© Peng Zhang 2018
Except where otherwise indicated, this thesis is my own original work.
Parts of the material in this thesis has previously appeared in the following publications:

Table 1: List of chapters that consists of past publications

<table>
<thead>
<tr>
<th>Thesis Chapter</th>
<th>Publication Title</th>
<th>Publication Status</th>
<th>Candidate’s contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 3</td>
<td>From Raw Sensor Data to Detailed Spatial Knowledge.</td>
<td>In IJCAI’15</td>
<td>Principal author. Jae Hee Lee helped with formalizing problems. Jochen Renz helped with designing experimental environment</td>
</tr>
<tr>
<td>Chapter 4</td>
<td>Trend-Based Prediction of Spatial Change</td>
<td>In IJCAI’16</td>
<td>Equal contribution author. Xiaoyu Ge designed algorithms on matching spatial regions at different states, and obtaining the outer approximation of the changes. Jae Hee Lee implemented Kalman Filter and helped with formalizing problems. Jochen Renz designed a qualitative reasoning method for region matching and helped with formalizing problems.</td>
</tr>
<tr>
<td>Chapter 5</td>
<td>Qualitative Spatial Representation and Reasoning in Angry Birds: the Extended Rectangle Algebra</td>
<td>In KR’14</td>
<td>Principal author. Jochen Renz helped with formalizing problems and refining the qualitative representation and rules</td>
</tr>
<tr>
<td>Chapter 6</td>
<td>Support Relation Analysis for Objects in Multiple View RGB-D Images</td>
<td>under submission to IJCAI’18</td>
<td>Principal author. Xiaoyu Ge helped with designing and implementing static equilibrium for force analysis, and helped with performing experiments. Jochen Renz helped with formalizing problems</td>
</tr>
</tbody>
</table>
To the memory of my grandfather, Yali Zhang.
Acknowledgments

First, I would like to express my sincere appreciation to my supervisor Professor Dr. Jochen Renz for his continuous support and encouragement during my Ph.D. study. It is my fortune to have such a supervisor who provided invaluable advices and support to my research and career, and showed warm care to my life like a family.

I would like to thank the Australian National University and Australian government for offering me scholarships to cover my living cost and tuition fee. I also thank IJCAI, KR, ECAI and ARC for providing me grants for attending conferences.

I would like to thank my colleagues and friends Xiaoyu Ge, Matthew Stephenson and Hua Hua for the uncountable interesting discussions and the late nights we were working together before the deadlines.

I would like to thank Dr. Jae Hee Lee for the help on mathematics and the advices on my research.

My thanks go to all my families for the love and solicitude.

Finally, I would like to show my deepest thanks to my parents for their endless love and unconditional support throughout my life. And to my beloved wife, thank you for your understanding and patience, for accompanying me in the sleepless busy nights. And most of all, thank you for being my best friend.
Abstract

Artificial intelligence (AI) systems that can interact with real world environments have been widely investigated in the past decades. A variety of AI systems are demonstrated to be able to solve problems robustly in a fully observed and controlled environment. However, there still remain problems for AI systems to effectively analyse and interact with semi-observed, unknown or constantly changing environments. One main difficulty is the lack of capability of dealing with raw sensory data in a high perceptual level. Low level perception cares about receiving sensory data and processing the data systematically so that it can be used in other modules in the system. Low level perception may be acceptable for AI systems working in a fixed environment, as additional information about the environment is usually available. However, the absence of prior knowledge in less observed environments produces a gap between raw sensory data and the high level information required by many AI systems. To fill the gap, a perception system which can interpret raw sensory input into high level knowledge which can be understood by AI systems is required.

The problems that limit the quality and capability of perception of AI systems are multitudinous. Although low level perception which concerns data reception and pre-processing is a significant component of a perception system, in this work, we focus on the subsequent high level perception tasks which relate to abstraction, representation and reasoning of the processed sensory data. There are several essential capabilities for high level perception of AI systems for analysing the requirement of critical information before a decision is made. First, the ability to represent spatial properties of the sensory data helps the perception system to resolve conflicts from sensory noise and recover incomplete information missed by the sensors. We develop an algorithm to combine raw sensory measurements from different view points of the same scene by resolving contradictory information and reconcile spatial features from different measurements. With spatial knowledge representation and reasoning (KRR), the ability of inferring and predicting changes of the environment from current and previous states will provide further guidance to the AI system for decision making. For this ability, we develop a general spatial reasoning system that predicts the evolution of constantly changing regions. However, in many situations where the AI system needs to physically interact with the entities, spatial knowledge is necessary but not sufficient. The ability of analysing physical properties of entities and their relations in the environment is required. For this task, we first develop a 2-dimensional reasoning system that analyse the support relations of rectangular blocks and predict the weak part of the structure of blocks. Then, we extend this idea to develop a method to reason about the support relations of real-world objects in a stack using RGB-D image data as input. With the above mentioned capabilities, the perception system is able to represent spatial properties of entities and their relations as well as predicting their evolutionary trend by discovering hidden information from the raw sensory data. Meanwhile, for manipulation tasks, supporting relations between objects can also be inferred by combining spatial and physical reasoning in the perception system.
List of Publications


## Contents

- **Acknowledgments** vii
- **Abstract** ix
- **List of Publications** xi

### 1 Introduction 1
1.1 Perception For AI Agents ........................................ 3
1.2 Spatial and Physical Reasoning in High Level Perception Systems ... 5
   1.2.1 Perception Problems Related to Spatial Reasoning ............ 5
   1.2.2 Perception Problems Related to Physical Reasoning .......... 7
1.3 Methodology .......................................................... 7
1.4 Contributions and Thesis Outline .................................. 8

### 2 Background and Related Work 11
2.1 Qualitative Physical Reasoning .................................... 11
2.2 Qualitative Spatial Reasoning ..................................... 12
2.3 Hybrid Reasoning ..................................................... 13
2.4 High Level Perception and Scene Understanding .................. 14

### 3 Spatial Knowledge Representation of Raw Sensory Data 17
3.1 Introduction .......................................................... 17
3.2 Our Approach ......................................................... 18
3.3 Related Work ........................................................ 21
3.4 Closing the Gap in Sensor Measurements .......................... 22
   3.4.1 Integration of Sensor Measurements .......................... 23
   3.4.2 Spatial Knowledge Extraction .................................. 25
3.5 Evaluation ............................................................ 26
3.6 Conclusions .......................................................... 28

### 4 Spatial Change Prediction of Evolving Regions 31
4.1 Introduction .......................................................... 31
4.2 Related Work ......................................................... 32
4.3 Problem Description and Approach ................................ 33
   Input ................................................................. 33
   Trend Analysis ...................................................... 33
   Prediction Model .................................................... 34
   Output .............................................................. 35
4.4 Trend-Based Prediction Algorithm ................................ 35
   4.4.1 Qualitative Adjustment of Trend .............................. 35

xiii
List of Figures

1.1  A stack of tableware that is common in daily life .......................... 2
1.2  Three-layer model of a perception system in an AI agent ..................... 4

2.1  Example of Top Supporter. Both A and C support B .......................... 15
2.2  Example of Core Supporter. B is a core supporter of A, but neither C nor D is a core supporter of A .......................... 16

3.1  A complex region measured by two sets of sensors ............................ 19
3.2  Integration of sensor measurements based on Algorithm 1 ..................... 22
3.3  An illustration of the augmented containment tree extracted from the merged measurements in Figure 3.2c .......................... 27
3.4  A randomly generated complex region ............................................. 28

4.1  A wildfire progression example. The boundaries of fire regions \( R_t \) are indicated in red ............................................................... 34
4.2  Depicted is region \( R_t \) for \( t = 0 \) and \( t = 1 \) ................................. 35
4.3  An example of the trend identification between evolving regions. The boundary points are indicated by the red dots. Each individual assignment between two points \( z_t \) and \( z_{t+1} \) is highlighted by a blue line .............................................. 36
4.4  A graphical representation of the assignment between \( \partial R_0 \) and \( \partial R_1 \). Each line with an arrow is an assignments before adjustment. Each dashed line stands for an assignment that violates a certain rule. We resolve these violations in breadth-first order .............................................. 37
4.5  Prediction and outer approximation of \( R_3 \) based on \( R_0, R_1, R_2 \) from Figure 4.1. The red dots are the ground truth in Figure 4.1d. It shows that the approximated boundary (shown in dark blue) based on \( d = 2.5 \) covers most of the actual boundary points. The one (shown in light blue) based on \( d = 5 \) covers all the actual boundary points .......................... 41
4.6  Comparasion between prediction results with and without qualitative adjustment ................................................................. 42

5.1  Two structures with the same RA relation (si,mi) .................................. 47
5.2  9 types of angular rectangles [Ge and Renz, 2013] ................................ 50
5.3  Examples for ERA-stable cases of regular rectangles .......................... 52
5.4  ERA-stable cases of angular rectangles .............................................. 54
5.5  Illustration of support structure .......................................................... 55
5.6  Two examples for case 2 ................................................................. 57
5.7  Examples for Cases 3 & 4 ................................................................. 57
5.8  Process of the proposed two step planner .............................................. 59
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.9</td>
<td>Level 1-16 in Chrome Angry Birds and agent output for this level</td>
</tr>
<tr>
<td>5.10</td>
<td>Actual result for the recommended shot</td>
</tr>
<tr>
<td>5.11</td>
<td>Actual result for the 3rd ranked shots</td>
</tr>
<tr>
<td>6.1</td>
<td>Segmentation of aligned images</td>
</tr>
<tr>
<td>6.2</td>
<td>Object matching result example. The top two objects are correctly matched and the bottom object is mismatched</td>
</tr>
<tr>
<td>6.3</td>
<td>Unique minimal bounding rectangle for region $r$, bounded by four straight lines $x = r_{x_l}, x = r_{x_r}, y = r_{y_u}, y = r_{y_d}$</td>
</tr>
<tr>
<td>6.4</td>
<td>Conceptual neighborhood graph for EIA based on horizontal view rotation</td>
</tr>
<tr>
<td>6.5</td>
<td>Cardinal direction relations</td>
</tr>
<tr>
<td>6.6</td>
<td>Results on core supporter detection</td>
</tr>
</tbody>
</table>
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>List of chapters that consists of past publications</td>
<td>iii</td>
</tr>
<tr>
<td>3.1</td>
<td>Evaluation based on 100 randomly generated complex regions</td>
<td>28</td>
</tr>
<tr>
<td>4.1</td>
<td>Six basic topological changes. The changes are given both pictorially and with containment trees. The left containment tree is before a change, and the right containment tree is after the change. The region boundaries where the change takes place are coloured in red.</td>
<td>39</td>
</tr>
<tr>
<td>5.1</td>
<td>27 EIA relations</td>
<td>48</td>
</tr>
<tr>
<td>5.2</td>
<td>Evaluation on first 21 levels. #RO is the number of reachable objects, $S_1$, $S_2$, and $S_3$ are the scores obtained when shooting at the highest, second-highest and third highest ranked object. HS is the rank of the reachable object with the highest score</td>
<td>60</td>
</tr>
<tr>
<td>6.1</td>
<td>View order results</td>
<td>79</td>
</tr>
<tr>
<td>6.2</td>
<td>Support relation results</td>
<td>79</td>
</tr>
</tbody>
</table>
Artificial intelligence (AI) agents have been demonstrated to be capable of assisting or superseding human on regular tasks which require repeatable and predictable actions. In other words, AI agents are adept in performing actions within a limited scope where most parameters are known or foreseeable. For example, expert systems have been widely used to assist human specialists on decision making by efficiently verifying constraints in a specific domain. As another example, the increasing magnitude of online shopping industry results in the rapid increasing of parcel post amount. In China, the amount of parcel posts increased by more than 20 times in the last decade $^1$ which caused a severe shortage of parcel sorting workers. Automotive logistic coordinating systems are able to sort and distribute thousands of parcels each day. For example, a sorting system developed by STO is able to sort more than 200,000 parcels per day $^2$ which breaks a bottleneck of the industry. Despite of the success of these domain-specific AI agents, one of their general characters is the requirement of a highly observed and relatively fixed environment. This requirement provides AI agents with significant benefits. In a highly observed environment, AI agents may receive additional information for recognising and modelling entities which weakens the requirement of a perception procedure. In a relatively fixed environment, the agents can accurately obtain relations between entities and motion planning and manipulation tasks will be simplified. However, this feature limits this kind of AI agents to involve in more general and complex tasks where the assumptions on the working environments are not valid.

Ideally, we would like AI agents to interact with an arbitrary environment where only partial information is available with uncertainty. By accomplishing this target, agents will be able to replace human from dangerous missions such as earthquake rescue, or to free people from daily repeatable but non-systematic tasks such as housework. So far, AI agents’ capability of dealing with such semi-observed or unknown environment still remains to be greatly improvable. An AI agent’s performance depends on multiple factors including perception, manipulation, motion planning and so on [Murphy 2000]. In unknown environments, perception systems may not provide an accurate result due to the absence of required input information; then manipulation will fail because of incorrect perception of entities in the environment; motion planning will also be of no sense as unexpected behaviours will occur when the information about the environment is incomplete. Thus, among those limitations,
Introduction

perception is the first and foremost step to be considered in order to have AI agents work in unknown environments. Intuitively, a perception system is responsible for sensing and collecting data from the surrounding environment and propagating it to other modules of AI agents. On the other hand, processing sensory data is also significant for perception systems as low quality and superficial output data will negatively affect decision making and manipulation steps. However, low level processing steps are necessary but insufficient for inferring implicit knowledge. Among all categories of knowledge hidden in the sensory data, spatial and physical knowledge are essential for AI agents to work in an unknown environment. For example, in a wildfire scenario, in order to provide guidance on fire-fighting strategies, fire regions are sensed and segmented at the very first step. But a more important step is to represent the fire regions in a way that their spatial properties and relations can be extracted and used for determining safe zones and predicting potential dangerous areas overtime. Spatial knowledge representation and reasoning capabilities allow AI agents to understand their surrounding environment and be prepared for potential changes. While interacting with the entities in the environment is required, capability of reasoning about physical knowledge plays an important role. Imagine a simple scenario that an agent would like to grasp a plate underneath other tablewares (see Figure 1.1). If the agent only recognises the objects but ignores their physical properties and relations, the decision of grasping may be on the bottom plate directly which will result in a collapse of the whole stack.

Simple commonsense physical reasoning about supporting relations and stability is required here to figure out the potential risk of the decision and satisfies both the explicit goal which is grasping the plate and the implicit goal which is preventing undesired and dangerous motions while attempting to achieve the first goal. This is a rather simple task for humans, but it is extremely hard for AI agents. AI agents are much better at quantitative computation than humans but are less capable of perceiving concepts qualitatively. This is the reason why AI agents can beat human
experts in deterministic games such as chess and go but are struggling to complete an everyday task such as the one in Figure 1.1. In order for machine perception systems to gain the capabilities of understanding spatial and physical properties and relations, a proper representation of the sensory data will produce organised data for efficient computation and reasoning. Also, reasoning on the sensory data will help discover implicit information that may be required by other modules of AI agents.

An ordinary perception system may pass through all sensory data without or with only minor processing such as noise filtering; a more advanced perception system will deal with high level data processing such as object recognition and clustering; rather intelligent perception systems are capable of reasoning about complex knowledge from sensory data. In a fixed and highly observed environment, perception systems are able to collect and process sensory data efficiently and accurately. However, in an arbitrary environment, perception systems will face several challenges. In this thesis, we specifically discuss the problems with respect to spatial and physical knowledge representation and reasoning. In an unknown environment, sensory data usually only covers a small amount of the required data. Recovering hidden information from incomplete input data needs to be solved first. Furthermore, spatial knowledge is usually considered with priority, as physical knowledge is based on spatial knowledge in most cases. In addition, determining the features of entities and understanding relations between entities are both challenging tasks. Similarly, the entities’ spatial properties and relations are important for discussing their physical properties and relations. In this thesis, we developed a set of algorithms to address the mentioned problems and tested them in both simulated and realistic environments to demonstrate the usefulness.

1.1 Perception For AI Agents

Perception is the ability of understanding the surrounding environment by receiving and processing raw input data such as eye vision and sound. Henderson and Hollingworth [1999] divided human perception of a scene into three layers, where low-level perception concerns data collection and initial process; intermediate-level perception deals with data representation in terms of the entities’ geometric and spatial features; high-level perception cares about mapping meanings to the data representation. Chalmers et al. [1992] divided perception tasks into two levels, where low level perception refers to non-semantic data reception and process and high level perception starts when concepts of entities and their relations are considered. Perception tasks for AI agents aim to enable machine perception system to perceive the environment in a way that is similar to human and animal perception [Nevatia 1982]. The techniques related to AI system’s perception cover many different fields of computer science and engineering. By analysing the functionalities of different components, we divide typical perception systems for AI agents into three layers (see Figure 1.2).

The first layer consists of sensor systems which collect data from the surrounding environment. This layer is highly hardware-related, therefore the performance of this layer is limited by physical implementation of the sensors. The second layer contains techniques that process raw sensory data from the first layer so that it can be used by other modules, though the output only represents explicit information from
Introduction

Sensors

Environment

Data Pre-Processing
(data filtering, data cleansing, entity detection, ...)

High Level Data Processing
(Spatial reasoning, Physical reasoning, Integration of data from different sources, ...)

Perception System

AI Agent

Decision Making Modules

Data Collection

Raw Sensory Data

Structured Data

Data With Implicit Information

Figure 1.2: Three-layer model of a perception system in an AI agent
raw data. The data processing step may vary with different sensors and perception tasks. However, the general purpose of this layer is to extract useful information and abandon redundant components from the sensory data while considering the processing capability of sensors. For example, a set of laser sensors which aims to detect spatial features of a complex region may need to be able to communicate with each other at a low power cost which requires the ability of filtering and compressing data. Visual sensors (e.g. cameras and depth sensors) may be able to detect interesting objects from the entire scene and filter out background layers. This layer focuses on providing structured data instead of raw data for the third level to perform high level reasoning. The third layer takes results from the second layer as input to further reason about hidden information from the explicit knowledge. Thus, although the last two layers both deal with data processing, the output from the third layer concerns more about implicit information extraction rather than the existing data itself.

