, and *neither object is subset of the other*:
  $(-1 < Dim(A^\circ \cap B^\circ) < \max(Dim(A), Dim(B))) \wedge A \nsubseteq B \wedge B \nsubseteq A$.
  Dimension more than -1 means the set is not empty. Only multiline–multiline and multiline–multiregion relations are possible (see Figure 4.3 for illustrations). All the other relations are always 'false'.

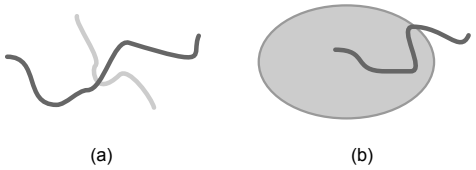

Figure 4.3: The Crosses relation between multilines and multiregions: (a) two crossing lines, (b) a line crossing a region.

- Overlaps(A, B) is true if *object interiors intersect in the same dimension* that the objects have themselves, but *neither object is subset of the other*:
  $(Dim(A^\circ \cap B^\circ) = Dim(A) = Dim(B)) \wedge A \nsubseteq B \wedge B \nsubseteq A$.
  Consequently, the relation is 'false' for objects of different dimension. Figure 4.4 illustrates the relation between multilines and multiregions.

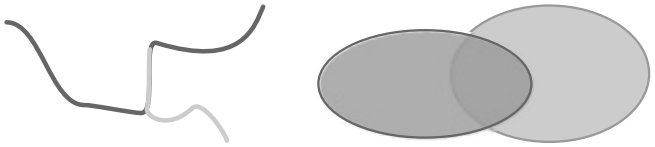

Figure 4.4: The Overlaps relation between multilines and multiregions: (a) two overlapping lines, (b) two overlapping regions.

- Within(A, B) is true if *the first object is a proper subset of the second*: $A \subset B$. Figure 4.5 illustrates the relation for different combination of objects. The Within multiregion–multiregion relation can be any of the 4-intersection relations CoveredBy or Inside.
- Equals(A, B) is true if objects are equal (*each object is subset of the other*): $A = B$. The relation is obviously 'false' for objects of different dimension.

Figure 4.6 shows the hierarchy of SQL/MM spatial relations. Relations in grey boxes are the SQL/MM relations. The symbol $\top$ (top) shows an arbitrary relation.

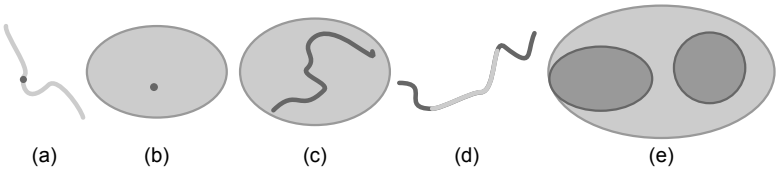

Figure 4.5: The Within relation between different dimension objects: (a) a point within a line, (b) a point within a region, (c) a line within a region, (d) two lines one within the other, (e) a multiregion within a region.

The double arrowed lines show equivalence, and are used for renaming a node of the tree to the corresponding SQL/MM spatial relation. All the (other) branches going out of a node form relations that are pairwise disjoint. The SQL/MM spatial relations are the leaf nodes of the tree, and are therefore pairwise disjoint. They are the only leaf nodes, therefore their union gives the arbitrary relation $\top$. That means, the relations are jointly exhaustive.

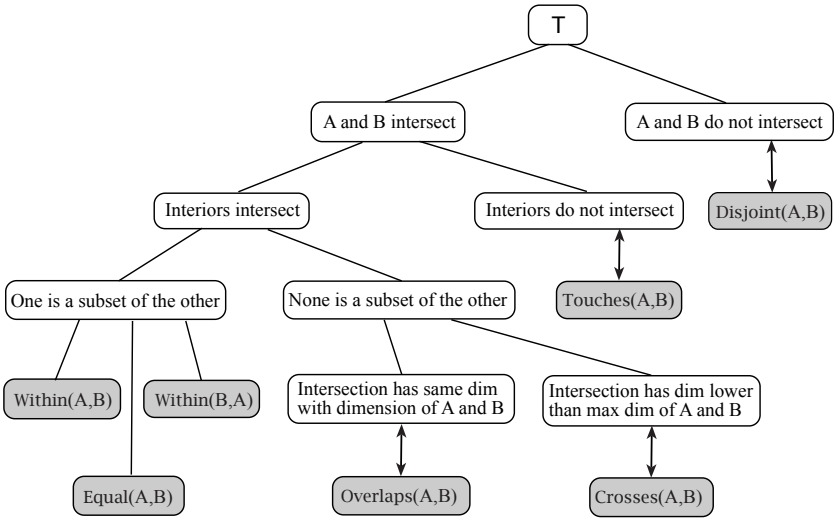

Figure 4.6: The hierarchy of SQL/MM spatial relations.

Region Connection Calculus

The RCC model captures the dependence between different spatial relations between regions. A connection relation ***C*** is the basic relation. Two regions A, B are in ***C*** relation if they share at least one location, i.e., one common point occurs in A and B. Other spatial relations are defined from ***C*** using first order logic formulas. Table 4.1 lists the relations identified by the model and the logical formulas

connecting them. Relation names are in bold face; non bold face A, B, and C, are RCC regions.

Relation	*Interpretation*	*Definition of relation*
DC(A, B)	A disconnected from B	$\neg \boldsymbol{C}(A, B)$
P(A, B)	A is part of B	$\forall C[\boldsymbol{C}(A, C) \Rightarrow \boldsymbol{C}(C, B)]$
PP(A, B)	A is proper part of B	$\boldsymbol{P}(A, B) \wedge \neg \boldsymbol{P}(B, A)$
EQ(A, B)	A coincides with B	$\boldsymbol{P}(A, B) \wedge \boldsymbol{P}(B, A)$
O(A, B)	A overlaps B	$\exists C[\boldsymbol{P}(C, A) \wedge \boldsymbol{P}(C, B)]$
DR(A, B)	A discrete from B	$\neg \boldsymbol{O}(A, B)$
PO(A, B)	A partially overlaps B	$\boldsymbol{O}(A, B) \wedge \neg[\boldsymbol{P}(A, B) \vee \boldsymbol{P}(B, A)]$
EC(A, B)	A externally connected to B	$\boldsymbol{C}(A, B) \wedge \neg \boldsymbol{O}(A, B)$
TPP(A, B)	A is tangential proper part of B	$\boldsymbol{PP}(A, B) \wedge \exists C[\boldsymbol{EC}(C, A) \wedge \boldsymbol{EC}(C, B)]$
NTPP(A, B)	A is non-tangential proper part of B	$\boldsymbol{PP}(A, B) \wedge \neg\exists C[\boldsymbol{EC}(C, A) \wedge \boldsymbol{EC}(C, B)]$

Table 4.1: Spatial relations definable in terms of connection ***C*** (taken from [20]).

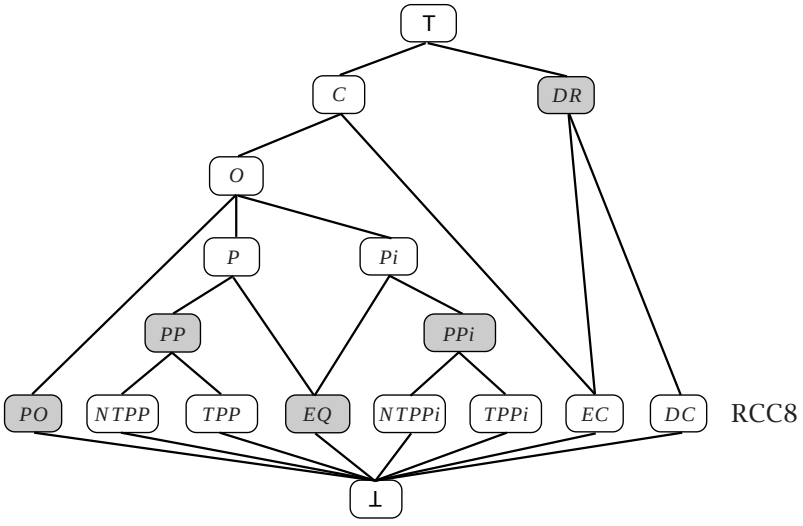

Figure 4.7: The hierarchy of RCC spatial relations (taken from [20]).

The complete set of RCC relations and their relationships is shown in Figure 4.7. The relations form a lattice, in which the order models an 'IS-A' relationship. For example, a proper part relation (***PP***) is a part-of relation (***P***). The symbol $\top$ (top) shows an arbitrary relation. The symbol $\bot$ (bottom) shows an impossible relation. Two regions are either connected or separated. This means they are either

in a ***C*** or a ***DR*** relation. The set of relations ***C*** and ***DR*** are jointly exhaustive, but they are not disjoint. The relation ***EC*** is a ***C*** and a ***DR*** relation. Two relations are pairwise disjoint if one excludes the other. Two models, RCC8 and RCC5, are built from these relations, each consisting of pairwise disjoint and jointly exhaustive relations. Relations of the level just above $\perp$ form the RCC8 model. They are pairwise disjoint: any combination or relations is impossible. They are jointly exhaustive, which can be seen by following up the hierarchy of Figure 4.7. Relations in grey boxes form the RCC5 model. Following the hierarchy, it can be seen that they are pairwise disjoint and jointly exhaustive as well.

The RCC8 and the 4–intersection model result intuitively in the same set of relations. Figure 4.8 illustrates RCC5, RCC8, 4–intersection relations, and the relationships between them.

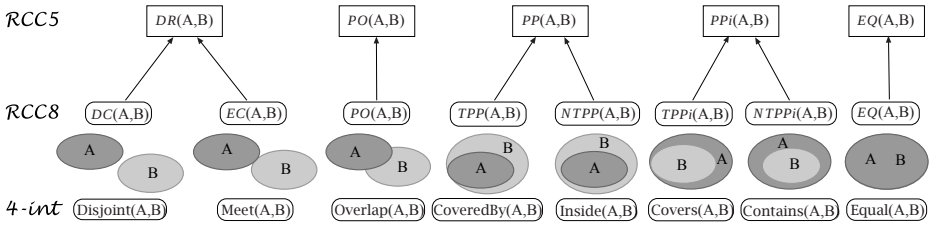

Figure 4.8: Relations identified by RCC5, RCC8, 4–intersection model, and their correspondence.

A formal correspondence between the two models cannot be established easily, because they are based on fundamentally different theories. Proofs exist, though, that point-sets satisfying certain topological properties can be used to build RCC models [45, 93]. Stell [93] proves that regular closed sets on a connected, regular topological space X can serve as a model for region connection calculus. Connection relation ***C*** is modelled by the connectedness property, as defined in Section 2.2.

4.1.2 Spatial relations between vague regions

Several models have been proposed to describe spatial relations between vague or indeterminate regions [18, 19, 21, 41, 79, 83, 84, 85, 94, 97, 112, 116, 117]. Some of these models can handle both indeterminacy and vagueness, whereas others are appropriate only for one of these characteristics. Vagueness can be ontological or linguistic. Ontological vagueness means no sharp boundaries but gradual transition. Linguistic vagueness and indeterminacy, on the other hand, mean impossibility of determining where the sharp boundary lies. The impossibility of knowing the sharp boundary for indeterminacy can be seen as lack of knowledge, which could be obtained in some way. This is typically not the case for linguistic vagueness. Models of spatial relations extend the number of known binary relations for crisp regions to allow additional configura-

tions [18, 19, 21, 97], or they keep the same relations as for crisp regions, but extend their values from two (true, false) to three [41, 79], to a six-valued lattice [94, 112], or to the $[0,1]$ interval [83, 84, 85, 116, 117].

The Egg-Yolk model [21] builds the spatial relations between egg-yolk regions from RCC5 relations applied to the eggs and yolks of the regions. Denoting by R a variable taking a value from the set of RCC5 relations, and by $\mathcal{y}(\mathrm{A})$ and $e(\mathrm{A})$ the yolk and egg of an egg-yolk region A, respectively, the egg-yolk relations between A and B are identified by considering four different combinations in the matrix:

$$\begin{bmatrix} R(\mathcal{y}(\mathrm{A}),\mathcal{y}(\mathrm{B})) & R(\mathcal{y}(\mathrm{A}),e(\mathrm{B})) \\ R(e(\mathrm{A}),\mathcal{y}(\mathrm{B})) & R(e(\mathrm{A}),e(\mathrm{B})) \end{bmatrix}.$$

Because yolks and eggs of any egg-yolk region are by definition in a part-of relation, only 46 of the 5^4 combinations are possible. This gives 46 different two-valued relations.

Clementini and di Felice [18, 19] define spatial relations between broad boundary regions using the 9–intersection model, replacing the line boundary with the broad boundary. Relations are identified by considering the empty/ non-empty values of the intersections:

$$\begin{bmatrix} \mathrm{A}^\circ \cap \mathrm{B}^\circ & \mathrm{A}^\circ \cap \triangle\mathrm{B} & \mathrm{A}^\circ \cap \mathrm{B}^- \\ \triangle\mathrm{A} \cap \mathrm{B}^\circ & \triangle\mathrm{A} \cap \triangle\mathrm{B} & \triangle\mathrm{A} \cap \mathrm{B}^- \\ \mathrm{A}^- \cap \mathrm{B}^\circ & \mathrm{A}^- \cap \triangle\mathrm{B} & \mathrm{A}^- \cap \mathrm{B}^- \end{bmatrix}.$$

Interior, broad boundary $\triangle$, and exterior of any region are related to each other. Therefore, only 44 of the 2^9 combinations of empty/non-empty values are possible. From those, 42 correspond to relations of the egg-yolk calculus. The other two are divided in two different relations from the egg-yolk calculus [79].

Tang [97] extends the 3×3 matrix of the 9–intersection model to a 4×4 matrix, which is also checked for empty/non-empty values of the intersections. Using the notation of Clementini and di Felice model, the broad boundary is decomposed into its interior, $(\triangle\mathrm{A})^\circ$, and its boundary, $\partial(\triangle\mathrm{A})$. The broad boundary is replaced by these two components forming a 4×4 matrix. Figure 4.9 shows a fuzzy set μ, and the elements extracted from it to define the matrix of topological relations: the interior of the core, the interior of $supp(\mu^b)$, and the boundary of $supp(\mu^b)$. The matrix is formed from these three elements, and the exterior of $supp(\mu)$. Tang identifies 152 topological relations for vague regions. The 9–intersection model is extended in a similar way for topological relations between the other object types, vague points and vague lines, and their relations with vague regions. The number of relations proposed by this model and the previous two is relatively big, which makes them somehow unpractical for use in real applications.

Erwig and Schneider [41] extend the values of known spatial predicates for crisp regions to three–valued predicates — true, maybe, false. They build spatial

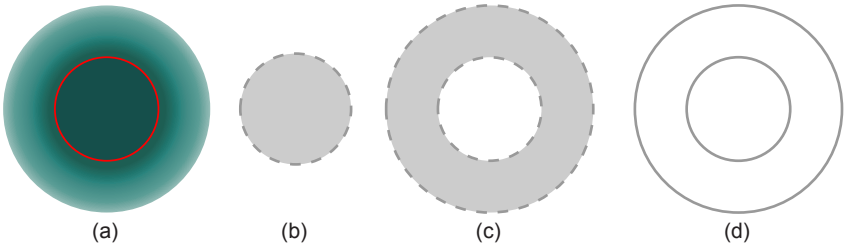

Figure 4.9: Elements of the 4×4 matrix: (a) a fuzzy set μ, and its core boundary drawn in red, (b) the interior of μ's core, (c) the interior of the support set of μ's boundary, (d) the boundary of the support set of μ's boundary.

relations between two vague regions from relations between their kernels and boundaries, coming up with three–valued relations between vague regions.

Vague regions are represented as rough sets in [79, 94, 112]. The RCC model is used to define spatial relations between vague regions, which extend the values of RCC relations to three, true, maybe, and false. Roy and Stell [79] propose a model for indeterminate regions and their spatial relations that is an extension of the RCC model. Logical formulas expressing the dependencies between RCC spatial relations, hold in a three–valued Łukasiewicz logic for vague relations proposed in [79]. These vague relations fall into RCC relations when applied to crisp regions. The model is appropriate for expressing relations between vague regions having a linguistic type of vagueness.

The previous two models propose spatial predicates that take three values. They are appropriate for vague regions which membership values are also in that range, true, maybe, and false. The membership values of fuzzy regions cover a wider range, the $[0,1]$ interval. Zhan [116, 117] and Schneider [83, 84, 85] propose relations that take values in the same range with their membership values, the $[0,1]$ interval. They define a fuzzy relation for each crisp relation of the 4–intersection model. A fuzzy relation R between two fuzzy regions μ and ν is calculated from the corresponding crisp relation R using a finite number of α-cuts, $\alpha_1 = 1 > \alpha_2 > \ldots > \alpha_n = 0$:

$$R(\mu,\nu) = \Sigma_{i=1}^{n-1}\Sigma_{j=1}^{n-1}(\alpha_i - \alpha_{i+1}) \cdot (\alpha_j - \alpha_{j+1}) \cdot \mathsf{R}(\mu_{\alpha_i}, \nu_{\alpha_j}).$$

The value of the relation R depends on the selected α-cuts, i.e., when the set of α values changes, the R value will change as well.

4.2 Spatial relations between vague objects

We restrict ourselves to a small set of relations, those proposed by the SQL/MM spatial. Our relations, *Disjoint*, *Touches*, *Crosses*, *Overlaps*, *Within*, and *Equal*,

follow the intuition behind SQL/MM spatial relations. They are dyadic operators with arguments that are vague objects of a general type. Their return value is a number in the unit interval $[0,1]$. We introduce a new type, $\mathit{TruthDegree} \equiv [0,1]$, to be the return type of spatial relations between vague objects.

Relations are presented below in the order mentioned above. First, the meaning of a relation is described, followed by its definition. Then, we prove that the corresponding crisp relation is a special case of the vague one. In the illustrations of this section, different colours indicate different objects. When an illustration uses two vague multiregions, we draw them in 3D using the membership value as the third coordinate (in the hope that it helps conveying better the idea).

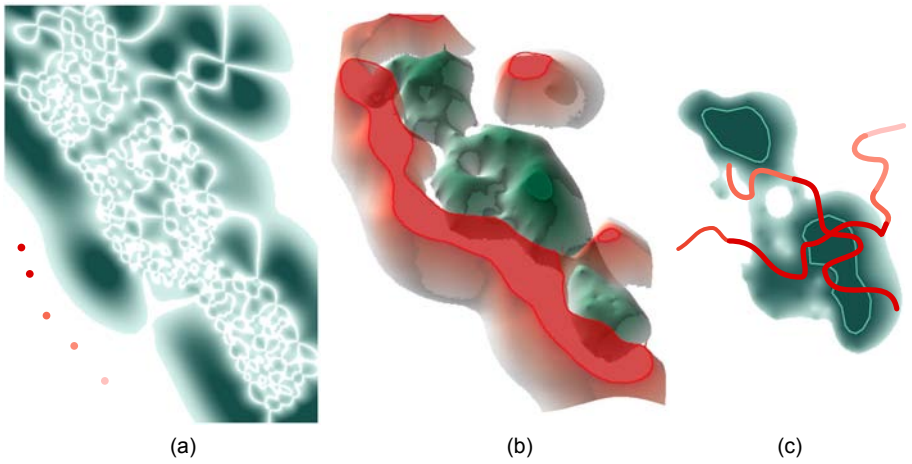

Figure 4.10: Three different cases of a *Disjoint* relation: (a) certainly disjoint vague multipoint and vague multiregion, (b) possibly disjoint vague multiregions, (c) certainly not disjoint vague multiline and vague region.

Two vague objects are certainly disjoint if their support sets are disjoint. They are certainly not disjoint if their cores intersect. In all other cases they are disjoint to some positive degree, dependent on the membership values of the intersection. Figure 4.10(a) illustrates a vague multiregion and a vague multipoint that are disjoint. The two vague multiregions of Figure 4.10(b) are possibly disjoint. Their cores, with boundaries drawn in red and green, respectively, do not intersect, but their support sets do. Figure 4.10(c) shows a vague multiline and a vague region that are certainly not disjoint, because their cores intersect. The boundary of the core of the vague region is drawn in light green, whereas the core of the multiline is shown in fully saturated red.

Two vague objects μ and ν are not disjoint if their intersection is not empty, i.e., $\mu \sqcap \nu \neq 0^{\mathbb{R}^2}$. The intersection disproves their disjointness. We consider the highest membership value of the intersection to be the degree of disapproval. The degree of the relation $\mathit{Disjoint}(\mu,\nu)$ is calculated as the substraction of the degree of disapproval from the full certainty value 1. The relation *Disjoint* is

thus defined as:

$$\mathit{Disjoint} : \mathit{GVSpatial} \times \mathit{GVSpatial} \to \mathit{TruthDegree}$$
$$\forall \mu, \nu \in \mathit{GVSpatial},\ \mathit{Disjoint}(\mu, \nu) = 1 - \sup_{p \in \mathbb{R}^2} \{(\mu \sqcap \nu)(p)\}\,.$$

If μ and ν are *crisp* objects, $\mu \sqcap \nu$ takes only values 0 or 1, as it is the minimum of two characteristic functions. $\mathit{Disjoint}(\mu, \nu)$ is equal to 0 iff there is p such that $\mu(p) = 1$ and $\nu(p) = 1$, that is the two objects share at least one location. It is equal to 1 if and only if there is no such p, i.e., the two objects have no common location. This is the behaviour of the crisp relation Disjoint.

The relations *Touches*, *Crosses*, and *Overlaps*, are defined using the notion of object interior. We first define what is the interior $\mu^\star$ of a vague object μ, for each type. The interior of a vague multipoint μ is the vague multipoint itself: $\mu^\star = \mu$. The interior of a vague line $\mu = \tilde{h}(\eta)$, is the image by h of the fuzzy interior of η for $(\mathbb{R}, \mathcal{T}_1)$: $\mu^\star = \tilde{h}(\eta^\circ)$. Figure 2.15(d) illustrates the membership function for a vague line, and Figure 2.15(c) shows its interior. The interior changes the value of the function only at its discontinuity points, by putting it to the lowest value. The interior of a vague multiline $\mu = \bigsqcup_{i=1}^{n} \mu_i$ is $\mu^\star = \bigsqcup_{i=1}^{n} \mu_i^\star$. The interior of a vague multiregion μ is its fuzzy interior for $(\mathbb{R}^2, \mathcal{T}_2)$: $\mu^\star = \mu^\circ$. It changes the value of the function along discontinuity lines, by putting it to the lowest value. If μ is a crisp object, $\mu^\star$ is the characteristic function of the crisp interior of the object.

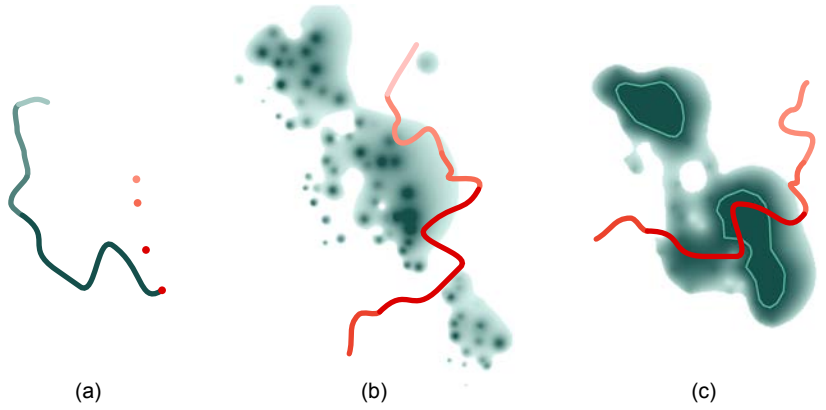

Figure 4.11: Three different cases of a *Touches* relation: (a) certainly touching vague multipoint and vague line, (b) possibly touching vague line and vague multiregion, (c) certainly not touching vague line and vague region.

Two vague objects touch if their boundaries intersect, but the interiors of their cores do not. Figure 4.11 illustrates three cases of *Touches* relation: certainly touching objects, possibly touching, and certainly not touching objects. The vague multipoint of Figure 4.11(a) has a vague point component with membership 1 located at an end node of the vague line, which is in the boundary of the line because it is the end node of the core. The vague line of Figure 4.11(b) passes

through the boundary of the vague multiregion, and their cores do not intersect. The two objects are possibly touching. The cores of the vague line and the vague region of Figure 4.11(c) intersect, therefore the objects are not touching.

The degree of the relation is calculated from the intersection of one object with the boundary of the other, provided that their core interiors do not intersect. The maximal degree of this intersection is considered to be the degree of the relation. Let us denote by $\delta\mu$ the boundary of a vague object μ. If μ is a vague multipoint $\delta\mu = \mathit{PBoundary}(\mu)$; if it is a vague multiline $\delta\mu = \mathit{LBoundary}(\mu)$; for a vague multiregion $\delta\mu = \mathit{RBoundary}(\mu)$. The relation *Touches* is defined as:

$$\mathit{Touches} : \mathit{GVSpatial} \times \mathit{GVSpatial} \rightarrow \mathit{TruthDegree}$$
$$\forall \mu, \nu \in \mathit{GVSpatial},$$

$$\mathit{Touches}(\mu,\nu) = \begin{cases} 0 & \text{if } \mu \sqsubseteq \nu \text{ or } \nu \sqsubseteq \mu \text{ or} \\ & \sup_p \{(\mu^\star \sqcap \nu^\star)(p)\} = 1, \\ \sup_p \{((\mu \sqcap \delta\nu) \sqcup (\delta\mu \sqcap \nu))(p)\} & \text{otherwise.} \end{cases}$$

The conditions in the right side force a *Touches* relation to be false when one object is a subset of the other, or when their cores intersect.

For *crisp* objects μ and ν, the *Touches* relation can only take values 0 and 1, because all the functions involved are characteristic functions. The relation may be true, i.e., its value is equal to 1, only if $\sup_p(\mu^\star \sqcap \nu^\star)(p) < 1$ that means the object interiors do not intersect. The boundaries $\delta\mu$ and $\delta\nu$ are the crisp boundaries $\partial(\mathit{supp}(\mu))$ and $\partial(\mathit{supp}(\nu))$, respectively. The expression $\sup_p \{((\mu \sqcap \delta\nu) \sqcup (\delta\mu \sqcap \nu))(p)\}$ translates to $(\mathit{supp}(\mu) \cap \partial\mathit{supp}(\nu)) \cup (\partial\mathit{supp}(\mu) \cap \mathit{supp}(\nu))$ is empty or not empty. Therefore, the relation $\mathit{Touches}(\mu,\nu)$ is equal to 1 iff $(\mathit{supp}(\mu) \cap \partial\mathit{supp}(\nu)) \cup (\partial\mathit{supp}(\mu) \cap \mathit{supp}(\nu)) \neq \emptyset$, given that their interiors do not intersect. Thus, $\mathit{Touches}(\mu,\nu)$ is true iff μ and ν intersect but their interiors do not, which is the behaviour of the crisp relation Touches.

Two vague objects cross each other to a positive degree if their support sets cross each other. The relation is possible only between vague multilines, or a vague multiline and a vague multiregion. It is always false for any other combination of object types. Two vague multilines possibly cross if their extensions intersect in a finite number of points. A vague multiline possibly crosses a vague multiregion if the line extension intersects the support set of the region without being fully inside it. Figure 4.12 illustrates different cases of the *Crosses* relation. It shows two vague lines that are certainly crossing, because their cores intersect. This is also the case for the vague line and the vague region of Figure 4.11(c). Figure 4.12(b) shows a possibly crossing vague line and vague region. The vague multiline and vague multiregion of Figure 4.12(c) are certainly not crossing, because their support sets are disjoint.

For two vague objects μ and ν, we take the highest value of the intersection of their interiors to be the degree of $\mathit{Crosses}(\mu,\nu)$ relation. The *Crosses* relation is defined in terms of crisp Crosses relation between the support sets of the

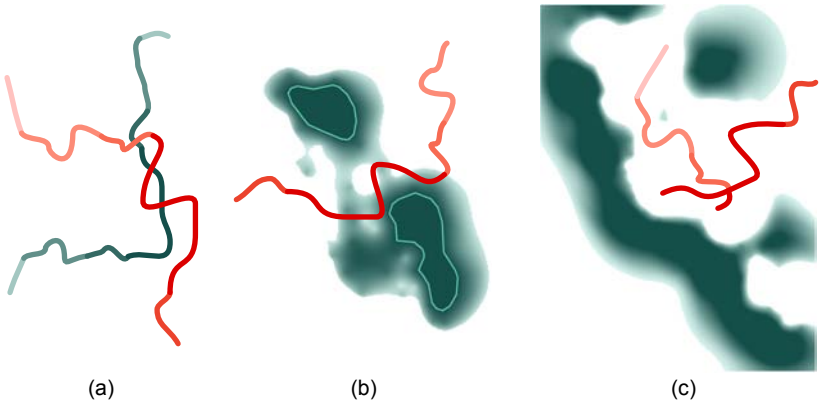

Figure 4.12: Three different cases of a *Crosses* relation: (a) certainly crossing vague lines, (b) possibly crossing vague line and vague region, (c) certainly not crossing vague multiline and vague multiregion.

objects:

$$\mathit{Crosses} : \mathit{GVSpatial} \times \mathit{GVSpatial} \to \mathit{TruthDegree}$$
$$\forall \mu, \nu \in \mathit{GVSpatial},$$
$$\mathit{Crosses}(\mu, \nu) = \begin{cases} \sup_p \{(\mu^\star \sqcap \nu^\star)(p)\} & \text{if } \mathsf{Crosses}(\mathit{supp}(\mu), \mathit{supp}(\nu)), \\ 0 & \text{otherwise.} \end{cases}$$

If μ and ν are *crisp* objects, the expression $\sup_p(\mu^\star \sqcap \nu^\star)(p)$ can only have values 0 or 1, and so does the *Crosses* relation. From the definition it can be seen that $\mathsf{Crosses}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ is false implies that $\mathit{Crosses}(\mu, \nu)$ is equal to 0. Also, if $\mathit{Crosses}(\mu, \nu)$ is equal to 1, then $\mathsf{Crosses}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ is true. There are two more implications that prove the equivalence: the relation Crosses is true implies that *Crosses* is equal to 1, and *Crosses* is equal to 0 implies that Crosses is false. Indeed, $\mathsf{Crosses}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ is true implies that the interiors of μ and ν intersect, which in turn implies that $\sup_p(\mu^\star \sqcap \nu^\star)(p) = 1$. Thus, $\mathit{Crosses}(\mu, \nu)$ is equal to 1. It can be shown similarly that if $\mathit{Crosses}(\mu, \nu)$ is equal to 0, then $\mathsf{Crosses}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ is false.