The proposed methods in this thesis is mainly related to the third layer which aims to improve the perceptual output by reasoning about relations among entities detected from raw sensory data. Undoubtedly, the first two layers determine if the perception result is valid for AI agents to produce reasonable outcomes, and the outputs from the first two layers demonstrate the perception system’s capability of interpreting the information of the surroundings of AI agents. However, the third layer decides to what extent the AI system is intelligent. Humans usually recognise concepts and relations between entities qualitatively. AI agents designed for many application areas require high level knowledge from this layer. For example, geographic information system (GIS) concerns high level spatial information of involved entities for several purposes including fast query processing, entity similarity assessment and visualization [Wolter and Wallgrün, 2012]; robotic manipulation requires accurate spatial and physical relations between objects. However, machine perception systems are far less advanced than human perception systems. Even intuitive perceptual capabilities for humans are difficult for machines. In the following section, research problems related to spatial and physical aspects in the third level will be discussed.

1.2 Spatial and Physical Reasoning in High Level Perception Systems

In this section, we start with the discussion of research problems related to spatial reasoning followed by a discussion of physical aspect.

1.2.1 Perception Problems Related to Spatial Reasoning

In humans’ daily life, different kinds of spatial knowledge provides humans with capabilities of perceiving properties of spatial entities and their relations. For example, a target entity may be localised by analysing its directional relations with different landmarks; topological information helps people identify properties such as connectivity and containment relations among spatial entities. Other spatial information includes distance, size, shape, morphology and spatial change are also useful for spatial reasoning tasks [Renz and Nebel, 2007]. Humans will then apply different reasoning mechanisms to infer useful information for further decision making. Spatial
knowledge representation and reasoning also plays an important role in machine perception. It is one of the essential components that makes GIS different from traditional map processing techniques [Berry, 1996]. Specifically, different aspects of spatial reasoning problems in GIS have been addressed, such as reasoning about directional information [Freksa, 1992; Frank, 1991, 1996], reasoning with topological information [Renz, 2002; Egenhofer, 1991; Cui et al., 1993; Egenhofer and Herring, 1990], reasoning about a combination of different kinds of spatial information [Frank, 1992; Zimmermann and Freksa, 1996; Zimmermann, 1993; Gerevini and Renz, 1998; Liu et al., 2009a].

In robotics, spatial reasoning is advocated to be an important capability of linking machine sensory data to human understandable spatial knowledge in order to accomplish human-robot interaction [Landsiedel et al., 2017]. Although high-accuracy sensors are available in recent years, the complex hardware is usually energy consuming and low in efficiency. Thus, reasoning and integrating relatively simple sensory data from different sources with low cost still remains as an important research topic in robotic perception [Smith et al., 2013]. For example, Moratz and Wallgrün [2003] discussed how spatial reasoning about topology, orientation and distance contributes to robot exploring in an indoor scene. In addition, qualitative spatial reasoning (QSR) approaches can adapt to uncertainty from incompleteness of sensory data, while pure quantitative models will fail due to insufficient input. Escrig and Toledo [1998] presented methods to solve different QSR tasks as constraint satisfaction problem which can be applied to robotics navigation.

Based on the discussion above, QSR mainly cares about representing properties of spatial entities and inferring their relations and potential changes. This characteristic allows us to reduce three dimensional (3D) entities into two dimensional regions by producing a projected map from a bird-eye view. For some problems that are not sensitive to manipulation but focusing on analysing overall trend of the input data, a bird-eye sensory input reduces unnecessary information but retains generally interesting parts. Although 3D spatial reasoning techniques have been studied recently [Albath et al., 2010; Cohen and Hegarty, 2012], these methods are usually with high complexity and will produce higher level of uncertainty. Therefore two dimensional (2D) spatial reasoning is still preferable as long as the projected dimension does not affect the reasoning tasks significantly.

In this thesis, we mainly focus on developing spatial reasoning methods for 2D entities. We are also interested in getting through the whole pipeline of spatial reasoning techniques in AI perception systems. As a start, we need to figure out how to represent spatial knowledge from raw sensory data for the usage of spatial reasoning. Previous spatial reasoning methods usually assume the availability of organised spatial information, thus research on this problem will fill the gap between many existing spatial reasoning approaches and real-world applications. Then, extraction of useful spatial information includes topology, orientation and distance relations among entities from the represented sensory data is the first step of reasoning tasks. By this step, the perception system is able to perform spatial reasoning on static spatial entities. Moreover, the capability of predicting changes of spatial regions is important for an AI agent that interacts with the physical world. For agent without advanced spatial reasoning capabilities, prediction tasks are processed with simulation methods. One of the disadvantages of simulators is the requirement of precise quantitative
data input which is usually not completely available in the real world scenario. With QSR methods, high flexibility of data requirement will provide the perception system an appropriate generality when quantitative methods with specific models does not work.

1.2.2 Perception Problems Related to Physical Reasoning

Reasoning about physical properties and relations is more complicated than reasoning about spatial information. Spatial information usually can be observed and represented from the perceptive of different categories. Even in an unknown environment where sensory data is incomplete, missing parts can be inferred from the collected information following general routines. However, the methods for reasoning about physical properties and relations of entities vary from case to case. Thus, most physical reasoning frameworks are designed to work in specific domains. In the past decades, qualitative physical reasoning have been extensively studied. In [Weld and de Kleer, 2013], most of the early work on qualitative physics has been discussed. Methods for modelling of naive physical processes and their causalities are covered in great detail. In this thesis, we focus on modelling a specific physical property of the objects, which is the supporting relation between connected objects in a structure. The discussion about research problems related to spatial reasoning in section 1.2.1 emphasises analytical capabilities of perception systems on sensory data rather than actually interacting with the world physically. Different from spatial reasoning tasks, the bird-eye view will lead to a considerable information loss when the entities’ physical properties and relations are taking into account, as gravity which always points to the earth’s core implies the importance of the vertical dimension. The first problems we would like to address is to validate a method that is able to model the supporting relation between objects and the overall stability of the structure in a simplified 2D environment with limited prior knowledge about physical properties of the objects. Then, we will generalise the 2D method to 3D environments. This will lead to more challenges. For example, the representation of 3D entities are more complicated than that of 2D models. Also, 3D physical relations bring larger magnitude of uncertainty. If the 2D supporting and stability reasoning method does not satisfy in 3D world, how could we improve or modify it to fit the 3D environment is worth exploring.

1.3 Methodology

The performance of AI agents highly depends on the capability of perception systems on mining implicit information from raw or preprocessed sensory data. From the perspective of high-level reasoning in perception systems, two topics are worth exploring:

- How to represent simple but massive sensory data?
  Sensory data which is collected from simple sensor nodes may be noisy and incomplete. The sensory data from AI perception systems is usually simpler than data sensed from human perception due to limitations of machine sensors. Raw sensory data is massive because typical machine sensors do not actively select potential useful information from the whole data collection. Differently,
Introduction

humans are good at observing and summarising rules from their observations to link and combine knowledge from different sources. For example, people are able to generate a map with reasonable landmarks in their mind after viewing a region from different angles. This problem is seemingly straightforward for humans, as humans have rich prior knowledge and reasoning capabilities to build linkage between different entities. It is however extremely difficult for AI systems. Compared with humans, AI agents do not have as extensive knowledge base and as strong capability of inducing knowledge from existing knowledge base. In order for AI agents to improve performance on real world perception tasks, sophisticated representations of sensory data which benefit reasoning procedures would be seriously considered.

• How to extract sufficient information by reasoning on the data representation for AI agents with special purposes? The principle of human inspiration still remains unclear and yet to be applied on AI agents. But it is important for AI agents to discover potential relations between scattered pieces of sensory data from sophisticated knowledge representation on specific domains. With incomplete input sensory data, a high uncertainty would be introduced while reasoning about underlying information. In order to reduce the uncertainty, novel reasoning mechanisms would be necessarily proposed.

Generally, we follow a bottom-up approach for the whole research procedure. First, we observe some practical unsolved problems related to spatial and physical aspects of AI perception systems. Then, we identify general characteristics of each kind of problem and formalise the problems. After that, we formulate hypotheses on how the problems could be solved potentially. Finally, we develop sophisticated methods to solve the problems in a systematic way with reasonable evaluation in both simulation and realistic environments.

1.4 Contributions and Thesis Outline

In this thesis, some essential problems related to spatial and physical aspects in AI perception systems are identified. For the spatial aspect, three problems are addressed:

1. How to integrate raw sensory data from different sources to suitable represent spatial information with noise and conflicts

2. How to extract spatial properties and relations between entities from the representation

3. How to predict spatial changes of entities with a tracking of their previous states

For the physical aspect, we focus on solving intuitive physical reasoning problems on identifying supporting relations between objects and determining the stability of both a single object and the whole structure. We first propose a rule-base reasoning approach to solve the problems in a 2D scenario to demonstrate the feasibility. Then we generalise the method to 3D real-world environment by combining qualitative reasoning with intuitive mechanical model about system stability.
This thesis consists of 7 chapters. In Chapter 2, background about related sensors, processing of sensing data and reasoning techniques related to the algorithms proposed in this thesis are discussed. In Chapter 3, we will present a method on merging multiple laser sensor measurements from different aspects at the same state in order to recover the missing information from single measurement and extracting high level spatial knowledge especially topological knowledge from the merged sensor data. In Chapter 4, we further develop an algorithm on analysing and predicting changing trends of the detected regions at different states. With the methods from these two chapters, we are able to perform reliable spatial reasoning on both a static and dynamic basis. The next two chapters focus on the problems related to formalising physical reasoning mechanism on analysing supporting relations between entities and the stability of both an individual object and the structure. In Chapter 5, a purely qualitative reasoning method is proposed to solve this problem in the Angry Birds game which is a perfect testbed for supporting and stability analysis in 2D scenario. Then, in Chapter 6, we extend the proposed representation in the previous chapter and combine the reasoning process with a quantitative mechanical model. The proposed hybrid reasoning approach successfully deals with the incompleteness of input data and reduces the uncertainty produced from pure qualitative reasoning method. The above four chapters demonstrate approaches to solve some essential spatial and physical reasoning problems in AI perception systems. In Chapter 7, the proposed perception methods are summarised first, following by a list of potentially interesting problems to be solved in the future research.
Introduction
Background and Related Work

An important difference between an AI agent and an autonomous system is that the AI agent has the ability to adapt to different situations and to make decisions based on the information it collected. Thus, perception of real world environment is essential for AI agents to analyse and interact with their surroundings. Many techniques will be involved in a perception system for AI agents. For the low level perception, different sensors are used to collect necessary information. A commonly used sensor type is a vision sensor for capturing visual data in the form of single frame or video stream. There are many other sensor types to be used in different AI systems. For example, GIS may require specific sensors such as laser sensor for region measurement and hygrothermograph for climatic condition monitoring. At a higher level, the collected data needs to be processed in a way that it can be used by the decision making modules. For example, raw visual input will be segmented and the entities will be labeled, so that the agent can have a basic idea about the environment. Properly representing sensory data and reasoning about the relations between the detected entities will help AI agents to further infer hidden information that can be potentially useful for other modules. For instance, spatial relations such as connectivity and relative direction are helpful for localization, and the supporting relation between entities is important for robotic manipulation.

Due to the noise and incompleteness of real world sensory data, quantitative representation of the data may be inaccurate and lead to incorrect decisions. Qualitative reasoning (QR) which aims to maintain valuable information while discarding unimportant part is needed for real world perception. In this chapter, related work on the proposed perception algorithms including qualitative spatial reasoning, qualitative physical reasoning, hybrid reasoning and scene understand will be discussed in turn.

2.1 Qualitative Physical Reasoning

Perceiving general physical knowledge in the real world for AI agents is challenging due to high level uncertainty produced by limited range of sensory data. Qualitative reasoning has been demonstrated to be capable of dealing with incomplete input data [Davis, 2008]. From onwards 1970s, qualitative physical reasoning has become an active research area with significant fundamental work emerged. NEWTON [de Kleer, 1977] which performs physical predictions of motions of objects in a controlled environment showed the capability of qualitative reasoning systems to
solve commonsense physical problems. [Hayes 1979] proposed to construct a general
large-scale formalism of commonsense physical knowledge about the world instead
of modelling trial toy problems individually. ENVISION [de Kleer and Brown 1984]
takes components as the primitives of physical state changes. By introducing a set
of interacting rules, the possible subsequent states of the component and the causal
analysis will be provided. In contrast, Qualitative Process Theory (QP) [Forbus 1984]
considers the physical processes as the cause of the behaviours. Therefore detailed
modelling of components is not required. Different from the behaviour primitives of
ENVISION and QP, QSIM [Kuipers 1986] applies pure mathematical equations as
constraints but not physical modelling of the components, processes and etc. These
three frameworks with different primitives of physical behaviours cover most parts of
early qualitative physics research.

In addition, the existence of massive physical domains makes it implausible to
have a universal framework to model all physical domains. [Forbus 1986] attempted
to identify a computational account of how humans learn new physical domain
knowledge but it is yet to be implemented. On the other hand, modelling of some
general physical domains are discussed in [Davis 2008]. For example, kinetic theory
and dynamic theory are two different methods for reasoning about motions of rigid
body with different expressiveness. The dynamic theory of rigid body, with a richer
expressiveness, is discussed in detail in [Davis 2007] where a pure qualitative method
is used to model the components and processes in a closed domain.

However, present potential applications of physical reasoning often require the
capability of performing reasoning in a complex domain, therefore dealing with
quantitative information together with qualitative information is required. Such
hybrid reasoning approaches will be discussed in Section 2.3.

2.2 Qualitative Spatial Reasoning

Spatial knowledge is one of the most important components of the physical world
as it provides entities with necessarily basic relations between each other such as
connectivity and relative location in order to further reason about their physical
properties. Early QSR work as one component of qualitative physics arose the
problem of poverty conjecture [Forbus et al. 1991]: no problem-independent, purely
qualitative spatial representation exists. Since then, more foundational work with
respect to different aspects of QSR including topology [Egenhofer and Franzosa 1991;
Randell et al. 1992], direction [Frank 1992; Ligozat 1998; Renz and Mitra 2004], size
and shape has emerged which weakened the poverty conjecture to some extent [Cohn
and Renz 2008]. In real world scenarios, there is usually more than one aspect of
spatial knowledge to be concerned in a single reasoning task. In order to reason
about realistic spatial problems, combination of different aspects of spatial calculi
has been studied. For example, reasoning about combination of topological and size
information was studied in [Gerevini and Renz 1998]. The combination of topological
and directional information was also investigated. In [Liu et al. 2009a], RCC8 [Randell
et al. 1992] was combined with Rectangle Algebra (RA) [Balbiani et al. 1999] and the
Cardinal Direction Calculus (CDC) [Goyal and Egenhofer 2001; Skiadopoulos and
Koubarakis 2005]. In [Li 2007], a combination between RCC8 and DIR9 (a subset
of RA) was demonstrated. Reasoning with multiple aspects of spatial knowledge
introduces more strict constraints, therefore reducing ambiguity of the reasoning results. However, the complexity is usually higher than reasoning with a single spatial aspect, thus the calculi to be combined should be selected and analysed carefully. In Chapter 3, a combination of size and directional information is used as constraints to decide the matching of the same set of spatial regions from different sensory input.

In addition, with the development of sensory techniques, 3D spatial representation starts to draw attention. Similar to 2D spatial reasoning, 3D topology as a fundamental aspect was first studied [Albath et al., 2010; Sabharwal and Leopold, 2014]. Mojtahedzadeh et al. [2013] proposed a relational scene representation with respect to contact relations between 3D blocks and applied this representation to analyse the support relation between cuboid-shaped objects. [Li et al., 2009] demonstrated a 3D spatial calculus combining topological and directional constraints and introduced an uncertainty model for dealing with large ambiguity of the 3D representation. A common limitation of the above mentioned calculi is that gathering precise quantitative data for constructing qualitative relations is difficult. Imprecise and incomplete input data may lead to much higher ambiguity or even mistakes in the reasoning outcomes. As a result, conflicts which are hard to resolve will occur during reasoning procedures (e.g. composition). Instead of typical 3D spatial reasoning methods, a 2D reasoning method that can deal with 3D data with high incompleteness by projecting the 3D entity onto a 2D plane is discussed in Chapter 6. In addition, in [Asl and Davis, 2014], a pure qualitative 3D rotation calculus was proposed to model the motion of robot arms. In this application, the robot arm is fully modeled, thus there is no concern with quality of the qualitative constraints. However, as a trade-off, this calculus can only be used in a narrow domain and is difficult to be extended.

Apart from reasoning about static spatial properties, spatial change is also of great interest. Reasoning with spatial change can be regarded as reasoning with a combination of spatial and temporal information [Galton, 2000]. It is emphasized that the assumption of continuous change is essential for reasoning about qualitative spatial change [Forbus, 2011]. Therefore the closeness of relations should be constructed with care.

### 2.3 Hybrid Reasoning

By satisfying qualitative constraints, pure QR techniques can efficiently deal with imprecise and incomplete data. In the real world, however, the nature of certain types of quantitative information make them difficult to benefit from qualitative representations. For example, "877 km (545 mi) NE of Melbourne" describes location of Sydney relative to Melbourne. "877 km" as a quantitative constraint accurately bound the entities in an exact range. In contrast, a qualitative abstraction (e.g. close, far) of such quantitative description does not provide sufficient magnitude of precision for reasoning purpose, whilst "NE" is accurate enough for describing directional relation in this case. A combination of both qualitative and quantitative calculus is intuitive and more appropriate for reasoning with such constraints. In order to apply of QR approaches to more realistic domains, hybrid reasoning which aims to integrate

1https://en.wikipedia.org/wiki/Sydney
Background and Related Work

qualitative and quantitative reasoning methods is attracting incremental attention\(^2\).

In the early days of QR research, the method of reasoning with a combination of qualitative and quantitative information has been applied in several frameworks (e.g. \cite{Forbus1980, deKleer1989}) as some quantitative properties of entities were believed to be of nonsense to translate to qualitative representation and it was argued that purely qualitative reasoning framework did not exist \cite{Forbus1991}. Then in the last decades, most theoretical research on knowledge representation and reasoning has adopted purely qualitative approaches which extensively extended the modelling power of QR methods. As a result, there has been several successful pure QR systems being applied in relatively narrow domains. Although pure QR approaches usually provide correct results, the results sometimes do not satisfy the requirement of certain applications which need a higher accuracy or cannot guarantee a necessarily closed domain. \cite{Dutta1991} proposed a tractable hybrid spatial reasoning model where the quantitative constraints are expressed with fuzzy logic. In addition, research from cognitive perspective suggests that there is likely to be a simulation-like schema in humans’ physical reasoning procedure \cite{Craik1943} which indicates that combining simulation with QSR is worth an attempt. \cite{Johnston2008} combined slick simulator and tableaux reasoning; nevertheless simulation plays a dominant role in the model. In a more recent work, \cite{Welschehold2016} proposed a model to learn manipulation actions from human demonstration by combining quantitative optimization and qualitative matching of motion trajectories of the hand and the object.

2.4 High Level Perception and Scene Understanding

High level perception systems that can infer implicit properties of entities and their relations is required by AI agents that are supposed to work in unknown environment. With the availability of affordable RGB-D cameras such as Kinect, machines are able to sense their surroundings with depth information which is similar to humans’ way of perceiving the world. Modelling scenes from RGB-D data attracts increasing attention nowadays. Random sample consensus (RANSAC) \cite{Fischler1981} is widely used for detecting 3d shapes from RGB-D images. Dense RGB-D scans of a scene can be reconstructed with a 3D version of simultaneous localization and mapping (SLAM) algorithm \cite{Salas-Moreno2013} by matching key descriptors (such as SHOT \cite{Tombari2010}, SIFT \cite{Lowe2002}, SURF \cite{Bay2008} and ORB \cite{Rublee2012}) detected from adjacent scans. However, for sparse scans, SLAM algorithms will occasionally fail as less key descriptors could be found. In order to deal with the incompleteness of the sensory data, inferring the hidden parts of the scene is essential. \cite{Ge2016} adopts a pure qualitative 2D stability analysis method to detect unknown objects in the world of the video game “Angry Birds”. \cite{Shao2014} proposed an algorithm to recover the occluded parts of a RGB-D image by assuming shapes and symmetry of the entities.

High level perception methods are based on the results of the above mentioned procedures and aims to provide knowledge about the scene from a higher level by reasoning about implicit relations between the detected entities. The support relation

\(^2\)https://www.hybrid-reasoning.org
as one of the most important physical relations is studied from different perspectives.

Silberman et al. [2012] proposed a framework for understanding indoor scenes from segmentation to support relation inference. With respect to support relation detection, first entities are labeled with different categories such as furniture and ground. Then a statistical model is trained to predict the likelihood of one object to be a supporter of the other object. This method is adopted and improved by [Zhuo et al., 2017; Panda et al., 2016] for more accurate support relation inference. These statistical methods are able to detect most obvious support relations that commonly appear in the real world. However, they can barely figure out some special but important support cases. For example, in Figure 2.1 the top object A also contributes to the stability of its underneath object B.

From a different perspective, there has been work on detecting support relations by reasoning about mechanical constraints. In [Gupta et al., 2010], objects in an outdoor scene are modeled with a volumetric representation. Taking the ground plane as the base, physical and geometric properties of objects are interpreted by validating a stable configuration of the objects. Mojtahedzadeh et al. [2013] qualitatively model the contact relation of cuboid-shaped objects and approximate contact points between them. Then, static equilibrium is used for a stability check. By explicitly assuming the shape of the objects, the contact points and the volume of the objects are approximated. As a result, this method cannot deal with complex shapes with an straightforward extension.

In addition, more detailed information about support relations cannot be extracted based on the statistical methods, as little or no mechanical analysis is involved in those methods. In [Ge et al., 2016], the concept “core supporter” is introduced. If the status of an object changed from stable to unstable after removing one of its connected objects, the removed object is called a “core supporter” of the unstable object. For example, in Figure 2.2 object B is a core supporter of A, but neither C nor D is core supporter of A. For the methods based on mechanical analysis, “core supporter” is
Background and Related Work

Figure 2.2: Example of Core Supporter. B is a core supporter of A, but neither C nor D is a core supporter of A.

not addressed as well. In Chapter 6 the detection of core supporter and the above mentioned problems are addressed in the proposed algorithms.
Chapter 3

Spatial Knowledge Representation of Raw Sensory Data

The primary task for perception systems is to represent raw sensory input data in the form that potential interesting information is retained and abstracted. In this chapter, we present a method for extracting detailed spatial information from sensor measurements of regions. We analyse how different sparse sensor measurements can be integrated and what spatial information can be extracted from sensor measurements. Different from previous approaches to qualitative spatial reasoning, our method allows us to obtain detailed information about the internal structure of regions. The result has practical implications, for example, in disaster management scenarios, which include identifying the safe zones in bushfire and flood regions.

3.1 Introduction

There is the need for an intelligent system that integrates different information such as sensor measurements and other observations with expert knowledge and uses artificial intelligence to make the correct inferences or to give the correct warnings and recommendations. In order to make such an intelligent system work, we need to monitor areas of interest, integrate all the available knowledge about these areas, and infer their current state and how they will evolve. Depending on the application, such an area of interest (we call it a spatial region or just a region) could be a region of heavy rainfall, a flooded region, a region of extreme winds or extreme temperature, a region with an active bushfire, a contaminated region, or other regions of importance.

Considering such a region as a big blob or as a collection of pixels on a screen is not adequate, because these regions usually have a complex internal structure. A bushfire region for example, might consist of several areas with an active fire, several areas where the fire has passed already but are still hot, several areas that are safe, or several areas where there has been no fire yet, etc. These areas form the components of the bushfire region. Knowing such detailed internal structure of a region can help predict how the region evolves, as well as for determining countermeasures to actively influence how a region evolves.

In this chapter, we present methods for

- integrating measurements of spatial regions from different sensor networks to obtain more accurate information about the regions;
- extracting spatial information from the integrated sensor measurements in the form of a symbolic representation.

Having a symbolic representation of the internal structure instead of a collection of pixels on a screen allows us to answer queries, derive new knowledge, identify patterns, or to express conditions and rules, for example, when a warning must be issued. It also allows us to obtain an accurate approximation in the first place: by being able to represent the internal structure of a spatial region, we can build up a partial representation of the internal structure of a monitored region, track the region over time, and refine the partial representation whenever we obtain new sensor measurements or other new information.

The remainder of this chapter is organised as follows: in Section 3.2 we motivate and formally define the problem in question. This is followed by a review of related literature in Section 3.3. In Section 3.4 we describe the methods for matching sensor measurements and for extracting spatial information and evaluate them in Section 3.5. We conclude the chapter in Section 3.6.

### 3.2 Our Approach

The problem we are trying to solve, i.e. extracting accurate spatial information about the internal structure of a region using sensor measurements, is a very general problem with many facets. It depends on (1) what assumptions we make about the regions, (2) what assumptions we make about the sensors, and (3) what kind of spatial information we are considering. We will make some restrictions on all three points to see if the desired outcome (i.e. accurate spatial information) can be achieved at all. In future work we plan to lift these restrictions and to allow greater generality. The restrictions and assumptions we make in this chapter are the following:

**Regions:** A complex region (cf. [Li, 2010]) \( R \) is a two-dimensional spatial entity in the plane that can have a finite number of components. Each component can have arbitrary shape, extent and location and can have a finite number of holes of arbitrary shape, extent and location. Each hole can recursively contain a complex region which is part of the original complex region. We say that each component, independent of whether it is inside a hole or not, is a positive component of \( R \), while each hole and each bounded or unbounded area of the plane that is not a positive component is a negative component of \( R \). See Figure 3.1a for an example of such a region.

**Restrictions on regions:** We assume that a given complex region is rigid, i.e. does not change shape, extent or relative location of any of its components. We allow regions to move, to change speed, direction of movement and their orientation. We further assume that during a sensor measurement, regions have constant speed, direction and orientation.

**Sensors:** A sensor \( s \) is a point in the plane that can measure whether it is inside a region or outside a region at time \( t \), that is it senses 1 if \( s \) is in a positive component of \( R \) and 0 if \( s \) is in a negative component of \( R \). As a practical example, consider a sensor that measures (absence of) direct sunlight. If a positive component of a region is between the sun and the sensor it measures 1, otherwise 0. We consider sets \( S \) of sensors \( s_i \) located on arbitrary but known locations on a plane (see Figure 3.1b).
§3.2 Our Approach

(a) A complex region consisting of positive components $r_1, r_2$ and negative components $r_0, r_3, r_4, r_5$.

(b) A complex region moving over a set of arbitrarily placed sensors on the plane.

(c) Normalised sensor measurement of a complex region. Thick lines represent positive intervals and thin lines negative intervals.

(d) Normalised sensor measurement of the same region using a different set of sensors.

(e) Integrating two measurements reveals partial information about which positive intervals belong to the same components.

Figure 3.1: A complex region measured by two sets of sensors.

Restrictions on sensors: We assume that sensors are static and do not change their locations. Sensor plane and region plane can be different, but must be parallel.

Sensor measurements: A single sensor measurement (in short: a measurement) $M$ occurs when a complex region $R$ moves over a set $\{s_1, \ldots, s_n\}$ of sensors. Given the above assumptions and restrictions on regions and sensors, each measurement $M$ corresponds to a set $\{\ell_1, \ldots, \ell_n\}$ of parallel directed lines, where each line $\ell_i$ corresponds to one sensor $s_i$. The lines (more precisely line segments) start at the beginning of the measurement $M$ and end at the end of the measurement. Each line may contain zero or more alternating positive and negative intervals, where each positive interval corresponds to the time when the sensor was in a positive component of $R$ and each negative interval corresponds to when the sensor was outside of $R$ (see Figure 3.1c). We assume that each measurement covers all positive components of the region. Note that the ordering of the parallel lines and the distances between lines depend on the direction of movement of the measured region and the actual location of the sensors on the plane relative to this direction.

A normalised measurement $M$ is a measurement where all sensors are located on a line orthogonal to the direction of movement of the measured region, with arbitrary
Spatial Knowledge Representation of Raw Sensory Data

distance between sensors. Any measurement can be converted to its corresponding normalised measurement \( M \) if the direction of movement of the measured region and the actual sensor locations are known. This can be achieved by the following steps: (1) select a line orthogonal to the direction of movement such that all sensors are on the side of the line where the region is coming from, (2) project all sensors orthogonally on this line, and (3) subtract the distance between the line and each sensor \( s_i \) at the beginning of each \( \ell_i \).

Restrictions on sensor measurements: We assume that the direction of movement of regions is known, which can be easily obtained if there is at least one sensor consisting of multiple pixels. Then we can always obtain a normalised measurement. Note that this is independent of the orientation of a region, i.e. how it is rotated, which is unknown. We further assume that different measurements of the same region are independent, in particular that distances between sensors of a normalised measurement vary. This could result from different sets of sensors or from a different direction of movement.

Consequences of our assumptions and restrictions: Our assumptions are relatively general, the strongest restriction we make is that complex regions are rigid. Except for direction of movement and location of sensors, everything else is unknown and unrestricted. Given two independent normalised measurements, these measurements correspond to two sets of parallel lines that cut through the measured region at independent and unknown angles \( \delta \) (due to unknown orientation of regions). Depending on the speed during a measurement, regions appear stretched/skewed along the direction of movement. The slower the movement, the more stretched is the appearance of the region as each sensor stays within the region longer. In addition, due to the unknown distance between sensor plane and region plane, the relative scale of each measurement is also unknown. In the example of a direct sunlight sensor, regions closer to the sensors appear larger in the measurement. Note that under our assumptions, it is interchangeable to have moving regions and static sensors, or moving sensors and static regions.

Spatial information: The spatial information we are considering is the topology of \( R \) (for a complex region this refers to the number of components and holes and their hierarchy; cf. [Li, 2010]), the relative size of the components of \( R \), the relative direction and the relative distance between components of \( R \).

Problem description and motivation: In the following, we take as input two different measurements \( M \) and \( M' \) as sets of parallel lines obtained in the way described above. The distance between the parallel lines is arbitrary and it does not matter which sensor corresponds to which line. That means that the only input we have available are the parallel lines and their intervals, we do not know what is happening between these lines. In order to obtain accurate spatial information about \( R \) we need to be able to infer this missing information. This will be achieved by integrating the different measurements through reconstructing the actual relationship between the measurements and the region.

Our approach is motivated by the following observations:

1. Given a single measurement \( M \) where two adjacent parallel lines have overlapping positive intervals \( i_1 \) and \( i_2 \), i.e. there is a time when both sensors are in a
positive component of $R$. Without knowledge about the area between the two adjacent lines, we cannot say for sure whether $i_1$ and $i_2$ refer to the same positive component of $R$ or to two different ones (see Figure 3.1c).

2. Given a second measurement $M'$ that measured $R$ using a different angle $\delta'$ (see Figure 3.1d). Assume there is a positive interval $i_3$ in $M'$ that intersects both $i_1$ and $i_2$, then we know for sure that $i_1$ and $i_2$ and also $i_3$ are part of the same positive component of $R$ (see Figure 3.1e).

It is clear that by integrating two or more measurements we obtain much more accurate information about a region than having one measurement alone. However, obtaining this information is not straightforward as the angle and the scale of the measurement is unknown. This results from our assumption that distance between region and sensors and orientation of the measured region are unknown. Therefore, our first task is to develop a method to identify a translation, rotation, and scaling that allows us to convert one measurement into another such that the intervals can be matched. Once we have obtained a good match, we will then extract detailed spatial information from the integrated sensor information. We then evaluate our method and determine how close the resulting spatial information is to the ground truth and how the quality depends on factors such as sensor density.

### 3.3 Related Work

Although the setting of our problem seems related to classical sensor fusion techniques [Crowley, 1993; LaValle, 2010], no such techniques are known to be adequate to tackle our problem. For example, a sensor fusion method such as the extended Kalman filter requires a state transition model with specific parameters for each individual region and measurements of system inputs for its implementation, which is in practice difficult to obtain. Our approach is generic and does not have any of these requirements. In particular, it does not require a description of region geometry in terms of equations.

In the context of qualitative spatial reasoning [Cohn and Renz, 2008], most research has taken the objects (e.g. regions or relations) of a symbolic reasoning process as given and paid only little attention to the problem of acquiring such information from the environment. In [Santos and Shanahan, 2003] a stereo vision sensor measures 2D depth profile of a 3D environment taken at a particular height. The objects in the environment are represented as peaks of the depth profile graph. The depth peaks from different snapshots are then matched to provide a qualitative interpretation of their transitions. [Inglada and Michel, 2009] use remote sensing to extract topological relations between complex objects. A graphical representation of the relations is built to allow recognising objects using graph-matching techniques. Both mentioned methods, however, are not suitable for capturing detailed internal structures of regions that result from environmental parameters, such as light, sound, chemicals, temperature, pressure or magnetism. Our work makes no assumptions about the sensor type and includes sensor networks based on microelectromechanical system (MEMS) [Gardner et al., 2005] sensors that overcome such deficiencies.

Similar to our work there are approaches that employ a MEMS based sensor network: [Worboys and Duckham, 2006; Duckham, 2013] describe a computational
Spatial Knowledge Representation of Raw Sensory Data

(a) An overlay of two sensor measurements with mismatches that are indicated by circles.
(b) A partition of the measurement from Figure 3.1c, which consist of two clusters \( c_1 \) and \( c_2 \).
(c) A partition of the measurement from Figure 3.1d, which consist of two clusters \( c'_1 \) and \( c'_2 \).
(d) Initial match based on the spatial structures of the clusters. There are 4 mismatches.
(e) Result after fine-tuning with a local search.

Figure 3.2: Integration of sensor measurements based on Algorithm 1

model for tracking topological changes in spatial regions monitored by a sensor network. Jiang and Worboys [2009] go further and provide a systematic formal classification of the possible changes to a region, based on point-set topology and graph homomorphism. All these algorithms, however, assume the regions to be constructed from connected subgraphs of adjacent sensor nodes that detect the region, which do not allow us to reason about the internal structure of the areas between the sensors measurements. By contrast, this chapter considers all aspects of the internal structure of regions and uses it to infer knowledge about the areas between sensor measurements.

Finally, there are interpolation techniques from computer vision [Szeliski, 2010] and spatial statistics [Cressie, 2011], which are not suitable for situations where the sensor network is sparse and only partial information about regions without further statistical information is known.

### 3.4 Closing the Gap in Sensor Measurements

In this section we present a method for integrating two independent sensor measurements of a complex region (Section 3.4.1) which allows us to extract more precise
3.4 Closing the Gap in Sensor Measurements

3.4.1 Integration of Sensor Measurements

For integrating two independent measurements we search for parameter values for translation $\tau$, rotation $\rho$ and scaling $\sigma$ of the second measurement that minimise mismatches with the first measurement. A mismatch occurs if, in the overlay of the sensor measurements, a positive interval from one measurement intersects with a negative interval from another measurement (see Figure 6.2). An uninformed local search in the parameter space will likely lead to local minima. In order to avoid such local minima, we first build clusters in the individual measurements that give us information about spatial structures of the measurements (Figure 3.2b and Figure 3.2c), and exploit this information to obtain a good initial match (Figure 3.2d). Once good initial values for $\tau, \rho, \sigma$ are found, we fine-tune and improve the match by using a local search method that further reduces the number of mismatches between the measurements (Figure 3.2e).

The details of our procedure are given in Algorithm 1. The main function is `IntegrateSensorMeasurements`, which takes two sensor measurements and first determines for each sensor measurement a set of possible partitions (lines 2–3) by calling `GenPartitions`. A partition $P$ of a measurement $M$ consists of clusters, where a cluster is again comprised of positive intervals in $M$ that potentially belong to the same positive component of underlying region $R$ (see Figure 3.2b and Figure 3.2c). To be precise, we say two positive intervals $i_1$ and $i_2$ are neighboured and write $i_1 \sim i_2$, if $i_1$ and $i_2$ are from adjacent sensor lines and temporally overlap. We furthermore say that positive intervals $i_1$ and $i_2$ belong to the same cluster (or to the same equivalence class), if $i_1 \sim^+ i_2$, where $\sim^+$ is the transitive closure of $\sim$. This allows us to obtain a partition (or quotient set) $M/\sim^+$ of the sensor measurement $M$.

Since some of these clusters could actually belong to the same component of $R$ (unless there exists a negative interval in $M$ that clearly separates the clusters), we consider all coarser partitions of $M/\sim^+$ induced by combining clusters that could potentially belong to the same connected component of $R$. This is done by calling `GenCoarserPartitions` (line 16), where two clusters are merged if the distance between the clusters is less than a threshold value. This value controls the number of possible coarser partitions which is at most $2^{|M/\sim^+|}$. Note that it is irrelevant if the detected clusters reflect the positive components of $R$ truthfully, as they are only used to find a good initial match and not for obtaining spatial information.