Two vague objects overlap to a positive degree if their support sets overlap. They certainly do not overlap if their support sets do not intersect. They certainly overlap if the interiors of their cores intersect. The relation is possible only between objects of the same type, vague multipoints, vague multilines, or vague multiregions. It is always false for the other combinations of vague objects. Figure 4.13(a) shows two vague lines that share some part of their cores, and are therefore certainly overlapping. The two vague regions of Figure 4.10(b) are possibly overlapping. The degree of overlap is the highest value of their intersection, which is shown by the grey line. The two vague multipoints of Figure 4.13(c) have disjoint support sets, therefore they certainly do not overlap. The *Overlaps* relation is also defined in terms of the crisp Overlaps between the

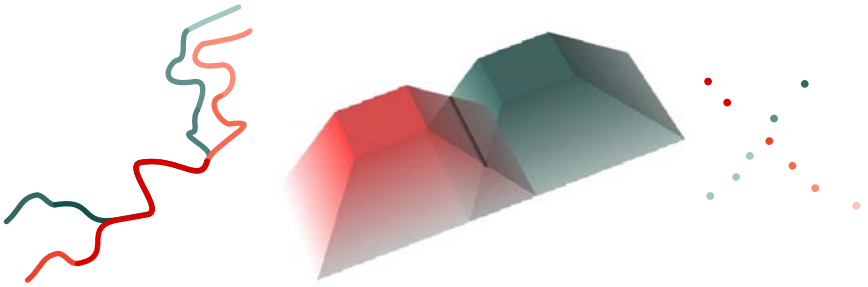

Figure 4.13: Different cases of an *Overlaps* relation: (a) certainly overlapping vague lines, (b) possibly overlapping vague regions, (c) certainly not overlapping vague points.

support sets of the objects:

$$\mathit{Overlaps} : \mathit{GVSpatial} \times \mathit{GVSpatial} \rightarrow \mathit{TruthDegree}$$
$$\forall \mu, \nu \in \mathit{GVSpatial},$$
$$\mathit{Overlaps}(\mu, \nu) = \begin{cases} \sup_p \{(\mu^\star \sqcap \nu^\star)(p)\} & \text{if } \mathsf{Overlaps}(\mathit{supp}(\mu), \mathit{supp}(\nu)), \\ 0 & \text{otherwise.} \end{cases}$$

For *crisp* objects μ and ν, $\mathit{Overlaps}(\mu, \nu)$ can only have values 0 or 1. From the definition, $\mathsf{Overlaps}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ is false implies that $\mathit{Overlaps}(\mu, \nu) = 0$, and $\mathit{Overlaps}(\mu, \nu) = 1$ implies that $\mathsf{Overlaps}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ is true. Inversely, the relation $\mathsf{Overlaps}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ is true means that $(\mathit{supp}(\mu))^\circ \cap (\mathit{supp}(\nu))^\circ \neq \emptyset$, which in turn implies $\sup_p(\mu^\star \sqcap \nu^\star(p) = 1$. It can be shown similarly that if $\mathit{Overlaps}(\mu, \nu) = 0$, then $\mathsf{Overlaps}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ is false.

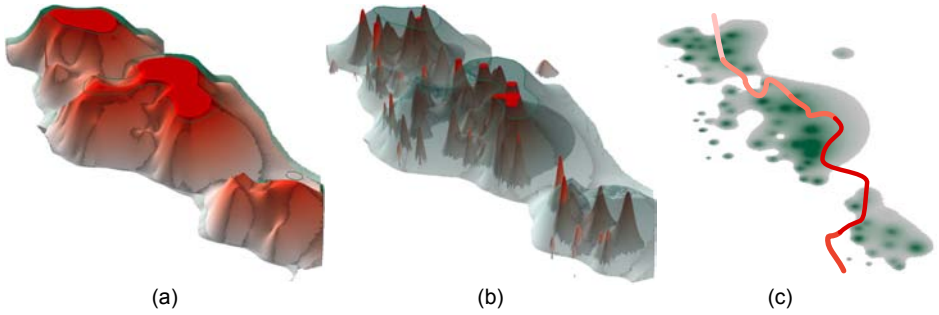

Figure 4.14: Three different cases of a *Within* relation: (a) vague regions one within the other, (b) multi vague regions possibly one within the other, (c) a vague multiline certainly not within the vague multiregion.

A vague object μ is certainly within another object ν if μ is a subset of ν, i.e., $\forall p, \mu(p) \leq \nu(p)$. If the core of μ is not fully inside the support set of ν, we consider that the two objects are certainly not in a *Within* relation. Otherwise, the

objects are in a *Within* relation to some positive degree. Figure 4.14 illustrates different cases of the within relation. The vague region in red in Figure 4.14(a) is fully within the region shown in transparent green. The vague region in red in Figure 4.14(b) is mostly within the region in green. There are location where the membership of the red region is higher than the membership of the green region: the peaks in saturated red that come out of the green surface. The *Within* relation is possible for the vague multiregions, but not certain. The vague line of Figure 4.14(c) is certainly not within the vague multiregion because a part of its core is outside the support set of the region.

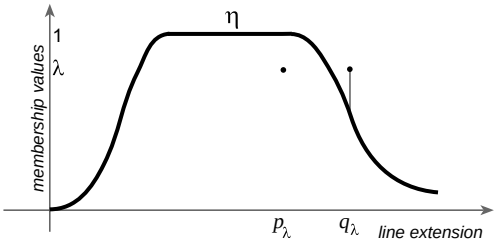

Figure 4.15: The membership function η at locations for a vague line. Vague points p_λ and q_λ lie at different locations in the vague line. The membership value of the vague line at p's location is higher than λ, its membership at q's location is lower than λ.

Figure 4.15 illustrates membership functions of two vague points p_λ and q_λ, and the membership function η of a vague line. Points p_λ and q_λ have the same membership value and their locations lie on the line extension. The value λ is lower than the membership value of the line at location p, but it is higher than the membership value of the line at location q. The vague point p_λ is certainly within the vague line. The vague point q_λ is possibly within the vague line. The value $\lambda - \eta(q)$ weakens the within relation. We consider this value to be the degree of disapproval of the within relation, and define the degree of the relation $\mathit{Within}(q_\lambda, \eta)$ to be $1 - (\lambda - \eta(q))$.

For any two vague objects μ and ν, if there are locations where $\mu(p) > \nu(p)$, then μ is within ν to some degree smaller than 1. A positive difference $\mu(p) - \nu(p)$ weakens (disapproves) the $\mathit{Within}(\mu, \nu)$ relation. The relation is the most disproved at the location(s) where this positive difference reaches the maximum. We are therefore interested in the positive difference. The bounded difference operator, $(\mu \bigtriangledown \nu)(p) = \max\{0, \mu(p) - \nu(p)\}$, provides what we need. We consider the maximum difference as the degree of disapproval of the *Within* relation, and calculate the value of the relation as the substraction of the disapproval degree from the truth value 1. The relation *Within* is defined as:

$$\mathit{Within} : \mathit{GVSpatial} \times \mathit{GVSpatial} \to \mathit{TruthDegree}$$
$$\forall \mu, \nu \in \mathit{GVSpatial},$$

$$\mathit{Within}(\mu, \nu) = \begin{cases} 0 & \text{if } \mu \sqcap \nu = 0^{\mathbb{R}^2} \text{ or } \mu = \nu \text{ or } \nu \sqsubseteq \mu \text{ or} \\ & \sup_p \{(\mu \sqcap \nu)(p)\} = 1 \text{ and } \mu \not\sqsubseteq \nu, \\ 1 - \sup_p \{(\mu \bigtriangledown \nu)(p)\} & \text{otherwise.} \end{cases}$$

The conditions in the right side force the *Within* relation between μ and ν to be false when objects are disjoint, or they are equal. Also, the relation is false if ν is a subset of μ, or the cores of μ and ν intersect (that means, one of the previous relations might be certain) but μ is not a subset of ν.

For *crisp* objects μ and ν, the bounded difference $\mu \nabla \nu$ takes only values 0 or 1, therefore $\mathit{Within}(\mu, \nu)$ can only take these values. The relation $\mathit{Within}(\mu, \nu)$ is equal to 1 means that $\sup_p(\mu \nabla \nu)(p) = 0$. That means there is no location p such that $\mu(p) = 1$ and $\nu(p) = 0$. That is to say that all locations of μ are also in ν. Thus, the relation $\mathsf{Within}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ is true. The inverse, $\mathsf{Within}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ is true implies that $\mathit{Within}(\mu, \nu) = 1$, can be shown similarly. $\mathit{Within}(\mu, \nu) = 0$ implies $\sup_p(\mu \nabla \nu)(p) = 1$, which means there is at least one location p such that $\mu(p) = 1$ and $\nu(p) = 0$. That is, there are locations from μ that are not in ν, therefore the $\mathsf{Within}(\mathit{supp}(\mu), \mathit{supp}(\nu))$ relation is false. The inverse can be shown similarly.

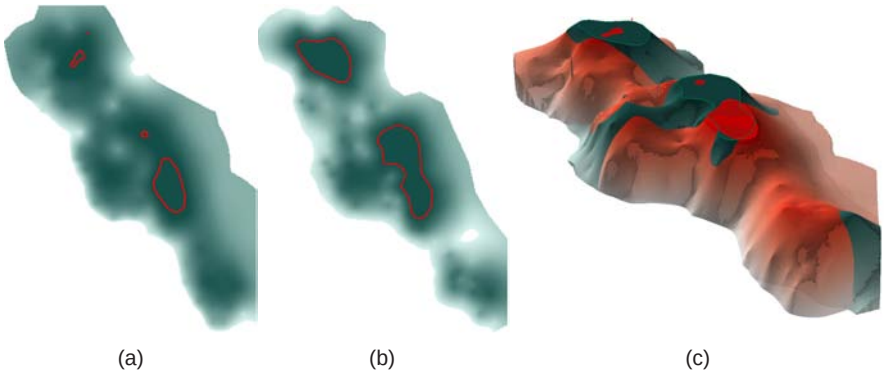

Figure 4.16: Two vague regions that are similar: (a) and (b) vague regions with their cores drawn in red, (c) the two regions shown in different colours, drawn in 3D.

Two vague objects are certainly equal if their membership functions are equal. If the core of one object lies partially outside the other object, the two objects are certainly not equal. Otherwise they are equal to some positive degree. Figure 4.16 shows two vague regions that are possibly equal. Part (a) and (b) show the two vague regions separately, in 2D. Part (c) shows the two vague regions as surfaces in 3D with their membership values as z coordinate. It can be seen that the surfaces are similar. The equality of the surfaces is violated from the locations where the difference between them is different from 0. The larger the absolute difference between memberships at the same locations, the lower the degree of being equal. A relation $\mathit{Equal}(\mu, \nu)$ is disapproved the most at the location(s) where $|\mu(p) - \nu(p)|$ reaches the maximum. We consider the maximum value of the absolute difference to be the degree of disapproval of the *Equal* relation. The value of the relation is then calculated by subtracting the disapproval degree from the truth value 1. The relation *Equal* is defined using the absolute

difference operator:

$$\begin{gathered}\mathit{Equal} : \mathit{GVSpatial} \times \mathit{GVSpatial} \to \mathit{TruthDegree} \\ \forall \mu, \nu \in \mathit{GVSpatial},\end{gathered}$$

$$\mathit{Equal}(\mu,\nu) = \begin{cases} 0 & \text{if } \mu \sqcap \nu = 0^{\mathbb{R}^2} \text{ or } \mu \neq \nu \text{ and} \\ & (\mu \sqsubseteq \nu \text{ or } \nu \sqsubseteq \mu \text{ or} \\ & \sup_p \{(\mu \sqcap \nu)(p)\} = 1), \\ 1 - \sup_p \{(\mu \mid - \mid \nu)(p)\} & \text{otherwise.} \end{cases}$$

The conditions force the *Equal* relation between μ and ν to be false if they are disjoint, or one is a subset of the other but the two are not equal. It is also false if μ and ν are not equal but their cores intersect, that means one of the relations *Touches*, *Crosses*, or *Overlaps* might be certain.

The absolute difference results in the symmetric difference when applied to the (characteristics functions of) classical sets. For *crisp* objects μ and ν, $\sup_p(\mu \mid - \mid \nu)(p) = 0$ iff their symmetric difference is empty. This means that *Equal* is equal to 1 if and only if the two objects are equal.

4.3 Some properties of spatial relations

The formulas for the spatial relations, as defined in Section 4.2, show that all relations, except *Within*, are symmetric, i.e., $R(\mu,\nu) = R(\nu,\mu)$. The relations *Disjoint*, *Crosses*, *Overlaps*, and *Within*, are anti-reflexive, that is $\forall\mu,\ R(\mu,\mu) = 0$. The *Equal* relation is reflexive, that is $\mathit{Equal}(\mu,\mu) = 1$. Being symmetric and reflexive, the *Equal* relation is a tolerance relation [61].

The relations are such that the total certainty of one relation excludes the total certainty of others. For some cases this property is stronger: if one relation is certain, all the others are false.

- The *Disjoint* relation between two object is certain, i.e., its value is 1, iff the intersection of the two objects is empty. All the other relations require a non-empty intersection to have a positive degree. Therefore a certain *Disjoint* relation forces all the others to be false.
- The *Touches* relation between μ and ν is positive if $\sup_p(\mu^\star \sqcap \nu^\star)(p) < 1$, which means both *Crosses* and *Overlaps* relations return a value less than 1. Total certainty of *Touches* relation implies that *Crosses* and *Overlaps* cannot be certain. The relation $\mathit{Touches}(\mu,\nu)$ is positive only if μ is not a subset of ν, or viceversa. Therefore, if *Touches* is certain, none of the relations *Within* and *Equal* can be certain.
- The relation $\mathit{Crosses}(\mu,\nu)$ is certain if $\sup_p(\mu^\star \sqcap \nu^\star)(p) = 1$, which implies the relations *Touches*, *Within* and *Equal* are equal to 0. A positive *Crosses*

relation requires a true Crosses relation between support sets, which excludes the Overlaps of support sets that is a requirement for a positive *Overlaps* relation. Therefore, if $Crosses(\mu, \nu)$ is positive, $Overlaps(\mu, \nu)$ is equal to 0.

- The *Overlaps* relation between μ and ν is certain if $\sup_p(\mu^\star \sqcap \nu^\star)(p) = 1$, which forces *Touches*, *Within* and *Equal* relations to be equal to 0. Condition Overlaps on support sets excludes Crosses between support sets, which means if $Overlaps(\mu, \nu)$ has positive value, then $Crosses(\mu, \nu)$ is equal to 0.
- The conditions for the *Within* and *Equal* relations are defined such that if one is certain the other is equal to 0. Both relations are positive (smaller than 1) only if $\sup_p \{(\mu \sqcap \nu)(p)\}$ is smaller than 1. Therefore, if *Within* or *Equal* are certain, then *Touches*, Overlaps, and Crosses cannot be certain.

4.4 Discussions

A spatial relation between vague objects was defined in Section 4.2 from extreme values that support or disapprove it. The continuity of membership functions of objects justifies this decision. The relations could also be defined from some averaging over membership values, instead of using the extreme values. An integration can perform the averaging process.

Using integrals instead of the superior, notations like $\sup_p \{(\mu \sqcap \nu)(p)\}$ will be replaced by $\int (\mu \sqcap \nu)(p)\, dp$. Because we want the relation values to be in the interval $[0, 1]$, we normalize an integral with the size of the support set. The superior $\sup_p \{(\mu \sqcap \nu)(p)\}$ would then be replaced by the ratio $\int (\mu \sqcap \nu)(p)\, dp / \int_{supp(\mu \sqcap \nu)} dp$. For example, the *Disjoint* relation between two vague multiregions μ and ν will be calculated as

$$Disjoint(\mu, \nu) = 1 - \frac{\iint (\mu \sqcap \nu)(x, y)\, dx\, dy}{\iint_{supp(\mu \sqcap \nu)} dx\, dy}.$$

The *Equal* relation between μ and ν will be calculated as

$$Equal(\mu, \nu) = \begin{cases} 0 & \text{if } \mu \sqcap \nu = 0^{\mathbb{R}^2} \text{ or } \mu \neq \nu \text{ and} \\ & (\mu \sqsubseteq \nu \text{ or } \nu \sqsubseteq \mu \text{ or } \sup_p(\mu \sqcap \nu)(p) = 1), \\ 1 - \frac{\iint (\mu \mid - \mid \nu)(x, y)\, dx\, dy}{\iint_{supp(\mu \mid - \mid \nu)} dx\, dy} & \text{otherwise.} \end{cases}$$

These example formulas can be modified in order to cover all different object types. The membership function μ of a vague object can be seen as a mass distribution over points, lines, or regions in space. The integrals in the above formulas calculate the mass of a vague multiregion object. The above formulas can be written for all object types by replacing the integrals with the mass of

the objects. The notion of mass and its calculation for vague multipoints, vague multilines, and vague multiregions is provided in Chapter 5.

The change of the formulas brings a change in the behaviour of the relations. The behaviour of the relations *Disjoint*, *Touches*, *Crosses*, and *Overlaps* changes similarly for the extreme value 1: the relation is certain only if the involved objects are crisp. The behaviour of *Within* and *Equal* remains the same for the extreme value 1, but it changes for the extreme value 0. The core of an object μ can be outside the support set of an object ν, but the objects can still be within the other, or be equal to a positive degree.

4.5 Summary

In this chapter we provided definitions of spatial relations between general vague objects. The provided relations, *Disjoint*, *Touches*, *Crosses*, *Overlaps*, *Within*, and *Equal*, follow the intuition behind the SQL/MM spatial relations. They extended the true/false set of truth values of the SQL/MM relations to the $[0,1]$ interval. That means that the truth of a relation is a matter of degree.

The relations *Disjoint*, *Touches*, *Crosses*, and *Overlaps* were defined so that a relation is certain if the corresponding crisp relation is true for their cores; a relation is certainly false if the corresponding crisp relation is false for their support sets. The total certainty of the other two relations, *Within*, and *Equal*, was modelled by the subset and equality relation for fuzzy sets, respectively. A $\mathit{Within}(\mu,\nu)$ relation is certainly false if the corresponding crisp relation between the core of μ and the support set of ν is false. Similarly, an *Equal* relation is certainly false if the corresponding crisp relation between the core of one object and the support set of the other is false

The relations have the property that only one relation can be certain at a time, i.e., if one relation is certain, all the others have a degree smaller than 1. For some of the relations this property is stronger: if the relation is certain, all the others are false. Each relation gives the corresponding crisp relation when applied to crisp objects.

Chapter 5

Metric operators for vague objects

Metric operators offer measures for spatial objects. They include operators like distance between two objects, length of a line, area of a region. The objective of this chapter is to provide a basic set of metric operators for vague objects. We aim at operators that generalize metric operators on crisp objects (called crisp operators hereafter), meaning when applied to crisp objects they obtain the same results as the crisp operators.

This chapter provides mathematical definitions of metric operators for vague objects of a general type. The operators we provide are distance between two vague objects of any type, length of a vague multiline, area, diameter, and perimeter of a vague multiregion. An operator on vague objects is such that for every α in $(0,1]$ it returns the value of the analogous crisp operator applied to the α-cut of the vague objects. For example, the area operator returns for every α in $(0,1]$ the area of the α-cut of the vague multiregion. We call these alpha operators. An alpha operator takes as argument one or two vague objects, and returns a function from an interval in $(0,1]$ to the non-negative real numbers $\mathbb{R}^+$. The returned function by an alpha operator is upper or lower semi-continuous. To provide the definition of an alpha operator we need a new type, vague measure:

$$\mathit{VMeasure} \equiv \{f : (0,\sigma] \to \mathbb{R}^+ \mid \sigma \in (0,1] \text{ and } f \text{ is semi-continuous}\}\,.$$

For each alpha operator we provide a corresponding operator that produces an average over all values of the return function of the alpha operator. We call the operators of this second group average operators. The integration performs an averaging process on functions [56]. We define an average operator as the integral over $[0,1]$ of the return function of the corresponding alpha operator. Such an integral exists, because the return function of an alpha operator is lower or upper semi-continuous. An average operator returns a non-negative real number that is a value of the type $\mathit{Measure} \equiv \mathbb{R}^+$. In addition to the two groups of operators, we provide two other operators: the centroid of a vague object, and a measure for the vagueness of an object. These last operators consider a vague object as a body with variable density, which is its membership degree, and use the concept of mass for the calculation.

The chapter is organized as follows. Section 5.1 summarizes previous work about metric operators for fuzzy objects. Section 5.2 provides the definitions of the distance operators on vague objects. Distance is the only diadic operator. Section 5.3 provides the definition of the unary operators: length for vague multilines, area, perimeter and diameter operators for vague multiregions. The centroid and the vagueness measure are presented together in Section 5.4. Section 5.5 compares our approach with existing measures for fuzzy objects, and discusses the choice of our operators. Section 5.6 summarizes the results of the chapter.

5.1 Previous work

There are several works in fuzzy image processing dealing with geometric measures for fuzzy sets. On the other hand, there exist fuzzy measures offered by the (general) fuzzy theory. We present first the geometric measures for being the main topic of this chapter, and close the section with two fuzziness measures offered by fuzzy theory.

Bloch [8] provides an overview of fuzzy distances proposed in the literature. The proposals can be separated into two groups. One group calculates a distance between two fuzzy sets by comparing their membership functions – mainly proposals from the fuzzy theory. The other group considers the spatial domain, by including the Euclidean distance d_2 in the distance measures they propose – mainly proposal from fuzzy image processing. The proposals of this second group are of our interest, and are discussed in the next paragraph.

A distance measure between fuzzy sets was proposed by Dubois and Prade [29], and modified by Rosenfeld [77]. They propose a distance between two fuzzy sets μ and ν that is a fuzzy number $\mathit{Dist} : \mathbb{R}^+ \to [0,1]$, such that for any $r \in \mathbb{R}^+$:

$$\mathit{Dist}_{\mu,\nu}(r) = \sup \left\{ \min \{\mu(p), \nu(q)\} \mid p, q \in \mathbb{R}^2, \mathsf{d}_2(p,q) \leq r \right\}.$$

Other proposals consider the distance between fuzzy sets to be a (positive) number. One way of building a distance between two fuzzy sets μ and ν is by applying a distance measure dist between α-cuts of the fuzzy sets, and averaging the values via an integral [8]: $\mathit{dist}(\mu,\nu) = \int_0^1 \mathsf{dist}(\mu_\alpha, \nu_\alpha)\, d\alpha$. For fuzzy sets with membership values from a finite set, e.g., fuzzy sets in digital images, the integral is replaced by a finite sum. Chaudhuri and Rosenfeld [16, 17] also propose averaging of α-cuts, employing a weighted average instead of the simple average, and the Hausdorff distance HD (see page 14) as the distance between the α-cuts. The distance between two fuzzy sets μ and ν is

$$\mathit{HD}(\mu,\nu) = \frac{\int_0^1 \alpha\, \mathsf{HD}(\mu_\alpha, \nu_\alpha)\, d\alpha}{\int_0^1 \alpha\, d\alpha}.$$

Their distance *HD* is a metric (as defined in Section 2.2), and it can be used as a measure for the similarity of fuzzy sets. Another proposal defines a distance *dist* by weighting an arbitrary distance d between points, with weights provided by their membership values:

$$dist(\mu,\nu) = \frac{\sum_p \sum_q d(p,q)\ \min\{\mu(p),\nu(q)\}}{\sum_p \sum_q \min\{\mu(p),\nu(q)\}}.$$

The geometric measures proposed by Rosenfeld and Haber in [76, 78] are area, height, width, intrinsic and extrinsic diameter, and perimeter of a fuzzy set. Each measure returns a non-negative real number, and produces a meaningful crisp measure when applied to a crisp set. It is calculated from an integral over the membership function of the fuzzy set. The *area* of a fuzzy set μ was defined as the volume between the surface of the function $\mu(x,y)$ and the x, y plane: $A(\mu) = \int\int \mu(x,y)\,dx\,dy$. The *height* of μ was calculated as $h(\mu) = \int \max_x \{\mu(x,y)\}\,dy$. This is the area of the projection of μ's surface on the y, z plane. The *width* was calculated as $w(\mu) = \int \max_y \{\mu(x,y)\}\,dx$. It is the area under μ's projection on the x, z plane. These three measures are in a relation $A(\mu^2) \le h(\mu) \cdot w(\mu)$. The *extrinsic diameter* is a kind of generalization of height and width. It was calculated as $E(\mu) = \max_u \{\int \max_v \{\mu(u,v)\}\,du\}$, where u and v denote any pair of orthogonal directions. The function under the integral is the projection of μ to a plane perpendicular to the x, y plane and the direction v. The integral produces the area of the projection. The extrinsic diameter is then taken from the projection with the maximal area. The *intrinsic diameter* was defined for fuzzy sets with connected α-cuts (Weiss connected). It is calculated from the shortest path ϱ_{pq} connecting any two locations p and q in the support set, for paths such that the membership value of any location s in the path is higher than the memberships of p and q. The intrinsic diameter of a fuzzy set μ is given by $I(\mu) = \max_{p,q}\{\min \varrho_{pq} \int_{\varrho_{pq}} \mu\}$. The *perimeter* of a smooth[1] fuzzy set μ was defined as the double integral of the magnitude of its gradient:

$$p(\mu) = \int\int \sqrt{\left(\frac{\partial\mu}{\partial x}(x,y)\right)^2 + \left(\frac{\partial\mu}{\partial y}(x,y)\right)^2}\,dx\,dy.$$

The perimeter of a piecewise constant fuzzy set is defined by a double summation, and it is shown that the above formula is the limit case for the piecewise constant set.

Bogomolny [10] modified the above definitions of height, width, diameter, and perimeter, so that they satisfy known interrelations for the crisp sets. One relation between the geometric measures is that area is smaller than the product of height and width. Another useful relation is the isoperimetric inequality: the squared perimeter is greater than the area multiplied by 4π. The isoperimetric property is utilized for the index $p^2/4\pi A$, characterizing the shape of a set.

[1] A function $\mu(x,y)$ is smooth if its first partial derivatives exist.

These two relations do not hold for Rosenfeld's measures. Bogomolny modified the *height* of a fuzzy set μ to $h(\mu) = \int \max_x \{(\mu(x,y))^{1/2}\}\, dy$. The *width* and the *diameter* were modified in a similar way. The definition of *perimeter* was modified to

$$p(\mu) = \iint \sqrt{\left(\frac{\partial \mu^{1/2}}{\partial x}(x,y)\right)^2 + \left(\frac{\partial \mu^{1/2}}{\partial y}(x,y)\right)^2}\, dx\, dy.$$

Schneider [82] proposed geometric measures for fuzzy regions and fuzzy lines: area, height, width, outer and inner diameter, perimeter, elongatedness, and roundedness for fuzzy regions, and length and strength for fuzzy lines. The *roundedness* of a fuzzy region μ is calculated as:

$$r(\mu) = \frac{\min_x \left\{\int \max_y \{(\mu(x,y))^{1/2}\}\, dx\right\}}{\max_x \left\{\int \max_y \{(\mu(x,y))^{1/2}\}\, dx\right\}}.$$

The *elongatedness* of μ is calculated as $1 - r(\mu)$. The other measures for fuzzy regions coincide with those proposed by Bogomolny [10]. The outer and inner diameter correspond to extrinsic and intrinsic diameter. The *length* of a fuzzy line was calculated from the same integral as the perimeter. The integration is performed over the support set of the line, instead of the whole $\mathbb{R}^2$. The *strength* of a fuzzy line was defined to be its minimum membership value. Schneider proposed another group of fuzzy–valued operators: area, height, width, diameter, perimeter, and length, which return fuzzy numbers. These operators apply the same principle, e.g., the fuzzy–valued area of a fuzzy region μ associates the area of an α-cut μ_α with the value α.

Bandemer and Gottwald [4] discussed two fuzziness measures for fuzzy sets in a finite space X, *entropy* and *energy*, summarizing definitions given by different authors. The *entropy* evaluates the deviation of a fuzzy set from a crisp set. The entropy for a crisp set is 0, whereas a maximum entropy is reached from a fuzzy set μ in which every location has a value $\mu(x) = 0.5$. Three *entropy* measures were presented. A first measure was $F_1(\mu) = \max\{(\mu \sqcap (1-\mu))(x) \mid x \in X\}$. The second measure was founded on the cardinal of a fuzzy set, $\mathit{card}(\mu) = \sum_{x \in X} \mu(x)$. The entropy was calculated as $F_2(\mu) = 2\, \mathit{card}(\mu \sqcap (1-\mu))/\mathsf{card}(X)$, where $\mathsf{card}(X)$ is the cardinal of X (i.e., number of its elements), and the factor 2 normalizes the values of F_2. The third measure is an analogue of the Shanon entropy: $F_3(\mu) = -c \sum_{x \in X} (\mu(x) \ln \mu(x) + (1-\mu(x)) \ln (1-\mu(x)))$. Two *energy* measures were presented: $E_1(\mu) = \mathit{card}(\mu)$, and $E_1(\mu) = \max\{\mu(x) \mid x \in X\}$.

5.2 Distance operators for vague objects

We define two distance operators, an alpha distance *Distance* and an average distance *AvDistance*, both taking as arguments two vague objects of a general

type. The alpha distance produces for every $\alpha \in [0,1]$ the nearest point distance between the α-cuts of the two objects. The average distance produces an average of distances between the α-cuts of the two objects, calculated as the integral of *Distance* values over all α values. We start by explaining the *Distance* operator, provide its definition, illustrate $\mathit{Distance}(\mu,\nu)$ for a vague point and a vague region, and show the properties of the *Distance* return function. Then we explain how the *AvDistance* is constructed as an average of values of *Distance* return function, and provide its definition.

Several measures exists for the distance between two sets in $\mathbb{R}^2$. We consider the nearest point distance to be the distance measure between two sets $A, B \subset \mathbb{R}^2$. It is based on the Euclidean distance between the elements of the sets: $\mathsf{Distance}(A,B) = \inf\{d_2(p,q) \mid p \in A, q \in B\}$. We define the alpha distance between two vague objects of a general type, *GVSpatial*, from the Distance between their α-cuts:

$$\mathit{Distance} : \mathit{GVSpatial} \times \mathit{GVSpatial} \rightarrow \mathit{VMeasure}$$
$$\forall \mu, \nu \in \mathit{GVSpatial},$$
$$\mathit{Distance}(\mu,\nu) = \left\{(\alpha, \mathsf{Distance}(\mu_\alpha, \nu_\alpha)) \mid 0 < \alpha \leq \min\{\max_p \mu(p), \max_q \nu(q)\}\right\}$$

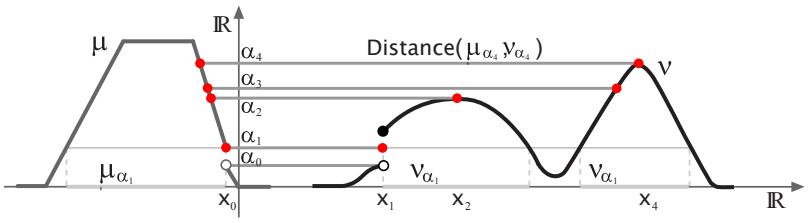

Figure 5.1: Distances between the α-cuts of two fuzzy sets μ and ν in $\mathbb{R}$. The nearest points between the α-cuts are shown in red. The α-cuts, μ_{α_1} and ν_{α_1}, are drawn in $\mathbb{R}$ with a thick grey line.

Figure 5.1 illustrates the distances between α-cuts of two fuzzy sets μ and ν in $\mathbb{R}$, both valid membership functions for vague lines. The nearest points between the α-cuts are shown in red. Location x_4 is a global maximum for ν: the value α_4 at this location is the maximum value of ν. For any $\alpha > \alpha_4$, ν_α is empty. The distance between α-cuts is undefined for such α values. The function ν has a local maximum at x_2: the value α_2 at this location is smaller than α_4. Location x_2 is the nearest location of ν_{α_2} to the α_2-cut of μ. The value α_3 is higher than α_2 with a small difference. The nearest point for the α_3-cut of ν has jumped further away. Both functions μ and ν, have a discontinuity at x_0 and x_1, respectively. Interval $[\alpha_0, \alpha_1]$ is in the discontinuity gap of both functions. Locations x_0 and x_1 are the nearest locations for the α-cuts of μ and ν for all α values in $[\alpha_0, \alpha_1]$. The distance between these α-cuts is constant.

To illustrate the distance between vague objects, let us consider the vague point and the vague region of Figure 5.2. The vague point, shown in black, has a

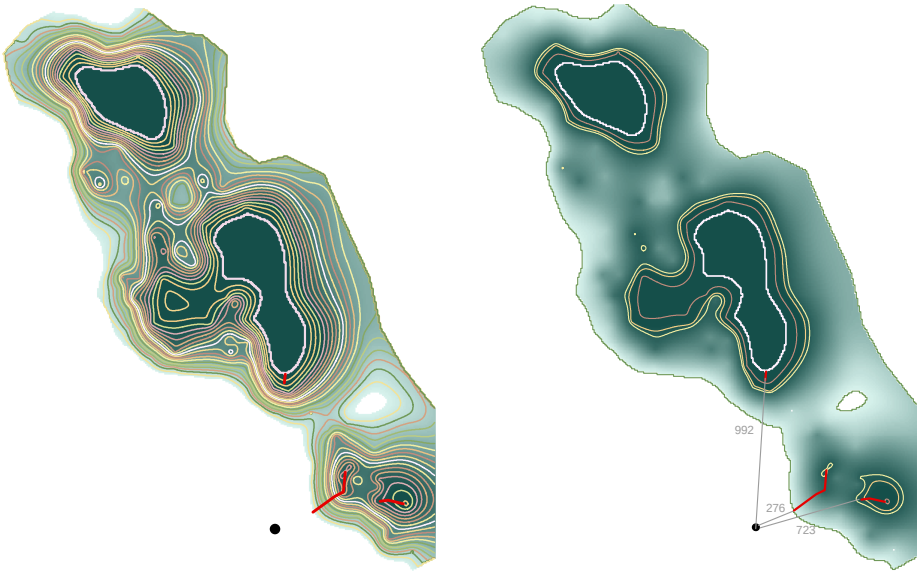

Figure 5.2: A vague point (in black) and a vague region (in green shades). Closest location of every α-cut to the point is shown in red. The left picture shows α levels at every 0.05 value, the right picture shows a selected subset of α levels.

membership value equal to 1. The core boundary of the vague region is drawn in white. For every α-cut of the region, the nearest location to the point is shown in red. The left picture shows the α levels of the vague region for every multiple of 0.05. The right picture shows α levels for α in $\{0, 0.75, 0.8, 0.9, 1\}$. The distance between the point and each α level was calculated using ArcGIS, and these values were used to build the graph in Figure 5.3. It is the graph of the alpha distance between the point and the region. The function has two discontinuities, near $\alpha = 0.75$ and $\alpha = 0.9$, both local maxima of the μ function. When a local maximum is the nearest location to the other object, for α values higher than its membership value, the nearest location jumps to another position. That happens twice for the vague region of Figure 5.2.

An increasing α value leads to an increasing distance between α-cuts: $\alpha_1 < \alpha_2$ implies that $\mu_{\alpha_2} \subseteq \mu_{\alpha_1}$ and $\nu_{\alpha_2} \subseteq \nu_{\alpha_1}$, which in turn implies $\mathsf{Distance}(\mu_{\alpha_1}, \nu_{\alpha_1}) \leq \mathsf{Distance}(\mu_{\alpha_2}, \nu_{\alpha_2})$. The function $\mathit{Distance}(\mu, \nu)$ may be constant if there is discontinuity in the objects membership functions, i.e., a vertical cliff for a region surface, or a vertical jump for a line graphic. The function $\mathit{Distance}(\mu, \nu)$ is a non-decreasing function. A local maximum of an object that is the nearest location to the respective α-cut of the other object, causes discontinuity in the distance function. An α-cut includes the locations with value α: $\mu_\alpha = \{p \in \mathbb{R}^2 \mid \mu(p) \geq \alpha\}$. For a local maximum value α_0, $\mathit{Distance}[\mu, \nu](\alpha_0)$ is the distance to the local maximum location. For α values higher than that α_0, yet very close to it, the nearest location is displaced further away, and the dis-

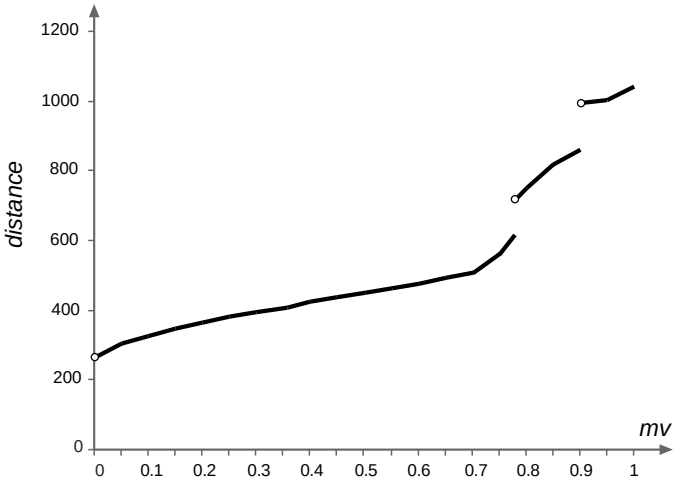

Figure 5.3: The graph of the *Distance* function between the vague point and the vague region of Figure 5.2.

tance value makes an upward jump. The distance function is thus lower semi-continuous. The distance $\mathit{Distance}(\mu, \nu)$ is a function from a semi-interval $(0, \sigma]$ to $\mathbb{R}^+$, with $\sigma = \min\left\{\max_p \mu(p), \max_q \nu(q)\right\} \in (0, 1]$. The function is non-decreasing and lower semi-continuous. The distance function between two crisp objects is constant.

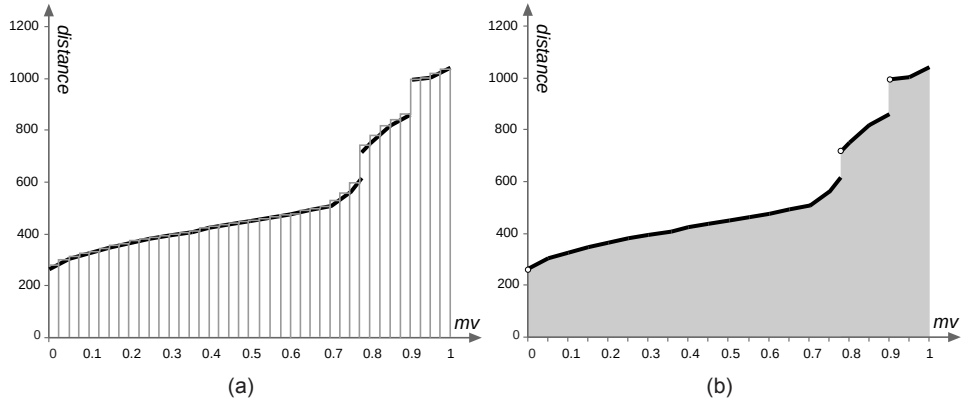

Figure 5.4: The *Distance* function and its integral in $[0, 1]$ that gives an averaging of its values.

For an integrable function $f : \mathbb{R} \to \mathbb{R}$, the average value of $y = f(x)$ in $[a, b]$ is $\overline{y} = \int_a^b f(x)\,dx$ [34]. This property is used to define the average distance, as well as the other average operators. Figure 5.4 illustrates this property for the $\mathit{Distance}(\mu, \nu)$ function of Figure 5.3. The area drawn in grey in Figure 5.4(b) is the integral of the function $\mathit{Distance}[\mu, \nu](\alpha)$, and it is the limit of the sum of

vertical bars in Figure 5.4(a), $\sum_1^n \mathit{Distance}[\mu,\nu](\alpha_i)/n$. The function $\mathit{Distance}(\mu,\nu)$ is integrable because it is lower semi-continuous. The average distance operator *AvDistance* is defined as:

$$\mathit{AvDistance} : \mathit{GVSpatial} \times \mathit{GVSpatial} \to \mathit{Measure}$$
$$\forall \mu,\nu \in \mathit{GVSpatial}, \sigma = \min\left\{\max_p \mu(p), \max_q \nu(q)\right\},$$
$$\mathit{AvDistance}(\mu,\nu) = \int_0^\sigma \mathit{Distance}[\mu,\nu](\alpha)\, d\alpha/\sigma.$$

The distance operator *AvDistance* is reflexive, and $\mathit{AvDistance}(\mu,\nu) = 0$ if $\mu = \nu$. Both properties are properties of a metric distance. The operator is not a metric though, because $\mathit{AvDistance}(\mu,\nu)$ could be equal to 0 even if $\mu \neq \nu$. The triangle inequality is not satisfied either.

The average distance between two crisp objects is the integral in $[0,1]$ of a constant function. The integral returns the constant, which is indeed the nearest distance between the objects. Thus, the average distance *AvDistance* applied to crisp objects returns the Distance value between the two objects.

5.3 Length, area, diameter, and perimeter

In this section we define geometric measures, length of a vague multiline, area, diameter, and perimeter of a vague multiregion. Two groups of corresponding operators are presented: alpha operators *Length*, *Area*, *Diameter*, and *Perimeter* that produce for every $\alpha \in (0,1]$ the value of the analogous crisp operator on the α-cut of the argument, and average operators *AvLength*, *AvArea*, *AvDiameter*, and *AvPerimeter* that produce an average over the values of the corresponding alpha operator. The operators are presented here as follows. First, an alpha operator is explained and defined, followed by the properties of its return function. Then the corresponding average operator is defined, sometimes together with an equivalent formula for the operator.

The *Length* of a vague multiline μ is calculated from the lengths of its α-cuts. An α-cut is a crisp multiline, composed of several lines that might intersect at their end nodes. The length of a simple line given by a parametric equation $l = \{(x(t), y(t) | t \in [0,1]\}$, is $\mathsf{length}(l) = \int_0^1 \sqrt{(x'(t))^2 + (y'(t))^2}\, dt$. The length of a multiline $\mathrm{L} = \bigcup_i l_i$ is the sum of lengths of its simple line components: $\mathsf{Length}(\mathrm{L}) = \sum_i \mathsf{length}(l_i)$. The length operator *Length* for vague lines is defined from Length by:

$$\mathit{Length} : \mathit{VMLine} \to \mathit{VMeasure}$$
$$\forall \mu \in \mathit{VMLine},\ \mathit{Length}[\mu](\alpha) = \begin{cases} \mathsf{Length}(\mu_\alpha) & 0 < \alpha \leq \max_p \mu(p), \\ 0 & \max_p \mu(p) < \alpha \leq 1. \end{cases}$$

The α-cuts of μ are empty for α values higher than the maximum membership value of μ. The alpha length of μ is taken to be 0 for such α's.

An increasing α leads to decreasing values of *Length*(μ). For $\alpha_1 < \alpha_2$ the respective α-cuts satisfy $\mu_{\alpha_2} \subseteq \mu_{\alpha_1}$, therefore $\mathsf{Length}(\mu_{\alpha_2}) \leq \mathsf{Length}(\mu_{\alpha_1})$. The function may remain constant if the membership function along the line has discontinuities. Function *Length*(μ) is thus a non-increasing function. Figure 5.5(a) shows the graph of the membership function of the vague line of Figure 3.8(b), and Figure 5.5(a) shows the graph of the *Length* function for this line. A discontinuity in the membership function of the line results in a constant value of the *Length* function for all α's in the discontinuity gap. The membership function in Figure 5.5(a) has a discontinuity at location l_0. The membership value at l_0 jumps from α_0 to 1. The *Length* function in 5.5(b) is constant for α's in $(\alpha_0, 1]$. A local maximum value reached on a piece of the line, not just in one location, will decrease abruptly the value of *Length*(μ) at α's smaller than this maximum. The function is discontinuous at such local maxima. Because an α-cut is closed, the alpha length for the local maximum α includes the line piece with membership value α. The value α_0 is a local maximum for the membership function of Figure 5.5(a). The *Length* of the line shown in Figure 5.5(b) is upper semi-continuous at α_0 (and continuous everywhere else). The function *Length*(μ) is non-increasing and upper semi-continuous.

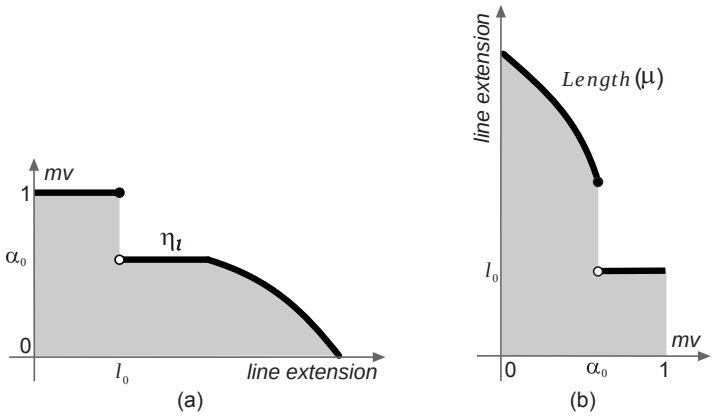

Figure 5.5: Calculation of the average length from the alpha length function.

The average length *AvLength* of a vague multiline μ is determined as the average of all *Length*(μ) values over the $[0,1]$ interval, and is calculated as the integral of *Length*(μ) in $[0,1]$. The integral exists because the function *Length*(μ) is upper semi-continuous. The operator *AvLength* is defined as:

$$AvLength : VMLine \rightarrow Measure$$

$$\forall \mu \in VMLine,\ AvLength(\mu) = \int_0^1 Length[\mu](\alpha)\, d\alpha.$$

The function *Length*(μ) is constant if μ is a crisp line. The average length *AvLength*(μ) is the integral in $[0,1]$ of this constant that is the Length of the crisp line μ. Thus, the *AvLength* operator returns the same value with the crisp operator Length when applied to a crisp line.

The areas shown in grey in Figure 5.5(a) and 5.5(b) are the integral of $\eta(l)\,dl$ over the line extension, and the integral of $\mathit{Length}(\mu)$ over $[0,1]$, respectively. The two integrals present two different ways of calculating the same area. This gives another way to calculate the average length of a vague line μ, independent from the $\mathit{Length}(\mu)$ function:

$$\mathit{AvLength}(\mu) = \int_0^1 \eta(t)\sqrt{(x'(t))^2 + (y'(t))^2}\,dt.$$

The average length *AvLength* of a vague multiline is the sum of *AvLength* values of its simple line components.

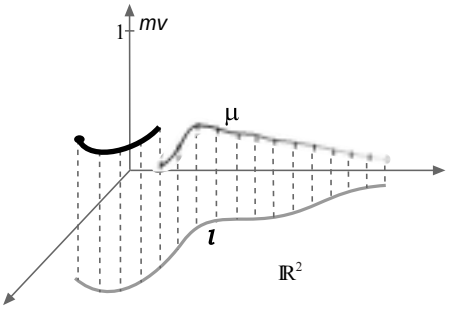

Figure 5.6: A vague line μ drawn in 3D using membership values as the third coordinate, and its projection l on the plane. The surface created from the projection is shown with vertical dashed lines.

This equivalent formula of the average length is the area of the surface delineated by the line μ put in 3D space, with membership values as z coordinate, and its projection on the plane, which is the line extension (see Figure 5.6). This gives a more practical way to calculate the average length of a vague line.

The alpha area of a vague multiregion μ is calculated from the areas of its α-cuts: $\mathsf{Area}(\mu_\alpha) = \int\int_{\mu_\alpha} dx\,dy$. If the global maximum of μ is lower than 1, we consider the area $\mathit{Area}(\mu)$ to be 0 for all α values higher than the maximum. The *Area* operator is defined as:

$$\mathit{Area} : \mathit{VMRegion} \to \mathit{VMeasure}$$

$$\forall \mu \in \mathit{VMRegion},\ \mathit{Area}[\mu](\alpha) = \begin{cases} \mathsf{Area}(\mu_\alpha) & 0 < \alpha \le \max_p \mu(p), \\ 0 & \max_p \mu(p) < \alpha \le 1. \end{cases}$$

For $\alpha_1 < \alpha_2$, the respective α-cuts satisfy $\mu_{\alpha_2} \subseteq \mu_{\alpha_1}$, therefore $\mathit{Area}(\mu)(\alpha_2) \le \mathit{Area}(\mu)(\alpha_2)$. A local maximum of μ reached at a region will cause discontinuity in the $\mathit{Area}(\mu)$ function. The alpha area of a vague multiregion is a non-increasing and upper semi-continuous function. The function $\mathit{Area}(\mu)$ is integrable. The average area of a region μ is calculated from the integral of $\mathit{Area}(\mu)$

over $[0,1]$. The operator *AvArea* is defined as:

$$AvArea : VMRegion \to Measure$$

$$\forall \mu \in VMRegion,\ AvArea(\mu) = \int_0^1 Area[\mu](\alpha)\, d\alpha.$$

If μ is a crisp region, $Area(\mu)$ is a constant function with value $\mathsf{Area}(supp(\mu))$. The operator *AvArea* is the integral of a constant value over $[0,1]$, and it is equal to that constant. Therefore, *AvArea* returns the same value as Area when applied to a crisp region.

Figure 5.7: A vague region shown in 3D with membership values as the third coordinate. Membership values are also used for colour saturation. The boundary of the core is shown in white.

Figure 5.7 shows the same region of Figure 5.2, using membership values as the third coordinate to build the surface in 3D. Memberships are also used for colouring: low saturation indicates low membership value. The average area calculated from the integral $Area[\mu](\alpha)\, d\alpha$ provides the volume under the μ function (see Figure 5.7). Therefore, the average area is equal to

$$AvArea(\mu) = \iint \mu(x,y)\, dx\, dy.$$

The integral exists because the function μ is upper semi-continuous.

The alpha diameter of a vague multiregion μ can be calculated from the diameters of its α-cuts. An α-cut may consist of several components. We consider the diameter of a component A to be $\mathsf{diam}(\mathrm{A}) = \max\{\mathrm{d}_2(p,q) \mid p,q \in \mathrm{A}\}$, and the diameter of an α-cut consisting of components $\{\mathrm{A}_i\}_1^n$, to be the sum of diameters of its components: $\mathsf{Diameter}(\mu_\alpha) = \sum_1^n \mathsf{diam}(\mathrm{A}_i)$. This is used to define the diameter operator for vague regions:

$$Diameter : VMRegion \to VMeasure$$

$$\forall \mu \in VMRegion,\ Diameter[\mu](\alpha) = \begin{cases} \mathsf{Diameter}(\mu_\alpha) & 0 < \alpha \le \max_p \mu(p), \\ 0 & \max_p \mu(p) < \alpha \le 1. \end{cases}$$

If a vague multiregion μ has a local maximum reached in an area, the value of the diameter function jumps down for α higher than that local maximum. The function $\mathit{Diameter}(\mu)$ is upper semi-continuous. The average diameter of μ is calculated from the integral of $\mathit{Diameter}(\mu)$ in $[0,1]$. The average diameter operator *AvDiameter* is defined as:

$$\mathit{AvDiameter} : \mathit{VMRegion} \rightarrow \mathit{Measure}$$

$$\forall \mu \in \mathit{VMRegion},\ \mathit{AvDiameter}(\mu) = \int_0^1 \mathit{Diameter}[\mu](\alpha)\, d\alpha.$$

Similarly to the other operators, the average diameter *AvDiameter* is equivalent to the crisp Diameter when applied to a crisp region.

The alpha perimeter of a vague multiregion is calculated from the perimeters of its α-cuts. The perimeter of an α-cut $\mathsf{Perimeter}(\mu_\alpha)$ is the length of the α-cut boundary $\mathsf{Length}(\partial\mu_\alpha)$. If a vague multiregion μ has a continuous function, the boundary of the α-cut is the α level. It is a curve in $\mathbb{R}^2$ determined by the the equation $\mu(x,y) - \alpha = 0$. If μ is continuous all its α levels are closed (looped), otherwise some α levels might not be closed. In such cases the boundary of the α-cut is completed by the intersection of the vertical cliff of μ with the horizontal plane $z = \alpha$. The alpha perimeter operator is defined as:

$$\mathit{Perimeter} : \mathit{VMRegion} \rightarrow \mathit{VMeasure}$$

$$\forall \mu \in \mathit{VMRegion}, \mathit{Perimeter}[\mu](\alpha) = \begin{cases} \mathsf{Perimeter}(\mu_\alpha) & 0 < \alpha \leq \max_{p\in\mathbb{R}^2} \mu(p), \\ 0 & \max_{p\in\mathbb{R}^2} \mu(p) < \alpha \leq 1. \end{cases}$$

Local maxima of a vague multiregion μ, reached in an area, cause discontinuities to its perimeter function, for the same reason as for the area and diameter function. The function $\mathit{Perimeter}(\mu)$ is upper semi-continuous.

Figure 5.8 shows the α levels of vague region displayed in 3D space. The average perimeter of a vague region μ is calculated from the integral over $\mathit{Perimeter}(\mu)$ in $[0,1]$. The average perimeter operator *AvPerimeter* is defined as:

$$\mathit{AvPerimeter} : \mathit{VMRegion} \rightarrow \mathit{Measure}$$

$$\forall \mu \in \mathit{VMRegion},\ \mathit{AvPerimeter}(\mu) = \int_0^1 \mathit{Perimeter}[\mu](\alpha)\, d\alpha.$$

From the formula it can be seen easily that when μ is a crisp multiregion, the average perimeter *AvPerimeter* is equivalent to the crisp operator Perimeter.

5.4 Centroid and vagueness degree

The membership function μ of a vague object can be seen as a mass distribution. A crisp object is a body with constant density, whereas a vague object has a variable density. The total mass of a body consisting of a finite set of locations

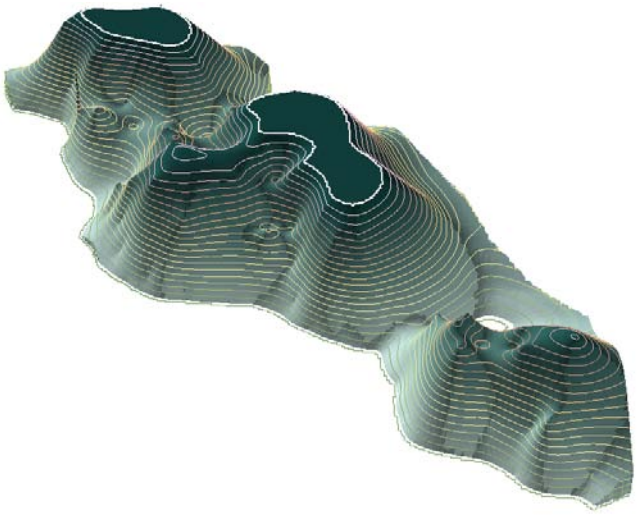

Figure 5.8: A vague region drawn in 3D, together with its α levels at every multiple of 0.05, drawn in different colours.

is calculated from the sum of masses at every location. A total mass distributed over a line and over an area is calculated from the integral of the density function, $\mu(x,y)\,dl$ and $\mu(x,y)\,dA$, respectively [3]. The notion of mass allows us to define the centroid of an object that is its centre of mass.

Let us first define the mass for each type of a vague object. The mass of a vague multipoint μ is

$$Mass(\mu) = \sum_{p \in supp(\mu)} \mu(p).$$

This is the cardinal of μ $card(\mu)$ as defined in [4]. If μ is crisp, its mass is the cardinal of the point set constituting μ. The mass of a vague multiline μ is $Mass(\mu) = \int_{supp(\mu)} \mu(x,y)\,dl$. Suppose the vague multiline is composed of vague lines $\left\{\mu_i = \tilde{h}_i(\eta_i)\right\}_1^n$, the mass of $\mu = \bigsqcup_{i=i}^n \mu_i$ is

$$Mass(\mu) = \sum_{i=i}^{n} \int_0^1 \eta_i(t) \sqrt{(x_i'(t))^2 + (y_i'(t))^2}\,dt.$$

This is the average length of the vague multiline μ. The mass of a crisp multiline is its length. The mass of a vague multiregion is

$$Mass(\mu) = \iint \mu(x,y)\,dx\,dy.$$

It is the average area of the vague multiregion. The mass of a crisp multiregion is its area.

The coordinates of the centroid of a body are calculated from its mass, and its moments M_x and M_y. The moments, M_x and M_y, are the sum or integral of $y_p\,\mu(p)$ and $x_p\,\mu(p)$, respectively. The moments of a vague multipoint μ are

$$M_x(\mu) = \sum_{(x,y)\in supp(\mu)} y\,\mu(x,y), \quad \text{and} \quad M_y(\mu) = \sum_{(x,y)\in supp(\mu)} x\,\mu(x,y).$$

The moments of a vague multiline $\mu = \bigsqcup \left\{\mu_i = \tilde{h}_i(\eta_i)\right\}_1^n$ are

$$M_x(\mu) = \sum_{i=i}^{n} \int_0^1 y_i(t)\,\eta_i(t)\,\sqrt{(x_i'(t))^2 + (y_i'(t))^2}\,dt,$$

and

$$M_x(\mu) = \sum_{i=i}^{n} \int_0^1 x_i(t)\,\eta_i(t)\,\sqrt{(x_i'(t))^2 + (y_i'(t))^2}\,dt.$$

The moments of a vague multiregion are

$$M_x(\mu) = \iint y\,\mu(x,y)\,dx\,dy \quad \text{and} \quad M_y(\mu) = \iint x\,\mu(x,y)\,dx\,dy.$$

The centroid of a vague object is a vague point. Its coordinates are calculated from its mass and moments. Its membership degree is the ratio between the mass of the object and the mass of its support set. The Mass of a support set is its cardinal if the object is a multipoint, its length if it is a multipoint, and its area if it is a multiregion. *Centroid* operator is then defined as:

$$\begin{gathered} \mathit{Centroid} : \mathit{GVSpatial} \to \mathit{VPoint} \\ \forall \mu \in \mathit{GVSpatial}, \\ \mathit{Centroid}(\mu) = \left(M_y(\mu)/\mathit{Mass}(\mu), M_x(\mu)/\mathit{Mass}(\mu), \mathit{Mass}(\mu)/\mathsf{Mass}(\mathit{supp}(\mu))\right). \end{gathered}$$

Centroid operator returns a crisp point when applied to a crisp object.

We propose an operator *Vagueness* to measure the vagueness of an object. It takes a value in the unit interval $[0,1]$. A low value shows low vagueness. The value 0 is reached when the object is crisp. A maximal value is reached when the object does not have a core or the core has a lower dimension than the object. The measure is derived from the ratio of masses of the vague object with the mass of its support set. *Vagueness* operator is defined as:

$$\begin{gathered} \mathit{Vagueness} : \mathit{GVSpatial} \to [0,1] \\ \forall \mu \in \mathit{GVSpatial},\ \mathit{Vagueness}[\mu] = 1 - \mathit{Mass}(\mu)/\mathsf{Mass}(\mathit{supp}(\mu)). \end{gathered}$$

For the three general vague types the vagueness measure is calculated as follows. If μ is a vague multipoint then $\mathit{Vagueness}(\mu) = 1 - \mathit{card}(\mu)/\mathsf{card}(\mathit{supp}(\mu))$. If μ is a vague multiline then $\mathit{Vagueness}(\mu) = 1 - \mathit{AvLength}(\mu)/\mathsf{Length}(\mathit{supp}(\mu))$. For a vague multiregion $\mathit{Vagueness}(\mu) = 1 - \mathit{AvArea}(\mu)/\mathsf{Area}(\mathit{supp}(\mu))$.

5.5 Discussions

The alpha operators proposed follow the same idea of the distance operator of Rosenfeld [77], and the fuzzy-valued operators of Schneider [82]. Their operators return fuzzy numbers, i.e., functions $f : \mathbb{R} \rightarrow [0,1]$, whereas our operators are functions $f : (0,1] \rightarrow \mathbb{R}^+$. A function returned by an alpha operator is the inverse of the function returned by the corresponding fuzzy-valued operator of Schneider [82].

The most commonly used distances between crisp sets are the nearest point distance and the Hausdorff distance. We chose the nearest point distance as the distance between α-cuts. It is commonly used as the distance between spatial objects in GIS packages and applications. The nearest point distance is not a metric (as defined in Section sec:topology), whereas the Hausdorff distance is a metric. The Hausdorff distance can be used to describe spatial distances, and at the same time it is a good measure for similarity. On the other hand, the nearest point distance is computationally less expensive than the Hausdorff distance [8].

The average operators we proposed give a similar or equivalent formula to Rosenfeld's measures. The area operator is equivalent to Rosenfeld's area. The diameter is bigger than Rosenfeld's extrinsic diameter. We could add height and width measures, following the same way of constructing the alpha and average operators. Both measures are similar to the diameter. The maximum distance between locations of an α-cut component, would be replaced by the height and width of the component bounding box. The average operators for height and width constructed in this way, would be equivalent with Rosenfeld's measures.

An average operator is defined in terms of the corresponding alpha operator. This makes explicit the relation between the two corresponding operators. Such a definition is not handy in the implementation stage: the execution of an average operator would require first the execution of its corresponding alpha operator. The independence of the operators is thus a desirable property. For this reason we proposed equivalent formulas for the calculation of average operators. We provided an equivalent formula for the average length and the average area. The average perimeter seems to be equivalent with Rosenfeld's perimeter. The proof is still to be provided. Two other remaining issues to be solved are equivalent formulas for the average distance and the average diameter.

The vagueness measure we define takes into account the membership values and their distribution. Rough set theory offers a measure for roughness of a set, which could also be used for fuzzy sets. Translated to vague objects, the roughness measure of an object μ would be the ratio between the size of the object core and the size of its support set: $\mathsf{Mass}(\mu_1)/\mathsf{Mass}(\mathit{supp}(\mu))$. As μ_1 and $\mathit{supp}(\mu)$ are (classical) sets, the measure Mass is the mass of a crisp object. This measure is coarser than the *Vagueness* measure we propose.

5.6 Summary

In this chapter we provided definitions of metric operators for vague objects of a general type. The operators were distance between two vague objects of any type, length of a vague multiline, area, diameter, and perimeter of a vague multiregion, centroid and vagueness degree of a vague object.

We proposed two different measures for distance, length, area, diameter, and perimeter, given by the alpha operators and the average operators. An alpha operator takes as argument one or two vague objects, and returns a function $f : (0,1] \to \mathbb{R}^+$. For every α in $(0,1]$ the function yields the value of the analogous crisp operator applied to the α-cuts of the vague objects. An average operator was calculated as the integral over $[0,1]$ of the return function f of the corresponding alpha operator. It produces an average over the returned values by the function f of the alpha operator. We provided equivalent formulas for the average length and the average area operators, which allow to perform their calculation independently from the corresponding alpha operator. An average operator gives its crisp analogue when applied to a crisp object.

The centroid operator calculates the centre of mass for a vague object. The vagueness operator provides a measure for the degree of vagueness of an object. The vagueness degree of a crisp object is equal to 0. The vagueness degree of a vague object is maximal when the object is equal to its boundary, i.e., it consists only of an uncertain part.

Chapter 6

Implementation of types and operators

This chapter provides an implementation of vague types and operators defined formally in the previous chapters. We use GRASS, an open source GIS software package, as the platform for our implementation. This allows the use of existing GIS functionality for the storage an manipulation of spatial objects. The structures we build to store the data about our vague objects make use of GRASS data structures. In addition, we use GRASS modules and vector functions for the manipulation of vague objects.