From collections $L, L'$ of all coarser partitions of measurements $M, M'$ that we obtained in the previous step, we now try to find a pair of partitions from $L \times L'$ that will lead to the best match of $M, M'$. As a heuristic for finding such a pair we rank all pairs $(P, P') \in L \times L'$ according to the structural similarity between $P$ and $P'$ by means of Function `StructDist` (line 4). More precisely, Function `StructDist` measures the structural distance between partitions $P$ and $P'$, for which we consider the relative sizes and relative directions (i.e. angles) of the clusters in each partition. Given two lists of clusters $(c_1, c_2, \ldots, c_k)$ of $P$ and $(c'_1, c'_2, \ldots, c'_l)$ of $P'$ sorted according to their sizes (the lower the index, the bigger the size), we determine angular information of salient clusters (the lower the index, the more salient is the cluster), and compare the
Algorithm 1: Integrating sensor measurements.

1. **Function** \textsc{IntegrateSensorMeasurements}(M, M')
   
   **Input**: Two different measurements M and M'.
   
   **Output**: The merged version of M and M'.

   // initial match
   2. \( L \leftarrow \text{GenPartitions}(M) \)
   3. \( L' \leftarrow \text{GenPartitions}(M') \)
   4. \( S \leftarrow \text{RANK}(L \times L', \text{key} = \text{StructDist}) \)
   5. \( v^* \leftarrow \infty, \ \tau^* \leftarrow 0, \ \rho^* \leftarrow 0, \ \sigma^* \leftarrow 0 \)
   
   **while** \( v^* > \epsilon \) and \( S \neq \emptyset \) **do**
   6. \( (P, P') \leftarrow \text{Pop}(S) \)
   7. \( (v, \tau, \rho, \sigma) \leftarrow \text{GetParameter}(P, P') \)
   8. **if** \( v < v^* \) **then**
   9. \( (v^*, \tau^*, \rho^*, \sigma^*) \leftarrow (v, \tau, \rho, \sigma) \)

   // local search
   10. \( (\tau^*, \rho^*, \sigma^*) \leftarrow \text{MinMismatches}(M, M', \tau^*, \rho^*, \sigma^*) \)
   11. return \( M \cup \text{Rotate} \cdot (\text{Translate}_{\tau^*} \cdot (\text{Scale}_{\sigma^*} \cdot (M'))) \)

12. **Function** \text{GenPartitions}(M)
   
   **Input**: A sensor measurements M.
   
   **Output**: A list \( L \) of partitions of M.

   13. **foreach** positive intervals \( i_1, i_2 \in M \) **do**
   14. \( \text{Set} \ i_1 \sim i_2, \text{if} \ i_1 \text{ and } i_2 \text{ are from adjacent sensors and temporally overlap.} \)
   15. \( L \leftarrow \text{GenCoarserPartitions}(M/\sim^+) \)
   16. return \( L \)

17. **Function** \text{StructDist}(P, P')
   
   **Input**: Two sets \( P, P' \) consisting of clusters.
   
   **Output**: The structural distance between \( P \) and \( P' \).

   18. \( (c_1, c_2, \ldots, c_k) \leftarrow \text{SortSize}(P) \)
   19. \( (c'_1, c'_2, \ldots, c'_k) \leftarrow \text{SortSize}(P') \)
   20. return \[ \sum_{i=1}^{\max\{k,k'\}} -2 \angle c_ic_{i+1}c_{i+2} - \angle c'_{i+1}c'_{i+2} \cdot w_i \]

similarity between \( P \) and \( P' \). This is achieved by the following formula:

\[
\sum_{i=1}^{\max\{k,k'\}} -2 \angle c_ic_{i+1}c_{i+2} - \angle c'_{i+1}c'_{i+2} \cdot w_i \tag{3.1}
\]

where \( w_i \) is a weight given by the maximum of the sizes of \( c_i \) and \( c'_i \). The formula in (3.1) captures the angles between salient clusters and returns a large value, if the angles between salient clusters of the two partitions are dissimilar, and returns a smaller value if the angles between salient clusters are similar.
Algorithm 2: Extracting spatial knowledge.

1 Function ExtractSpatialKnowledge(M)
   Input : Sensor measurement M.
   Output: An augmented containment tree of M.
2 C ← GetConnectedComponents(M)
3 T ← ContainmentTree(C)
4 $T^+$ ← AUGMENT(T)
5 return $T^+$

6 Function GetConnectedComponents(M)
   Input : Sensor measurement M
   Output: List of connected pos. and neg. components
7 foreach positive intervals $i_1, i_2 \in M$ do
8   Set $i_1 \sim i_2$, if $i_1$ intersects $i_2$.
9 foreach negative intervals $i_1, i_2 \in M$ do
10  Set $i_1 \sim i_2$, if $i_1$ intersects $i_2$.
11 return $M/\sim^+$

After ranking the pairs of partitions according to the similarity criteria in formula (3.1), we start with the pair with the highest similarity from the ranked list $S$, and calculate initial parameter values $\tau, \rho, \sigma$ respectively for translation, rotation and scaling. To overcome local minima we choose the pair of partitions with the next highest similarity from $S$ until either the number $v$ of mismatches are below a threshold $\epsilon$ or no more pairs of partitions are left in $S$ (lines 6–10). Obtaining the number $v$ of mismatches and the parameter values from a pair of partitions is achieved by calling function GetParameter (line 8), which compares the two most salient and clearly distinguishable clusters $(c_1, c_2)$ and $(c'_1, c'_2)$ from the partitions and determines the translation, rotation and scale between those pairs. After obtaining the initial parameters, function MINMISMATCHES further reduces the number of mismatches and fine-tunes the result by performing a local search around the initial parameter values (line 11). This is done by means of point set registration, which is a local search technique for matching 2D point sets [Jian and Vemuri, 2011]. As point set registration requires two sets of points as its input, we uniformly sample points from positive sensor intervals of both of the sensor measurements. The final outcome is the union of two sensor measurements, where the second sensor measurement is transformed using the parameters found.

3.4.2 Spatial Knowledge Extraction

After integrating sensor measurements, it is now possible to extract more precise spatial information. For example, from only one sensor measurement it was impossible to extract precise information about the connected components of a complex region, as all positive intervals are disconnected. From the merged sensor measurements, however, it is possible to extract this information, as overlaying different measurements
connects the intervals that were previously disconnected. In what follows we present an algorithm (Algorithm 2) for extracting spatial information from the merged sensor measurements.

The main function ExtractSpatialKnowledge takes as its input a merged sensor measurement and extracts spatial information. The extracted spatial information is represented in the form of a tree, which is called an augmented containment tree, where each node is either a positive component or a negative component and the parent-child relation is given by the containment relation of the components, i.e. a negative (resp. positive) component is a child of a positive component (resp. negative component), if the former is inside the latter. Additionally, for each node we provide spatial relations (directions, distances, and sizes) between its children nodes and between parent and children nodes. Most other relations can be inferred or they can be extracted as well. This allows us to obtain information about the internal structure of any connected component of a complex region. Which spatial relations we extract is actually not relevant anymore as we now have a reasonably fine grained actual representation of the complex region which allows us to obtain spatial relations of most existing spatial representations.

Function ExtractSpatialKnowledge first determines connected components in the sensor measurement (line 2). We not only detect positive components, but also negative components (i.e. holes), as holes play a role in several applications as mentioned in the introduction, e.g. disaster scenarios. We first identify the class of all intervals that are connected by the intersection relation ∼, and obtain the partition M/∼⁺, where ∼⁺ is the transitive closure of ∼ (line 11).

From the connected components we build a tree that reflects the containment relations between the components (line 3). We first detect the root r₀ of the tree, i.e. the negative component in the background which contains all other components. This can be easily done by finding the first negative interval of any sensor measurement and returning the negative component that contains the negative interval. Afterwards, we recursively find the children nodes of a parent node ri, which are components “touching” ri. This too can be realised easily by determining the interval (and the component containing the interval) that is immediately next to an interval of the parent component. We generate the containment tree using breadth-first search.

Once the containment tree is built, we augment the tree with spatial relations from the spatial calculi that are most suitable for a given application. We recursively detect the internal structure of each component by going through every node of the containment tree using breadth-first search and extract spatial information (directions, distances, and sizes) of its children. The final outcome is then an augmented containment tree (see Figure 3.3). Having such a tree, one can answer spatial queries such as “right_of(x, A) ∧ size(x, B)” by finding nodes in the tree that satisfy the queries. These relations can then be used for spatial reasoning in the standard way [Cohn and Renz 2008].

3.5 Evaluation

The purpose of this chapter was to obtain accurate information about the internal structure of a complex region by integrating sensor measurements. In order to evaluate the quality and accuracy of our method, we developed a tool that can
randomly generate arbitrarily complex spatial regions (see Figure 3.4). The generated regions are initially bounded by polygons, but the edges are then smoothened. We also randomly generate a line of sensors (corresponding to a normalised measurement), with random orientation and gap between sensors and random region move-over speed and take measurements for each choice. The regions are 800 × 600 pixels with 8–9 components on average. Notably, the setting of moving sensor and static region works in the same way as the moving region and static sensor setting that is used throughout the thesis.

For our experiments, we selected equidistant sensors, since we want to analyse how the gap between sensors influences the accuracy of the outcome. It does not affect our method. The chosen orientation, scale and speed of the sensor measurement is unknown to our method and will only be used to compare accuracy of the resulting match with the ground truth. We are mainly interested in how good the resulting match is to the correct match. The closer orientation and scale is to the true values, the more accurate the spatial information we can extract, provided the sensors are dense enough. A perfect match (i.e. orientation and scale differences are 0) leads to the best possible information for a given sensor gap. We therefore measure the difference in orientation and scale to the correct values. We also measure the percentage of mismatches of all possible intersections. While orientation and scale differences can only be obtained by comparing to the ground truth, mismatches can be obtained directly. The denser the sensors, the more mismatches are possible.

We created 100 random complex regions and two measurements with random angle per region and per sensor density and applied our method to match the measurements and to extract the spatial relations from the resulting match. We then evaluated the effectiveness of the initial match and the additional fine-tuning step by treating them separately, and also compared the result against the point set registration method from [Jian and Vemuri, 2011], which is used for the local search in our algorithm.

As shown in Table 3.1, the initial matches before the fine-tuning with local search are already close to the correct match and are significantly better than point set registration alone. However, the number of mismatches is as expected fairly large for sparser sensor density (i.e. higher gap size). This is significantly reduced by adding the fine-tuning step and could be further reduced by using more advanced local search heuristics. The point set registration method requires more computational
resources when the sensor network is dense, because there are more points to match. By contrast, the runtime in the initial match increases with growing gap size. This is because sparse sensors lose more information than dense sensors. The possibility for building different combinations of clusters of a measurement increases as the gap size grows, which significantly increases the runtime.

The number of components detected by our algorithm is generally greater than the ground truth. There are two main reasons for this observation: first, some components are only connected by very slim parts which are usually missed by the sensors; second, some mismatches causes some very small components to be detected. However, the difference between the number of components we detected and the ground truth is reasonable and can be improved by integrating further measurements.

### 3.6 Conclusions

In this chapter we presented a sophisticated method for obtaining accurate spatial information about the internal structure of complex regions from two independent relatively sparse sensor measurements. The task was to develop a method that can infer information about what happens in the areas the sensors cannot sense by integrating different independent measurements of the same complex region. The intention was to obtain more information than just the sum of individual measurements, each of which does not contain much accurate information. Furthermore, we wanted to obtain
§3.6 Conclusions

A symbolic representation of the sensed region that allows intelligent processing of the knowledge we obtained, such as query processing, change detection and prediction, or the ability to specifying rules, for example when warnings are issued. One of the benefits of our method is that it is very generic and does not require any complicated state transition models or object parameterisations that are typically used in the area of sensor fusion.

As our evaluation demonstrates, our method can successfully integrate different independent sensor measurements and can obtain accurate spatial information about the sensed complex regions. We make some restrictions about the sensor setup and the complex regions we sense. In the future we plan to lift these restrictions and analyse if we can obtain similar accuracy when using a less restricted sensor setup, such as, for example, using mobile phones as sensors which can follow any path. Allowing non-rigid regions and to detect their changes also seems a practically relevant generalisation of our method. In addition, although the evaluation data used in this chapter is pixel-wised discrete data, the proposed algorithms will also work on real plane, because the input points are treated as sampled points from the real world.

In this chapter we restricted ourselves to the detection of the internal structure of a single complex region. Obviously, this restriction is not necessary and we can easily apply our method to the detection of multiple complex regions and the relationships between the regions. In that case we can modify our sensors so they detect in which regions they are in at which time. Using the same procedure as for an individual complex region, we can obtain an integration of sensor measurements that allows us to infer the relations between multiple regions. What is not as straightforward is to change what sensors can measure, for example, a degree of containment, or continuous values rather than 1/0 values. Integrating such measurements requires significant modifications to our algorithms.

In summary, the method proposed in this chapter demonstrates a concrete example of how to extract high level information from raw sensory data from different sources. It combines both spatial reasoning method to prone large search space and local optimisation technique for further improve the integrated measurements. Once the result is satisfied, high level abstraction of the integrated data can be more accurately extracted. Although the setting of sensors is specific in this chapter, the pipeline is explicit for further developing algorithms of processing sensory data from other sources, which is essential for extracting high level knowledge of high quality in a perception system. The method and pipeline in this chapter focuses on extracting information from a static state of sensory input. In the following chapter, we will discuss how to extract high level information from consecutive sensory input by predicting the changing trend of the input data.
Spatial Change Prediction of Evolving Regions

The capability to predict changes of spatial regions is important for an intelligent system that interacts with the physical world. For example, in a disaster management scenario, predicting potentially endangered areas and inferring safe zones is essential for planning evacuations and countermeasures. Existing approaches usually predict such spatial changes by simulating the physical world based on specific models. Thus, these simulation-based methods will not be able to provide reliable predictions when the scenario is not similar to any of the models in use or when the input parameters are incomplete. In this chapter, we present a prediction approach that overcomes the aforementioned problem by using a more general model and by analysing the trend of the spatial changes. The method is also flexible to adopt to new observations and to adapt its prediction to new situations.

4.1 Introduction

Spatial changes that characterise events and processes in the physical world are ubiquitous in nature. As such, spatial changes have been a fundamental topic in different research areas such as computer vision, knowledge representation and geographic information science.

In this chapter, we are interested in predicting the spatial changes of evolving regions, which is an important capability for decision makers. For example in a disaster management scenario, it is essential to predict potentially endangered regions and to infer safe regions for planning evacuations and countermeasures. A more commercial example are solar power providers who need to estimate solar power output based on predicted cloud coverage.

Existing approaches to predicting changes of spatial regions typically use simulations tailored to a specific application domain, e.g., wildfire progression and urban expansion. The models behind such simulations are based on physical laws that require a detailed understanding of microscopic phenomena of the given application domain and rely on specific domain parameters that need to be known. If not all parameters are known, or if the underlying model is not fully understood, predictions based on simulations might fail.

This shortcoming is particularly problematic in disaster situations, where time
is often the most critical factor. An immediate response can save lives and prevent damage and destruction. An immediate response might not leave enough time to build a domain model, to adjust it to the situation at hand, or to collect and apply the required domain parameters. What is needed is a system that provides fast and reliable predictions, that can be quickly applied to any new domain and new situation, that can adopt to new observations, and that does not require intimate domain knowledge.

We present a general prediction method with all these desired properties. Our method can be applied to any kind of evolving regions in any application domain, as long as the regions follow some basic principles of physics. Our method only requires a sequence of “snapshots” of evolving regions at different time points that allow us to extract the boundaries of the observed regions. This could be, for instance, aerial or satellite images, but also data from sensor networks, or a combination of different sources. Our method then samples random points from the region boundaries of each snapshot and infers the trend of the changes by trying to assign boundary points between snapshots. We developed a hybrid method that combines a probabilistic approach based on a Kalman filter with qualitative spatial reasoning in order to obtain accurate predictions based on the observed trend.

In the following we first present some related work before discussing the specific problem we are solving and the assumptions we are making. We then present our method for inferring the trend of changes and for predicting future changes. We evaluate our method using real world data from two completely different domains, which demonstrates the accuracy of our predictions. We conclude the chapter with a discussion.

### 4.2 Related Work

Reasoning about spatial changes has been a fundamental topic in geographic information science, in knowledge representation as well as in computer vision.

In geographic information science, most prediction methods in the literature are tailored to a specific application area and are not based on trend-analysis of spatial change. For example [Alexandridis et al., 2008; Ghisu et al., 2015] predict wildfire progression and [Santé et al., 2010; Singh et al., 2015] urban expansion based on cellular automata simulations which cannot adopt sensor data to guide the prediction. Rochoux et al. [2014] on the other hand use a data assimilation method (extended Kalman filter) to estimate certain environmental parameters in a wildfire simulator, the parameters including seven biomass fuel properties and the wind. Peng et al. [2014] track cloud movement based on a constant velocity model for forecasting short-term solar irradiance.

From the perspective of a general prediction method, Junghans and Gertz [2010] propose a general method to track evolving regions in disaster scenarios which uses minimum bounding rectangles to represent the tracked regions. The method estimates the bounding rectangles of the future regions by means of linear regression.

In knowledge representation and reasoning, qualitative spatial change [Galton, 2000; 2009] has been identified as a vehicle for commonsense reasoning about space, where predictive reasoning plays an essential role [Bhatt, 2014]. Despite this importance,
research mainly focused on detecting spatial changes \cite{Worboys2006, Jiang2009, Liu2011} but not on predicting them. In computer vision, the active contour model has been used to track and predict regions with smooth and parametrisable shapes \cite{Blake1998}. In this model, region boundaries are approximated by splines and their positions and shapes are tracked by means of a Kalman filter. The active contour model approach is similar to our approach in that the target objects are region boundaries and the Kalman filter is used for tracking and prediction. However, the active contour model is more focussed on tracking splines by using a default shape information of the target region and is not suitable for prediction tasks where no such default shape information is given (e.g. in wildfire or cloud movement prediction).

\section*{4.3 Problem Description and Approach}

Our aim in this chapter is to predict future changes of evolving regions by only considering information about past changes. A region \( R \) at time \( t \) in this context is defined as a set \( R_t \subset \mathbb{R}^2 \) that can consist of multiple connected components with holes. Each hole can recursively contain other connected components of the region. We assume that the interior of regions is homogeneous and that changes of regions are entirely captured by changes to their boundaries. We now describe the problem and the assumptions in more detail.