We assume that data for vague objects comes from point measurements or processed remote sensing images. The recognition and extraction of a vague object from such input data is a required preliminary step to their storage and manipulation. This pre-process is of different complexity for different object types. Extraction of vague regions is the more complex, and it is given more attention. It is followed by the storage process, and the modules for the manipulation and analysis of data.

The presentation of the work is organized as follows: Section 6.1 is dedicated to the creation of vague objects from input data points, and to their storage. Section 6.2 describes the visualization techniques used to display vague objects and their information. Section 6.3 discusses the modules that implement set operators on vague objects. Sections 6.4 and 6.5 close the chapter with discussions and conclusions.

6.1 Storage of vague objects

The GRASS vector format[1] is used to store vague objects. Feature types supported by GRASS are *point*, *line*, *boundary*, *area* (without holes), and *centroid*. All data is stored using line representations. A line is a sequence of (x, y) coordinates forming types *point*, *line*, *boundary*, or *centroid*. A *point* or a *centroid* is constructed as a sequence of two identical elements. A *line* or a *boundary* is constructed as a sequence with at least two elements. A *line* type is used to store linear features, whereas a *boundary* type is used to store areal features. An *area* is formed by a set of lines of type *boundary*, which constitute its boundary. GRASS allows the storage of three dimensional (3D) features, i.e., a line can be a series of (x, y, z) coordinates. It can only build the complete topology for 2D features. It builds connectivity of 3D linear features, but ignores the third coordinate when creating (and building the topology of) areal features.

GRASS data is stored in layers. These contain objects conceptually belonging to a theme, e.g. road lines, vegetation classes. A theme generally consists of several classes. A data layer is physically stored as a directory that contains several files, `coor`, `topo`, `sidx`, etc., each containing specific information. For example, vegetation data is stored in a directory `vegetation`. Location information is stored in the `coor` file in that directory, while topology information is stored in the `topo` file. One or more attributes can be attached to objects of a data layer.

Objects of a simple vague type are the basic objects that we need to store. An identifier is given to such an object and used to compile the complete information at any time the object is needed. We can only collect and store a finite set of point data, while a vague line and a vague region represent infinite point sets. An interpolation method is used in both cases to complete (approximate) information about objects. A class of vague objects, represented by a general type, is a collection of simple objects, and it is stored as a data layer. Layers of vague classes belonging to a theme are bound together. The separation of a theme data into several layers, one for each class, is needed for a correct storage of topology data.

We use the third coordinate to store membership values. A *vague point* is implemented in GRASS by a VPoint that is a triple (x, y, mv), where $(x, y) \in \mathbb{R}^2$ provides the location and $mv \in (0, 1]$ provides the membership value. A *vague line* is implemented by a VLine that is a sequence of triples $\langle (x_1, y_1, mv_1), \ldots, (x_n, y_n, mv_n) \rangle$ each triple providing the x, y location of a point in the line, associated by the membership value of that point to the line. An approximation of the *vague line* is achieved by linear interpolation between consecutive points. A *vague region* is a surface embedded in $\mathbb{R}^2$. It is implemented by a VRegion that is a triangulation. An approximation of the *vague region* is achieved by a linear interpolation within each triangle of VRegion.

[1] We used GRASS 6.0 that was the latest stable version at the moment.

A *vague multipoint* is implemented as a VMPoint layer. It is a set of triples (x, y, mv) with different locations. A *vague multiline* is implemented as a VMLine layer. It is a collection of VLine objects intersecting only at end points. A *vague multiregion* is implemented as a VMRegion layer, which is a set of triangulations that do not overlap each other. We call such layers vague data layers, for containing vague information. For example, all VRegion objects belonging to the class 'forest' of a vegetation theme are stored in a vague layer `forest` of type VMRegion.

A vague layer is created by input containing membership information, or such information is derived from attributes in input data by applying membership functions. We provide a module `v.vague.membership` that applies trapezoidal membership functions to a numerical attribute of a data layer. A VMPoint layer may derive from any point data containing membership information, or an attribute to which we could apply membership functions. Each input point associated with the membership value creates a VPoint object in the vague point layer. A VMLine layer is derived from measurements along linear features, e.g., level of congestion at locations along roads in a road network data layer. Such measurements can be either direct membership values, or we may apply functions on the measurements to derive membership values. A VLine object is created in the vague layer for each line object in the input data.

We assume that data about a vague region layer VMregion either comes from points associated with membership values, or from points with an attribute to which we apply a membership function. These points may be irregularly distributed, e.g., coming from field measurements. They may also be regularly distributed, e.g., coming from processed images. The only information we can get from such input is a membership value to a certain vague class, which may be indeed composed of several VRegion objects. The input needs to be interpreted to create the simple objects. We cluster input points, and consider that each cluster establishes a separate object. Points of each cluster are then used to create a triangulation for each simple object identified.

The next two sections, Section 6.1.1 and Section 6.1.2, are dedicated to clustering the input to form separate VRegion objects, and creating objects from triangulation of clusters, respectively. Up to here, we have covered the creation and storage of a vague layer that is the implementation of a vague class. It is convenient to bind together all classes belonging to the same theme. This is discussed in Section 6.1.3.

6.1.1 Clustering input and delineating boundaries

Several techniques exist that can be used for clustering points in space: K-means (K-median) clustering, self organizing maps, hierarchical clustering, alpha shapes (see [6] for an overview). All the techniques are based on a distance measure between points. The first two techniques require the number of clusters to

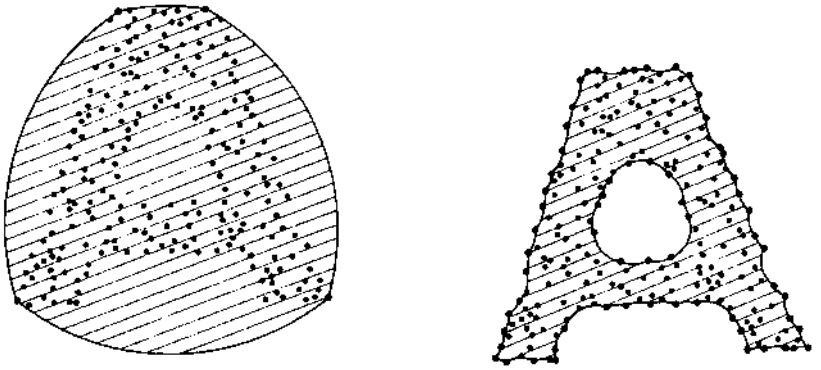

Figure 6.1: Positive α-hull in the left, negative α-hull in the right (taken from [32]).

be determined beforehand. This is usually not known for our case. Hierarchical clustering with Euclidean distance could be used for clustering the input. The technique is used for different inputs, not just points in Euclidean space, by employing different distance measures or different group (linkage) distances. It is a quite general, but slow technique. Alpha shapes (α-shapes from here onwards) work only with points in Euclidean space. They detect separate clusters from a given point set and delineate boundaries of the detected point clusters. The technique is faster than hierarchical clustering, and its output is richer as it contains a boundary for each cluster. We use α-shapes to cluster the input data and delineate the boundary of each cluster.

The α-shapes are a generalization of convex hulls. The *convex hull* of a point set S is the intersection of all closed half-planes that contain all points of S. This notion is generalized to α-*hulls* in [32]. For positive (yet sufficiently small) α, the α-hull of S is the intersection of all closed discs with radius $1/\alpha$ that contain all points of S. Large α produce a curved hull of which the boundary consists of parts of circles (with radius $1/\alpha$) that pass through extreme points of S. As α approaches zero, the curved hull converges to the convex hull. An α-hull is smoother than the convex hull, but its approximation of the intuitive shape of points is coarser than that of the convex hull. The shape of the hull can be refined by considering negative α values. For negative α, the α-hull of S is taken to be the intersection of all closed complements of discs with radius $-1/\alpha$ that contain all points of S. Figure 6.1 shows the positive α-hull (left) and the negative α-hull (right) of a point set resembling the letter 'A'.

The α-hull can be defined more concisely using the notion of a generalized disc. A generalized disc of radius $1/\alpha$ is a disc of radius $1/\alpha$ for $\alpha > 0$, it is the complement of a disc of radius $-1/\alpha$ for $\alpha < 0$, and a half-plane for $\alpha = 0$. For a point set S and a real value α, the α-hull of S is the intersection of all closed generalized discs of radius $1/\alpha$ that contain all points of S. Two other concepts, α-extreme and α-neighbour, are used in [32] to construct α-shapes. A point p

of a set S is termed an α-extreme in S if there exists a closed generalized disc of radius $1/\alpha$ such that p lies on its boundary, and it contains all points of S. Two α-extremes p and q are said to be α-neighbours if there exists a closed generalized disc of radius $1/\alpha$ with both points on its boundary, and which contains all points of S. An α-shape of S is then the straight line graph whose vertices are the α-extremes and whose edges connect the α-neighbours [32]. The assumption is that no four points of S are cocircular and no three points are collinear.

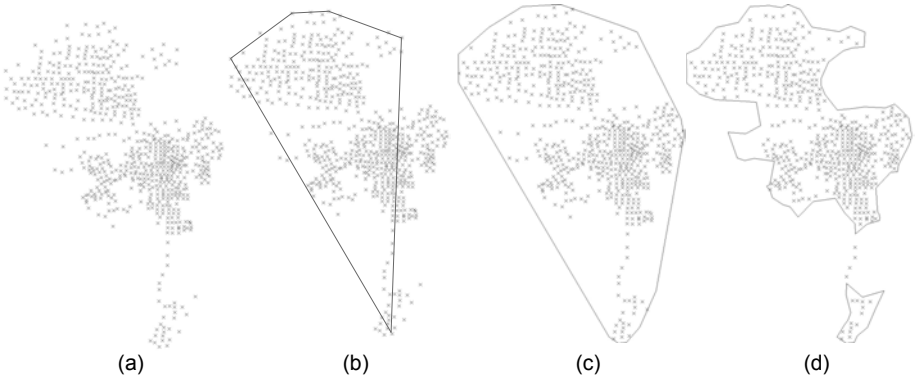

Figure 6.2: The effect of the α value on the alpha-shape: (a) input points, (b) α-shape for a positive α, (c) α-shape for α equal to 0, (d) α-shape for a negative α.

Figure 6.2(b)–(d) illustrate α-shapes produced for decreasing α values on the point set of Figure 6.2(a). A decreasing α value results in a finer shape. Depending on the α value, a single point can be a cluster (and its own boundary at the same time), as is the case in Figure 6.2(c) for four points. The set of α-extremes becomes larger when α decreases: the set of α_1-extreme points of a point set S is a subset of α_2-extreme points of S if $\alpha_1 > \alpha_2$ [32]. We are interested in shapes finer than the convex hull, therefore we work only with negative α values. This simplifies the checks for α-extremes and α-neighbours, and also the complete algorithm for building the α-shape.

A point is an α-extreme of a point set if and only if a circle with radius $-1/\alpha$ (α-circle hereafter) can be constructed such that the point is on the circle and no other point of the set lies inside it. Figure 6.3 illustrates α-extremes in a point set A. Point m from A is an α-extreme. Point n is not an α-extreme, because any α-circle passing through n contains at least another point of A. We use the same set A for illustrations 6.3–6.7 in this section. The set is chosen such that it covers all cases treated by the α-shape algorithm.

Two points are α-neighbours, if and only if an α-circle can be constructed through them, with no other points of the set within that circle. Figure 6.4 illustrates α-neighbours in A. Points v and w are α-neighbours: there is a circle passing through v and w that does not contain any other point of A. Points s and v are

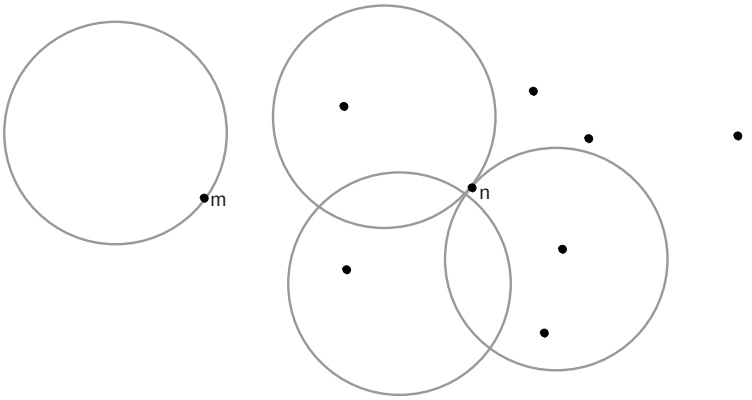

Figure 6.3: Examples of α-extreme in a set A: point m is an α-extreme, point n is not.

not α-neighbours: there are only two circles with radius $-1/\alpha$ passing through both of them, and both circles contain point t.

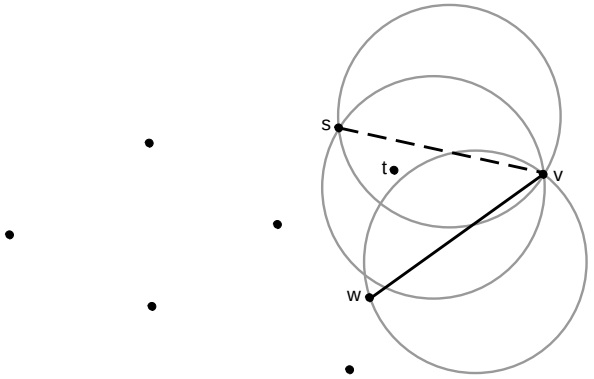

Figure 6.4: Examples of α-neighbours: v and w are α-neighbours, s and v are not.

The tests for building the α-shape of a point set are based on the Delaunay triangulation of the set, its dual (in the graph theoretical sense), the Voronoi diagram, and relations between the two. A Delaunay triangulation is a triangulation that maximizes the minimum angle [25]. A Voronoi diagram of a point set is a partition of the plane into convex polygons, one for each point of the set, such that each polygon contains only one point from the set that is its central point, and every point in a polygon is closer to its central point than to any other point of the set. Figure 6.5 shows the Delaunay triangulation (in light grey) and the Voronoi diagram (in thick light grey lines) of the point set A.

The α-shape of a point set S is a subset of the Delaunay triangulation of S [32]. It is built by first constructing the Delaunay triangulation of S, then testing the

Delaunay edges whether they are in the α-shape. The complete algorithm is:

```
create a Delaunay triangulation DT of S
create Shape as an empty set of edges
for every edge (p, q) in DT
    if p is α-extreme and q is α-extreme and p and q are α-neighbours
        add (p, q) to Shape
    fi
rof
```

To build the Delaunay triangulation we use TRIANGLE, an open source tool created by Shewchuk [86]. Testing for α-extremes and α-neighbours of S, based on relations between the Delaunay triangulation and the Voronoi diagram of S, is explained in the next few paragraphs. Tests are translated into properties of the Delaunay triangulation and its convex hull, therefore this is the only structure needed for constructing the α-shape.

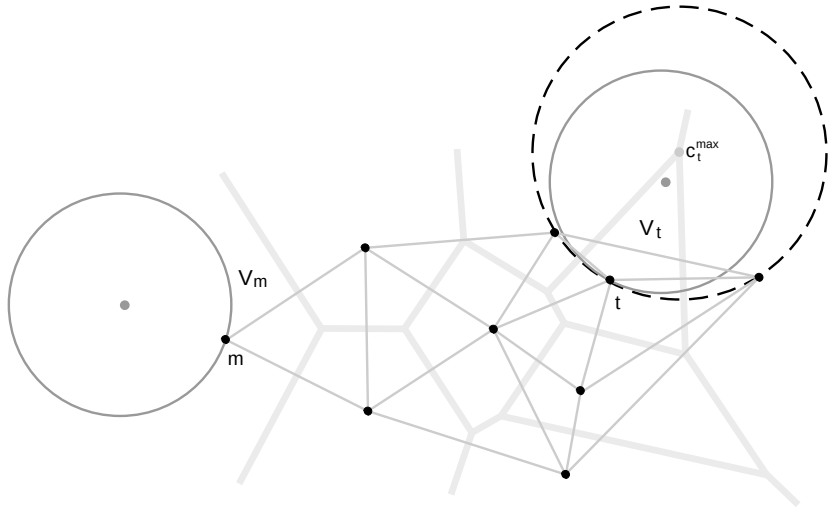

Figure 6.5: Test for α-extremes: unbounded Voronoi polygon V_m and an α-circle (in grey) passing through m; bounded Voronoi polygon V_t, an α-circle passing through t, and the maximal circle of t (with dashed line).

An α-extreme of a point set is a point on the boundary of the convex hull of the set, or a point which maximal circumradius[2] of triangles of which the point is a vertex, is bigger than the radius $-1/\alpha$. These properties are used to test for α-extremes. We explain them using Figure 6.5. Points on the boundary of the convex hull of a point set have unbounded Voronoi polygons. Any other

[2]The circumradius is the radius of the triangle's circumscribed circle, i.e., the unique circle that passes through each of the triangles vertices. The circumscribed circle is called circumcircle of the triangle.

point has a bounded Voronoi polygon. The point m is on the boundary of the convex hull of A: its Voronoi polygon V_m is unbounded. Any α-circle centred inside V_m that passes through m, does not contain any other point of A. Point t is inside the convex hull of A: it has a bounded Voronoi polygon V_t. The Voronoi polygon V_t of t contains all points that are closer to t than to any other point of A. This means that any circle passing through t and centred at a point inside V_t does not contain any other point of A. The maximal circle having this property (shown with dashed line) is the one centred at the furthest Voronoi vertex c_t^{max} from t. An α-circle passing through t, and lying inside the maximal circle of t, does not contain other points of A. Vertices of the Voronoi polygon V_t are the circumcentres of Delaunay triangles of which t is a vertex. The maximal circle of t is one of the circumcircles of triangles of which t is a vertex. It is the circumcircle with the maximum radius.

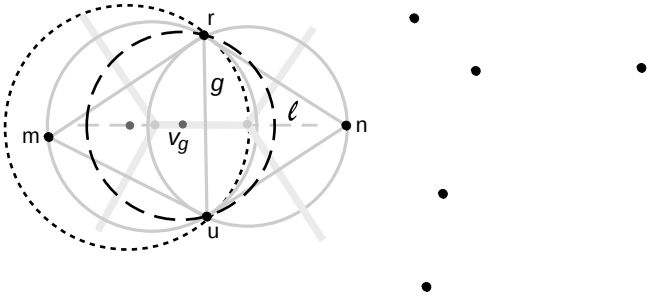

Figure 6.6: Delaunay triangles sharing edge g and their circumcircles (in light grey), the corresponding Voronoi edges, and circles centred inside and outside edge v_g passing through end points of g (with dashed and dotted line, respectively).

The test for α-neighbours is based on the relation between Delaunay edges (D-edge) and Voronoi edges (V-edge). Figure 6.6 shows a partial Delaunay triangulation and Voronoi diagram of A. D-edges are drawn in light grey, V-edges are drawn with thick light grey lines. Every D-edge has a corresponding V-edge, e.g. D-edge g has its corresponding V-edge v_g. V-edge v_g joins the centres of the circumcircles of the two Delaunay triangles of which g is an edge. Circumcircles of Delaunay triangles are drawn in light grey. Any circle passing through the end points r and u of g has the centre in l, the perpendicular line to g at its midpoint (the dashed grey line). The V-edge v_g lies on this line. A Delaunay triangle has the property that its circumcircle does not contain any other point of the set. The two circumcircles of triangles sharing edge g have their centres at the end points of edge v_g. Any circle passing through end points of g and centred inside v_g has no other point from A, e.g., the circle in dashed line. Moving the centre of the circle along line l, outside v_g, produces a circle that encloses m or n, and possibly other points from A, e.g., the circle with dotted line in Figure 6.6 encloses point m.

Two α-neighbours are connected through a D-edge, and the centre of one of the α-circles passing through them is on the corresponding V-edge. No other point

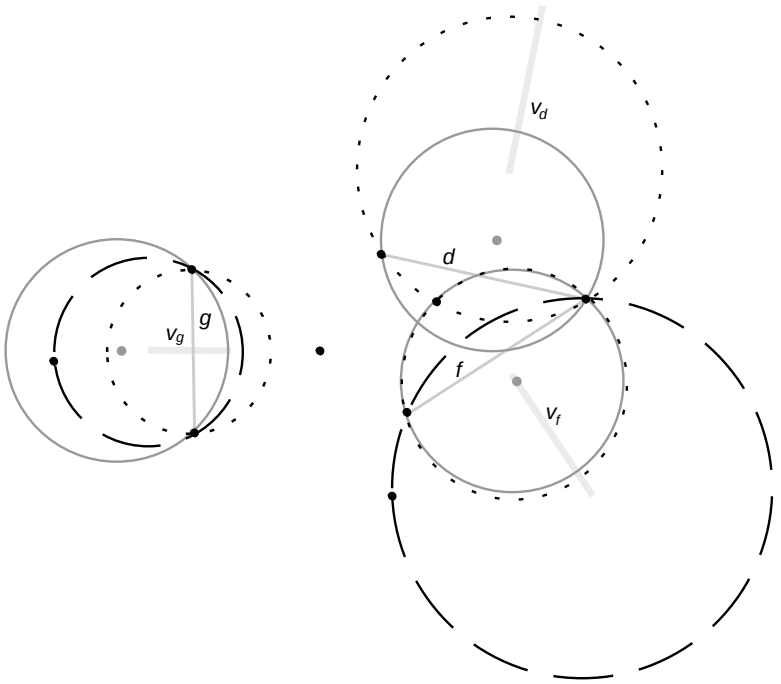

Figure 6.7: Testing Delaunay edges for being in the α-shape by using min and max circles (with dotted and dashed lines, respectively), and α-circles passing through end points of D-edges (in grey).

of the set lies inside the α-circle that has its centre in the V-edge. This property is used to test a D-edge whether its end points are α-neighbours, that is the D-edge is in the α-shape. The centre of an α-circle passing through end points of a D-edge is on the corresponding V-edge, if the radius $-1/\alpha$ is between the minimum and maximum distance from any of these points to the V-edge. If a D-edge is in the boundary of the convex hull, its corresponding V-edge is a half line, in which case the maximum distance is infinite. The test reduces to checking whether the radius $-1/\alpha$ is bigger than the minimum distance. Figure 6.7 shows three different cases for calculating the minimum and maximum distance for a D-edge. Edge d is in the boundary of the convex hull A; its corresponding V-edge v_d is a half line. The minimum distance of an endpoint of d to the V-edge v_d is the circumradius of the triangle of which d is an edge. The circumcircle of this triangle is shown with dotted line. D-edge g intersects with its V-edge v_g. The minimum distance of a g endpoint to v_g is the half length of g (v_g is perpendicular with g at its midpoint). The minimum circle is shown with dotted line. The maximum distance is the distance from an endpoint of g to one of v_g endpoints. Endpoints of v_g are circumcentres of triangles of which g is an edge. The maximum distance is the maximum circumradius of the two triangles. The maximum (circum)circle is shown with dashed line. D-edge f does not intersect

with its V-edge v_f. The minimum and maximum distance for f are respectively, the minimum and maximum circumradius of the two triangles of which f is an edge. The minimum and maximum (circum)circles of triangles sharing edge f are drawn with dotted and dashed line, respectively. The α-circles and their centres are shown in grey. Only D-edge f is in the α-shape.

Figure 6.8: (a) α-shape for the initial α-value, (b) α-shape for decreased α-value.

The value of α defines the level of detail of the α-shape. The criterion we use to set an initial α value is that every point is at least vertex of one triangle in the α-shape interior. Figure 6.8(a) shows the α-shape taken from such α value, and 6.8(b) shows the shape after decreasing the α value. In Figure 6.8(b) we see that there are loose points, not part of any shaped object. A triangle is part of the α-shape if and only if the radius of its circumscribed circle is smaller than or equal to $-1/\alpha$ [32]. To set an initial α value for a point set S, we calculate for every point $s \in S$ the minimum radius r_s of circumscribed circles of all Delaunay triangles that have s as a vertex. The radius $-1/\alpha$ is set to the maximum of r_s values for all points of S. The α value calculated from that is used to estimate the shape. This initial value can be further adjusted by the user, if needed.

The α-shape of regularly distributed point data (grid data) resulting from the use of the initial α-value is shown in Figure 6.9. Grid points with membership value equal to zero have been excluded from the input.

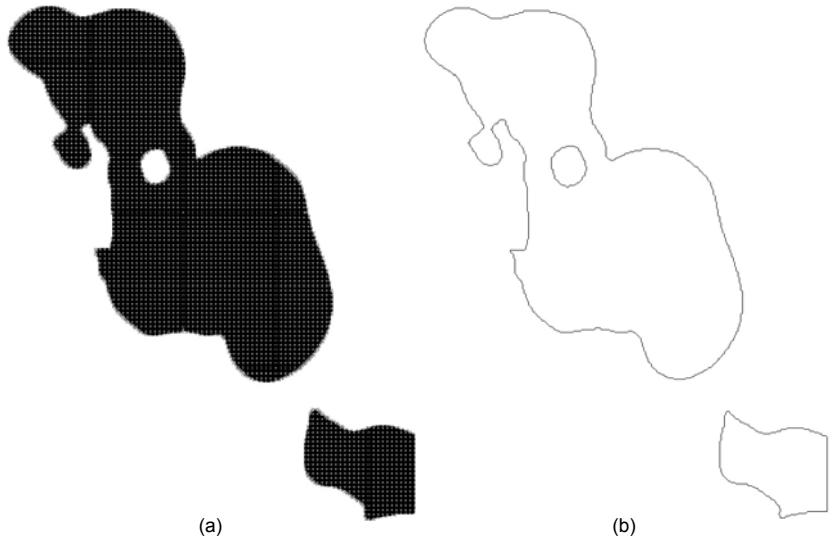

Figure 6.9: The α-shape of a regularly distributed point set: (a) input point set, (b) object boundaries detected with the initial α-value.

6.1.2 Creating and storing triangulations

The output of α-shapes consists of boundary lines together with the input data points. The lines constitute boundaries of the support sets of vague regions. They are processed to identify objects by their boundary. An identifier is assigned to every region object detected. Each region is then built from the constrained Delaunay triangulation performed on boundaries of the region and points inside the boundary. Its triangulation data is stored together with the identifier. Module `v.vague.triangle` performs the whole procedure. Each step of the procedure is explained in more detail in the following paragraphs.

Vague regions are identified from cluster boundaries. Each region may contain holes. Because GRASS does not support areas with holes, we have to check for them. A hole inside a vague region may contain other regions. Therefore, from cluster boundaries we have to detect which boundary is an outer boundary of a region and which is a hole boundary. We check for every area boundary if it lies inside other areas. It is the outer boundary of a vague region if it lies inside an even number of areas. It is the boundary of a hole if it lies inside an odd number of areas. Figure 6.10 shows two different configurations of outer boundaries and holes: area A is inside area B and it is a hole. Area C is inside area D which in turn is inside area E. The boundary of E and the boundary of D form (the boundaries of) a vague region. The boundary of C gives another vague region.

After identifying all vague regions, we perform a constrained Delaunay triangulation for every region. A triangulation is the division of a surface or a polygon

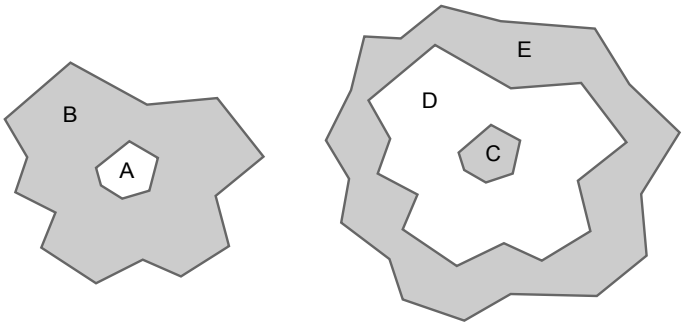

Figure 6.10: Identifying vague regions (shown in grey) from areal boundaries: one vague region in the left; two vague regions in the right, one enclosing the other.

into a set of triangles, such that each triangle edge is shared by two adjacent triangles. A constrained Delaunay triangulation is a triangulation of vertices with predefined edges. It consists of four steps [86]:

1. creating a Delaunay triangulation from the input vertices;
2. inserting missing line segments from the boundary and deleting the Delaunay edges that overlap with them;
3. removing triangles at concavities and holes;
4. adding more points in order to improve the quality of the triangulation.

Figure 6.11 illustrates the first three steps of the constrained Delaunay triangulation. Part (a) shows boundaries of a vague region. For simplicity of illustration we consider a constrained Delaunay triangulation with only line boundaries as input. Figures 6.11(b)–(d) show the results from the first, second and third step, respectively. To visualize change in different steps, the boundary edges are always drawn in black. In the two intermediate steps, the other edges are shown in grey, and the edges to be removed are shown with dashed grey lines.

We use TRIANGLE by Shewchuk [86] to perform the constrained Delaunay triangulation. GRASS data is transformed into the TRIANGLE data format. The program is run on the transformed data, and its output is transformed back to GRASS vector format. The program cannot handle holes when the input is big (i.e., more than 50,000 points). We use TRIANGLE to perform the triangulation constrained only on the outer boundaries. Then we remove all edges inside holes. We expect the core of a region to have many input points, therefore many flat triangles will be created. We remove the redundant core triangles, by first constructing the boundary of the core from all flat triangles of value 1, then performing the constrained Delaunay on the boundary. Figure 6.12 shows triangulation of a vague region after the simplification of its core triangulation. Points

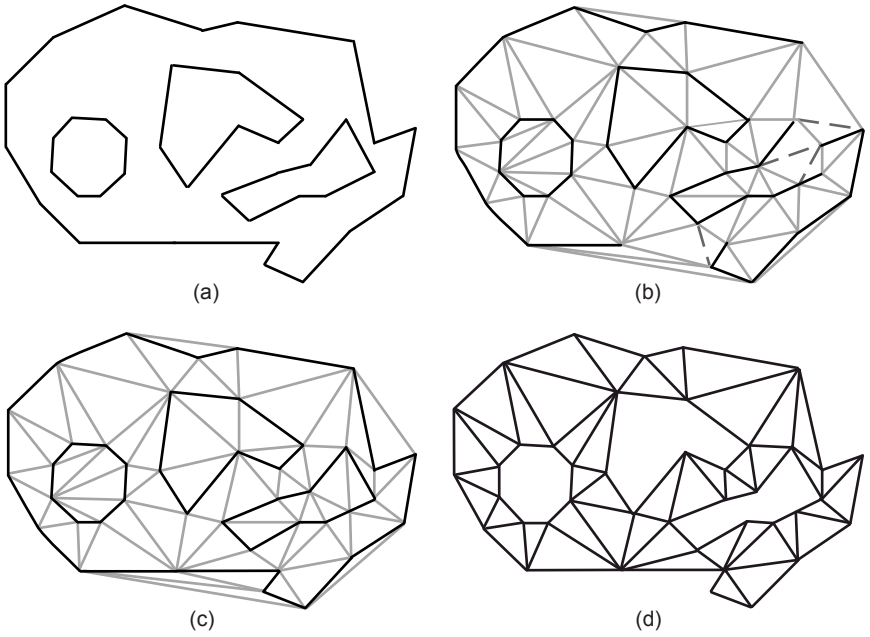

Figure 6.11: Results of three steps constrained Delaunay triangulation for a vague region: (a) region boundaries, (b) Delaunay triangulation, (c) insertion of missing boundary lines, (d) removal of triangles outside the boundary and inside holes.

are very dense outside the core, which makes the triangulation very dense. Core triangles are visible after the simplification phase.

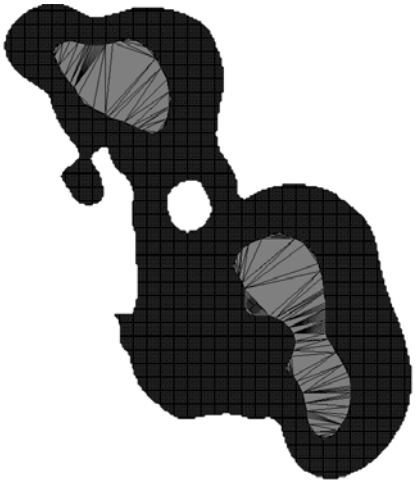

Figure 6.12: The triangulation of a vague region after reducing core triangles. Triangulation is very dense outside the core; only core triangles are distinguishable.

For each vague region, triangulation edges are stored as *boundary* lines with (x, y, z) coordinates in the `coor` file. An attribute is used to store the identifier of the vague region to which the edge belongs. Topology data of triangles and holes, e.g., indices of line boundaries, is stored in the `topo` file.

The output of module `v.vague.triangle` is a data layer that contains information about a vague class. The algorithm of the module is presented below:

```
create a list VR of vague regions from cluster boundaries
for every region r in VR
    find all points that are inside its boundary
    build constrained Delaunay triangulation from r's boundary and points
    if r has holes
        remove edges inside every hole
    store triangulation edges together with r index in VR
    store topology for triangles and holes
rof
```

Every time a layer of vague regions is used, its information is compiled from stored data, and put in memory in a list Vague_region of vague regions. Each element of Vague_region contains a list of triangles and a list of holes. Information about triangles and holes is built using several GRASS data structures that are inside a main structure, Map_info [62]. Every time a vector layer is used, data from `coor` and `topo` files is read and put into these structures. The indexed lists Area and Line inside a Map_info structure contain topology information of areas and lines, respectively. Each element of Area contains a list of indices of lines that constitute its boundary. The index of the Line list is used to connect each element with the corresponding element in another list that contains attribute values. Figure 6.13 shows relations between these data structures.

6.1.3 Creating themes from several layers

A data layer keeps information about one class. For example, a layer `forest` keeps data about vague regions that belong to a class 'forest' of a 'vegetation' theme. The vegetation theme consists of several classes: forest, grassland, shrubs, etc. Vague classes could overlap with each other, e.g., forest and shrubs, in which case triangulations representing them will overlap. Overlapping lines or areas are not allowed (handled) by the topology, which we need for compiling object information. Therefore we cannot store all classes together. However, we often need to have all classes of a theme together, to operate with all or a selected subset of them.

We create two tables to store the relation class and theme it belongs to. Figure 6.14 shows their schemas. Tables are stored in DBF format, which is a format integrated in GRASS, meaning that no connection to an external database is

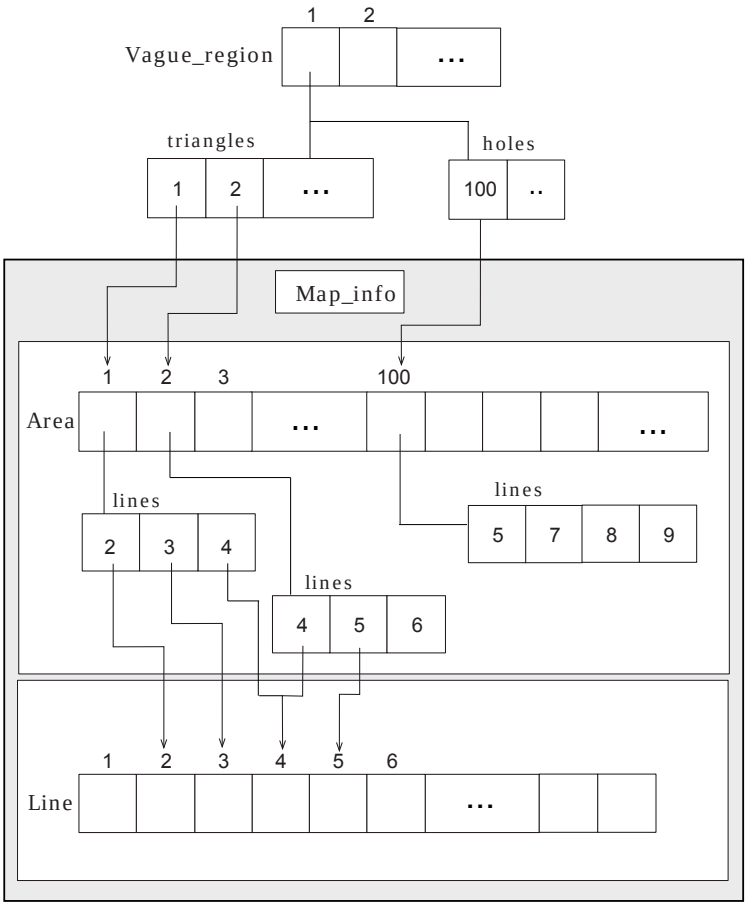

Figure 6.13: The GRASS structures (inside Map_info) used to store information about vague regions.

needed. Table `Themes.dbf` stores theme name and description, respectively in

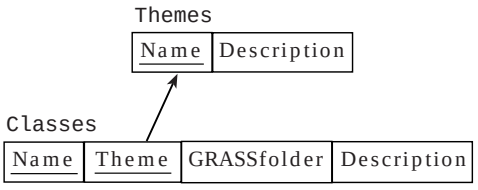

Figure 6.14: Tables storing the relation between a vague theme and its vague classes.

Name and Description. Name is the identifier. The table `Classes.dbf` stores class name, the name of the theme it belongs to, the name of the GRASS folder where (layer) data is stored, and class description, respectively in Name, Theme,

GRASSfolder, Description. The combination Name, Theme is the identifier of the table. Theme refers to `Themes.dbf` table.

Module `v.vague.combine` binds several layers in a theme. The created theme is added to the `Themes.dbf` table and its list of layers is added to the `Classes.dbf` table (each layer as a new record in the table). The module allows users to add or remove layers from an existing theme, or to delete a theme. Changes are then reflected in `Classes.dbf` and `Themes.dbf` table.

A theme of vague region layers forms a vague partition. Objects in a layer do not overlap with each other. This is assured by the triangulation process. When adding layers to a theme with the `v.vague.combine` we check if objects from different layers overlap only in their uncertain part. We give a warning when this criterion is violated, and store a report for the violating cases (object identifiers, layers they belong to, and the theme).

6.2 Visualizing objects and displaying information

Visualization of objects in a layer is done using colour brightness to display levels of membership values, e.g., using grey scales. Different layers are displayed using different colour hue for each layer, and brightness for membership value on each layer. Objects can be selected by clicking. Selected objects are shown in a separate colour, not used for displaying layers. Figure 6.15 shows a theme of vague regions having two classes. Different colour hue is used for each class. Darker colour shows higher membership value. An object is selected and shown in yellow colour.

Module `v.vague.what` performs visualization of layers. It allows to select objects from a layer, and displays information for a given location in a layer. A theme (created by `v.vague.combine` module) is the input for this module. Vague regions can be displayed by drawing only triangle edges of their triangulations. The set of triangles is drawn with a different colour for each layer. The interpolated values inside triangles can be used to colour the full extent of objects in a layer, as it is shown in Figure 6.15. Section 6.2.1 explains more thoroughly the techniques used for this visualization.

The interface of `v.vague.what` uses two windows: the *control window* that contains the list of layers, one layer in a separate tab, and the *display window* for drawing the layers. Layers are first drawn in the order in which they are stored in the theme. The selected layer (tab) in the control window becomes the top layer in the display window. An object of the top layer can be selected by clicking at a location inside the object. The triangle containing the given location is found first, then the object the triangle belongs to. The object is highlighted. The information of the given location is displayed in the control window. This information contains the membership value of the location, the directory stor-

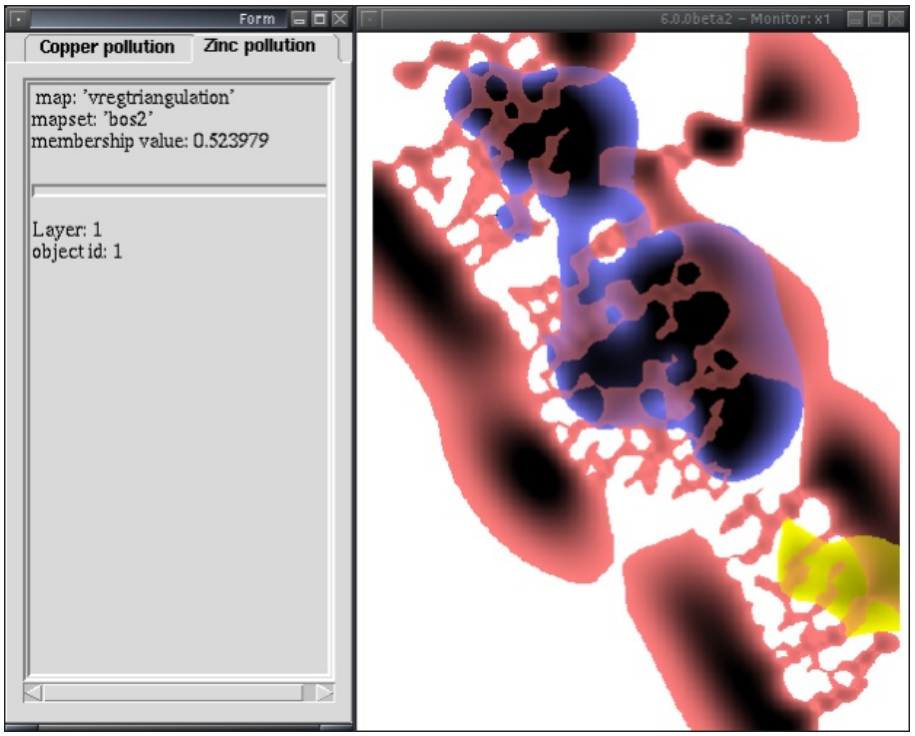

Figure 6.15: The visualization of several layers, each drawn in separate colour hue. A selected object is shown in yellow.

ing the layer data, and the identifier of the object (it falls in). The membership value is calculated by using linear interpolation inside the triangle containing the location. The user can select several objects and output them to a new layer.

6.2.1 Visualization techniques

A GRASS module, `d.vect`, is used to draw vague region layers with triangle edges. Each layer is drawn in a different colour. A new module, `d.vague`, is created to fill triangles with interpolated values and draw them. The module uses a scanline rendering algorithm [107]. The rest of the section describes how this module works.

GRASS functions are used to map the coordinates of triangle vertices to screen coordinates. A screen is a raster map consisting of pixels. Every pixel has a red, green, and blue value, specifying its colour. Scanline rendering fills a triangle by drawing a line at a time, starting at the top of the triangle. All pixels of a line are drawn, which are part of this triangle. The line is then moved down and

the drawing is repeated until the bottom of the triangle is reached. Figure 6.16 illustrates how the algorithm draws a triangle. Linear interpolation is used to calculate the membership value at each location. Every triangle lies on a plane defined by the three triangle vertices. Any point (x, y, z) of the plane satisfies the equation $z = ax + by + c$. The a, b, and c values are calculated from the coordinates of the three triangle vertices. The membership value for any point in the triangle is calculated by replacing its (x, y) location in the above equation.

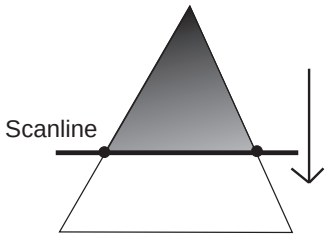

Figure 6.16: Scan-line visualization technique.

A different colour is used for each layer, selecting only colours that have the same saturation, so no colour draws more attention than others. The brightest colour is specified for every layer. The saturation and hue are kept constant for a layer. The brightest colour is used for the lowest membership value of the layer. The colour with brightness equal to 0 is used for the membership value equal to 1. The corresponding brightness value for any other membership value is calculated by first inverting the $[0, 1]$ interval (of membership values) then stretching it linearly to the range of brightness values. Because the monitors work with RGB colours, we use a function that applies that idea in an RGB model. The function that maps memberships value to RGB colours is

$$f : [0, 1] \rightarrow \mathbb{R}^3 \text{ such that for every } \lambda \in [0, 1]$$
$$f(\lambda) = (1 - \lambda) \times (R_{max}, G_{max}, B_{max}),$$

where $(R_{max}, G_{max}, B_{max})$ is the specified brightest colour of the layer.

If multiple layers overlap, transparency is used to draw them together. This is calculated with alpha-blending [107]. The colour $(R, G, B)_{new}$ at every location in the overlapping part is calculated as

$$(R, G, B)_{new} = 0.5 \times (R, G, B)_{S_1} + 0.5 \times (R, G, B)_{S_2},$$

where $(R, G, B)_{S_1}$ and $(R, G, B)_{S_2}$ are the colours at that location for the top and bottom layer respectively. When more than two layers are to be drawn, first the two bottom layers are calculated. Then for every layer to be added on top of them, the above formula is reused, replacing $(R, G, B)_{S_2}$ with the calculated colour for the previous layers, and $(R, G, B)_{S_1}$ the colour of the new layer.

6.3 Operators on vague objects

The operators we implemented are union, intersection, and difference. This section describes shortly these operators for vague points and vague lines, for being simple, and concentrates more on the operators for vague regions.

The operators for vague points take two VMPoint layers as input, and output a new VMPoint layer. The three operators check first for simple point objects in the two input layers that have identical location. Union operator selects from each pair (of identical location points) the point with the higher membership value, and puts it in the result layer. It also adds all other (unmatched) points from both layers to the result layer. The intersection operator selects the point with the lower membership from each pair of matched objects, and inserts them in the result layer. For each pair of matched points, the difference operator calculates a new membership value as their fuzzy difference, and inserts in the result layer a point with the common location and the calculated membership value. It also adds to the result layer all unmatched points from the first layer.

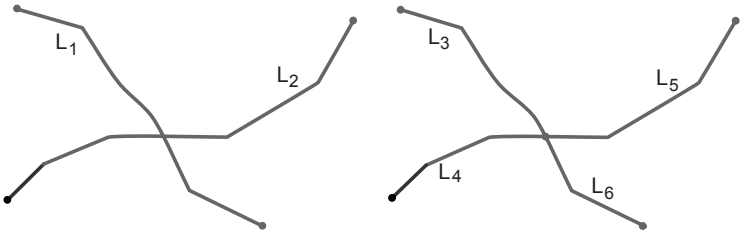

Figure 6.17: Union of two vague lines.

The union operator for vague lines takes two VMLine layers as input, and outputs a new VMLine layer. It first checks for simple lines that intersect. The operator splits the intersecting lines at the intersection point, which becomes common node for the newly created lines. The membership value at the common node is the maximum value of the memberships at this location in the two initial lines. The new lines are inserted in the result layer. Lines that do not intersect are directly added to the result layer. Figure 6.17 illustrates union of vague lines: lines L_1 and L_2 from input layers are intersecting; lines L_3, L_4, L_5, L_6 are created and added in the result layer. The intersect operator for vague lines takes two VMLine layers as input, and outputs a VMPoint layer. It checks for intersecting lines, creates a point at the intersection location with membership value the minimum of the two lines at this location, and inserts these points in the result layer.

Operators for vague regions take two VMRegion layers as input, and output a VMRegion layer. A VMRegion layer is a surface that consists of several non-overlapping triangulations. Triangulations belonging to different surfaces might overlap. The overlapping zone between two surfaces is important for the opera-

tors, because they treat differently triangles inside and outside the overlapping zone. The overlapping zone may consist of several separate areas. Operators start by detecting the intersection line between triangulations. The line separates the input triangulations into parts that are used to construct the output surface. Union is built by taking the higher triangulation inside an overlapping area, and adding unchanged triangulation parts that are outside the overlapping zone. Intersection is built by taking the lower triangulation inside any overlapping area. The difference operator re-calculates values at triangulations inside an overlapping area, and adds unchanged triangulations of the first surface that are outside the overlapping zone. The basic steps for the operators are:

1. Add vertical faces along boundaries of triangulations on both surfaces;
2. Detect the intersection line;
3. Re-triangulate both surfaces with this intersection line;
4. Select the right triangles from the re-triangulated surfaces to build the result.

The first three steps are the same for union and intersection operators. The forth step, selecting triangles for the output, is different. We explain the first three steps, and give the algorithm of the forth step for union. This algorithm can be used for the intersection with only few changes, which are described shortly. Difference operator requires specific treatment in most of the steps, therefore we explain it separately.

To add vertical faces along a triangulation boundary we consider every edge of the boundary. Two vertical triangles are created for each edge and added to the triangulation. Suppose the edge is defined by points $p_1 = (x_1, y_1, z_1)$ and $p_2 = (x_2, y_2, z_2)$. The face determined by p_1, p_2 and their projections in plane, $p_3 = (x_1, y_1, 0)$ and $p_4 = (x_2, y_2, 0)$ is split into two triangles, and these are added to the triangulation.

Detection of the intersection line is performed using the GNU triangulated surface library (`http://gts.sourceforge.net`). It produces the set of looped lines where the two surfaces intersect. The intersection between two triangles can be a line segment, or a polygon if the triangles lie in the same plane. Therefore, the intersection line between two surfaces consists only of straight-line segments.

The intersection line is added as constraint to the triangulations of both surfaces. On each surface, only triangulations containing a part of the intersection line are to be re-triangulated. For every triangulation (to be changed), the triangles containing a line segment from the intersection are split into several new triangles. The other triangles are added unchanged to the new triangulation. The splitting of triangles is done by greedy triangulation, as described in [111]. Figure 6.18 illustrates the splitting of three triangles. The algorithm below explains

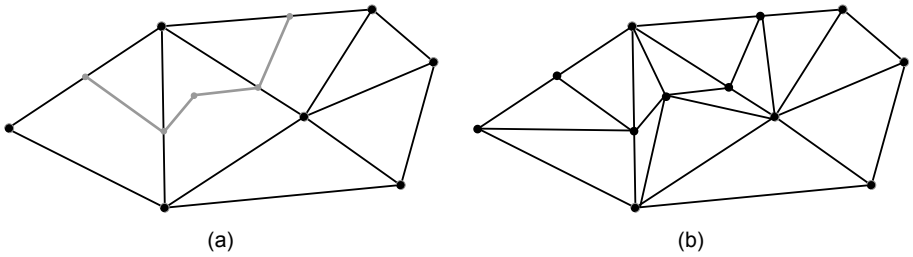

Figure 6.18: Re-triangulation: (a) old triangles and intersection line in grey, (b) new triangles after re-triangulation.

how splitting is done. Let V_i be the set of vertices of the triangle i, and L_i the set of line segments of the intersection line passing through the triangle i. Let us denote by E_i the set of edges of the new triangulation in triangle i, and P_i the union of V_i with the end nodes of L_i.

```
set E_i = L_i
for all points p from P_i
    create D_p as edges from p to any other point in P_i
      sorted in ascending order on length
    for every edge d = (p,q) in D_p
        if d ∉ E_i and no edge e ∈ E_i intersects with d
          and no point r ∈ P_i − {p, q} lies on d
            add d to E_i
        fi
    rof
rof
```

After the re-triangulation, the relation of a triangle from one surface to any triangle in the other surface is one of the three cases:

- The three vertices of the triangle are on the other triangle: the triangle is part of an area that is contained in both surfaces.
- One or two vertices are on the other triangle: the triangle intersects the other surface only at a point or along a line (the edge joining the two vertices). The rest of the triangle is above or below the other surface.
- No vertex of the triangle is on the other surface: The triangle does not intersect with the other surface. It is completely above or below the other surface.

A point is above a surface when its z coordinate is higher than the z value of the surface at the point location. A point in the interior of a triangle from one

surface determines the relation of this triangle with the other surface. When an interior point is above, on, or below the other surface, the whole triangle is above, on, or below the other surface, respectively. The centre of the incircle[3] of a triangle is always in the interior the triangle, and can therefore be used for such testing.

The intersection line consist of several looped lines. Triangles of one surface that are inside a looped line are all above the other surface, or all below the other surface. So the union of two surfaces is formed by groups of triangles bounded by these looped lines. The algorithm that performs union of two re-triangulated surfaces S_1 and S_2 and outputs a surface S is given below:

```
set S to an empty surface
for any triangle t from surface S1
    generate a point p(x, y, z) in the interior of t
    if S2 exist in location (x, y)
        if p is on or above S2
            add t to S
    else
        add t to S
    fi
rof
for any triangle t from surface S2
    generate a point p(x, y, z) in the interior of t
    if S1 exist in location (x, y)
        if p is above S1
            add t to S
    else
        add t to S
    fi
rof
```

After the output surface is created, separate triangulations are detected and given an object identifier. They are the VRegion objects of the result layer.

The algorithm for calculating the intersection is quite similar. The difference is that a triangle in one surface is discarded if the other surface does not exist at the interior location of the triangle. Also, the testing for the z value of a point and a surface is reversed: the point should be on or below the surface.

Difference between surfaces S_1 and S_2 is the intersection of S_1 with the complement of S_2. Outside S_2 its complement is equal to 1. The values of S_1 are everywhere smaller or equal to 1, therefore the intersection outside S_2 is equal to S_1. Thus, we only need to build the complement of S_2 inside its boundaries.

[3] The incircle is the inscribed circle of a triangle, i.e. the unique circle that is tangent to each of the triangle's edges.

Surface S_2' is built from S_2 triangulation by inverting the values of each triangle vertex. The core of S_2 will be a hole for S_2', therefore core triangles are not put in S_2'. Holes of S_2 constitute the core of S_2'. Boundary of each hole is triangulated and included in S_2', all as flat triangles with value 1.

To build the result surface we pass through the same steps of union and intersection. First vertical faces are added along boundaries of S_1 and S_2'. For S_1 faces are added from the boundary line to its projection in the horizontal plane, that is down to membership value 0. For S_2' vertical faces are built along the boundary up to membership value 1. The intersection line between the two surfaces (extended by the vertical faces) is detected, and both surfaces are re-triangulated with this line. The forth step, selecting the right triangles, is quite similar with the intersection operator, except that every triangle of the first surface is included in the output if the second surface does not exist at its location(s). We keep to notations S_1 and S_2' to denote now the surfaces taken after re-triangulation. The algorithm for the last step, selecting the right triangles for the output surface, is

```
set S to an empty surface
for any triangle t from surface S1
    generate a point p(x, y, z) in the interior of t
    if S2' exist in location (x, y)
        if p is on or below S2'
            add t to S
    else
        add t to S
    fi
rof
for any triangle t from surface S2'
    generate a point p(x, y, z) in the interior of t
    if S1 exist in location (x, y)
        if p is below S1
            add t to S
rof
```

As for the other operators, after the output surface is created, separate triangulations are detected and given an object identifier. Triangle edges of every triangulation are stored together with this identifier in the new layer.

6.4 Discussions

The current implementation of α-shapes gives good results for data sets with more or less regular sample density. If the sample density changes too much,

the α-shapes do not work very well. Density-scaled α-shapes [98] provide a solution to this. Implementation of their algorithm would make the clustering process more robust. On the other hand, density-based clustering algorithms, e.g., DBSCAN, OPTICS, would be another possible solution to clustering of data points. It seems though that the technique aims at clustering of points, and does not provide for delineation of cluster boundaries.

The only input type we consider for vague regions is data points. Data about vague regions could also be (vague) lines, e.g., lines having the same membership value (α levels). A part of the procedure to create vague regions from line input would be the same as for point input. They both need the constrained Delaunay triangulation. The identification of objects would need another technique.

The operators we implemented are only part of the set of operators we defined in the previous chapters. Some of the other operators will be built upon these implemented basic operators. For example, overlay and fusion of vague partitions will make use of the modules for the intersection and union of vague regions, respectively. The implementation of boundary operator is straightforward, identification and then removal of the core. Frontier operator is also straightforward for vague points and lines. Frontier of vague regions extracts their discontinuity lines, of which we have not taken care in this implementation. The next paragraph discusses this issue. Some of the spatial relations are based on the intersection of vague objects. Other spatial relations are based on operators like bounded difference, and absolute difference. They would therefore need an extension of these basic operators for their implementation. Metric operators are independent of the above operators. They would, though, make use of existing GIS functions for geometric measures. For example, the average area will use the functions for the calculation of volumes.

The membership function of a *vague region* can have discontinuities along lines. These discontinuities result in vertical faces in triangulations. The work presented in the chapter does not consider discontinuity lines. We build and store the topology of triangulations using GRASS topology, which ignores vertical faces and areas adjacent to them. We do not need to store vertical faces, but we do need triangles adjacent to them. To allow discontinuity lines we could build separate functions for the topology of triangulations from GRASS topology functions, modifying the last. GRASS topology builds the connectivity of 3D lines (through their nodes) correctly, but considers only their x, y coordinates when building areas from lines. We can use line connectivity to build the adjacent triangles to vertical faces, changing the existing functions to consider the special cases we need. Operators on vague regions are to be modified in order to consider discontinuities.

6.5 Summary

The chapter showed how vague spatial objects can be stored and manipulated using existing GIS functionality for vector data format. GRASS spatial features and data structures were employed for storing information about vague objects. Points, lines, and triangulations were used to store vague points, vague lines, and vague regions, respectively. These simple types represent identifiable objects. Classes of simple objects were stored in separate vague layers. Classes of a theme were bound together via relations stored on database tables. A theme of vague region classes forms a vague partition, which allows for a soft classification of space that is important for many spatial applications.

A few modules were offered to handle vague objects: a module that creates layers of vague objects from input data points; a module that visualizes vague layers in the screen, and allows to retrieve and display information about their objects; some modules that perform different operations on vague layers. Union, intersection, and difference operators were implemented for vague objects. These are basic operators, on which other spatial operators can be built.

Chapter 7

Introducing time in the system

This chapter considers vague objects describing natural phenomena that are changing over time. The change can be continuous, or abrupt at discrete moments of time. A forest is well described as a vague object, because of the transition zones to other land uses resulting from the vagueness of the defining concepts. In time, forest is subject to *continuous* change. To the contrary, a state or a province has well defined boundaries by some legal agreement. The change to these objects happens at *discrete* times, e.g., due to wars or legal agreements. We expect continuous changes to be the most frequent ones for vague objects. We are often interested to know the history of change, and not just only the current state of a phenomenon. The history of change of a phenomenon might then help to predict the future.

The system of vague spatial types and operators introduced in the previous chapters can only handle the current state of a vague spatial phenomenon, i.e., a static phenomenon. In this chapter, we describe how this system can be extended in order to handle dynamic vague objects. The objective is to show the possibilities for extending the system of vague types and operators with the time dimension. We do not aim at providing a complete and fully formal system of temporal types and operators. Looking at what is offered by spatiotemporal databases, we discuss a possible implementation of dynamic objects. Considering two applications, we show the use of the proposed types and operators for monitoring and analysis of dynamic vague phenomena.

The material of the chapter is presented in three parts. Section 7.1 looks at how time can conceptually affect the proposed object types. It suggests several dynamic types and operators constructed from the previously defined vague types and operators. Section 7.2 is dedicated to the implementation issues. It describes how a spatiotemporal system can be used to handle vague objects changing over time. Section 7.3 describes two spatial applications, and their analysis performed by our operators. Section 7.4 summarizes the results of this chapter.

7.1 Types and operators for dynamic vague objects

The system we proposed for describing spatial phenomena under vagueness, takes an object view of space. This means that we are interested in distinct entities in space. The proposed object types categorize the different spatial entities that one may encounter. These object types were defined to describe entities at one moment in time. Including time in this object view of space, we distinguish two approaches towards modelling vague spatial phenomena. The first approach considers vagueness as spatial, and we are interested how a vague spatial phenomenon changes in time. The second approach generalizes towards vagueness in space *and* time. This occurs when the properties defining the objects are in the space–time domain, and they are described in vague terms. For example, a riparian area has a vague extent that depends on the season, dry or rainy. The concept of a 'season' is vague in the time domain.

In this section we introduce dynamic vague object types, and several operators on these types. We emphasize types and operators for vague objects changing over time, i.e., objects exhibiting spatial vagueness, as we believe there are more applications needing such objects. Types and operators proposed for spatiotemporal databases provide a basis for this. We follow the system of types and operators proposed by Güting et al. [47, 48] for moving objects, adjusting it to our vague objects changing over time. At the end of the section, we introduce shortly objects exhibiting vagueness in space and time.

A vague object of a simple or general type is represented by a fuzzy set in $\mathbb{R}^2$. To describe a vague object changing over time we need to know for any moment in time the fuzzy set representing the object. In general, a changing vague object can be represented as a partial function from the time domain to the set of fuzzy sets in $\mathbb{R}^2$, $ch : \mathbb{R} \nrightarrow \mathcal{F}(\mathbb{R}^2)$, satisfying semi-continuity conditions. To formally define the continuity, we need a metric in a function space. We will not elaborate on this, except for noting that the conditions allow continuous and discrete change in time. For any vague type introduced in Chapter 3, we build a new type to represent the change over time. For example, a changing vague object of a general type is an element of the set *chGVSpatial*, defined as

$$chGVSpatial \equiv \left\{ o : \mathbb{R} \nrightarrow \mathit{GVSpatial} \mid o \text{ is semi-continuous} \right\}.$$

The naming for each changing object type follows the same rule as above, *ch* proceeding the name of the corresponding spatial type. In this fashion, the type of vague regions changing in time is *chVRegion*.

A collection of operators can be defined over the general *chGVSpatial* type, or over specific types. These operators should be able to express the most common questions one may ask about changing objects in time. We present two groups of operators: one that performs questions in the time domain, and a group of operators that extend with time the (static) operators proposed in Chapters 3, 4, and 5. Operators of the first group are a subset of those proposed in [47, 48].

We select one operator from each group of static operators, and show how they can be extended in the time domain. The other operators can be constructed similarly. The operators that are presented here are chosen such that they provide an understanding of the analysis that can be performed on changing vague objects.

The operator $\mathcal{DefTime}$ returns the time intervals during which a $\mathcal{chGVSpatial}$ object exists. This is the domain of the function representing the $\mathcal{chGVSpatial}$ object. The $\mathcal{DefTime}$ operator is defined as

$$\mathcal{DefTime} : \mathcal{chGVSpatial} \to \mathbb{P}(\mathbb{R})$$
$$\forall o \in \mathcal{chGVSpatial},\ \mathcal{DefTime}(o) = \mathrm{dom}(o).$$

The operator $\mathcal{DefTime}$ provides the lifetime of a changing vague object.

The operator $\mathcal{AtInstant}$ returns the state of a $\mathcal{chGVSpatial}$ object at a given instant. It is defined as

$$\mathcal{AtInstant} : \mathcal{chGVSpatial} \times \mathbb{R} \to \mathit{GVSpatial} \times \mathbb{R}$$
$$\forall (o,t) \in \mathcal{chGVSpatial} \times \mathbb{R},\ \mathcal{AtInstant}(o,t) = (o(t),t).$$

The $\mathcal{AtInstant}$ operator provides snapshots of a changing object as a pair of (static) object state and instant of time.