**Input** As input we use a series of snapshots of evolving regions at different times. Each snapshot must allow us to detect the boundary of the observed regions (see Figure 4.1). Hence, snapshots could be obtained, for example, from remote sensing images, sensor networks (cf. \cite{Duckham2013}), or a combination of different sources. We denote the boundary of a region \( R_t \) as \( \partial R_t \), which is a finite set of closed curves, one for each connected component and one for each hole.

**Trend Analysis** Given these snapshots and the corresponding boundaries, we now try to infer the trend of observed changes. In order to do this, we randomly sample a set of \( k \) boundary points \( z_t \in \partial R_t \) and are interested in the movement of these boundary points over time. To this end, for every pair of consecutive snapshots we assign to each boundary point \( z_t \in \partial R_t \) a boundary point \( z_{t+1} \in \partial R_{t+1} \). This assignment is given by a map \( f_t : \partial R_t \rightarrow \partial R_{t+1} \) that minimises the total distances\(^1\) of the point movements, i.e.

\[
\hat{f}_t = \arg \min_{f} \sum_{z_t \in \partial R_t} \| f(z_t) - z_t \|. \tag{4.1}
\]

The function \( \hat{f}_t \) yields a reliable assignment if each of the boundaries \( \partial R_t \) and \( \partial R_{t+1} \) consist of one closed curve as is the case in Figure 4.1. However, it does not always lead to a meaningful assignment, if their topological structures are more complex. In

\(^1\)The rationale behind equation (4.1) is that physical entities seek to move as efficiently as possible, which is also known in physics as the principle of least action \cite{Feynman1965}.\)
Section 4.4.1 we will resolve this issue by analysing the topological structures of the boundaries and adjusting the assignment accordingly.

Once we have $f_t$, we can repeatedly apply $f_t$ to $\partial R_t$ and construct a sequence $z_0, z_1, \ldots, z_T$ of observed points from region boundaries $\partial R_0, \partial R_1, \ldots, \partial R_T$. This sequence is, however, only an approximation of the true movement of boundary points $p_t$ and the locations of $z_t$ are therefore subject to noise

$$z_t = p_t + b_t,$$

where $b_t$ is a zero mean Gaussian noise. We assume that this noise is negatively correlated with the density $D$ of sampled boundary points in a given area, i.e. the noise decreases with increasing $D$.

**Prediction Model** The main assumption we make about the change of region boundaries is the following: *If there is no external cause, any point $p$ of a region boundary $\partial R \subset \mathbb{R}$ moves with a constant speed.* We assume that there is always a noise in the model owing to the ignorance of physical parameters such as acceleration. Formally, for each
4.4 Trend-Based Prediction Algorithm

In this section we present the prediction algorithm in detail. We first describe how the trend of a change is identified from two consecutive snapshots of a region boundary. Then we describe how this information is used to predict changes.

4.4.1 Qualitative Adjustment of Trend

Equation (4.1) in Section 4.3 is an instance of the assignment problem that can be solved by the Hungarian method [Kuhn, 1955] in $O(n^3)$ time. However, as mentioned in Section 4.3, equation (4.1) does not always lead to meaningful assignments, particularly if the topology of the boundaries is complex. An example is illustrated in Figure 4.2. In this example region boundary $\partial R_t$ consists of four closed curves $C^1_t, C^2_t, C^3_t, C^4_t$, where $C^1_t$ contains $C^2_t$ and $C^3_t$ contains $C^4_t$ at time $t = 0$ (Figure 4.2a). At time $t = 1$ the topological structure of region boundary $\partial R_t$ remains the same, but each of the
Spatial Change Prediction of Evolving Regions

(a) An overlay of $R_0$ and $R_1$. Assignments are without adjustment.

(b) An overlay of $R_0$ and $R_1$. Assignments are after adjustment.

Figure 4.3: An example of the trend identification between evolving regions. The boundary points are indicated by the red dots. Each individual assignment between two points $z_t$ and $z_{t+1}$ is highlighted by a blue line.

curves $C_1^t$, $C_2^t$, $C_3^t$, $C_4^t$ has either shrunken or expanded (Figure 4.2b). As a result a one-to-one assignment of points in $C_1^t$, $C_2^t$, $C_3^t$, $C_4^t$ to their counterparts in $C_1^{t+1}$, $C_2^{t+1}$, $C_3^{t+1}$, $C_4^{t+1}$ is not possible, leading to false assignments of boundary points (Figure 4.3a).

Our method solves this problem qualitatively by analysing the topological relations between closed curves of the boundaries, and by ensuring that all assignments will only be between corresponding curves.

To this end we first construct a containment tree $G_{R_t}$ that represents the topological relations between closed curves of $\partial R_t$. In this tree, each curve $C$ is represented by a vertex $v_C$ and the background is represented by the root. A vertex $v_C$ is a parent of $v_{C'}$, if $C$ contains $C'$ and no other curve $C''$ lies in between, i.e. there is no $C''$ that is contained by $C$ and at the same time contains $C'$.

Then, we compute the initial assignment $f_t$ between $\partial R_t$ and $\partial R_{t+1}$ by solving the assignment problem in equation (4.1), and represent the assignment graphically using the containment trees $G_{R_t}$ and $G_{R_{t+1}}$ (Figure 4.4). We say that there is an assignment between a vertex $v_{C_t}$ and $v_{C_{t+1}}$ when there is an assignment between the underlying curves $C_t$ and $C_{t+1}$, i.e. $f_t(C_t) \cap C_{t+1} \neq \emptyset$. We then check whether there are any conflicts or ambiguities in the assignment by traversing the containment tree $G_{R_t}$ in breadth-first order and examining its vertices. Given an assignment $f_t$ between $v_{C_t}$ and $v_{C_{t+1}}$

- a conflict arises when $v_{C_t}$ and $v_{C_{t+1}}$ are not at the same level in the containment tree, or when there is no assignment between their parents. The conflict is resolved by preventing such an assignment;

- an ambiguity arises when there are assignments between $v_{C_t}$ and $v_{C_{t+1}}$ and between $v_{C_t}$ and $v_{C'_{t+1}}$ with $C'_{t+1} \neq C_{t+1}$. Such an ambiguity is natural, if a topological change (e.g. merge, split) occurs between the transition from $R_t$ to
Figure 4.4: A graphical representation of the assignment between $\partial R_0$ and $\partial R_1$. Each line with an arrow is an assignment before adjustment. Each dashed line stands for an assignment that violates a certain rule. We resolve these violations in breadth-first order.

$R_{t+1}$ and is sometimes hard to resolve. Ambiguities allow us to classify and to explain topological changes (cf. Table 4.1).

After identifying all conflicts and ambiguities, we recalculate the assignment $f_t$ between $R_t$ and $R_{t+1}$ while preventing assignments between conflicting curves. This is obtained by setting the distances between boundary points of conflicting curves so high that the Hungarian algorithm will avoid these assignments. Note that we specifically designed our method in this way as the correspondence of nodes between the two containment trees is not necessarily known.

4.4.2 Boundary Point Prediction

Once we obtained the trend analysis $z_0, \ldots, z_T$ of each boundary point $p_t$, we can then estimate its current location as well as predict its future location. To this end, we apply the Kalman filter [Kalman, 1960], which is a data fusion method that has been widely used for guidance, navigation, and control of vehicles and for object tracking in computer vision. The Kalman filter is an ideal choice as it is the optimal estimator that satisfies previous assumptions. For more details about Kalman Filter see [Kalman, 1960] or [Russell and Norvig, 2003a, Section 15.4].

The Kalman filter requires a state transition model as well as an observation model. The state transition model describes how the state of a system at time $t$ evolves linearly to a state at time $t+1$. Our state transition model is given in equation (4.3) and the observation model, which describes the observation (or measurement) $z_t$ of the true location $p_t$, is given in equation (4.2). The detailed integration and prediction procedure is implemented in Algorithm 3. In line 8 of the algorithm, $Q_t$ is the covariance of the noise $a_t$ in the state transition model, where $I$ is the $2 \times 2$ identity matrix.
Algorithm 3: Boundary point prediction

Input: A sequence \( z_0, z_1, \ldots, z_T \) of observed boundary points and \( \Delta t \in \mathbb{N} \).
Output: The location of \( p_{T+\Delta t} \) and the covariance \( \Sigma_{T+\Delta t} \) of its noise.

1. \( p_0 \leftarrow z_0, \quad p_1 \leftarrow z_1, \quad \Sigma_1 \leftarrow I \)  // initialise
2. for \( t \leftarrow 1 \) to \( T \) do  // Update current state
   3. \( (p_{t+1}, \Sigma_{t+1}) \leftarrow \text{Predict}(p_t-1, p_t, 1, \Sigma_t) \)
   4. \( (p_{t+1}, \Sigma_{t+1}) \leftarrow \text{Update}(p_{t+1}, \Sigma_{t+1}, z_{t+1}) \)
5. return \( \text{Predict}(p_{T-1}, p_T, \Delta t, \Sigma_T) \)  // \( p_{T+\Delta t}, \Sigma_{T+\Delta t} \)

Function \( \text{Predict}(p_{t-1}, p_t, \Delta t, \Sigma_t) \)

7. \( v_t \leftarrow p_t - p_{t-1} \)
8. \( Q_t \leftarrow \Delta t \cdot \|v_t\| \cdot \|z_t - p_t\| \cdot I \)
9. \( p_{t+1} \leftarrow p_t + \Delta t \cdot v_t \)
10. \( \Sigma_{t+1} \leftarrow \Sigma_t + Q_t \)
11. return \( p_{t+1}, \Sigma_{t+1} \)

Function \( \text{Update}(p_t, \Sigma_t, z_t) \)

12. \( D \leftarrow r / \sqrt{|B_r(z_t) \cap \partial R_t|} \)
13. \( S_t \leftarrow D \cdot I \)
14. \( K_t \leftarrow \Sigma_t (\Sigma_t + S_t)^{-1} \)  // Kalman gain
15. \( p_t \leftarrow p_t + K_t(z_t - p_t) \)
16. \( \Sigma_t \leftarrow \Sigma_t - K_t \Sigma_t \)
17. return \( p_t, \Sigma_t \)

matrix. The term \( \|z_t - p_t\| \) measures the agreement between the prediction model and the observed boundary points as addressed in Section 4.3. In line 14, \( S_t \) is the covariance of the noise \( b_t \) in the observation model, where \( D \) reflects the density of boundary points in a neighbourhood of \( z_t \) and is defined as \( D = r / \sqrt{|B_r(z_t) \cap \partial R_t|} \), where \( B_r(z_t) \) is an open disk with center \( z_t \) and radius \( r > 0 \).

4.4.3 Outer Approximation and Qualitative Prediction

Given the predicted boundary points \( \partial R_{T+\Delta t} \) whose locations have Gaussian noise, we can now approximate \( \partial R_{T+\Delta t} \) by building an outer boundary that most likely contains the region at time \( T + \Delta t \). We call this the outer approximation of a \( \partial R_{T+\Delta t} \). Using our method we are also able to predict certain topological changes that might occur, as is described below.

Outer approximation To outer-approximate \( R_{T+\Delta t} \) we need to adapt the outer-approximation to the noise of each predicted boundary point of \( \partial R_{T+\Delta t} \), i.e. if the noise has a large covariance then its outer-approximation has to be large and vice
Table 4.1: Six basic topological changes. The changes are given both pictorially and with containment trees. The left containment tree is before a change, and the right containment tree is after the change. The region boundaries where the change takes place are coloured in red.

<table>
<thead>
<tr>
<th>Appearance</th>
<th>Merge</th>
<th>Self-merge</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Appearance" /></td>
<td><img src="image2" alt="Merge" /></td>
<td><img src="image3" alt="Self-merge" /></td>
</tr>
</tbody>
</table>

versa. For this purpose we use the Mahalanobis distance

\[ D_M(p) = \sqrt{(p - p_{T+\Delta t})^T \Sigma_{T+\Delta t}^{-1} (p - p_{T+\Delta t})} \]

which measures how far a point \( p \) is away from a boundary point \( p_{T+\Delta t} \) relative to its covariance \( \Sigma_{T+\Delta t} \), and build an \( \alpha \)-concave hull of \( \partial R_{T+\Delta t} \) in a way that the hull does not intersect points with \( D_M(p) \leq d \) for a radius \( d \). Thus the threshold value \( d \) controls the size of the outer approximated area and can be varied to suit different prediction requirements.

An admissible concave hull should enclose the entire region while including as few points that are outside the region as possible. We use [Edelsbrunner et al., 1983] to obtain an admissible \( \alpha \)-concave hull. The outer approximation of a region \( R_t \) is denoted as \( \alpha(R_t) \)

**Qualitative Prediction** The method predicts topological changes of the region during the transition from \( R_T \) to \( R_{T+1} \) by detecting changes of their containment trees. We detect 6 types of topological changes according to [Jiang et al., 2009], namely, appear, disappear, merge, split, self-merge and self-split (Table 4.1).

**Additional Layer Information Integration** Our method is able to integrate additional information. Specifically, due to the nature of Kalman filter, we can trace the path along each predicted point and its previous states. This allows us to adjust the resulting points based on domain-specific models and their previous states. Integrating different models may require different methods. For example, during a bushfire, isolation areas such as a highway or water body can prevent fire from burning across. Here we show the integration of isolation areas in a bushfire as an example. We assume fire cannot burn on or jump over isolation areas but has to extend along
the edge. Then the predicted points will be filtered out under the following two circumstances:

1. if \( p_t \) is on the isolation area
2. if either of the path between \( p_t \) and \( p_{t-1} \) and between \( p_{t-1} \) and \( p_{t-2} \) intersects with the isolation areas.

While this is intuitive, it demonstrates the ability of our method to integrate with domain specific models.

4.5 Evaluation

We now evaluate our proposed method using real-world data from two different domains as well as generated data. We built our own simulator for generating test data as that allows us to generate data that is more complicated and has more erratic changes than what we saw in the real data. For each evaluation, our method takes a sequence \( R_{T-w+1}, \ldots, R_T \) of \( w \) recent snapshots to predict the outer approximation of regions at \( T+1 \). For our analysis we varied both \( w \) and the threshold \( d \) as defined in Section 4.3. We measure the accuracy of the predicted outer approximation \( \alpha(R_t) \) using the metrics of Recall (\( R \)) and Precision (\( P \)):

\[
R = \frac{|\alpha(R_t) \cap R_t|}{|R_t|}, \quad P = \frac{|\alpha(R_t) \cap R_t|}{|\alpha(R_t)|}
\]

where \( |R_t| \) refers to the size of the area enclosed by \( R_t \).

Given our outer approximation task, recall is more important than precision because we care more about how our method can cover the region rather than if the prediction points are true positive. However, precision should stay in a reasonable range, as a prediction that covers the full map is useless.

4.5.1 Experimental Results on Real-world Data

We applied the method to two different spatial domains, namely wild fire progression and also cloud movement, where it is more likely that sudden changes or topological changes of a region occur. We use a real-world data set for each domain and the average precision and recall under different settings are summarized in Table 4.2.

Wild Fire Progression: From the data base of the USDA forest service\(^2\), we obtained a sequence of progression maps of a real wild fire. Each map depicts the boundary of the wild fire on a certain date. The time gap between consecutive maps varied between 1 and 15 days. As our method is able to take only two snapshots as input for prediction, data with different time gap between adjacent snapshots can be dealt with. There have been several sudden changes in the boundary of the fire and the method can successfully deal with these changes. Figure 4.5 shows an example where

\(^2\)http://www.fs.fed.us/nwacfire/ball/
Figure 4.5: Prediction and outer approximation of $R_3$ based on $R_0, R_1, R_2$ from Figure 4.1. The red dots are the ground truth in Figure 4.1d. It shows that the approximated boundary (shown in dark blue) based on $d = 2.5$ covers most of the actual boundary points. The one (shown in light blue) based on $d = 5$ covers all the actual boundary points.

the method predicts the fire boundary based on the snapshots displayed in Figure 4.1. Although there is a significant non-linear spatial change from $R_2$ to $R_3$, the method is still able to overcome it with a reliable outer approximation.

There are several reasons why we could not compare our approach with other wild fire prediction methods. One problem was that fire maps used to test these approaches were not publicly available, relevant papers only contain a few images of the data that was used. Another problem is that these existing tools require a detailed specification of all kinds of parameters about fuel and weather, which was not available to us. While this made it impossible for us to compare our method, it clearly demonstrates one major advantage of our method, namely that it can be applied to any types of changing regions without having to know any domain parameters.

Cloud Movement We also obtained video data of moving clouds that is recorded using a fisheye lens (see Wood-Bradley et al., 2012 for detail). Predicting cloud movement is particularly interesting for estimating power output of solar power plants. Before using the data, we had to flatten all images to reflect the actual shapes and movement of clouds. Unlike the wild fire case, the clouds movement is almost linear.

However, the problem is still challenging because the topological structure is very complex in such a data set. Therefore adjusting the assignment of matching points with qualitative information is useful here. We compared the results with and without
Spatial Change Prediction of Evolving Regions

(a) The prediction before qualitative adjustment. We can see that it incorrectly matches two holes into one.

(b) The result after adjustment. The two holes can be detected separately.

(c) A part of the cloud picture that contains the two holes.

Figure 4.6: Comparison between prediction results with and without qualitative adjustment.

The result shows that after adjusting the point assignment, our algorithm performs better on detecting the inner structure of the region (Figure 4.6). Moreover, both the recall and precision are generally better than the result without adjustment.

In the end, we compared our method with a state-of-the-art method developed in [Junghans and Gertz, 2010] described in Section 4.2. The results show that our method outperforms their method on all the data sets (see Table 4.2 for a summary).

4.5.2 Experimental Results on Generated Data

We applied the proposed method to a simulated environment of forest fire where fire regions undergo extensive change. The results confirm that the method can correctly identify the spatial changes of multiple evolving regions in an environment with unknown dynamics.

The underlying structure of the simulator is a two-dimensional cellular automaton (CA). It has been shown that CA are expressive enough to simulate highly complex phenomena from simple rules. Each cell has two properties: fuel and heat consumption. The amount of fuel determines how long the cell will remain on fire and the amount of heat consumption determines how long it takes for a cell of forest to become a cell of fire. The CA model simulates a global wind that has the eight cardinal directions. The wind speeds up the progression of the fire along its direction and slows down the progression in the other directions.

We evaluate the method in scenarios where the global wind randomly changes its direction and magnitude every $\Delta t$ time steps. Therefore, there can be sudden changes in the wind’s direction (e.g., a north wind becomes a south wind) or magnitude (e.g., a strong wind suddenly disappears). We use this setting to test the capability of the method in handling uncertainty within a highly dynamic system.