The operators $\mathcal{Initial}$ and $\mathcal{Final}$ return, respectively, the state of a $\mathcal{chGVSpatial}$ object at the first and last instant of its existence, respectively. The $\mathcal{Initial}$ operator is defined as

$$\mathcal{Initial} : \mathcal{chGVSpatial} \to \mathit{GVSpatial} \times \mathbb{R}$$
$$\forall o \in \mathcal{chGVSpatial},\ t_0 = \min \mathcal{DefTime}(o), \mathcal{Initial}(o) = (o(t_0), t_0).$$

The $\mathcal{Final}$ operator is defined in a similar way.

The operator $\mathcal{Val}$ extracts the object from a pair of object state and time. It is defined as

$$\mathcal{Val} : \mathit{GVSpatial} \times \mathbb{R} \to \mathit{GVSpatial}$$
$$\forall (\mu,t) \in \mathit{GVSpatial},\ \mathcal{Val}(\mu,t) = \mu.$$

The operator $\mathcal{Present}$ returns true if a $\mathcal{chGVSpatial}$ object exists at a given time instant, and returns false otherwise. It is defined as

$$\mathcal{Present} : \mathcal{chGVSpatial} \times \mathbb{R} \to \mathrm{Boolean}$$
$$\forall (o,t) \in \mathcal{chGVSpatial} \times \mathbb{R},\ \mathcal{Present}(o,t) = \begin{cases} \text{true} & \text{if } t \in \mathcal{DefTime}(o), \\ \text{false} & \text{otherwise.} \end{cases}$$

We might be interested if a specific object exists at a particular moment. $\mathcal{Present}$ operator provides for this.

A collection of static (spatial) operators can be extended in the time domain through a process called lifting [47, 48]. The idea is to allow any argument of a spatial operator to be made spatiotemporal, i.e., to become a changing object, and to return a temporal type [47]. For example, we can perform the intersection between two changing vague multiregions *chVMRegion*, or between a *chVMRegion* object and a *VMRegion* object, both returning a *chVMRegion* object. A *VMRegion* object is constant and exists all the time. It can thus be presented as a *chVMRegion* object that is a constant total function. All lifted operators are defined for changing objects, or a combination of changing objects with static objects. Considering the above interpretation of a static object as a changing object, we can use the same definition of a time-dependent operator for changing and static object arguments. Under this remark, the intersection operator between changing or static vague regions is defined as

$$\mathcal{RIntersection} : \mathit{chVMRegion} \times \mathit{chVMRegion} \rightarrow \mathit{chVMRegion}$$
$$\forall o_1, o_2 \in \mathit{chVMRegion},$$
$$\forall t \in \mathrm{dom}(\mathcal{RIntersection}(o_1, o_2)) = \mathcal{DefTime}(o_1) \cap \mathcal{DefTime}(o_2),$$
$$(\mathcal{RIntersection}(o_1, o_2))(t) = \mathit{RIntersection}(o_1(t), o_2(t)).$$

The result object from the $\mathcal{RIntersection}$ operator is defined in the periods of existence of both input objects. At any instant of its existence, the result object is calculated from the intersection between the states of the two input objects at that instant. All the other operators returning spatial types can be extended similarly to the time domain.

To perform lifting of spatial relations we need a changing truth degree that is a partial function $f : \mathbb{R} \nrightarrow \mathit{TruthDegree}$. The type of changing truth degrees is $\mathit{chTruthDegree} \equiv \left\{ f : \mathbb{R} \nrightarrow \mathit{TruthDegree} \right\}$. Each of the spatial relations is extended to a time-dependent relation between two *chGVSpatial* objects that returns a *chTruthDegree* value. For example, a $\mathcal{D}isjoint$ relation is defined as

$$\mathcal{D}isjoint : \mathit{chGVSpatial} \times \mathit{chGVSpatial} \rightarrow \mathit{chTruthDegree}$$
$$\forall o_1, o_2 \in \mathit{chGVSpatial}, \mathrm{dom}(\mathcal{D}isjoint(o_1, o_2)) = \mathcal{DefTime}(o_1) \cap \mathcal{DefTime}(o_2),$$
$$\forall t \in \mathrm{dom}(\mathcal{D}isjoint(o_1, o_2)),\ (\mathcal{D}isjoint(o_1, o_2))(t) = \mathit{Disjoint}(o_1(t), o_2(t)).$$

It produces for any instant of the concurrent existence of the two objects, the degree of the *Disjoint* relation between the states of the two objects at that instant.

The lifting of metric operators requires two new types for changing vague measures: $\mathit{chVMeasure} \equiv \{f : \mathbb{R} \nrightarrow \mathit{VMeasure}\}$ for time-dependent alpha operators, and $\mathit{chMeasure} \equiv \{f : \mathbb{R} \nrightarrow \mathit{Measure}\}$ for time-dependent average operators. The time-dependent distance $\mathcal{D}istance$ between two *chGVSpatial* objects is defined as

$$\mathcal{AvDistance} : \mathit{chGVSpatial} \times \mathit{chGVSpatial} \rightarrow \mathit{chMeasure}$$
$$\forall o_1, o_2 \in \mathit{chGVSpatial}, \mathrm{dom}(\mathcal{D}istance(o_1, o_2)) = \mathcal{DefTime}(o_1) \cap \mathcal{DefTime}(o_2),$$
$$\forall t \in \mathrm{dom}(\mathcal{D}istance(o_1, o_2)),\ (\mathcal{D}istance(o_1, o_2))(t) = \mathit{Distance}(o_1(t), o_2(t)).$$

The changing measure function is defined for the periods of concurrent existence of the two objects.

The time-dependent average area $\mathcal{AvArea}$ of a changing vague multiregion, returns a changing vague measure defined in the periods of existence of the $ch\mathcal{VMRegion}$ object. For any instant in the lifetime of the object, it gives the *AvArea* of the state of the $ch\mathcal{VMRegion}$ object at that instant. The operator is defined as

$$\mathcal{AvArea} : ch\mathcal{VMRegion} \rightarrow ch\mathcal{VMeasure}$$
$$\forall o \in ch\mathcal{VMRegion}, \mathrm{dom}(\mathcal{AvArea}(o)) = \mathcal{DefTime}(o),$$
$$\forall t \in \mathrm{dom}(\mathcal{AvArea}(o)),\ (\mathcal{AvArea}(o))(t) = \mathit{AvArea}(o(t)).$$

Objects characterized from space-time properties expressed in vague terms, can be represented as fuzzy sets in the space-time domain, $\xi : \mathbb{R}^2 \times \mathbb{R} \rightarrow [0,1]$. The vague object types, points, lines, and regions, will be defined using semi-continuity of function in the $\mathbb{R}^3$ space. Set operators are regularized fuzzy sets operators for $(\mathbb{R}^3, \mathcal{T}_3)$. Spatial relations *Disjoint*, *Touches*, *Within*, and *Equal* can be used for these objects without change in their formulas. The *Crosses* and *Overlaps* will need change, as they are founded on (crisp) relations between 2D objects. Metric operators might require more fundamental change, being measures defined over space and time, while the two generally have different measures, not comparable to each other.

7.2 Implementation of dynamic vague objects

To store vague spatial phenomena that change over time, we can use the solutions offered by the spatiotemporal databases. Spatiotemporal data can be seen as objects embedded in a three-dimensional space, the cross product of (2D) space and time, considering only the valid time dimension. Figure 7.1 illustrates discrete and continuous change of (crisp) spatial objects. The point and the region in Figure 7.1(a) change at discrete moments of time, whereas the point and region of Figure 7.1(b) change continuously. A historical DBMS with tuple timestamping (see Section 2.3.3), and extended with spatial types, can be used for storing changing vague objects at discrete moments of time.

A vague object that changes continuously is presented by a function from (the continuous) time to a spatial object type, for the period of existence of that object. This assumes that the changing process is fully known and modelled, which however is generally not the case. Spatial objects are extracted from acquired data. This is often done at discrete moments of time. A more realistic solution is to store an object at various discrete moments of validity, together with functions that model the change (transformation) from a stored instant to the consecutive one. Modelling change between short periods of time is less prone

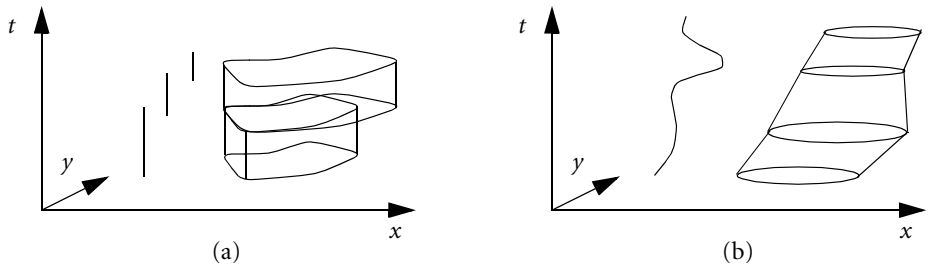

Figure 7.1: Spatial objects changing over time (taken from [51]): (a) discretely changing point and region, (b) continuously changing point and region.

to errors, and can be accepted as a better approximation of the real change. A changing vague object can thus be represented by a sequence of vague objects at discrete moments of valid time, together with a function for each of these moments, giving the change from that moment to the next one. In the coming paragraphs we propose general interpolation functions that assume a linear change through time. The change functions can also be built from knowledge about the change process.

The objects to be stored are simple type objects, that is objects of type *chVPoint*, *chVLine*, or *chVRegion*. A changing object is to be stored as a sequence of the corresponding static simple objects. The complete (approximate) information about a changing object is then compiled from a linear interpolation method.

A *chVPoint* object is a (time) sequence of *VPoint* objects, each associated with a time stamp, i.e., $\langle((x_1, y_1, mv_1), t_1), \ldots, ((x_n, y_n, mv_n), t_n)\rangle$. Linear interpolation should be performed between two consecutive elements of the sequence in order to get the state of the *chVPoint* object at any moment of time. A *chVPoint* object is thus a piecewise linear feature in a four-dimensional space.

A *chVLine* object is a (time) sequence of *VLine* objects, each associated with a time stamp. A *VLine* object itself is a sequence of vague points. To perform interpolation between two consecutive *VLine* objects, the vague point sequences representing them should have the same length, i.e., the same number of elements. If this is not the case, vertices are added in both lines in such a way they they divide each line in proportional parts. Linear interpolation is then performed between corresponding vague points elements of the *VLine* objects. This is the same interpolation as the one for a *chVPoint* object. Figure 7.2 illustrates the stored states of a changing line o at two consecutive moments t_i and t_{i+1}. The vague line $o(t_i)$ is stored as a sequence of three elements, $\langle(p_1, v_1^p), (p_3, v_3^p), (p_5, v_5^p)\rangle$. Any location p_j is composed of x and y coordinates. The value v_j^p is the membership value of the line at that location. The vague line $o(t_{i+1})$ is stored as a sequence $\langle(q_1, v_1^q), (q_2, v_2^q), (q_4, v_4^q), (q_5, v_5^q)\rangle$. Vertices $p_2^{'}$ and $p_4^{'}$ are added to the vague line $o(t_i)$. Their locations divide the line $o(t_i)$ in the same proportion as the vertices q_2 and q_4 divide $o(t_{i+1})$. The

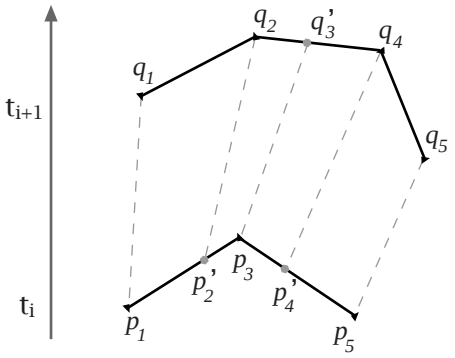

Figure 7.2: Two consecutive stored states of a changing vague line. Black triangles are stored vertices for each vague line state. Grey hexagons are vertices added in each line, dividing them in proportional parts.

membership values at $p_2^{'}$ and $p_4^{'}$ are calculated from linear interpolation along the line $o(t_i)$. The vertex $q_3^{'}$ is added to the vague line $o(t_{i+1})$ in a similar way. For any moment $t \in (t_i, t_{i+1})$, the state of the changing vague line $o(t)$ is presented by a sequence $\langle (r_1, v_1^r), (r_2, v_2^r), (r_3, v_3^r), (r_4, v_4^r), (r_5, v_5^r) \rangle$. Locations r_j and their membership values v_j^r are calculated from linear interpolation between the jth elements of the vague lines $o(t_i)$ and $o(t_{i+1})$.

A *chVRegion* object is a (time) sequence of *VRegion* objects, each associated with a time stamp. A *Vregion* object itself is a triangulation, stored as a set of triangle nodes. To have the state of *chVRegion* object at any moment of time t, an interpolation is to be performed between the two consecutive stored states at t_i, t_{i+1}, such that $t \in (t_i, t_{i+1})$. The state of *chVRegion* object at time t is a triangulation derived from the interpolation between the nodes of the stored triangulations presenting the states of *chVRegion* object at moments t_i and t_{i+1}. To perform the interpolation, the triangulations of time t_i and t_{i+1} should have the same set of triangle nodes. When this is not the case, other nodes are added to both triangulations to form a corresponding set of nodes. Re-triangulation is performed for both objects, and these are used for the interpolation. The state of the *chVRegion* object at time t is then determined as a set of triangle nodes calculated from the interpolation between the corresponding nodes of the two re-triangulated *Vregion* objects of times t_i and t_{i+1}.

The construction of a corresponding set of triangulation nodes is more complex than the construction of corresponding vertices between two lines (presenting states of a changing vague line). The addition of line vertices was done from scaling on line extensions. For triangulations, scaling should be done in two dimensions. Boundaries of the triangulations can be compared to find a transformation that brings them to a strong similarity. A correspondence is thus to be found first between nodes of the boundary. Parameters of an affine transfor-

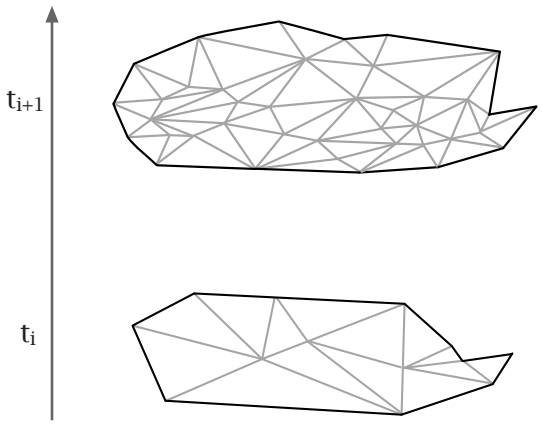

Figure 7.3: Correspondence between boundaries of vague region triangulations. Boundary edges are drawn in black, other triangulation edges are drawn in grey.

mation can be calculated from the best[1] correspondence. The transformation parameters can then be used to calculate missing nodes (within a threshold) from both triangulations. Figure 7.3 illustrates the stored states of a changing vague region o at two consecutive moments t_i and t_{i+1}. Boundaries of the stored triangulations for region states at t_i and t_{i+1} are to be be used for finding the best transformation.

The $\mathcal{A}tInstant$ operator extracts the state of a changing object for a given time. The interpolation is performed from this operator, then used by all the lifted operators. All the lifted operators can indeed be expressed in terms of $\mathcal{A}tInstant$ operator.

Vague objects in space and time are functions of three arguments, $\mu(x, y, t)$, with x, y providing location in space, and t location in time. Structures used for curves, surfaces, and solid bodies in a three-dimensional (3D) space would be a solution for the storage of these objects.

7.3 Monitoring and analysis of dynamic phenomena

We consider two applications, land cover change as discussed in [53], and beach erosion as discussed in [101]. Both applications deal with dynamic phenomena that are of a vague nature. The land cover application is interested in discrete changes happening in a period of a few years, whereas the beach erosion application is interested in the process of sand movement that is a continuous change.

[1] Different conditions can be put to define what is the best correspondence.

7.3.1 Application: land cover change

The study area of the land cover application is Shimla district of Himachal Pradesh in India. The land cover is composed of six classes: forest, agriculture, barren land, water, settlements, and snow cover. The area is quite hilly, and even classes like agriculture are vague, while in flat areas agricultural fields normally have crisp boundaries. The changes in the area are slow, and images of around 10 years difference have been chosen for change detection. A fuzzy classification is performed on satellite images from years 1987 and 1999, using membership functions defined from field knowledge. The result of the classification for each year is a vague partition, consisting of six vague land cover classes. The intersection of any pair of classes in two different years is used for the change detection. The change for each class can be measured by comparing the unchanged area of the class, with the area that has moved to another class.

Let us denote by $\mathcal{P}^1 = \left\{C_i^1\right\}_1^6$, and $\mathcal{P}^2 = \left\{C_i^2\right\}_1^6$, the vague partitions presenting the land cover of the first and second year, respectively. The overlay of the two partitions produces the intersection of any pair of classes in two different years. Let $\mathcal{P} = \left\{A_{i,j}\right\}_{i,j=1}^6$ be the result of the overlay, in which a vague class $A_{i,j}$ is the intersection of the class C_i in the first year with the class C_j in the second year. The change of a class C_i is measured by the ratio $R(C_i) = \sum_{j \neq i} \mathit{AvArea}(A_{i,j}) / \mathit{AvArea}(A_{i,i})$. A high value of this ratio demonstrates a big change for the class.

The two classes with the highest ratio R were the forest and the agriculture. Considering areas of the overlay classes, we can conclude which is the direction of change. It was seen that the biggest changes happened from forest to agriculture, and viceversa. Therefore, in an overall view, there was no considerable loss in any of these classes.

7.3.2 Application: beach erosion

The beach erosion application has its study area in the island of Ameland, in the north of the Netherlands (see Figure 7.4). Strong tidal currents cause the movement of sand around the island. Subsequent erosion and sedimentation in turn cause major changes along the shore. Figure 7.5 shows the digital elevation models (DEM) of the northern part of Ameland in the years 1989–1995, in a grey scale picture. The dark part is the sea, the greyish part is the beach, and the white part are the dunes. The Ministry of Public Works of The Netherlands is interested in stabilizing the process of change in the Ameland shore. To neutralize the erosion, they carry out beach nourishment. A model describing the process of changes would help to decide where the interaction is needed.

Changes along the shore can be described as transformation from and to foreshore, beach, and foredune. Studying these transformations can help under-

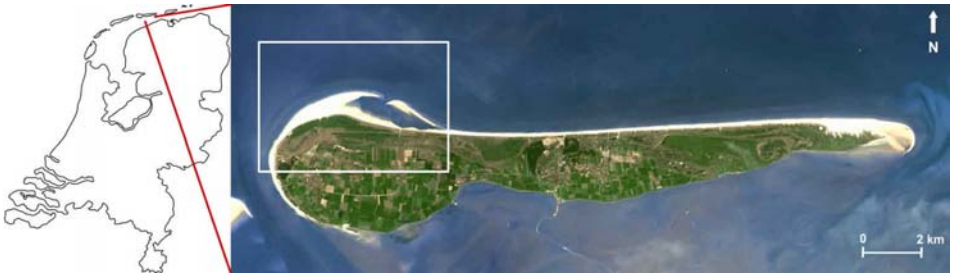

Figure 7.4: The study area: the north-west part of the Ameland island, northern of The Netherlands (taken from [100]).

standing the movement of sand. Transition zones occur between foreshore, beach, and foredune, hence justifying an approach by vague objects. These three classes are defined from the elevation values applying the membership functions shown in figure 7.6. The red bars show the elevation values for the crisp classification of the zone performed in the methodology of the Dutch National Institute for Coastal and Marine Management (RIKZ) for the preservation of the coast [75] (a short description is provided in [101]).

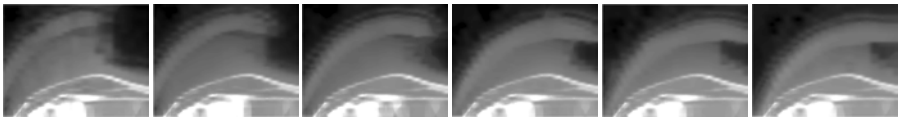

Figure 7.5: DEM of the northern part of Ameland for years 1990–1995. Low elevation values are shown in black colour, high elevation values in white. The dark part is the sea, the grey part is the beach, and the white part are the dunes.

The coastline position on 1st January 1990 is considered the basal coastline, and the state of the beach on that day is taken as a reference for the calculations. Figure 7.7 shows the results of the crisp and fuzzy classifications for the year 1990. Part (a) is a map of elevations in the study area, part (b) shows the result of the crisp classification of elevations, and parts (d)–(e) are the vague objects created from the membership functions of Figure 7.6. The three objects, vague shore, vague beach, and vague dune, form a vague partition of the zone (in each year), assured from the choice of their membership functions.

The decision for executing beach nourishment is based on detection of structural erosion in the beach area in a period of 10 years, reference to the basal coastline, and the volume of sand that is to be deposited. The calculation for this last condition is performed on compartments that are division of the beach by the profile lines shown in Figure 7.7(b). In the next paragraphs we show how our operators can be used for monitoring beach erosion and perform analysis on the data, without going to the detailed level of compartments.

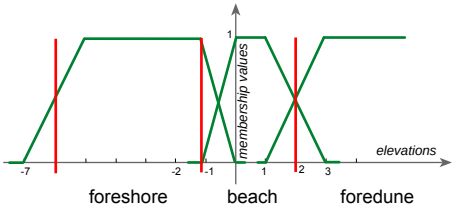

Figure 7.6: Membership functions for shore, beach, and dune, drawn in green. The red bars show the boundaries for a crisp classification of elevations.

In the Ameland data (see Figure 7.5) we notice a lake that emerges at the north-eastern side of the area, as a consequence of sedimentation. Following tidal movements, it starts existing around the year 1995, and may disappear in the future, becoming either beach or open sea. The *Deftime* operator will provide us with the time period of its existence.

Figure 7.7: (a) DEM of Ameland for the year 1990, and (b)–(e) objects created from crisp and fuzzy classification; (b) crisp classification of elevations, and profile lines along the beach; (c) vague shore, (d) vague beach, (e) vague dune. Saturated colour shows high membership for vague objects.

A beach plain has to obey specific elevation requirements that are characterized by a membership function. Its position and shape at a certain moment of time is given by the *AtInstant* operator. The operator provides, indeed, the membership function for that particular object, as well as the time.

An object might be selected in the screen, using an interface like the the one of Figure 6.15. Tracking the object to prior or later moments is done by the object identifier. Applying the *Initial* operator to a selected object gives the state of the object at the first moment of its existence in the database. The *Initial* operator provides also the time of the first occurrence in the database. The *Final* operator similarly provides the characteristics of an object at its final occurrence in the database.

The *Present* operator provides a 'yes' or 'no' answer about the presence of an object at a given moment in time. It gives, for example, a 'no' when the lake on the north-eastern part of the area does not exist, if e.g., asked for its existence in 1990.

The *RIntersection* operator describes the co-existence between two objects. We know that the Ameland beach can be vegetated by small, salt resisting plants,

thus creating patches of vegetation that grow after some period of the existence of a beach object only. The coexistence of the beachplain with vegetation patches is given by the *RIntersection* operator. It is an object 'vegetated beachplain' taken as the intersection of the beach object with a vegetated object extracted from an image of NDVI[2] values.

The *Disjoint* operator provides the 'disjointness' degree for any moment of the existence of two objects. Shore and dune are certainly disjoint at all times, whereas the two combinations shore and beach and beach and dune both have a 'disjointness' degree smaller than 1 (Figure 7.6). The lowest value that may occur equals 0.5.

The *Distance* operator gives a function that is changing in time. The distance between the vague dune and the vague shore may change in time. The *Distance* operator then provides for every year the distance between the α-cuts of the two objects.

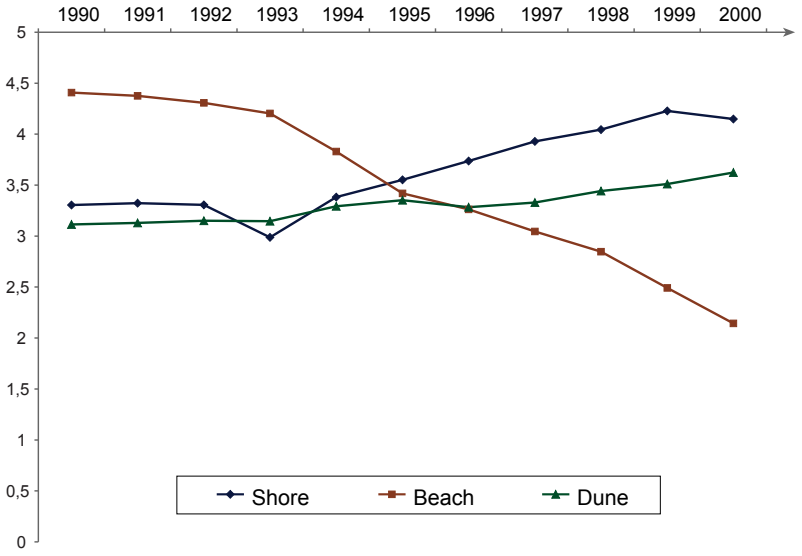

Figure 7.8: Graphs of average areas of vague objects in Ameland for years 1990–2000.

The *AvArea* operator provides for each of the years the average area of a vague object, derived from the averaging of its α areas over all the α values. Figure 7.8 shows graphs of the average areas of vague shore, vague beach, and vague dune for years 1990–2000. This graph is used to check one of the conditions for the detection of beach erosion.

The zone is said to have structural erosion if there is a negative trend-line for the beach areas through ten consecutive years. From the graph of Figure 7.8 it

[2]The NDVI index is a vegetation index calculated from different bands of satellite imagery.

can be seen that beach areas have a negative trend-line. The areas are compared with the beach area of the reference year, Area[beach90]. This value determines the lower limit for a beach area. Figure 7.9 shows vague beach objects in years 1990–1995 under a transparent layer of crisp objects in the reference year 1990. It can be seen how the beach area is shrinking towards a south-easterly direction.

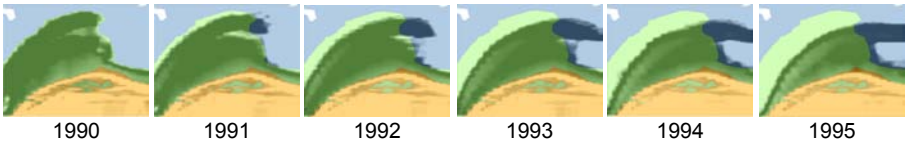

Figure 7.9: The vague beach in the Ameland island for years 1990–1995. The crisp classification of the year 1990, Figure fig:fuzzy-class(b), is overlayed to all beach images.

The last condition for beach nourishment is the volume of sand to be deposited. This cannot be expressed directly by our operators. It can, however, be calculated after some modifications. Appendix A describes fuzzy Markov chains as a possible model for the change process of vague objects. It uses sand volumes to build the model, and it can as result help to predict the possible loss on sand volumes.

7.4 Summary

The system of vague types and operators provided in the previous chapters was sketchily extended in this chapter with temporal types and operators. A temporal type is a changing object defined as a function from the time domain to any of the object types proposed in Chapter 3. A set of temporal operators was defined to answer simple questions related to time, e.g., the period of existence of an object, if the object exists at a given moment, its state at the first or last moment of existence. All spatial operators defined in the previous chapters were extended to temporal operators. For example, the relation disjoint between changing objects compares the objects at any moment of their existence towards the static disjoint relation.

Functionality of existing spatiotemporal databases can be used to store information about dynamic vague objects at discrete moments of time. We proposed linear function to interpolate between two consecutive stored states of a dynamic vague object. That provides the complete (approximate) information of a dynamic object. At the end, we considered two dynamic vague applications, and performed their analysis using our operators.

Chapter 8

Conclusions

This thesis provides a system of types and operators that can handle vague spatial information. Research concentrates on objects in $\mathbb{R}^2$. We distinguish between vague point, line, and region objects. The proposed types are formal representations of spatial objects that exhibit thematic vagueness. Regions often result from a classification of space. The vagueness of concepts defining the classes makes vague the extent of regions. This means that locational vagueness results from thematic vagueness. Therefore, the proposed region types are appropriate for representing locational vagueness as well. The operators allow reasoning with the proposed vague object types. These operators correspond to the analysis procedures that are commonly performed with spatial data.

The thesis can be divided into three parts: formal definitions of vague types and operators for static spatial objects, presented in Chapters 3, 4, and 5; implementation of these types and operators in a GIS software package, presented in Chapter 6; extension of types and operators with the time domain to allow representation of and reasoning with dynamic vague objects, presented in Chapter 7. The coming sections summarize the results of each part.

8.1 Definitions of static vague types and operators

In the first part of the thesis we consider static objects and operators to handle those. The objects are separated into simple and general types. A *simple type* represents an identifiable object that is not divisible into individual components. Attribute values can be attached to these objects, enriching the geometry by giving importance for use in applications. We distinguish between vague points, vague lines, and vague regions that are objects of type *VPoint*, *VLine*, and *VRegion*, respectively. A *vague point* is a site with a known location, but with an uncertain belonging to a phenomenon of interest. A *vague line* is a linear feature with known position, but with an uncertain extent, i.e., any point belongs to the line to some degree. A *vague region* is a broad boundary region, such that loca-

tions in the broad boundary typically have different degree of belonging to the region. The *general type* objects assure closure under operators. A general object is a class of simple objects that have the same value on a particular attribute. We distinguish between vague multipoints, vague multilines, and vague multiregions that are objects of type *VMPoint*, *VMLine*, and *VMRegion*, respectively. A *vague multipoint* is a finite collection of disjoint vague points. A *vague multiline* is a finite collection of vague lines that intersect only at their end nodes, and have the same membership value at the common end nodes. A *vague multiregion* is a finite collection of disjoint vague regions. Two other types were needed to assure closure under operators: *VLDim*, which values are collection of vague points and vague lines, and *VExt*, which values are collection of vague lines and vague regions. A *vague partition* allows for a soft classification of space. It is a collection of vague multiregions that might intersect each other only at their uncertain parts. The vague types proposed are such that they include crisp objects as special cases. Figure 8.1 shows the hierarchy of these spatial types, together with a few other data types that are the return types of operators.

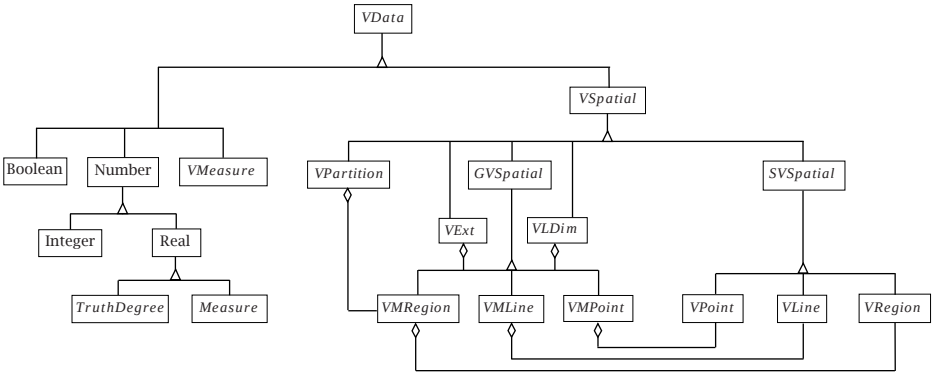

Figure 8.1: Hierarchy of types for static vague spatial data. *VSpatial* is the super-type of all spatial types. The left branch of the hierarchy consists of numerical types that are returned from spatial operators.

Provided operators are a standard set of operators that allow spatial analysis and reasoning. They are divided into three groups: operators returning spatial types, spatial relations, and metric operators.

Operators returning spatial types are regularized fuzzy set operators, boundary operators, and operators on partitions. The regularized set operators are *union*, *intersection*, and *difference* between objects of the same general type. They behave in the same way with their crisp correspondents when applied to crisp objects. Other operators can be built from those, such as the symmetric difference. Two boundary operators are proposed for objects of a general type: *boundary* and *frontier*. The boundary operator extracts the transition zone of a vague object. The frontier extracts the locations where abrupt changes occur on the membership values of vague objects. Both the boundary and the frontier

give the crisp boundary operator when applied to crisp objects. The operators on vague partitions are *overlay* and *fusion*. The overlay operator combines two vague partitions to form a new, more refined, vague partition. The fusion operator allows for generalization of a vague partition.

Spatial relations are diadic operators between general vague objects. The proposed relations, *disjoint*, *touches*, *crosses*, *overlaps*, *within*, and *equal*, follow the intuition behind the SQL/MM spatial relations. They extend the true/false set of truth values of the SQL/MM relations to the $[0,1]$ interval. Hence, the truth of a relation is a matter of degree (a value of type *TruthDegree*). The relations have the property that only one can be certain at a time. Again, they give the corresponding crisp relations when applied to crisp objects. Relations *disjoint*, *touches*, *crosses*, and *overlaps* between vague objects are defined such that the relation is certain if the corresponding crisp relation is true for the object cores. It is certainly false if the corresponding crisp relation is false for the support sets of the objects. The total certainty of the other two relations, *within* and *equal*, is modelled by the subset and equality relation for fuzzy sets, respectively. The within relation between two vague objects is certainly false if the corresponding crisp relation between the core of the first object and the support set of the second is false. Similarly, the equal relation between two vague objects is certainly false if the corresponding crisp relation between the core of one object and the support set of the other is false

Metric operators provided in this thesis are *distance*, *length*, *area*, *diameter*, *perimeter*, *centroid*, and *vagueness degree*, all defined for vague objects of a general type. We provided two groups of corresponding operators for distance, length, area, diameter, and perimeter measures: *alpha operators* and *average operators*. An alpha operator takes as argument one or two vague objects, and returns a function $f : (0,1] \to \mathbb{R}^+$ (a value of type *VMeasure*). For every $\alpha \in (0,1]$ the function yields the value of the analogous crisp operator applied to the α-cuts of the vague objects. An average operator is calculated as the integral over $[0,1]$ of the return function f of the corresponding alpha operator, hence producing an average over the returned values by the function f of the alpha operator. We provide equivalent formulas for the average length and the average area operators. This allows one to perform their calculation independently from the corresponding alpha operator. An average operator gives the analogous crisp operator when applied to a crisp object. The centroid operator calculates the centre of mass for a vague object. The vagueness operator provides a measure for the vagueness degree of an object. As a special case, the degree of vagueness of a crisp object is equal to 0. It reaches the maximal value if a vague object consists only of an uncertain part.

8.2 Implementation of static types and operators

We implemented the proposed types and the set operators in GRASS, using its functionality for vector data format. GRASS spatial features and data structures have been employed for storing information about vague objects. *Points*, *lines*, and *triangulations* have been used to store vague points, vague lines, and vague regions, respectively. An identifier is attached to an object of a simple type. Classes of simple objects are stored in separate vague *layers*. A vague partition is stored as a collection of vague region layers that form a *theme*. Classes of a theme were bound together via relations saved on database tables.

Several *modules* were developed to handle vague objects. A layer of vague regions is created from input data points via module `v.vague.triangle`. Module `v.vague.what` visualizes vague layers in the screen, and allows to retrieve and display information about their objects. Module `v.vague.combine` binds together layers of a theme. Other implemented modules perform union, intersection, and difference between layers of vague points, vague lines, and vague regions. These are basic operators, on which other spatial operators can be built.

8.3 Types and operators for dynamic vague information

The third part of the thesis addresses dynamic vague information. We distinguish two approaches. We first consider spatial vagueness, where interest focuses on changes of vague spatial phenomena over time. Second, we generalize towards vagueness in space *and* time. This second approach becomes relevant when objects are defined by properties in the space–time domain that are described by vague terms.

This part mainly considers the first approach. Each type and operator defined for static objects has been extended with time. A changing vague object type is defined as a function from the time domain to the set of static objects of a particular type. A time-dependent operator is constructed from a static operator by replacing its static argument(s) with changing object(s) of the corresponding type. The time-dependent operator returns for each moment of existence of its argument(s) the value of the static operator on the states of the changing object(s).

As concerns implementation, functionality offered by spatiotemporal databases is adequate to store discrete states of a changing vague object. The state of an object between two consecutive states can then be calculated from functions describing the change.

Finally, we considered two dynamic spatial applications to show how their monitoring and analysis can be performed with our system of types and operators.

Vague region and vague partition types were employed to describe objects of both applications. The detection of change was performed by overlaying the vague partitions, and estimation of change was calculated by the average area operator. The other temporal operators were used for monitoring changes, e.g., time of appearance, or the whole existence of an object. The analysis performed with such operators can help understanding the process of change, and fine tune parameters of dynamic models describing vague spatial phenomena.

Appendix A

Fuzzy Markov chains to model beach erosion

Markov chains are used to model the dynamics of processes with short memory: the future state of the system depends only on the current state. However, they assume that the states the system is passing through are crisp. In Section A.1 we introduce fuzzy Markov chains to model systems with fuzzy states. The transition law between fuzzy states is given by fuzzy conditional probability. Section A.2 applies the fuzzy Markov chains to the beach erosion application.

A.1 Fuzzy Markov chains

Markov processes are processes where the future development is completely determined by the present state, and is independent of the way in which the present state has developed [42]. This is called Markovian property. We call a process a fuzzy Markov process[1] if it has fuzzy states and satisfies the Markovian property. A fuzzy Markov chain is modelled by the set of possible states and the transition law of passing from one state to another, in the same way as a (classical) Markov chain. We use fuzzy conditional probabilities to build the transition law.

Let us define what is a fuzzy state and a fuzzy conditional probability. As we are working with real spaces, we provide definitions for the $\mathbb{R}^n$ space. Let P be a probability measure in $\mathbb{R}^n$. A fuzzy event in $\mathbb{R}^n$ is a fuzzy set μ in $\mathbb{R}^n$, which is Borel measurable [114]. We identify a *fuzzy state* with a fuzzy event. The probability of a fuzzy event μ is defined [114] by the integral

$$P(\mu) = \int_{\mathbb{R}^n} \mu(q) dP, \tag{A.1}$$

[1] The terms fuzzy Markov process and fuzzy Markov chain are used interchangeably.

and is called *fuzzy probability*. It gives the expectation of the membership function of the fuzzy event. The conditional probability of two fuzzy events μ and ν is defined from their algebraic product [113]: $\mu \cdot \nu = \{(q, \mu(q) \cdot \nu(q)) \mid q \in \mathbb{R}^n\}$. It is defined as [114]

$$P(\mu \mid \nu) = \frac{P(\mu \cdot \nu)}{P(\nu)} \tag{A.2}$$

provided that $P(\nu) > 0$. The conditional probability on fuzzy events is called *fuzzy conditional probability*.

A (discrete-time) fuzzy Markov process is a set of random variables, $\{X_t\}_{t \in \mathbf{N}}$ (t showing time), from the same probability space of fuzzy events, satisfying the criteria

$$P(X_t = x_t | X_{t-1} = x_{t-1}, \ldots, X_1 = x_1) = P(X_t = x_t | X_{t-1} = x_{t-1}).$$

Transition law from fuzzy state i to j is given by the fuzzy conditional probability of having j in the coming step knowing that the current state is i. Assuming stationarity, this value is independent of the time this transition happens. This is called transition probability, and it is given by the formula

$$p(i, j) = P(X_{k+1} = j | X_k = i) \text{ for all } k \geq 0, \tag{A.3}$$

for every two fuzzy states i and j.

Transition probabilities of a process with finite number of states n, are put in a matrix notation, such that $p(i, j)$ is the element on the ith row and jth column of the matrix, called the transition matrix:

$$\mathbf{T} = \begin{pmatrix} p(1,1) & p(1,2) & \cdots & p(1,n) \\ p(2,1) & p(2,2) & \cdots & p(2,n) \\ & \cdots & & \\ p(n,1) & p(n,2) & \cdots & p(n,n) \end{pmatrix}$$

A fuzzy probability measure defined on fuzzy states (of the process) and the transition matrix completely describe a fuzzy Markov process.

A.2 Modelling beach erosion

The changes happening along the shore are described as transformation from and to foreshore, beach and fore dune. We consider these three classes: foreshore, beach and fore dune, to be the states of a fuzzy Markov chain. We calculate conditional probabilities in two consecutive years, then use these values to calculate transition probabilities. Let C_i^t be class $i \in \{\text{foreshore}, \text{beach}, \text{foredune}\}$ in year t. The conditional probability that we will have class j on year $t+1$, knowing that we had class i on year t, is calculated as ratio of areas:

$$p^{t,t+1}(i, j) = \frac{\mathit{AvArea}(C_i^t \cdot C_j^{t+1})}{\mathit{AvArea}(C_i^t)}. \tag{A.4}$$

We calculate the transition probability as the average of the conditional probabilities in consecutive years for the period 1990–2000:

$$p(i,j) = \frac{p^{90,91}(i,j) + p^{91,92}(i,j) + \cdots + p^{99,00}(i,j)}{10}. \tag{A.5}$$

The transition matrix **T** is a 3×3 matrix, which elements are the $p(i,j)$s. Initial probabilities can be defined from the ratio of areas of the 1990 classes

$$P(C_i) = \frac{AvArea(C_i^{90})}{\sum_{j=1}^{3} AvArea(C_j^{90})}. \tag{A.6}$$

Initial probabilities and the transition matrix **T**, model the process of sand movement as a fuzzy Markov chain.

The conditional probability $p^{t,t+1}(i,j)$ is

$$p^{t,t+1}(i,j) = \frac{\iint C_i^t(x,y) \cdot C_j^{t+1}(x,y)\, dx\, dy}{\iint C_i^t(x,y)\, dx\, dy}. \tag{A.7}$$

Instead of using areas to calculate probabilities, we might use volumes. This is immediately related to volumes of sand needed for beach nourishment. To perform calculations in raster data format, we apply the formula

$$p^{t,t+1}(i,j) = \frac{\sum_{pixel} \mathrm{DEM}^t(pixel) \cdot C_i^t(pixel) \cdot C_j^{t+1}(pixel)}{\sum_{pixel} \mathrm{DEM}^t(pixel) \cdot C_i^t(pixel)}. \tag{A.8}$$

Table A.1 shows the calculated values for the conditional probabilities of changes between shore, beach, and dune, for any two consecutive years in the period 1990–2000. The probabilities are calculated using the formula A.8.

Years	p(1,1)	p(1,2)	p(1,3)	p(2,1)	p(2,2)	p(2,3)	p(3,1)	$p(3,3)$	$p(3,3)$
90–91	0.83	0.15	0.0	0.07	0.81	0.12	0.0	0.11	0.89
91–92	0.81	0.16	0.0	0.09	0.80	0.11	0.0	0.10	0.90
92–93	0.79	0.14	0.0	0.09	0.80	0.11	0.0	0.11	0.89
93–94	0.89	0.08	0.0	0.08	0.78	0.14	0.0	0.10	0.90
94–95	0.88	0.05	0.0	0.11	0.75	0.15	0.0	0.11	0.89
95–96	0.94	0.02	0.0	0.07	0.78	0.15	0.0	0.13	0.87
96–97	0.93	0.03	0.0	0.06	0.76	0.18	0.0	0.12	0.88
97–98	0.90	0.05	0.0	0.05	0.76	0.19	0.0	0.10	0.90
98–99	0.88	0.05	0.0	0.12	0.70	0.18	0.0	0.09	0.91
99–00	0.87	0.04	0.0	0.11	0.70	0.19	0.0	0.07	0.93

Table A.1: Conditional probabilities for changing between shore, beach, and dune, for any two consecutive years during the period 1990–2000.

The transition matrix with probabilities calculated via the formula A.5 is

$$\mathbf{T} = \begin{pmatrix} 0.87 & 0.08 & 0.0 \\ 0.08 & 0.77 & 0.15 \\ 0.0 & 0.10 & 0.90 \end{pmatrix}$$

These transition probabilities can be used to predict the volume of sand for the next year.

Bibliography

[1] ISO/IEC 13249-3:1999(E). Information technology – database languages – sql multimedia and application packages – part 3: Spatial. Working Draft Text 1.32.04.02.03.00, International Organization for Standardization, October 2000.

[2] Pavel Sergeevich Alexandrov. *Combinatorial Topology*, volume 1,2, and 3. Dover Publications, June 1998.

[3] Frank Jr. Ayres. *Theory and Problems of Differential and Integral Calculus*. Schaum Publishing Co., New York, 2nd edition, 1964.

[4] Hans Bandemer and Siegfried Gottwald. *Fuzzy Sets, Fuzzy Logic, Fuzzy Methods: with Applications*. John Wiley & Sons, 1995.

[5] Robert G. Bartle. *The Elements of Real Analysis*. Number (Library of Congress) 64–20061. John Wiley & Sons, Inc., 1964.

[6] Pavel Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, Inc., San Jose, 2002.

[7] Thomas Bittner and John G. Stell. Rough sets in approximate spatial reasoning. In *Second International Conference on Rough Sets and Current Trends in Computing*, volume 2005 of *Lecture Notes in Computer Science (LNCS)*, pages 445–453. Springer-Verlag, 2000.

[8] Isabelle Bloch. On fuzzy distances and their use in image processing under imprecision. *Pattern Recognition*, 32:1873–1895, 1999.

[9] I. Bogàrdi, A. Bárdossy, and L. Duckstein. *Optimizing the Resources for Water Management*, chapter Risk Management for Groundwater Contamination: Fuzzy Set Approach, pages 442–448. ASCE, New York, 1990.

[10] A. Bogomolny. On the perimeter and area of fuzzy sets. *Fuzzy Sets and Systems*, 23:257–269, 1987.

[11] D. G. Brown. Mapping historical forest types in Baraga County Michigan, USA as fuzzy sets. *Plant Ecology*, 134(1):97–111, 1998.

[12] James J. Buckley and Esfandiar Eslami. *An Introduction to Fuzzy Logic and Fuzzy Sets*. Advances in Soft Computing. Physica-Verlag, 2002.

[13] Peter A. Burrough and Andrew U. Frank. *Geographic objects with indeterminate boundaries*. Number 2 in GISDATA. Taylor & Francis, London, 1996.

[14] Peter A. Burrough, Pauline F. M. van Gaans, and R. A. MacMillan. High-resolution landform classification using fuzzy k-means. *Fuzzy Sets and Systems*, 113(1):37–52, July 2000.

[15] C. L. Chang. Fuzzy topological spaces. *Journal of Mathematical Analysis and Applications*, 24:182–190, 1968.

[16] Bidyut Baran Chaudhuri and Azriel Rosenfeld. On a metric distance between fuzzy set. *Pattern Recognition Letters*, 17:1157–1160, 1996.

[17] Bidyut Baran Chaudhuri and Azriel Rosenfeld. A modified hausdorff distance between fuzzy set. *Information Sciences*, 118:159–171, 1999.

[18] Eliseo Clementini and Paolino di Felice. *Geographic objects with indeterminate boundaries*, chapter An Algebraic Model for Spatial Objects with Indeterminate Boundaries, pages 171–187. In *GISDATA* [13], 1996.

[19] Eliseo Clementini and Paolino di Felice. A spatial model for complex objects with a broad boundary supporting queries on uncertain data. *Data & Knowledge Engineering*, 37(3):285–305, June 2001.

[20] Anthony G. Cohn, Brandon Bennett, John Gooday, and Nicholas Mark Gotts. Qualitative Spatial Representation and Reasoning with the Region Connection Calculus. *GeoInformatica*, 1(3):275–316, 1997.

[21] Anthony G. Cohn and Nicholas Mark Gotts. *Geographic objects with indeterminate boundaries*, chapter The 'egg-yolk' representation of regions with indeterminate boundaries, pages 171–187. In *GISDATA* [13], 1996.

[22] Anthony G. Cohn and Nicholas Mark Gotts. Representing Spatial Vagueness: A Mereological Approach. In Luigia Carlucci Aiello, Jon Doyle, and Stuart Shapiro, editors, *Principles of Knowledge Representation and Reasoning (KR'96)*, pages 230–241. Morgan Kaufmann, 1996.

[23] OpenGIS Consortium. Opengis simple features specification for sql. Revision 1.1, Open GIS Consortium, Inc., 1999.

[24] E. Cuchillo-Ibáñez and Juan Tarrés. On the boundary of fuzzy sets. *Fuzzy Sets and Systems*, 89:113–119, 1997.

[25] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry-Algorithms and Applications*. Springer, Berlin, 2nd edition, 1997.

[26] Rolf A. de By. *Principles of Geographical Information Systems*. ITC Educational Textbook series; 1. International Institute for Geo-Information Science and Earth Observation (ITC), Hengelosestraat 99, P.O. Box 6, 7500 AA Enschede, The Netherlands, third edition, 2005.

[27] J.J. de Gruijter, D.J.J. Walvoort, and P.F.M. van Gaans. Continuous soil maps — a fuzzy set approach to bridge the gap between aggregation levels of process and distribution models. *Geoderma*, 77:169–195, 1997.

[28] Arta Dilo. Mapping an extended ER model to a spatial relational model. Master's thesis, International Institute for Geo-Information Science and Earth Observation (ITC), June 2000.

[29] Didier Dubois and Henri Prade. *Fuzzy Sets and Systems: Theory and Applications*. Academic Press, New York, 1980. pp. 38–40.

[30] Didier Dubois and Henri Prade. *Fundamentals of Fuzzy Sets*, volume 1 of *Handbook of Fuzzy Sets*, chapter Fuzzy interval analysis, pages 483–581. Kluwer Academic Publisher, 2000.

[31] Michael Dummett. Wang's paradox. *Synthese*, 30:301–324, 1975.

[32] H Edelsbrunner, D. G. Kirkpatrick, and Seidel R. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, IT-29(4):551–559, July 1983.

[33] Gerald A. Edgar. *Measure, topology, and fractal geometry*. Undergraduate texts in mathematics. Springer-Verlag, Berlin, 1990.

[34] Charles Henry Jr. Edwards and David E. Penney. *Calculus and Analytic Geometry*. Prentice Hall, Inc., 1982.

[35] Max J. Egenhofer. A model for detailed binary topological relationships. *Geomatica*, 47(3&4):261–273, 1993.

[36] Max J. Egenhofer, Eliseo Clementini, and Paolino Di Felice. Topological relations between regions with holes. *International Journal of Geographical Information Systems*, 8(2):129–142, 1994.

[37] Max J. Egenhofer and Robert D. Franzosa. Point-set topological spatial relations. *International Journal of Geographical Information Systems*, 5(2):161–174, 1991.

[38] Max J. Egenhofer and Robert D. Franzosa. On the equivalence of topological relations. *International Journal of Geographical Information Systems*, 8(6):133–152, 1994.

[39] Max J. Egenhofer and John Herring. Categorizing binary topological relationships between regions, lines, and points in geographic databases. Technical report, University of Maine, Department of Surveying Engineering, 1991.

[40] Martin Erwig and Ralf Hartmut Güting. Explicit graphs in a functional model for spatial databases. *IEEE Transactions on Knowledge and Data Engineering*, 6(5):787–804, October 1994.

[41] Martin Erwig and Markus Schneider. Vague regions. In *5th International Symposium on Advances in Spatial Databases (SSD'97)*, volume LNCS 1262, pages 298–320, 1997.

[42] William Feller. *An Introduction to Probability Theory and Its Applications*. Wiley series in probability and mathematical statistics. John Wiley & Sons Ltd., England, third edition, 1968.

[43] Kit Fine. Vagueness, truth and logic. *Synthese*, 30:265–300, 1975.

[44] Peter Fisher. Sorites paradox and vague geographies. *Fuzzy Sets and Systems*, 113(1):7–18, July 2000.

[45] Nicholas Mark Gotts. An axiomatic approach to topology for spatial information systems. Research report series 96.25, University of Leeds. School of Computer Studies, August 1996.

[46] Ralf Hartmut Güting. An introduction to spatial database systems. *VLDB Journal*, 3(4):357–399, 1994.

[47] Ralf Hartmut Güting, Michael H. Böhlen, Martin Erwig, Christian S. Jensen, Nikos Lorentzos, Enrico Nardelli, Markus Schneider, and Jose R.R. Viqueira. *Spatio-Temporal Databses: The CHOROCHRONOS Approach*, chapter Spatio-temporal Models and Languages: An Approach Based on Data Types, pages 117–176. Lecture Notes in Computer Science. Springer, September 2003.

[48] Ralf Hartmut Güting, Michael H. Böhlen, Martin Erwig, Christian S. Jensen, Nikos Lorentzos, Markus Schneider, and Michalis Vazirgiannis. A foundation for representing and querying moving objects. *ACM Transactions on Databases Systems*, 25(1):1–42, March 2000.

[49] Ralf Hartmut Güting, Thomas de Ridder, and Markus Schneider. Implementation of the rose algebra: Efficient algorithms for realm-based spatial data types. In *SSD*, pages 216–239, 1995.

[50] Ralf Hartmut Güting and Markus Schneider. Realm-based spatial data types: The ROSE algebra. *VLDB Journal*, 4(2):243–286, 1995.

[51] Ralf Hartmut Güting and Markus Schneider. *Moving Objects Databases*. Data Management Systems. Morgan Kaufmann, August 2005.

[52] Petr Hájek. *Metamathematics of Fuzzy Logic*, volume 4 of *Trends in Logic*. Kluwer Academic Publishers, 1998.

[53] Santosh Hedge. Modelling Land Cover Change using Fuzzy Based Geo-spatial Approach. Master's thesis, International Institute for Geo-Information Science and Earth Observation, December 2003.

[54] H.J.W.M. Hendricks Franssen, A.C. van Eijnsbergen, and Alfred Stein. Use of spatial prediction techniques and fuzzy classification for mapping soil pollutants. *Geoderma*, 77:243–262, 1997.

[55] Tomislav Hengl, Dennis J. J. Walvoort, Allan Brown, and David G. Rossiter. A double continuous approach to visualization and analysis of categorical maps. *International Journal of Geographical Information Science*, 18(2):183–202, March 2004.

[56] Edwin Hewitt and Karl Stromber. *Real and Abstract Analysis*. Springer-Verlag, 1969.

[57] Dominic Hyde. *The Stanford Encyclopedia of Philosophy*, chapter Sorites Paradox. The Metaphysics Research Lab CSLI, fall 2002 edition, 2002.

[58] Jürgen Jost. *Postmodern analysis*. Universitext. Springer, Berlin, 1998.

[59] John L. Kelley. *General topology*. Number (Library of Congress) 55–6495 in The University series in Higher Mathematics. D. Van Nostrand Company, Inc., March 1955.

[60] John L. Kelley. *General topology*. Number (Library of Congress) 75–14364 in Graduate texts in mathematics. Springer, New York, 1975.

[61] George J. Klir and Bo Yuan. *Fuzzy Sets and Fuzzy logic: Theory and Application*. Prentice-Hall, New Jersey, 1995.

[62] Pawalai Kraipeerapun. Implementation of vague spatial objects. Master's thesis, International Institute for Geo-information Science and Earth Observation (ITC), March 2004.

[63] Arko Lucieer, Peter Fisher, and Alfred Stein. *GeoDynamics*, chapter Texture-based Segmentation of Remotely Sensed Imagery to Identify Fuzzy Coastal Objects. CRC Press LLC, 2004.

[64] Arko Lucieer, Alfred Stein, and Peter Fisher. Texture-based segmentation of high-resolution remotely sensed imagery for identification of fuzzy objects. In *Proceedings of GeoComputation*, Southampton, UK, 2003.

[65] Chris Meyer, Abraham Kandel, and Dewey Rundus. The triad of fuzzy theory. *ACM SIGAPP Applied Computing Review*, 1(2):12–15, September 1993.

[66] I.O.A. Odeh, A.B. McBratney, and Chittleborough D.J. Soil pattern recognition with fuzzy cŰ-means: application to classification and soilŰ-landform interrelationships. *Soil Science Society of America Journal*, 65:505–Ű516, 1992.

[67] Pu Pao-Ming and Liu Ying-Ming. Fuzzy topology. I. Neighbourhood structure of a fuzzy point and Moore-Smith convergence. *Journal of Mathematical Analysis and Applications*, 76:571–599, 1980.

[68] Simon Parson and Anthony Hunter. A review of uncertainty handling formalisms. In *Applications of Uncertainty Formalisms*, pages 8–37, 1998.

[69] Simon Parsons. Current approaches to handling imperfect information in data and knowledge bases. *Knowledge and Data Engineering*, 8(3):353–372, 1996.

[70] Zdzislaw Pawlak. Rough sets: present state and further prospects. In *Third International Workshop on Rough Set and Soft Computing (RSSC '94)*, pages 72–76, November 1994.

[71] Zdzislaw Pawlak. Vagueness and uncertainty: a rough set perspective. *Computational Intelligence*, 11(2):227–232, 1995.

[72] David A. Randell and Anthony G. Cohn. Modelling topological and metrical properties of physical processes. In R. J. Brachman, H. J. Levesque, and R. Reiter, editors, *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning (KR'89)*, pages 357–368. Morgan Kaufmann, 1989.

[73] David A. Randell, Zhan Cui, and Anthony G. Cohn. A spatial logic based on regions and connection. In Bernhard Nebel, Charles Rich, and William Swartout, editors, *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR'92)*, pages 165–176. Morgan Kaufmann, San Mateo, California, 1992.

[74] Vincent B. Robinson. A perspective on the fundamentals of fuzzy sets and their use in geographic information systems. *Transactions in GIS*, 7(1):3–30, 2003.

[75] P. Roelse. Water en zand in balans: Evaluatie zandsuppleties na 1990; een morfologische beschouwing. Technical Report RIKZ/2002.003, Rijksinstituut voor Kust en Zee/RIKZ, 2002.

[76] Azriel Rosenfeld. The diameter of a fuzzy set. *Fuzzy Sets and Systems*, 13:241–246, 1984.

[77] Azriel Rosenfeld. Distance between fuzzy sets. *Pattern Recognition Letters*, 3:229–233, 1985.

[78] Azriel Rosenfeld and Seymour Haber. The perimeter of a fuzzy set. *Pattern Recognition*, 18(2):125–130, 1985.

[79] Antony J. Roy and John G. Stell. Spatial relations between indeterminate regions. *International Journal of Approximate Reasoning*, 27(3):205–234, September 2001.

[80] Markus Schneider. Uncertainty management for spatial data in databases: Fuzzy spatial data types. In *SSD '99: Proceedings of the 6th International Symposium on Advances in Spatial Databases*, volume 1651 of *Lecture Notes in Computer Science*, pages 330–351. Springer-Verlag, 1999.

[81] Markus Schneider. Finite Resolution Crisp and Fuzzy Spatial Objects. In *9th Symposium on Spatial Data Handling (SDH)*, pages 3–17, 2000.

[82] Markus Schneider. Metric Operations on Fuzzy Spatial Objects in Databases. In *8th ACM Symposium on Geographic Information Systems (ACM GIS)*, pages 21–26, 2000.

[83] Markus Schneider. A design of topological predicates for complex crisp and fuzzy regions. In *ER '01: Proceedings of the 20th International Conference on Conceptual Modeling*, pages 103–116. Springer-Verlag, 2001.

[84] Markus Schneider. Fuzzy Topological Predicates, their Properties, and their Integration into Query Languages. In *9th ACM Symposium on Geographic Information Systems (ACM GIS)*, 2001.

[85] Markus Schneider. Design and implementation of finite resolution crisp and fuzzy spatial objects. *Data & Knowledge Engineering*, 44(1):81–108, January 2003.

[86] Jonathan Richard Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In *First Workshop on Applied Computational Geometry*, pages 124–133, Philadelphia, Pennsylvania, May 1996.

[87] Rodrigo S. Sicat, Emmanuel John M. Carranza, and Uday Bhaskar Nidumolu. Fuzzy modelling of farmers' knowledge for land suitability classification. *Agricultural Systems*, 83(1):49–75, January 2005.

[88] Philippe Smets. Varieties of ignorance and the need for well-founded theories. *Information Sciences*, pages 135–144, 1991.

[89] Philippe Smets. Imperfect information: Imprecision and uncertainty. *Uncertainty Management in Information Systems*, pages 225–254, 1996.

[90] Mike J. Smithson. *Ignorance and Uncertainty: Emerging Paradigms*. Cognitive Science. Springer Verlag, New York, 1989.

[91] Roy Sorensen. *Blindspots*. Oxford: Clarendon Press, 1988.

[92] Roy Sorensen. Vagueness. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. The Metaphysics Research Lab CSLI, fall 2003 edition, 2003.

[93] John G. Stell. Boolean connection algebras: A new approach to the region-connection calculus. *Artificial Intelligence*, 122(1–2):111–136, September 2000.

[94] John G. Stell. Part and complement: Fundamental concepts in spatial relations. *Annals of Artificial Intelligence and Mathematics*, 41(1):1–18, 2004.

[95] Knut Stolze. Sql/mm spatial - the standard to manage spatial data in a relational database system. In *BTW*, pages 247–264, 2003.

[96] Kauzo Tanaka. *An Introduction to Fuzzy Logic for Practical Applications*. Springer, 1997.

[97] Xinming Tang. *Spatial Object Modelling In Fuzzy Topological Spaces: with Application to Land Cover Change*. PhD thesis, International Institute for Geo-Information Science and Earth Observation (ITC), Hengelosestraat 99, 7500 AA Enschede, The Netherlands, January 2004.

[98] M. Teichmann and M. Capps. Surface reconstruction with anisotropic density-scaled alpha-shapes. In *In IEEE Visualization '98 Proceedings*, pages 67–72, San Francisco, CA, October 1998. ACM/SIGGRAPH Press.

[99] Michael Tye. *Philosophical Perspectives: Logic and Language*, chapter Sorites paradoxes and the semantics of vagueness. Atascadero, 1994.

[100] Daniël van de Vlag. *Modeling and visualizing dynamic landscape objects and their qualities*. PhD thesis, Wageningen University, Enschede, The Netherlands, 2006. ITC Dissertation No. 132.

[101] Daniel van de Vlag, Alfred Stein, and Bérengère Vasseur. Concepts and Representation of Beach Nourishment by Spatio-Temporal Ontology. In Andrew Frank and Eva Grum, editors, *Proceedings of the International Symposium on Spatial Data Quality (ISSDQ04)*, volume 2 of *GeoInfo*, pages 353–369, April 2004.

[102] Achille Varzi. *Encyclopedia of Cognitive Science*, chapter Vagueness. Macmillan and Nature Publishing Group, London, 2002.

[103] Achille Varzi. Mereology. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. The Metaphysics Research Lab CSLI, 2004.

[104] Gregory Vert, Molly Stock, and Ashley Morris. Extending erd modeling notation to fuzzy management of gis data files. *Data & Knowledge Engineering*, 40:163–179, 2002.

[105] Fangju Wang. A fuzzy grammar and possibility theory-based natural language user interface for spatial queries. *Fuzzy Sets and Systems*, 113:147–159, 2000.