4.5.3 Discussion

The results in Table 4.2 show that our method ($M_1$) performed better than the state-of-the-art method ($M_2$) on all the data sets. Table 4.2 shows the average recall and precision obtained using $M_1$ with different pairs of $w$ and $d$ on each data set.

<table>
<thead>
<tr>
<th></th>
<th>Fire</th>
<th></th>
<th>Cloud</th>
<th></th>
<th>Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$d$</td>
<td>$R$</td>
<td>$P$</td>
<td></td>
<td>$R$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w = 2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_2$</td>
<td>–</td>
<td>0.80</td>
<td>0.62</td>
<td>0.84</td>
<td>0.4</td>
</tr>
<tr>
<td>$M_1$</td>
<td>0.5</td>
<td>0.86</td>
<td>0.83</td>
<td>0.98</td>
<td>0.55</td>
</tr>
<tr>
<td>$M_1$</td>
<td>1.5</td>
<td>0.93</td>
<td>0.73</td>
<td>0.98</td>
<td>0.47</td>
</tr>
<tr>
<td>$M_1$</td>
<td>2.5</td>
<td>0.97</td>
<td>0.56</td>
<td>0.98</td>
<td>0.44</td>
</tr>
<tr>
<td>$w = 3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_2$</td>
<td>–</td>
<td>0.87</td>
<td>0.61</td>
<td>0.85</td>
<td>0.35</td>
</tr>
<tr>
<td>$M_1$</td>
<td>0.5</td>
<td>0.88</td>
<td>0.76</td>
<td>0.97</td>
<td>0.54</td>
</tr>
<tr>
<td>$M_1$</td>
<td>1.5</td>
<td>0.94</td>
<td>0.66</td>
<td>0.97</td>
<td>0.45</td>
</tr>
<tr>
<td>$M_1$</td>
<td>2.5</td>
<td>0.98</td>
<td>0.52</td>
<td>0.98</td>
<td>0.41</td>
</tr>
<tr>
<td>$w = 4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_2$</td>
<td>–</td>
<td>0.89</td>
<td>0.61</td>
<td>0.87</td>
<td>0.31</td>
</tr>
<tr>
<td>$M_1$</td>
<td>0.5</td>
<td>0.85</td>
<td>0.76</td>
<td>0.98</td>
<td>0.43</td>
</tr>
<tr>
<td>$M_1$</td>
<td>1.5</td>
<td>0.94</td>
<td>0.62</td>
<td>0.97</td>
<td>0.39</td>
</tr>
<tr>
<td>$M_1$</td>
<td>2.5</td>
<td>0.98</td>
<td>0.47</td>
<td>0.97</td>
<td>0.37</td>
</tr>
</tbody>
</table>

$M_1$ achieved an average recall of more than 90% on all the data sets while $M_2$ obtained an average of 83%. $M_2$ approximates a region using the oriented minimum bounding rectangle. Compared with the proposed method, this outer-approximation tends to include more false positive points. Besides, $M_2$ is not able to handle holes and sudden spatial changes such as disappearances of regions. Therefore, the precision rate of $M_2$ is much lower than $M_1$.

The results clearly reflect the relationship between $d$ and the recall as well as the precision rate. Because a larger $d$ results in a larger outer approximation, the recall increases and the precision decreases when $d$ increases. The result also indicates that running $M_1$ with a larger window size ($w$) may lower down the precision. Since the regions in those data sets underwent strong spatial changes over time, a larger window size introduced more uncertainties in the point prediction. As a result, the covariance of the prediction has been made larger by the method to accommodate these uncertainties, which eventually relaxed the outer approximation.

The precision rate obtained on the cloud movement data set is significantly lower than the one obtained on the wild fire data set. This is mainly because the cloud data contains many small connected components and holes. The concave hull of the outer approximation may merge a hole with a connected component if they are too close. However, the recall rate is generally higher than the wild fire data set. This is because the cloud movement can be considered as linear in most cases if the wind is not strong, which fits our assumption well.
4.6 Conclusions

We developed a general method that is able to predict future changes of evolving regions purely based on analysing past changes. The method is a hybrid method that integrates both a probabilistic approach based on the Kalman filter and a qualitative approach based on the topological structure of a region. It first identifies the trend of changes using qualitative analysis, and then predicts the changes with the Kalman filter. It does not need detailed a parameter setting as is required by existing domain specific prediction tools. We evaluated our method using both real world data from two different application domains and also by using generated data that allowed us to test our method under more difficult scenarios.

The main motivation of developing a general prediction method is the ability to use it in situations where predictions are required without having detailed knowledge about a domain or about the particular parameter setting of a specific situation at hand. This is particularly useful in disaster scenarios where time is the most critical factor and where fast and reasonably accurate predictions can save lives and prevent further destruction. As our experimental evaluation demonstrates, our general method is very good at providing accurate predictions at different application domains and thus can be used as the method of choice in time critical situations and in situations where detailed knowledge about a given scenario is not available. However, any available domain parameters can be integrated into our method as an additional layer.

Such model-free prediction method of the perception system contributes to the generality of the entire AI agent while exploring in an unknown environment meanwhile providing the perception system with the capability of analysing input data sequence from a holistic view rather than treating different states individually. This feature is especially significant for agents working in an non-fully observed environment. For example, a robot with this capability may be able to predict potential collision with a moving object and move away from the object moving trend direction in advance. Chapter 3 and this chapter have discussed about extracting high level spatial knowledge from static to dynamic perspective. In the next two chapters, we will discuss the extraction of physical features based on spatial information from visual input.
Understanding supporting relations between objects is of great interest especially in cluttered scenes with objects connected with each other [Mojtabazadeh et al., 2013; Silberman et al., 2012]. However, precise calculation of support relations between connected objects are often impossible due to the lack of sufficient information (e.g., density distribution, mass of the objects) from sensory input. In this chapter we use a qualitative spatial and physical reasoning approach for this task in 2D environment. We develop a novel qualitative spatial calculus for representing and analysing the structure. The method is tested with the Angry Birds game where the task is to kill pigs which are usually placed on a stack of block-shaped objects. Our calculus allows us to express and evaluate structural properties and rules, and to infer for each building block which of these properties and rules are satisfied.

5.1 Introduction

Qualitative spatial representation and reasoning has numerous applications in Artificial Intelligence including robot planning and navigation, interpreting visual inputs and understanding natural language [Cohn and Renz, 2008]. In recent years, plenty of formalisms for reasoning about space were proposed [Freksa, 1992; Frank, 1992; Renz and Mitra, 2004]. An emblematic example is the RCC8 algebra proposed by [Randell et al., 1992]. It represents topological relations between regions such as “x overlaps y”, or “x touches y”; however, it is unable to represent direction information such as “x is on the right of y” [Balbiani et al., 1999]. The Rectangle Algebra (RA) [Mukerjee and Joe, 1990; Balbiani et al., 1999], which is a two-dimensional extension of the Interval Algebra (IA) [Allen, 1983], can express orientation relations and at the same time represent topological relations, but only for rectangles. If we want to reason about topology and direction relations of regions with arbitrary shapes, we could for example combine RCC8 and RA, which was analysed by [Liu et al., 2009b]. However, if we only consider the minimum bounding rectangles (MBR) of regions, RA is expressive enough to represent both direction and topology.

One common characteristic of most qualitative spatial representations that have been proposed in the literature, and in particular for two-dimensional calculi such as the RA, is that the represented world is assumed to be represented from a birds-eye
Spatial and Physical Reasoning in 2D Environment

view, that is from above like a typical map. This is particularly remarkable since qualitative representation and qualitative reasoning are usually motivated as being very close to how humans represent spatial information. But humans typically don’t perceive the world from a birds-eye view, the “human-eye view” is to look at the world sideways. When looking at the world sideways, the most important feature of the world is the distinction of “up” and “down” and the concept of gravity. What goes up must come down, everything that is not supported by something and is not stable will invariably fall to the ground. This gives a whole new meaning to the concept of consistency which is typically analysed in the context of qualitative spatial reasoning: a qualitative representation $\Theta$ is consistent if there is an actual situation $\theta$ that can be accurately represented by $\Theta$. But if any possible $\theta$ is not stable and would fall apart under gravity, then $\Theta$ cannot be consistent. Notably, there is existing work on applying qualitative calculi on side view scenarios. For example, Tayyub et al. [2014] propose a hybrid reasoning method to recognize activities of humans in everyday scenarios; Gatsoulis et al. [2016] develop a software library named QSRlib to efficiently extract qualitative spatio-temporal relations from human-eye view video input. However, the gravity has not been considered in the above mentioned work.

In this chapter we are concerned with the human-eye view of the world, i.e., we want to be able to have a qualitative representation that takes into account gravity and stability of the represented entities.

Ideally, we want a representation that allows us to infer whether a represented structure will remain stable or whether some parts will move under the influence of gravity or some other forces (e.g., the structure is hit by external objects). Additionally, if the structure is regarded as unstable, we want to be able to infer the consequences of the instability, i.e., what is the impact and resulting movement of the unstable parts of the structure. Gravity and stability are essential parts of our daily lives and any robot or AI agent that is embedded in the physical world needs to be able to reason about stability. Obvious examples are warehouse robots who need to be able to safely stack items, construction robots whose constructions need to be stable, or general purpose household robots who should be able to wash and safely stack dishes. But in the end, any intelligent agent needs to be able to deal with stability, whenever something is picked up, put down or released. While this problem is clearly of general interest, the motivating example we use in this chapter is the popular Angry Birds game. There we have a two-dimensional snapshot of the world, ironically in the human-eye view, and inferring how objects move and fall down under gravity and other forces is of crucial importance when trying to build an AI agent that can successfully play the game.

The Rectangle Algebra is not expressive enough to reason about the stability or consequences of instability of a structure in a human-eye view. While it is possible to some degree, we could simply impose that a structure can only be consistent if each rectangle is supported from below, this is only a necessary condition, it is clearly not sufficient. For example, in Figure 5.1(a) and (b), assume the density of the objects is the same and object 2 is on the ground. The RA relation between object 1 and object 2 in these two figures are both $<\text{start inverse, meet inverse}>$, but obviously the structure in Figure 5.1(a) is stable whereas object 1 in (b) will fall. In order to make such distinctions, we need to extend the granularity of RA and introduce new relations that enable us to represent these differences. In this chapter, we introduce an Extended...
Interval Algebra (EIA) which contains 27 relations instead of the original 13. We use
the new algebra as a basis for an Extended Rectangle Algebra (ERA), which is obtained
in the same way as the original RA. Depending on the needs of an application, we
may not need to extend RA to 27 relations in each dimension. Sometimes we only
need the extended relations in one axis. Thus, the extended RA may include 13 \times 27,
27 \times 13 or 27 \times 27 relations depending on the requirement of different tasks.

![Figure 5.1: Two structures with the same RA relation (si,mi)](image)

In the following, we will introduce EIA and ERA and then develop a rule-based rea-
soning method that allows us to infer the stability of a structure and the consequences
of the detected instability. Our method is focused towards our target application,
namely, to build an agent that can play the Angry Birds game automatically and
rationally. This requires us to identify a target that will make a given stable structure
maximally unstable and will lead to the greatest damage to the structure. In order to
do so, we take a stable structure (typically everything we see is by default stable) and
for each possible target object we remove the object and analyse the consequences of
the structure minus this object. Other applications may require a slightly different
method, but the basic stability detection we develop is a general method. The results
of our evaluation show that the agent based on this method is able to successfully
detect stability and to infer the consequences of instability.

5.2 Interval Algebra and Rectangle Algebra

Allen’s Interval Algebra defines a set $B_{int}$ of 13 basic relations between two intervals.
It is an illustrative model for temporal reasoning. The set of all relations of IA is the
power set $2^{B_{int}}$ of $B_{int}$. The composition ($\circ$) between basic relations in IA is illustrated
in the transitivity table in Allen [1983]. The composition between relations in IA is
defined as $R \circ S = \cup \{ A \circ B : A \in R, B \in S \}$.

RA is an extension of IA for reasoning about 2-dimensional space. The basic
objects in RA are rectangles whose sides are parallel to the axes of some orthogonal
basis in 2-dimensional Euclidean space (see Figure 5.1). The basic relations of RA
can be denoted as $B_{rec} = \{(A, B) | A, B \in B_{int} \}$. The basic relations RA
relations is defined as $(A, B) \circ (C, D) = (A \circ C, B \circ D)$. 
Table 5.1: 27 EIA relations

<table>
<thead>
<tr>
<th>Relation</th>
<th>Illustration</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>EIA(A, B) = eq</td>
<td></td>
<td>A is equal to B</td>
</tr>
<tr>
<td>EIA(A, B) = b;</td>
<td></td>
<td>A takes place before B; B takes place after A</td>
</tr>
<tr>
<td>EIA(B, A) = a</td>
<td></td>
<td>A meets B; ‘i’ stands for inverse</td>
</tr>
<tr>
<td>EIA(A, B) = m;</td>
<td></td>
<td>less part of A overlaps with less part of B</td>
</tr>
<tr>
<td>EIA(B, A) = mi</td>
<td></td>
<td>A meets B; ‘i’ stands for inverse</td>
</tr>
<tr>
<td>EIA(A, B) = lol;</td>
<td></td>
<td>less part of A overlaps with less part of B</td>
</tr>
<tr>
<td>EIA(B, A) = loli</td>
<td></td>
<td>less part of A overlaps with most part of B</td>
</tr>
<tr>
<td>EIA(A, B) = lom;</td>
<td></td>
<td>most part of A overlaps with less part of B</td>
</tr>
<tr>
<td>EIA(B, A) = lomi</td>
<td></td>
<td>most part of A overlaps with most part of B</td>
</tr>
<tr>
<td>EIA(A, B) = ms;</td>
<td></td>
<td>A starts B and covers most part of B</td>
</tr>
<tr>
<td>EIA(B, A) = msi</td>
<td></td>
<td>A starts B and covers most part of B</td>
</tr>
<tr>
<td>EIA(A, B) = ls;</td>
<td></td>
<td>A starts B and covers less part of B</td>
</tr>
<tr>
<td>EIA(B, A) = lsi</td>
<td></td>
<td>A starts B and covers less part of B</td>
</tr>
<tr>
<td>EIA(A, B) = mf;</td>
<td></td>
<td>A finishes B and covers most part of B</td>
</tr>
<tr>
<td>EIA(B, A) = mfi</td>
<td></td>
<td>A finishes B and covers most part of B</td>
</tr>
<tr>
<td>EIA(A, B) = lf;</td>
<td></td>
<td>A finishes B and covers less part of B</td>
</tr>
<tr>
<td>EIA(B, A) = lfi</td>
<td></td>
<td>A finishes B and covers less part of B</td>
</tr>
<tr>
<td>EIA(A, B) = hd;</td>
<td></td>
<td>A is in B and only covers the head half part of B</td>
</tr>
<tr>
<td>EIA(B, A) = hdi</td>
<td></td>
<td>A is in B and only covers the head half part of B</td>
</tr>
<tr>
<td>EIA(A, B) = td;</td>
<td></td>
<td>A is in B and only covers the tail half part of B</td>
</tr>
<tr>
<td>EIA(B, A) =aldi</td>
<td></td>
<td>A is in B and only covers the center point of B</td>
</tr>
</tbody>
</table>

5.3 The Extended Rectangle Algebra (ERA)

In order to express the stability of a structure and reason about the consequences of the instability in a situation which observes physical rules, we extend the basic relations of IA from 13 relations to 27 relations denoted as $B_{\text{int}}$. The relations consider the importance of the centre of mass of an object for its stability, which corresponds to its centre point for rectangular objects.

**Definition 1** (The extended IA relations). We introduce the centre point of an interval as a new significant point in addition to the the start and end points. For an interval $a$, denote centre point, start point and end point as $c_a$, $s_a$ and $e_a$, respectively.

1. The ‘during’ relation has been extended to ‘left during’, ‘centre during’ and ‘right during’ (ld, cd & rd).

   - “x ld y” or “y ldi x” : $s_x > s_y, e_x \leq c_y$
   - “x cd y” or “y cdi x” : $s_x > s_y, s_x < e_y, e_x > c_y, e_x < e_y$
   - “x rd y” or “y rdi x” : $s_x \geq c_y, e_x < e_y$
2. The 'overlap' relation has been extended to 'most overlap most', 'most overlap less', 'less overlap most' and 'less overlap less' (mom, mol, lom & lol).
   - “x mom y” or “y momi x” : $s_x < s_y, c_x \geq s_y, e_x \geq c_y, e_x < e_y$
   - “x mol y” or “y moli x” : $s_x < s_y, c_x \geq s_y, e_x < c_y$
   - “x lom y” or “y lomi x” : $c_x < s_y, e_x \geq c_y, e_x < e_y$
   - “x lol y” or “y loli x” : $c_x < s_y, e_x > s_y, e_x < e_y$
3. The 'start' relation has been extended to 'most start' and 'less start' (ms & ls).
   - “x ms y” or “y msi x” : $s_x = s_y, e_x \geq c_y$
   - “x ls y” or “y lsi x” : $s_x = s_y, e_x > s_y, e_x < c_y$
4. Similarly, the 'finish' relation has been extended to 'most finish' and 'less finish' (mf & lf).
   - “x mf y” or “y mfi x” : $s_x > s_y, s_x \leq c_y, e_x = e_y$
   - “x lf y” or “y lfi x” : $s_x > c_y, s_x < e_y, e_x = e_y$

Denote the set of relations of extended IA as the power set $2^{B_{eint}}$ of the basic relation set $B_{eint}$. Denote the set of relations of extended RA as the power set $2^{B_{erec}}$ of the basic relation set $B_{erec}$. The composition table of EIA and ERA is straightforward to compute.

Note that EIA can be expressed in terms of INDU relations [Pujari et al., 2000] if we split each interval $x$ into two intervals $x_1$ and $x_2$ that meet and have equal duration. However, this would make representation of spatial information very awkward and unintuitive. There is also some similarity with Ligozat’s general intervals [Ligozat, 1991] where intervals are divided into zones. However, the zone division does not consider the centre point.

### 5.4 Inferring stability using ERA rules

In the example in Figure 5.1 we have seen a simple case where a situation is unstable. However, since instability is not a concept which is contained in ERA itself, but is rather a consequence of certain ERA cases, compositional reasoning which is typically used in qualitative spatial reasoning is not helpful for determining instability. Instead, we express standard physical stability rules using ERA relations and use rule-based reasoning to infer stability.