[106] Richard H. Warren. Boundary of a fuzzy set. *Indiana University Mathematical Journal*, 26(2):191–197, 1977.

[107] A. Watt and F. Policarpo. *3D Games, Real-time Rendering and Software Technology*. Pearson Education Limited, Essex, England, 1st edition, 2001.

[108] Michael D. Weiss. Fixed Points, Separation, and Induced Topologies for Fuzzy Sets. *Journal of Mathematical Analysis and Applications*, 50:142–150, 1975.

[109] Stephen Willard. *General Topology*. Addison-Wesley, University of Alberta, April 1970.

[110] Timothy Williamson. *Vagueness*. The problems of philosophy. Routledge, 1994.

[111] Michael F. Worboys. *GIS, a Computer Perspective*. Taylor & Francis, 1st edition, 1995.

[112] Michael F. Worboys. *Data Quality in Geographic Information*, chapter Some Algebraic and Logical Foundations for Spatial Imprecision, pages 151–156. Hermes, 1998.

[113] Lofti A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[114] Lofti A. Zadeh. Probability measures of fuzzy events. *Journal of Mathematical Analysis and Applications*, 23:421–427, 1968.

[115] Lofti A. Zadeh. Fuzzy logic and approximate reasoning. *Synthese*, 30:407–428, 1975.

[116] Benjamin F. Zhan. Topological relations between fuzzy regions. In *Proceedings of the 1997 ACM Symposium on Applied Computing*, pages 192–196. ACM Press, 1997.

[117] Benjamin F. Zhan. Approximate analysis of topological relations between geographic regions with indeterminate boundaries. *Soft Computing*, 2(2):28–34, 1998.

Biography

Arta Dilo was born on 12 June 1968 in Vlorë, Albania. From 1986 to 1991 she studied mathematics at the University of Tirana, Albania. From 1992 until 1993 she was an assistant professor at the department of mathematics of the Faculty of Geology, Polytechnic University of Tirana. From 1993 until 1998 she was working as a programmer and/or database administrator at a few institutes in Albania, Institute of Informatics and Applied Mathematics, Geographic Studies Centre, and the Patent Office of Albania. From 1992 until 1995 she was also teaching probability and mathematical programming at the Faculty of Natural Sciences, University of Tirana.

From January 1999 until July 2000 she completed studies for a Master of Science in Geoinformatics at the International Institute for Geo-Information Science and Earth Observation (ITC), Enschede, the Netherlands. Her MSc thesis was entitled 'Mapping an extended ER model to spatial relational model'. From September 2000 until November 2001 she worked as a database administrator at the general Taxation Department of Albania, and later at the Ark IT software company in Tirana. In December 2001 she started as a PhD researcher at ITC. Her research focused on representation and reasoning with vagueness in spatial information. She is currently working as a postdoctoral research fellow at the Technical University of Delft in the field of spatial and spatiotemporal modelling.

Publications of the author

[1] Arta Dilo. Mapping an extended ER model to a spatial relational model. Master's thesis, International Institute for Geo-Information Science and Earth Observation (ITC), June 2000.

[2] Arta Dilo, Pieter Bos, Pawalai Kraipeerapun, and Rolf A. de By. *Flexible Databases supporting Imprecision and Uncertainty*, chapter Storage and manipulation of vague spatial objects using existing GIS functionality. Studies in Fuzziness and Soft Computing. Springer, 2006. In press.

[3] Arta Dilo, Rolf A. de By, and Alfred Stein. A spatial algebra for vague objects. International Journal of Geographical Information Science. In review.

[4] Arta Dilo, Rolf A. de By, and Alfred Stein. Definition and implementation of vague objects. In Andrew Frank and Eva Grum, editors, *Proceedings of the 3rd International Symposium on Spatial Data Quality ISSDQ'04*, GeoInfo, pages 139–145, Bruck an der Leitha, Austria, April 2004. Technical University of Vienna.

[5] Arta Dilo, Rolf A. de By, and Alfred Stein. A proposal for spatial relations between vague objects. In Lun Wu, Wenzhong Shi, Yu Fang, and Qingxi Tong, editors, *Proceedings of the International Symposium on Spatial Data Quality ISSDQ'05, Beijing, China*, pages 50–59, Peking University, Beijing, China, August 2005. The Hong Kong Polytechnic University.

[6] Arta Dilo, Pawalai Kraipeerapun, Wim Bakker, and Rolf A. de By. Storing and handling vague spatial objects. In *15th International Workshop on Database and Expert Systems Applications (DEXA 2004)*, pages 945–950, Zaragoza, Spain, September 2004. IEEE Computer Society.

[7] Tamara Luarasi and Arta Dilo. *Gjuha e Programimit C*. Shtepia Botuese Eurorilindja, Tiranë, 2000. Adapted translation from "The C Programming Language" by B. Kernighan and D. Ritchie.

[8] Alfred Stein, Arta Dilo, Arko Lucieer, and Daniel van de Vlag. Definition and identification of vague spatial objects and their use in decision ontologies. In Andrew Frank and Eva Grum, editors, *Proceedings of the 3rd International*

Symposium on Spatial Data Quality ISSDQ'04, Geo-Info, pages 99–115, Bruck an der Leitha, Austria, April 2004. Technical University of Vienna.

[9] Bérengère Vasseur, Daniel van de Vlag, Alfred Stein, Robert Jeansoulin, and Arta Dilo. Quality-aware ontologies for dynamic spatial objects with indeterminate boundaries. Applied Ontology. In review.

[10] Bérengère Vasseur, Daniel van de Vlag, Alfred Stein, Robert Jeansoulin, and Arta Dilo. Spatiotemporal ontology for defining the quality of an application. In Andrew Frank and Eva Grum, editors, *Proceedings of the 3rd International Symposium on Spatial Data Quality ISSDQ'04*, Geo-Info, pages 67–81, Bruck an der Leitha, Austria, April 2004. Technical University of Vienna.

ITC Dissertations

[1] **Akinyede, Joseph O.**, 1990, *Highway Cost Modelling and Route Selection Using a Geotechnical Information System*, Delft University of Technology.

[2] **Pan, Ping He**, 1990, *A Spatial Structure Theory in Machine Vision and Applications to Structural and Textural Analysis of Remotely Sensed Images*, University of Twente, 90-9003757-8.

[3] **Bocco Verdinelli, Gerardo H. R.**, 1990, *Gully Erosion Analysis Using Remote Sensing and Geographic Information Systems: A Case Study in Central Mexico*, Universiteit van Amsterdam.

[4] **Sharif, Massoud**, 1991, *Composite Sampling Optimization for DTM in the Context of GIS*, Wageningen Agricultural University.

[5] **Drummond, Jane E.**, 1991, *Determining and Processing Quality Parameters in Geographic Information Systems*, University of Newcastle.

[6] **Groten, Susanne M.E.**, 1991, *Satellite Monitoring of Agro-ecosystems in the Sahel*, Westfälische Wilhelms-Universität.

[7] **Sharifi, Ali**, 1991, *Development of an Appropriate Resource Information System to Support Agricultural Management at Farm Enterprise Level*, Wageningen Agricultural University, 90-6164-074-1.

[8] **van der Zee, Dick**, 1991, *Recreation Studied from Above: Air Photo Interpretation as Input into Land Evaluation for Recreation*, Wageningen Agricultural University, 90-6164-075-X.

[9] **Mannaerts, Chris**, 1991, *Assessment of the Transferability of Laboratory Rainfall-runoff and Rainfall—Soil Loss Relationships to Field and Catchment Scales: A Study in the Cape Verde Islands*, University of Ghent, 90-6164-085-7.

[10] **Wang, Ze Shen**, 1991, *An Expert System for Cartographic Symbol Design*, Utrecht University, 90-3930-333-9.

[11] **Zhou, Yunxuan**, 1991, *Application of Radon Transforms to the Processing of Airborne Geophysical Data*, Delft University of Technology, 90-6164-081-4.

[12] **de Zuviría, Martín**, 1992, *Mapping Agro-topoclimates by Integrating Topographic, Meteorological and Land Ecological Data in a Geographic Information System: A Case Study of the Lom Sak Area, North Central Thailand*, Universiteit van Amsterdam, 90-6164-077-6.

[13] **van Westen, Cees J.**, 1993, *Application of Geographic Information Systems to Landslide Hazard Zonation*, Delft University of Technology, 90-6164-078-4.

[14] **Shi, Wenzhong**, 1994, *Modelling Positional and Thematic Uncertainties in Integration of Remote Sensing and Geographic Information Systems*, Universität Osnabrück, 90-6164-099-7.

[15] **Javelosa, Ricarte S.**, 1994, *Active Quaternary Environments in the Philippine Mobile Belt*, Utrecht University, 90-6164-086-5.

[16] **Lo, King-Chang**, 1994, *High Quality Automatic DEM, Digital Elevation Model Generation from Multiple Imagery*, University of Twente, 90-9006-526-1.

[17] **Wokabi, Stanley M.**, 1994, *Quantified Land Evaluation for Maize Yield Gap Analysis at Three Sites on the Eastern Slope of Mt. Kenya*, University Ghent, 90-6164-102-0.

[18] **Rodríguez Parisca, Oscar S.**, 1995, *Land Use Conflicts and Planning Strategies in Urban Fringes: A Case Study of Western Caracas, Venezuela*, University of Ghent.

[19] **van der Meer, Freek D.**, 1995, *Imaging Spectrometry & the Ronda Peridotites*, Wageningen Agricultural University, 90-5485-385-9.

[20] **Kufoniyi, Olajide**, 1995, *Spatial Coincidence: Automated Database Updating and Data Consistency in Vector GIS*, Wageningen Agricultural University, 90-6164-105-5.

[21] **Zambezi, Paul**, 1995, *Geochemistry of the Nkombwa Hill Carbonatite Complex of Isoka District, North-east Zambia, with Special Emphasis on Economic Minerals*, Vrije Universiteit Amsterdam.

[22] **Woldai, Tsehaie**, 1995, *The Application of Remote Sensing to the Study of the Geology and Structure of the Carboniferous in the Calañas Area, Pyrite Belt, South-west Spain*, Open University, United Kingdom.

[23] **Verweij, Pita A.**, 1995, *Spatial and Temporal Modelling of Vegetation Patterns: Burning and Grazing in the Páramo of Los Nevados National Park, Colombia*, Universiteit van Amsterdam, 90-6164-109-8.

[24] **Pohl, Christine**, 1996, *Geometric Aspects of Multisensor Image Fusion for Topographic Map Updating in the Humid Tropics*, Universität Hannover, 90-6164-121-7.

[25] **Bin, Jiang**, 1996, *Fuzzy Overlay Analysis and Visualization in Geographic Information Systemes*, Utrecht University, 90-6266-128-9.

[26] **Metternicht, Graciela I.**, 1996, *Detecting and Monitoring Land Degradation Features and Processes in the Cochabamba Valleys, Bolivia. A Synergistic Approach*, University of Ghent, 90-6164-118-7.

[27] **Chu Thai Hoanh**, 1996, *Development of a Computerized Aid to Integrated Land Use Planning (CAILUP) at Regional Level in Irrigated Areas: A Case Study for the Quan Lo Phung Hiep region in the Mekong Delta, Vietnam*, Wageningen Agricultural University, 90-6164-120-9.

[28] **Roshannejad, Ali**, 1996, *The Management of Spatio-Temporal Data in a National Geographic Information System*, University of Twente, 90-9009-284-6.

[29] **Terlien, Mark T. J.**, 1996, *Modelling Spatial and Temporal Variations in Rainfall-triggered Landslides: The Integration of Hydrologic Models, Slope Stability Models and GIS for the Hazard Zonation of Rainfall-triggered Landslides with Examples from Manizales, Colombia*, Utrecht University, 90-6164-115-2.

[30] **Mahavir, J.**, 1996, *Modelling Settlement Patterns for Metropolitan Regions: Inputs from Remote Sensing*, Utrecht University, 90-6164-117-9.

[31] **Al-Amir, Sahar**, 1996, *Modern Spatial Planning Practice as Supported by the Multi-applicable Tools of Remote Sensing and GIS: The Syrian Case*, Utrecht University, 90-6164-116-0.

[32] **Pilouk, Morakot**, 1996, *Integrated Modelling for 3D GIS*, University of Twente, 90-6164-122-5.

[33] **Duan, Zengshan**, 1996, *Optimization Modelling of a River-Aquifer System with Technical Interventions: A Case Study for the Huangshui River and the Coastal Aquifer, Shandong, China*, Vrije Universiteit Amsterdam, 90-6164-123-3.

[34] **de Man, Erik W. H.**, 1996, *Surveys: Informatie als Norm: Een Verkenning van de Institutionalisering van Dorp-surveys in Thailand en op de Filippijnen*, University of Twente, 90-9009-775-9.

[35] **Vekerdy, Zoltan**, 1996, *GIS-based Hydrological Modelling of Alluvial Regions: Using the Example of the Kisaföld, Hungary*, Lorand Eotvos University of Sciences, 90-6164-119-5.

[36] **Gomes Pereira, Luisa M.**, 1996, *A Robust and Adaptive Matching Procedure for Automatic Modelling of Terrain Relief*, Delft University of Technology, 90-407-1385-5.

[37] **Fandiño Lozano, Martha T.**, 1996, *A Framework of Ecological Evaluation oriented at the Establishment and Management of Protected Areas: A Case Study of the Santuario de Iguaque, Colombia*, Universiteit van Amsterdam, 90-6164-129-2.

[38] **Toxopeus, Bert G.**, 1996, *ISM: An Interactive Spatial and Temporal Modelling System as a Tool in Ecosystem Management: With Two Case Studies: Cibodas Biosphere Reserve, West Java Indonesia: Amboseli Biosphere Reserve, Kajiado District, Central Southern Kenya*, Universiteit van Amsterdam, 90-6164-126-8.

[39] **Wang, Yiman**, 1997, *Satellite SAR Imagery for Topographic Mapping of Tidal Flat Areas in the Dutch Wadden Sea*, Universiteit van Amsterdam, 90-6164-131-4.

[40] **Saldana Lopez, Asunción**, 1997, *Complexity of Soils and Soilscape Patterns on the Southern Slopes of the Ayllon Range, Central Spain: a GIS Assisted Modelling Approach*, Universiteit van Amsterdam, 90-6164-133-0.

[41] **Ceccarelli, Tomaso**, 1997, *Towards a Planning Support System for Communal Areas in the Zambezi Valley, Zimbabwe: A Multi-criteria Evaluation Linking Farm Household Analysis, Land Evaluation and Geographic Information Systems*, Utrecht University, 90-6164-135-7.

[42] **Peng, Wanning**, 1997, *Automated Generalization in GIS*, Wageningen Agricultural University, 90-6164-134-9.

[43] **Mendoza Lawas, Cora**, 1997, *The Resource Users' Knowledge, the Neglected Input in Land Resource Management: The Case of the Kankanaey Farmers in Benguet, Philippines*, Utrecht University, 90-6164-137-3.

[44] **Bijker, Wietske**, 1997, *Radar for Rain Forest: A Monitoring System for Land Cover Change in the Colombian Amazon*, Wageningen Agricultural University, 90-6164-139-X.

[45] **Farshad, Abbas**, 1997, *Analysis of Integrated Soil and Water Management Practices within Different Agricultural Systems under Semi-arid Conditions of Iran and Evaluation of their Sustainability*, University of Ghent, 90-6164-142-X.

[46] **Orlic, Bogdan**, 1997, *Predicting Subsurface Conditions for Geotechnical Modelling*, Delft University of Technology, 90-6164-140-3.

[47] **Bishr, Yaser**, 1997, *Semantic Aspects of Interoperable GIS*, Wageningen Agricultural University, 90-6164-141-1.

[48] **Zhang, Xiangmin**, 1998, *Coal fires in Northwest China: Detection, Monitoring and Prediction Using Remote Sensing Data*, Delft University of Technology, 90-6164-144-6.

[49] **Gens, Rudiger**, 1998, *Quality Assessment of SAR Interferometric Data*, University of Hannover, 90-6164-155-1.

[50] **Turkstra, Jan**, 1998, *Urban Development and Geographical Information: Spatial and Temporal Patterns of Urban Development and Land Values Using Integrated Geo-data, Villaviciencio, Colombia*, Utrecht University, 90-6164-147-0.

[51] **Cassells, Craig James Steven**, 1998, *Thermal Modelling of Underground Coal Fires in Northern China*, University of Dundee.

[52] **Naseri, Mohammad Y.**, 1998, *Monitoring Soil Salinization, Iran*, Ghent University, 90-6164-195-0.

[53] **Gorte, Ben G. H.**, 1998, *Probabilistic Segmentation of Remotely Sensed Images*, Wageningen Agricultural University, 90-6164-157-8.

[54] **Ayenew, Tenalem**, 1998, *The Hydrological System of the Lake District Basin, Central Main Ethiopian Rift*, Universiteit van Amsterdam, 90-6164-158-6.

[55] **Wang, Donggen**, 1998, *Conjoint Approaches to Developing Activity-Based Models*, Technical University of Eindhoven, 90-6864-551-7.

[56] **Bastidas de Calderon, María**, 1998, *Environmental Fragility and Vulnerability of Amazonian Landscapes and Ecosystems in the Middle Orinoco River Basin, Venezuela*, University of Ghent.

[57] **Moameni, Aziz**, 1999, *Soil Quality Changes under Long-term Wheat Cultivation in the Marvdasht Plain, South-central Iran*, University of Ghent.

[58] **van Groenigen, Jan-Willem**, 1999, *Constrained Optimisation of Spatial Sampling: A Geostatistical Approach*, Wageningen Agricultural University, 90-6164-156-X.

[59] **Cheng, Tao**, 1999, *A Process-oriented Data Model for Fuzzy Spatial Objects*, Wageningen Agricultural University, 90-6164-164-0.

[60] **Wolski, Piotr**, 1999, *Application of Reservoir Modelling to Hydrotopes Identified by Remote Sensing*, Vrije Universiteit Amsterdam, 90-6164-165-9.

[61] **Acharya, Bijnan**, 1999, *Forest Biodiversity Assessment: A Spatial Analysis of Tree Species Diversity in Nepal*, Leiden University, 90-6164-168-3.

[62] **Abkar, Ali Akbar**, 1999, *Likelihood-based Segmentation and Classification of Remotely Sensed Images*, University of Twente, 90-6164-169-1.

[63] **Yanuariadi, Tetra**, 1999, *Sustainable Land Allocation: GIS-based Decision Support for Industrial Forest Plantation Development in Indonesia*, Wageningen University, 90-5808-082-X.

[64] **Abu Bakr, Mohamed**, 1999, *An Integrated Agro-Economic and Agro-Ecological Framework for Land Use Planning and Policy Analysis*, Wageningen University, 90-6164-170-5.

[65] **Eleveld, Marieke A.**, 1999, *Exploring Coastal Morphodynamics of Ameland (The Netherlands) with Remote Sensing Monitoring Techniques and Dynamic Modelling in GIS*, Universiteit van Amsterdam, 90-6461-166-7.

[66] **Hong, Yang**, 1999, *Imaging Spectrometry for Hydrocarbon Microseepage*, Delft University of Technology, 90-6164-172-1.

[67] **Mainam, Félix**, 1999, *Modelling Soil Erodibility in the Semiarid Zone of Cameroon*, University of Ghent, 90-6164-179-9.

[68] **Bakr, Mahmoud I.**, 2000, *A Stochastic Inverse-Management Approach to Groundwater Quality*, Delft University of Technology, 90-6164-176-4.

[69] **Zlatanova, Siyka**, 2000, *3D GIS for Urban Development*, Graz University of Technology, 90-6164-178-0.

[70] **Ottichilo, Wilber K.**, 2000, *Wildlife Dynamics: An Analysis of Change in the Masai Mara Ecosystem*, Wageningen University, 90-5808-197-4.

[71] **Kaymakci, Nuri**, 2000, *Tectono-stratigraphical Evolution of the Cankori Basin (Central Anatolia, Turkey)*, Utrecht University, 90-6164-181-0.

[72] **Gonzalez, Rhodora M.**, 2000, *Platforms and Terraces: Bridging Participation and GIS in Joint-learning for Watershed Management with the Ifugaos of the Philippines*, Wageningen University, 90-5808-246-6.

[73] **Schetselaar, Ernst**, 2000, *Integrated Analyses of Granite-gneiss Terrain from Field and Multisource Remotely Sensed Data. A Case Study from the Canadian Shield*, University of Delft, 90-6164-180-2.

[74] **Mesgari, M. Saadi**, 2000, *Topological Cell-Tuple Structure for Three-Dimensional Spatial Data*, University of Twente, 90-3651-511-4.

[75] **de Bie, Cees A. J. M.**, 2000, *Comparative Performance Analysis of Agro-Ecosystems*, Wageningen University, 90-5808-253-9.

[76] **Khaemba, Wilson M.**, 2000, *Spatial Statistics for Natural Resource Management*, Wageningen University, 90-5808-280-6.

[77] **Shrestha, Dhruba**, 2000, *Aspects of Erosion and Sedimentation in the Nepalese Himalaya: Highland-lowland Relations*, Ghent University, 90-6164-189-6.

[78] **Asadi Harouni, Hooshang**, 2000, *The Zarshuran Gold Deposit Model Applied in a Mineral Exploration GIS in Iran*, Delft University of Technology, 90-6164-185-3.

[79] **Raza, Ale**, 2001, *Object-Oriented Temporal GIS for Urban Applications*, University of Twente, 90-3651-540-8.

[80] **Farah, Hussein O.**, 2001, *Estimation of Regional Evaporation under Different Weather Conditions from Satellite and Meteorological Data. A Case Study in the Naivasha Basin, Kenya*, Wageningen University, 90-5808-331-4.

[81] **Zheng, Ding**, 2001, *A Neuro-Fuzzy Approach to Linguistic Knowledge Acquisition and Assessment in Spatial Decision Making*, University of Vechta, 90-6164-190-X.

[82] **Sahu, Bijay K.**, 2001, *Aeromagnetics of Continental Areas Flanking the Indian Ocean; with Implications for Geological Correlation and Gondwana Reassembly*, University of Capetown, South Africa.

[83] **Alfestawi, Yahia Ahmed M.**, 2001, *The Structural, Paleogeographical and Hydrocarbon Systems Analysis of the Ghadamis and Murzuq Basins, West Libya, with Emphasis on Their Relation to the Intervening Al Qarqaf Arch*, Delft Technical University, 90-6164-198-5.

[84] **Liu, Xuehua**, 2001, *Mapping and Modelling the Habitat of Giant Pandas in Foping Nature Reserve, China*, Wageningen University, 90-5808-496-5.

[85] **Oindo, Boniface Oluoch**, 2001, *Spatial Patterns of Species Diversity in Kenya*, Wageningen University, 90-5808-495-7.

[86] **Carranza, Emmanuel John M.**, 2002, *Geologically-constrained Mineral Potential Mapping: Examples from the Philippines*, Technical University of Delft, 90-6164-203-5.

[87] **Rugege, Denis**, 2002, *Regional Analysis of Maize-based Land Use Systems for Early Warning Applications*, Wageningen University, 90-5808-584-8.

[88] **Liu, Yaolin**, 2002, *Categorical Database Generalization in GIS*, Wageningen University, 90-5808-648-8.

[89] **Ogao, Patrick**, 2002, *Scientific Visualization*, Utrecht University, 90-6164-206-X.

[90] **Abadi, Abdulbaset Musbah**, 2002, *Tectonics of the Sirt Basin: Inferences from Tectonic Subsidence Analysis, Stress Inversion and Gravity Modeling*, Vrije Universiteit Amsterdam, 90-6164-205-1.

[91] **Geneletti, Davide**, 2002, *Ecological Evaluation for Environmental Impact Assessment*, Vrije Universiteit Amsterdam, 90-6809-337-1.

[92] **Sedogo, Laurent D.**, 2002, *Integration of Local Participatory and Regional Planning for Resources Management Using Remote Sensing and GIS*, Wageningen University, 90-5808-751-4.

[93] **Montoya, Ana Lorena**, 2002, *Urban Disaster Management: A Case Study of Earthquake Risk Assesment in Cartago, Costa Rica*, Utrecht University, 90-6164-2086.

[94] **Mobin-ud Din, Ahmad**, 2002, *Estimation of Net Groundwater Use in Irrigated River Basins Using Geo-information Techniques: A Case Study in Rechna Doab, Pakistan*, Wageningen University, 90-5808-761-1.

[95] **Said, Mohammed Yahya**, 2003, *Multiscale Perspectives of Species Richness in East Africa*, Wageningen University, 90-5808-794-8.

[96] **Schmidt, Karen S.**, 2003, *Hyperspectral Remote Sensing of Vegetation Species Distribution in a Saltmarsh*, Wageningen University, 90-5808-830-8.

[97] **López Binnqüist, Citlalli**, 2003, *The Endurance of Mexican Amate Paper: Exploring Additional dimensions to the Sustainable Development Concept*, University of Twente, 90-3651-900-4.

[98] **Huang, Zhengdong**, 2003, *Data Integration for Urban Transport Planning*, Utrecht University, 90-6164-211-6.

[99] **Cheng, Jianquan**, 2003, *Modelling Spatial and Temporal Urban Growth*, Utrecht University, 90-6164-212-4.

[100] **Campos dos Santos, José Laurindo**, 2003, *A Biodiversity Information System in an Open Data-Metadatabase Architecture*, University of Twente, 90Ű6164Ű214Ű0.

[101] **Hengl, Tomislav**, 2003, *Pedometric Mapping: Bridging the Gaps Between Conventional and Pedometric Approaches*, Wageningen University.

[102] **Barrera Bassols, Narciso**, 2003, *Symbolism, Knowledge and management of Soil and Land Resources in Indigenous Communities: Ethnopedology at Global, Regional and Local Scales*, University of Ghent.

[103] **Zhan, Qingming**, 2003, *A Hierarchical Object-based Approach for Urban Land-use Classification from Remote Sensing Data*, Wageningen University, 90-5808-917-7.

[104] **Daag, Arturo Santos**, 2003, *Modelling the Erosion of the Pyroclastic Flow Deposits and the Occurrences of Lahars at Mt. Pinatubo, Philipines*, Utrecht University, 90-6164-218-3.

[105] **Bacic, Ivan Luiz Zilli**, 2003, *Demand Driven Land Evaluation: With Case Studies in Santa Catarina, Brazil*, Wageningen University, 90-5808-902-9.

[106] **Murwira, Amon**, 2003, *Scale matters! A New Approach to Quantify Spatial Heterogeneity for Predicting the Distribution of Wildlife*, Wageningen University.

[107] **Mazvimavi, Dominic**, 2003, *Estimation of Flow Characteristics of Ungauged Catchments: A Case Study in Zimbabwe*, Wageningen University, 90-5808-950-9.

[108] **Tang, Xinming**, 2004, *Spatial Object Modeling in Fuzzy Topological Spaces: With Applications to Land Cover Change*, University of Twente, 90-6164-2205.

[109] **Kariuki, Patrick C.**, 2004, *Spectroscopy to measure the swelling potential of expansive soils*, Technical University of Delft, 90-6164-221-3.

[110] **Morales G., Javier M.**, 2004, *Model-driven Design of Geo-Information Services*, University of Twente, 90-6164-222-1.

[111] **Mutanga, Onisimo**, 2004, *Hyperspectral remote sensing of tropical grass quality and quantity*, Wageningen University, 90-5808-981-9.

[112] **Šliužas, Ričardas V.**, 2004, *Managing informal settlements: a study using geo-information in Dar es Salaam, Tanzania*, Utrecht University, 90-6164-223-X.

[113] **Lucieer, Arko**, 2004, *Uncertainties in segmentation and their visualisation*, Wageningen University, 90-6164-225-6.

[114] **Corsi, Fabio**, 2004, *Applications of existing biodiversity information: capacity to support decision-making*, Wageningen University, 90-8504-090-6.

[115] **Tuladhar, Arbind M.**, 2004, *Parcel-based geo-information system: concepts and guidelines*, Technical University of Delft, 90-6164-224-8.

[116] **van Elzakker, Corné P.**, 2004, *The use of maps in the exploration of geographic data*, Utrecht University, 90-6809-357-6.

[117] **Nidumolu, Uday B.**, 2004, *Integrating geo-information models with participatory approaches: applications in land use analysis*, Wageningen University, 90-8504-138-4.

[118] **Koua, Etien L.**, 2005, *Computational and visual support for exploratory geovisualization and knowledge construction*, Utrecht University, 90-6164-229-9.

[119] **Blok, Connie A.**, 2005, *Dynamic visualization variables in animation to support monitoring of spatial phenomena*, Utrecht University, 90-6809-367-3.

[120] **Meratnia, Nirvana**, 2005, *Towards database support for moving object data*, University of Twente, 90-365-21521.

[121] **Yemefack, Martin**, 2005, *Modelling and monitoring soil and land use dynamics within shifting agricultural landscape mosaic systems in Southern Cameroon*, Utrecht University, 90-6164-233-7.

[122] **Kheirkhah, Masoud Zarkesh**, 2005, *Decision support system for floodwater spreading site selection in Iran*, Wageningen University, 90-8504-256-9.

[123] **Nangendo, Grace**, 2005, *Changing forest-woodland-savanna mosaics in Uganda: with implications for conservation*, Wageningen University, 90-8504-200-3.

[124] **Mohamed, Yasir Abbas**, 2005, *The Nile hydroclimatology: impact of the Sudd wetland*, Technical University of Delft, 0-415-38483-4.

[125] **Duker, Alfred**, 2005, *Spatial analysis of factors implicated in Mycobacterium ulcerans infection in Ghana*, Wageningen University, 90-8504-243-7.

[126] **Ferwerda, Jelle G.**, 2005, *Charting the quality of forage: measuring and mapping the variation of chemical components in foliage with hyperspectral remote sensing*, Wageningen University, 90-8504-209-7.

[127] **Martinez, Javier A.**, 2005, *Monitoring intra-urban inequalities with GIS-based indicators: with a case study in Rosario, Argentina*, Utrecht University, 90-6164-235-3.

[128] **Saavedra, Carlos P.**, 2005, *Estimating spatial patterns of soil erosion and deposition in the Andean region using geo-information techniques: a case study in Cochabamba, Bolivia*, Wageningen University, 90-8504-289-5.

[129] **Vaiphasa, Chaichoke**, 2006, *Remote sensing techniques for mangrove mapping*, Wageningen University, 90-8504-353-0.

[130] **Porwal, Alok**, 2006, *Mineral potential mapping with mathematical geological models*, Utrecht University, 90-6164-240-X.

[131] **van der Werff, Harald M.A.**, 2006, *Knowledge-based remote sensing of complex objects: recognition of spectral and spatial patterns resulting from natural hydrocarbon seepages*, Utrecht University, 90-6164-238-8.

[132] **van de Vlag, Daniël**, 2006, *Modeling and visualizing dynamic landscape objects and their qualities*, Wageningen University, 90-8504-384-0.

[133] **Joshi, Chudamani**, 2006, *Mapping cryptic invaders and invisibility of tropical forest ecosystems:* Chromolaena odorata *in Nepal*, Wageningen University, 90-8504-470-7.

The work of this thesis was funded by

International Institute for Geo-Information Science and Earth Observation
Enschede, the Netherlands

The work was carried out according to the PhD regulations of

Wageningen University
the Netherlands

This research was funded by the European Community project REV!GIS (IST-1999-14189).