For our analysis we assume that the centre of mass of all objects we represent is at the centre of their MBR. This is always the case for circles and rectangles with a uniform density, even if the rectangles are not parallel to the ground. This covers most objects used in Angry Birds, with the exception of triangular shaped objects and non-structurally significant objects. The MBRs of all Angry Birds objects are obtained using the object recognition module of the basic Angry Birds game playing software that is provided by the organisers of the Angry Birds AI competition [AIBIRDS]. In
In order to get the real shapes of objects, we used an improved version of the object recognition module [Ge et al., 2014], which will be added to the basic software package soon. This module also detects the ground, which is important for stability analysis. In accordance with Ge and Renz [Ge and Renz, 2013] we call rectangles that are equivalent to their MBRs regular rectangles and otherwise angular rectangles. Figure 5.2 gives an overview of the different kinds of angular rectangles that Ge and Renz distinguish. The distinctions are mainly whether rectangles are fat or slim, or lean to the left or right. Interestingly, these distinctions can be expressed using ERA relations by comparing the projections of each of the four edges of the angular rectangle with the projections of the corresponding MBR.

We first translate all objects we obtain using the object recognition modules into a set of ERA constraints $\Theta$. Since all these objects are from real Angry Birds scenes, all ERA constraints in $\Theta$ are basic relations and the identified objects are a consistent instantiation of $\Theta$. This is a further indication that compositional reasoning is not helpful for determining stability. The next step is that we extract all contact points between the objects. This is possible for us to obtain since we have the real shapes of all objects. The contact points allow us to distinguish different types of contact between two objects, which we define as follows:
Definition 2 (Contact Relation). Two rectangles $R_1$ and $R_2$ can contact in three different ways:

- If one corner of $R_1$ contacts with one edge of $R_2$, then $R_1$ has a *point to surface contact* with $R_2$, denoted as $CR(R_1, R_2) = ps$
- If one edge of $R_1$ contacts with one edge of $R_2$, then $R_1$ has a *surface to surface contact* with $R_2$, denoted as $CR(R_1, R_2) = ss$
- If one corner of $R_1$ contacts with one corner of $R_2$, then $R_1$ has a *point to point contact* with $R_2$, denoted as $CR(R_1, R_2) = pp$
- If $R_1$ does not touch $R_2$, then $R_1$ has no contact with $R_2$, denoted as $CR_{R_1, R_2} = n$

Definition 3 (Contact Dimension). The *contact dimension* expresses in which dimension (horizontal or vertical) two rectangles $R_1$ and $R_2$ contact each other. Specifically, if $CR(R_1, R_2) \in \{ps, ss, pp\}$, then:

- $R_1$ and $R_2$ contact horizontally (denoted as $CD(R_1, R_2) = hc$), if $ERA(R_1, R_2) \in (\{m, mi\}, \cup_{EIA})$, where $\cup_{EIA}$ is the set of all EIA relations.
- In all other cases $R_1$ and $R_2$ contact vertically, denoted as $CD(R_1, R_2) = vc$.

If $CR(R_1, R_2) = n$, then $CD(R_1, R_2) = n$, too.

5.4.1 Stability of Regular Rectangles

We start with stability for regular rectangles as the rules are obvious consequences of Newtonian physics. Throughout this chapter, we will never consider what is happening above an object when determining its stability. As an example how this can influence stability, refer again to Figure 5.1(b). Suppose we put a very heavy object 3 on object 1, then object 1 might be stable if the combined centre of mass of objects 1 and 3 is above object 2. Since this requires numerical calculations and cannot be obtained qualitatively, we will always ignore these cases, and consequently will only be able to obtain an approximation of stability. We therefore call our stability concept *ERA-stability*. ERA-stability is based on the simple physical rule that an object is stable and will not topple if the vertical projection of the centre of mass of an object falls into the area of support base.

Given a set of ERA-constraints $\Theta$ and a consistent instantiation $\theta$, where the centre of mass of all elements $x \in \theta$ coincides with the centre of the MBR of $x$. ERA-stability for regular rectangles $x \in \theta$ is determined recursively using the following rules, until no more objects are determined ERA-stable:

A regular rectangle $x \in \theta$ is ERA-stable, if one of the following rule applies:

Rule 1.1: $x$ lies on the ground.

Rule 1.2: $\exists y, z \in \theta$ such that $y$ and $z$ are both ERA-stable and $CD(y, x) = CD(z, x) = vc$: $ERA_y(x, y) \in \{moni, moli, lomi, lol, msi, lsi, rdi\} \land \land ERA_z(x, z) \in \{mom, mol, lom, lol, mfi, lfi, rdi\}$. 

\[\]
Rule 1.3: \( \exists y \in \theta \) such that \( y \) is ERA-stable and \( CD(y,z) = vc \): \( ERA_x(x,y) \in \{ms, mf, msi, ls, mfi, lf, cd, cdi, ld, rd, mom, momi, lomi, mol\} \).

Rule 1.4: \( \exists y, z \in \theta \) such that \( y \) and \( z \) are both ERA-stable, \( CD(x,y) = hc \) and \( CD(x,z) = hc \): \( (ERA_x(x,y) \in \{mi\} \land ERA_x(x,z) \in \{mom, mol, lom, lol, mfi, lfi, rdi\}) \lor (ERA_x(x,y) \in \{m\} \land ERA_x(x,z) \in \{mom, momi, lomi, lol, msi, lsi, ldi\}) \).

Rule 1.2 covers the cases where the vertical projection of the mass centre of an object is located between two ERA-stable supporters \( y \) and \( z \). Rule 1.3 covers the cases where it is located above the contact area of an ERA-stable supporter \( y \). Rule 1.4 covers cases where \( x \) is horizontally contacting with an ERA-stable object on one side and has an ERA-stable supporter on the other side. Figure 5.3 shows examples of the four ERA-stable cases for a regular rectangle. In all four rules \( x \) is always ERA-stable.

5.4.2 Stability of Angular Rectangles

The case of angular rectangles is more complicated than regular rectangles, but it follows the same physical rule that if an object is stable, the vertical projection of the centre of mass must fall into a support area. While the support area for regular rectangles is defined by two supporting objects, for angular rectangles it can be defined by the ground, acting as a supporting object, and one supporting object. Angular rectangles always have a momentum to become regular rectangles. The rectangles in Figure 5.2(a,b,e) have a momentum in counter-clockwise direction, the rectangles in
§5.4 Inferring stability using ERA rules

Figure 5.2(c,d,f) have a momentum in clockwise direction. The three rectangles in the bottom row probably never occur in reality as the top point has to be exactly above the bottom point and we ignore these cases. The momentum means that an angular rectangle always needs support to prevent it from becoming a regular rectangle. In cases where the bottom point of an angular rectangle is supported, for example by the ground, the second support must always be on the opposite side of the centre of mass. This leads to the following rules, which again are applied recursively, until no more objects are determined ERA-stable:

An angular rectangle $x \in \theta$ is ERA-stable, if one of the following rule applies:

**Rule 1.5:** $\exists y, z \in \theta$ such that $y$ and $z$ are both ERA-stable and $CD(y, x) = CD(z, x) = vc$: $ERA_x(x, y) \in \{\text{moni, moli, lomi, loli, msi, lsi, ldi}\} \land ERA_x(x, z) \in \{\text{mom, mol, lom, lol, mfi, lfi, rdi}\}$.

**Rule 1.6:** $\exists y \in \theta$ such that $y$ is ERA-stable and $CD(y, x) = vc$ and $CR(x, y) = ss$: $ERA_x(x, y) \in \{ms, mf, msi, ls, mfi, lf, cd, cdi, ld, rd, mom, momi, lomi, mol\}$.

**Rule 1.7:** $\exists y, z \in \theta$ such that $y$ and $z$ are both ERA-stable and $CD(y, x) = CD(z, x) = hc$ and $CD(z, x) = vc : ((ERA_x(x, y) \in \{mi\} \land ERA_x(x, z) \in \{mom, mol, lom, lol, mfi, lfi, rdi\}) \lor (ERA_x(x, y) \in \{m\} \land ERA_x(x, z) \in \{moni, moli, lomi, loli, msi, lsi, ldi\}))$.

**Rule 1.8:** $\exists y, z \in \theta$ such that $y$ and $z$ are both ERA-stable and $CD(y, x) = CD(z, x) = hc : ERA_x(x, y) \in \{m\} \land ERA_x(x, z) \in \{mi\}$

The principle of Rule 1.5 is similar to Rule 1.2 which tests if the centre of mass falls between two vertical supporters. Rule 1.6 is similar to Rule 1.3, the difference is that it requires the two objects to be surface-to-surface contact, otherwise the angular rectangle will topple. The supporter in this case also needs sufficient support to remain stationary. Rule 1.7 is a similar rule as Rule 1.4 which describes horizontal support for angular rectangles. Rule 1.8 describes a different property of angular rectangles, which is an angular rectangle can remain stable with the support of two horizontally contacting objects on both left and right sides. This is because the two objects on the sides can provide upward frictions to support the angular rectangle.

Figure 5.4 shows example cases of the above four rules where angular rectangles are ERA-stable.

All these rules only consider objects as support that are already stable. Cases where two rectangles are stable because they support each other are not considered. Applying rules 1.1-1.8 recursively to all objects in $\theta$ until no more objects are detected as ERA-stable identifies all ERA-stable objects. If all objects in $\theta$ are ERA-stable, then $\theta$ is ERA-stable. The recursive assignment of ERA-stability to objects allows us to build a stability hierarchy. Regular objects that lie on the ground are objects of stability level 1. Then, objects that are stable because they are supported by an object of stability level $\ell$ and possible objects objects of stability level smaller than $\ell$ are considered to be of level $\ell + 1$. Conversely, we could define the support depth of a target object (see Figure 5.5). Those object that are directly responsible for the ERA-stability of a target object $x$ as they support it have support depth 1. Objects that are directly responsible
for the ERA-stability of an object $y$ of support depth $\ell$ as they support it have support depth $\ell + 1$. This can be calculated while recursively determining stability of objects. The support level can be helpful in order to separate the support structure of an object from the larger structure, or when determining which object could cause the most damage to a structure when destroyed.

5.5 Applying ERA-stability to identify good shots in Angry Birds

The goal of the Angry Birds game is to shoot birds using a slingshot in a way that all pigs are killed with the fewest number of used birds. The pigs are typically supported by complicated structures that have to be destroyed in order to kill the pigs. Instead of shooting at a pig directly, it might be better to destroy the whole structure that supports the pig. This could be optimised by using the support structure we defined in the previous section.

Another useful substructure is the shelter of the pigs. The reason is straightforward, if a pig is not reachable, there must be some objects that protect it; these objects form the sheltering structure of the pigs. Destroying the sheltering structure can either kill the pig or make the pig directly reachable to the bird. We define the sheltering structure of a pig as the closest objects that protect the pig from a direct hit from each direction. Specifically, a sheltering structure of an object (usually a pig) consists of left, right and top sheltering objects. In order to obtain the sheltering structure of a certain object, the first step is to get the closest object from the left side of the
5.5 Applying ERA-stability to identify good shots in Angry Birds

Figure 5.5: Illustration of support structure

queried object; then, get the supportee list of the object (similar process as getting the supporter list); after that, get the right closest object with its supportee list. The next step is to check if the two supportee lists have objects in common. If so, pick the one with smallest depth as the roof object of the sheltering structure; if not, there is no sheltering structure for the queried object. If a roof object is found, also put the supportees of both the left and right closest objects with smaller depth than the roof object into the sheltering structure. Finally, put the supporters of both left and right closest objects which are not below the queried object into the sheltering structure. This can also be described using rules expressed in ERA (actually RA is sufficient).

5.5.1 The integration of the rules to evaluate a shot

With the rules described above, we are able to dynamically analyse the possible consequences after a shot has been made. In order to predict the final consequence of an external influence on the structure (usually the impact from a bird), the direct consequence and its following subsequences should be analysed in detail. Funt [1987] suggested a similar method to simulate the consequence of a structure with a changed object which assumes that the changed object disappears and chooses the most significant unstable object to simulate the consequence.

In the following, we analyse different cases of how a structure can be impacted from an external force and, in particular, what other objects will be affected by an impact to the target object. We separately analyse regular rectangles and angular rectangles, each of which can be affected in four different ways. First we consider the four cases where the affected objects are regular rectangles:

Case 1: The target object is directly hit by another object. The direct consequence to the hit object will be one of three types: destroyed, toppling, remaining stationary. Empirically, the way to determine the consequence of the hit depends on the height and width ratio of the target. For example, if an object hits a target with the height and width ratio larger than a certain number (such as 2), the target will fall down. And this ratio can be changed to determine the conservative degree of the system. In other words, if the ratio is high, the system tends to be conservative because many hits will have no influence on the target. Moreover, if the external object affects a target with the height and width ratio
Algorithm 4: Direct hit estimation

1 Function HitEstimation(o, pendingList, affectedList)
   Input: the object o that is hit, a list of objects pendingList that have not
   been evaluated and a list of objects affectedList that are predicted as
   affected objects

2 if o ∈ affectedList then
3     pendingList.remove(o)
4     return
5
6 if o.isRegular then
7     if o.affectedMethod = “Hit” then
8         if o.isDestroyed then
9             affectedList.add(o)
10            pendingList.add(supporteesOf(o))
11     else if !o.isDestroyed then
12         if o.height/o.width > threshold then
13             affectedList.add(o)
14             pendingList.add(supporteesOf(o))
15             pendingList.add(supportersOf(o))
16         else
17             pendingList.add(supportersOf(o))

18     pendingList.remove(o)

less than one, the target itself will remain stable temporarily because the system
will also evaluate its supporter to determine the final status of the target. After
deciding the direct consequence of the hit, the system should be able to suggest
further consequences of the status change of the direct target. Specifically, if the
target is destroyed, only its supportees will be affected. If the target falls down,
the case will be more complex because it may influence its supporters due to the
friction, as well as its supportees and neighbours. If the target remains stable
temporarily, it will also influence its supporters, and its supporters may again
affect its behaviour. Algorithm 4 demonstrates the evaluation process of an
object that has been hit by another object.

Case 2: The supportee of the target object topples down. The target object’s stabil-
ity is also determined by its height to width ratio, but the number should be
larger (about 5) as the influence from a supportee is much weaker than from a
direct hit. If the target is considered as unstable, it will fall down and affect is
neighbours and supporters; otherwise, it will only influence its supporters (see
Figure 5.6).

Case 3: The supporter of the target object topples down. Here a structure stability
check process (applying Rules 1.1 - 1.4) is necessary because after a supporter
5.5 Applying ERA-stability to identify good shots in Angry Birds

falls, the target may have some other supporters and if the projection of its mass centre falls into the areas of its other supporters, it can stay stable. Then, if the target remains stable, it will again only affect its supporters due to the friction; otherwise, it may fall and affect its supporters, supportees and neighbours (see Figure 5.7(a)).

Case 4: The supporter of the target is destroyed. This is more like a sub case of the previous one. If the target cannot remain stable after its supporter is destroyed, it may fall and affect its supporters, supportees and neighbours (see Table 5.7(b)).

Next we consider the four cases where the target objects are angular rectangles:

Case 5: The target object is directly hit. As analysed in Section Inferring stability using ERA rules, angular rectangles can be classified into 9 classes. If we only consider the direction of the objects, there only exists 3 classes, which are “lean to left”, “lean to right” and “stand neutrally”. Assume the hit is from left to right. Then if a “lean-to-left” object is hit and the force is strong enough, the object will rotate around the lowest pivot, i.e. the lowest corner of the object. However, in the Angry Birds game, before the force becomes sufficient to make the object rotate, the object will be destroyed first. Thus, the two possibilities here are either the target object affects its supporters or it is destroyed and affects its supportees. The case of “lean-to-right” object is the same as above. For a “stand-neutrally” object, it cannot stand by itself and at least one point-to-surface touched supporter on each side is necessary. If it is hit from left, there will be no friction applied to its left supporter, thus it will either be destroyed or affect its right supporter.

Case 6: The supportee of the target object topples down. Angular objects will not topple due to the fall of their supportees. Because their supporters restrict
their spatial location, only the supporters will be affected. For "stand-neutrally" objects, if the friction given by their supportees is from left to right, the left supporters will not be affected, and vice-versa for the right supporter.

**Case 7: The supporter of the target object topples down.** First check the stability using Rules 1.5 - 1.8. If the object still has sufficient support, it only affects its supporters, otherwise it will rotate and affect its supportees, other supporters and neighbours.

**Case 8: The supporter of the target object is destroyed.** Again, a stability check is applied first. However if it remains stable, it will not affect its supporters because no friction applies to the target from the destroyed supporter. If it is not stable, than it will affect its supportees, other supporters and neighbours.

By applying these cases recursively for each potential target object, we obtain a list of objects that are affected when hitting a particular target object. Then, with all the affected objects in a list, the quality of the shot can be evaluated by calculating a total heuristic value depending on the affected objects. The scoring method is defined as follows and is based on experimenting with different heuristic values: if an affected object belongs to the support structure or the sheltering structure of a pig, 1 point will be added to this shot; and if the affected object is itself a pig, 10 points will be added to the shot. After assigning scores to shots at different possible target objects, the target with the highest score is expected to have the largest influence on the structures containing pigs when it is hit. The heuristic value roughly corresponds to the number of affected objects plus a higher value for destroyed pigs. Then, based on different strategies, the agent can choose either to hit the reachable object with highest heuristic value or generate a sequence of shots in order to hit the essential support object of the structure.

We first extract the ERA relations between all objects and then match the rules for all relevant combinations of objects. Thus, the process of evaluating the significance of the targets is straightforward and fast.

### 5.6 Planning in Angry Birds

The previous analysis only looks at the effects of one shot, while taking into account reachability of target objects. Reachability is calculated using the impact different bird types have on different block types. For example, a yellow bird can hit through 2 layers of wood, a blue bird can go through 2 layers of ice, etc. We can use this knowledge to reason about the reachability of a bird and roughly estimate the damage of the direct consequence of a shot. Ideally, we would like to plan a sequence of shots. Due to the inability to accurately predict the state after a shot (we can currently only predict affected objects), successful planning of shot sequences seems very hard.

However, in order to demonstrate how planning in Angry Birds could work, we have implemented one special case of planning, which can be used when the shot with highest score is not reachable due to some blocks in the path. We can identify which blocks are on the trajectory and block the target. Then a planner could be used to generate a shot to first clear the blocks in the path and then hit the primary target with the second shot. Specifically, the planner will first check if the highest-ranked
shot is reachable. If not, it will retrieve the block objects in the calculated trajectory. Then it will search for a shot that can affect most of the objects in the block list. Finally, the planner will suggest using the first shot to clear the blocks if applicable. However, as we cannot exactly simulate the consequence of a shot, sometimes the first shot may lead to an even worse situation. Initial experiments we did with this method showed good results though. Figure 5.8 shows the process of the planner.

5.7 Evaluation

We evaluate our approach by applying it to the first 21 Poached Eggs levels, which are freely available at chrome.angrybirds.com. These levels are also used to benchmark the participants of the annual Angry Birds AI competition. We take each level and calculate the heuristic value of all reachable objects and rank them according to the heuristic value. We then fire the bird at each of the reachable objects and record the actual game score resulting from each shot. We reload the level before each shot, so each bird always fires at the initial structure. We restricted our analysis to initial shots only, as we found that it is not possible to consistently obtain the same intermediate game state simply by repeating the same shot. However, our method works equally well for intermediate game states.

The results of our evaluation are shown in Table 5.2, where we listed for each of the 21 levels how many reachable objects there are, the resulting game scores of shooting...
Table 5.2: Evaluation on first 21 levels. #RO is the number of reachable objects, S1, S2, and S3 are the scores obtained when shooting at the highest, second-highest and third highest ranked object. HS is the rank of the reachable object with the highest score.

<table>
<thead>
<tr>
<th>Level</th>
<th>#RO</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>Time(s)</th>
<th>HS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>29030</td>
<td>30370</td>
<td>27330</td>
<td>1.34</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>16240</td>
<td>6040</td>
<td>11250</td>
<td>1.06</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>5410</td>
<td>41850</td>
<td>6790</td>
<td>0.77</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>36890</td>
<td>1850</td>
<td></td>
<td>1.45</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>22380</td>
<td>66240</td>
<td>16800</td>
<td>5.11</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td>1610</td>
<td>6770</td>
<td>7710</td>
<td>1.83</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td>46200</td>
<td>6230</td>
<td>6210</td>
<td>3.81</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>17310</td>
<td>17310</td>
<td>6010</td>
<td>1.08</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>14030</td>
<td>10140</td>
<td>13210</td>
<td>5.37</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>22920</td>
<td>3860</td>
<td>5920</td>
<td>3.88</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>57110</td>
<td>22640</td>
<td>9680</td>
<td>6.60</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>26570</td>
<td>15010</td>
<td>21800</td>
<td>3.47</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>8</td>
<td>12200</td>
<td>12650</td>
<td>14460</td>
<td>4.02</td>
<td>3</td>
</tr>
<tr>
<td>14</td>
<td>10</td>
<td>30330</td>
<td>28590</td>
<td>37670</td>
<td>2.43</td>
<td>7</td>
</tr>
<tr>
<td>15</td>
<td>9</td>
<td>16880</td>
<td>17550</td>
<td>4580</td>
<td>3.97</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>11</td>
<td>15090</td>
<td>22430</td>
<td>23860</td>
<td>4.24</td>
<td>3</td>
</tr>
<tr>
<td>17</td>
<td>12</td>
<td>48850</td>
<td>12240</td>
<td>19850</td>
<td>7.18</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>10</td>
<td>9460</td>
<td>14590</td>
<td>3240</td>
<td>3.02</td>
<td>2</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>5870</td>
<td>3290</td>
<td></td>
<td>5.83</td>
<td>1</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
<td>7420</td>
<td>8170</td>
<td>6820</td>
<td>4.46</td>
<td>2</td>
</tr>
<tr>
<td>21</td>
<td>8</td>
<td>14270</td>
<td>4650</td>
<td>7170</td>
<td>12.70</td>
<td>1</td>
</tr>
</tbody>
</table>

at the four highest ranked reachable objects, as well as the rank of the object that received the highest game score. It turns out that in about half of the levels (10 out of 21), our method correctly identified the best object to hit. In a further 5 cases, the second highest ranked object received the highest score, in 2 cases the 3rd highest ranked object. On average there are about 7.8 reachable objects. The average score of the shooting at the highest ranked object over all 21 levels is about 21,700, shooting at the second ranked object gives an average of about 16,700 and the third ranked object an average of about 13,100. This clearly demonstrates the usefulness of our method in identifying target objects.

However, the resulting game score is not always a good measure, as shots can have unexpected side effects that lead to higher scores than expected. These side effects are typically unrelated to the stability of a structure and, hence, are not detected by our method. As an example, consider level 16 where the second and third ranked objects resulted in a higher score than the first ranked object. Figure 5.9 shows the level and output of our stability analysis. Our method predicts that shooting at object 30 will kill the two pigs on the left and result in the toppling of the left half structure.

Figure 5.10 shows the actual result after the recommended shot has been made. We can see that the prediction of the system is very accurate in this case. The left two
pigs are killed and the left part of the structure toppled. However, the shots at the second and third ranked objects each killed three pigs and therefore obtained higher scores. It turns out that two of the three pigs were killed when they bounced to the wall without actually destroying much of the structure (see Figure 5.11). Therefore, the next shot is much less likely to solve the level as more of the structure remains. So the shot at the highest ranked object first is likely still the best shot overall as the second shot can easily solve the level. This example nicely shows that Angry Birds requires even more sophisticated methods to correctly predict the outcome of shots.

5.8 Related Work

Qualitative spatial representation and reasoning has been applied to some physical systems to do common sense reasoning [Klenk et al., 2005]. Physical reasoning has two important theories, namely kinematics and dynamics [Forbus, 1988]. Kinematics mainly concerns about the position of objects which may change continuously over time and the shape which is strictly rigid and static over time [Davis, 2008]. However it
considers less about the forces between objects and the type of motions. These features make kinematics easy to be formulated, on the other hand models using kinematics are usually limited by the context and appear to be less expressive. The CLOCK Project [Forbus et al., 1991] is a typical success which uses a variety of techniques but mainly kinematics to represent and reason about the inner motions of a mechanical clock. Under some given environments and constraints, this approach can successfully analyse the mechanisms; however, as this system requires restricted constraints such as the exact shape of the parts and the range of the motion, it may not be applicable in the cases with high degrees of freedom such as the world of Angry Birds.

In contrast, dynamics takes force analysis into account which allows reasoning about more complex motions and the transfer of motion. However, the limitation is that precisely predicting a consequence of a motion in an arbitrary environment is almost impossible due to uncertainty. However, it is possible to reason about some simpler physical features of a structure, such as stability. [Fahlman, 1974] implemented a stability verification system dealing with the stability of blocks with simple shapes (e.g. cuboid). This system can produce qualitative output to determine whether a structure is stable without external forces, but it requires some quantitative inputs (e.g. magnitude of forces and moments). The stability check in our work is partially inspired by this system, however without quantitative inputs about forces, we use a purely qualitative way to analyse the probability of the balance of forces and moments in a structure. Although our approach may produce a few false predictions in very complex cases, it is a good estimation of humans’ reasoning approach. Our work also focuses on reasoning about the impacts of external forces (a hit from an external object) on a structure, which has not been discussed much in previous work.

Additionally, a method for mapping quantitative physical parameters (e.g. mass, velocity and force) to different qualitative magnitudes will be very useful for the situation where the input information is often incomplete. [Raiman, 1991] developed a formalization to define and solve order of magnitude equations by introducing comparison operators such as \( \approx \) (of the same magnitude) and \( \ll \) (far less than). Raiman’s method is of particular significance for our approach, for example, we will be able to infer that a hit from a \( 1 \times 1 \) block will not change the motion state of a \( 100 \times 100 \) block.
with the same density.

## 5.9 Discussion and Future Work

In this chapter we have introduced an extended rectangle algebra useful for representing and reasoning about stability under gravity and other properties of 2-dimensional structures. By splitting some basic interval relations into more detailed ones, we obtained 27 interval relations in each dimension that can express the physical relations between rectangular objects more precisely. Our algebra is very useful when representing a human-eye view of the physical world, rather than the birds-eye view which is surprisingly common in qualitative spatial representation and reasoning. A nice example of a two-dimensional human-eye view of the world is Angry Birds, where we can represent the world using the new algebra.

In addition to representing what we see using the extended interval algebra, we also used it for defining some useful structural rules regarding properties such as stability, reachability, support, and shelter that allow us to perform rule-based reasoning about these properties. We tested the usefulness of our rules by designing an agent that performs a structural analysis of Angry Birds levels. Based on these rules, we predict for each block the consequences if it gets hit and calculate a heuristic value that determines the usefulness to hit the block. We then shoot at the block with the highest value that is reachable with the current bird. Our evaluation shows that our method is very successful in determining good target objects and good shots.

Despite its success, some parts of our method offer significant room for improvement. For example, our current motion analysis can only vaguely predict which objects may be affected by a shot, i.e., we do not have accurate predictions about a possible successor state at this stage. Because of this, we currently cannot plan ahead and evaluate sequences of shots in advance. Also we only consider stability of an object with respect to its supporters, but not with respect to objects that are on top of it, which means we do not analyse the interactions between substructures. In the future, we will try to design a feedback loop between the dynamic analysis and ERA, specifically, so we can use the results of the dynamic analysis to restrict a range of possible subsequent positions of an object and use ERA rules to suggest several possible successor cases. Then we can use dynamic analysis again to check the stability of these cases and finally decide on desirable successor states. Moreover, we will find a method to determine useful substructures in order to increase the reliability of the dynamic analysis.

Although qualitative stability analysis can be applied in a 2D simulated environment such as Angry Birds and other video games with gravity enabled \cite{Ge et al., 2016}, integrating this method into a 3D perception framework is not straightforward. The extra dimension brings a significant magnitude of uncertainty in terms of object volume, mass, contact type, etc. for qualitative stability analysis. In the next chapter, we will discuss a hybrid method that combines qualitative reasoning and static equilibrium for determining structural stability.
Chapter 6

Spatial and Physical Reasoning in 3D Environment

In a 3D perception system, understanding physical relations between objects, especially their support relations, is crucial for the AI agent to make rational and safe decision. There has been work on reasoning about support relations and structural stability of simple configurations in RGB-D images. In this chapter, we proposed a method on extracting more detailed physical knowledge from a set of RGB-D images taken from the same scene but different views using qualitative reasoning and intuitive mechanic models. Rather than providing a simple contact relation graph and approximating stability over convex shapes, our method is able to provide a detailed supporting relation analysis based on a volumetric representation. Specifically, true supporting relations between objects (e.g., if an object supports another object by touching it on the side or if the object above contributes to the stability of the object below) are identified. In the experiments, we apply our method to real-world structures captured by Kinect V2 and show our method works as desired on the structures with irregular shaped objects.

6.1 Introduction

Scene understanding for RGB-D images has been extensively studied recently with the availability of affordable cameras with depth sensors such as Kinect [Zhang, 2012].

Among various scene understanding aspects [Chen et al., 2016], understanding spatial and physical relations between objects is essential for robotics manipulation tasks [Mojtabazedeh et al., 2013] especially when the target object belongs to a complex structure, which is common in real-world scenes. Although most research on robotics manipulation and planning focuses on handling isolated objects [Ciocarlie et al., 2014][Kemp et al., 2007], increasing attention has been paid to the manipulation of physically connected objects (for example [Stoyanov et al., 2016][Li et al., 2017]). To be able to perform the manipulation in those complex situations successfully, we have to solve different problems. For example, connected objects can remain stable due to support from their adjacent objects rather than simple surface support from the bottom for stand-alone objects. Actually, support force may come from an arbitrary direction in real-world situations. Additionally, real-world objects tend to cover or hide behind other objects from many view points. Given that the real-world objects
Spatial and Physical Reasoning in 3D Environment

often have irregular shapes, correctly segmenting the objects and extracting their contact relations are challenging tasks. With the results of the above-mentioned tasks, an efficient physical model which can deal with objects with arbitrary shapes is required to infer precise support relations.

In this chapter, we propose a framework that takes raw RGB-D images as input and produces a support graph about objects in the scene. Most existing work on similar topics either assumes object shapes to be simple convex shapes [Shao et al., 2014], such as cuboid and cylinder or make use of previous knowledge of the objects in the scene [Silberman et al., 2012; Song et al., 2016] to simplify the support analysis process. Although reasonable experimental results were demonstrated, those methods usually lack the capability of dealing with scenes that contain a lot of unknown objects. As a significant difference from the existing methods, our proposed method does not assume any knowledge of the appearing objects. After individually segmenting point cloud of each view of the scene, a volumetric representation based on Octree [Laboratory and Meagher, 1980] will be built for each view with information about hidden voxels. The octree of the whole scene combined from the input views will be constructed by reasoning under spatial constraints among the objects. This process allows us to correct mismatches of objects from different views and finally recover a reliable contact graph of all views for support relation analysis.

We adopt an intuitive mechanic model to determine the overall stability of the structure. By iteratively removing contact force between object pairs, we can figure out supporters of each object in the structure and then build the support graph. During this process, we can also reason about the solidity of objects by preserving the initial stability of the structure. For example, to validate the structure stability through the mechanic model, sometimes two objects with the same volume have different mass, thus they have different solidity internally.

6.2 Related Work

In this section, we will talk about related work on qualitative spatial and physical reasoning as well as relevant work about scene understanding on spatial and physical relations in general.

There has been work on scene understanding on support relations from a single view RGB-D image in both computer vision and robotics. In computer vision, scene understanding helps to produce more accurate detailed segmentation results. The work described in [Jia et al., 2013] applied an intuitive physical model to qualitatively infer support relations between objects and the experimental results showed the improvement of segmentation results on simple structures. In [Silberman et al., 2012], the types of objects in indoor scenes were determined by learning from a labeled data set to get more accurate support relations. The above-mentioned papers both took a single image as input which limited the choice of the physical model since a significant amount of hidden information was not available. Shao et al. [2014] attempt to recover unknown voxels from single view images by assuming the shape of the hidden objects to be cuboid and use static equilibrium to approximate the volume of the incomplete objects. In robotics, Mojtahedzadeh et al. [2013] proposed a method to safely de-stack boxes based on geometric reasoning and intuitive mechanics, and this method was shown to be effective in their later work [Stoyanov et al., 2016].
a simulation based method was proposed to infer stability during robotics manipulation on cuboid objects. This method includes a learning process using a large set of generated simulation scenes as a training set.

However, people look at things from different angles to gather a complete information for a better understanding. For example, before a snooker player proceeds a shot, the player will usually look around the table from several critical views to have an overall understanding of the situation. This case also applies to robots which take images as the input source. A single input image provides incomplete information. Even when the images are taken from the same static scene from different views, the information may be still inadequate for scene understanding on detailed physical and spatial information while using quantitative models which require precise inputs. Qualitative reasoning is demonstrated to be suitable for modelling incomplete knowledge [Kuipers, 1989]. There are various qualitative calculi devised to represent entities in a space with focuses on different properties of the entities [Randell et al., 1992; Liu et al., 2009a; Ligozat, 1998; Guesgen, 1989; Mukerjee and Joe, 1990; Lee et al., 2013]. Reasoning using the combination of rectangle algebra and cardinal direction relations, two well-known examples of qualitative spatial calculi, was also studied [Navarrete and Sciacchitano, 2006]. In our previous work [Zhang and Renz, 2014], we developed an Extended Rectangle Algebra (ERA) which simplifies the idea in [Ge and Renz, 2013] to infer stability of 2D rectangular objects. It is possible to combine ERA with an extended version of basic cardinal direction relations to qualitatively represent detailed spatial relations between the objects, which helps to infer the transformation between two views. It is worth mentioning that [Panda et al., 2016] proposed a framework to analyze support order of objects from multiple views of a static scene, yet this method requires relatively accurate image segmentation and the order of the images for object matching.

Models for predicting stability of a structure have been studied in the past several decades. Fahlman [1974] proposed a model to analyze system stability based on Newton’s Laws. A few simulation based models were also presented in recent years [Cholewiak et al., 2015; Li et al., 2017]. However, Davis and Marcus [2016] argues that probabilistic simulation based methods are not suitable for automatic physical reasoning due to some limitations including the lack of capability to handle imprecise input information. Thus in our approach, we aim to apply qualitative reasoning to combine raw information from multiple views to extract understandable and more precise relations between objects in the environment.

### 6.3 Method Pipeline

In this section, the overall pipeline of the support relation extraction method will be described. The method consists of three modules, namely image segmentation, view registration and stability analysis.

The image segmentation module takes a set of RGB-D images taken from different views of a static scene as input. To retain generality of our method, we do not assume any pre-known shapes of the objects in the scene (that is why we do not use template matching methods that can provide more accurate segmentation results). This setting makes our method applicable in unknown environments. In the implementation, we use LCCP [Stein et al., 2014] for point cloud segmentation. LCCP first represents the
point cloud as a set of connected supervoxels [Papon et al., 2013]. Then the supervoxels are segmented into larger regions by merging convexly connected supervoxels. We identify and correct the segmentation errors by comparing segmentation from different views (see section 6.4.3). Each point cloud of a view will be segmented into individual regions. We use connected graph to represent relations between the regions. Each graph node is a segmented region. There is an edge between two nodes if the two regions are connected. We use Manhattan world [Furukawa et al., 2009] assumption to find the ground plane. The entire scene will then be rotated such that the ground plane is parallel to the flat plane. Details about segmentation and ground plane detection will not be discussed in this chapter as we used this method with little change. Figure 6.1 shows a typical output from this module.

In the view registration module, we use an object matching algorithm to find an initial match between objects from two views. Panda et al. [2016] used a similar process to match objects across multiple views. However, their method assumes the geometries of objects are known so that a template matching segmentation can be applied. Also, the input views are assumed to be in either clockwise or anti-clockwise order and taken with a camera at the same height. While these settings lead to a more accurate object matching result, it is less applicable to some real-world situations where the robots cannot take pictures freely, or the inputs are from multiple agents. To handle these situations, we develop an efficient method that can register objects from different views without those restrictions. In this chapter, we assume no prior knowledge to ensure the generality of the method.

To achieve this, we need to solve some problems from a cognitive aspect. For example, due to occlusion, a large object may be segmented into different regions. Thus the match between objects in two views may not be a one-to-one match. Also, as the order of the input images is unknown, we need a method to determine if a match is acceptable by detecting conflicts regarding spatial relations among all
matched objects. The conflicts and ambiguities of object matching will be resolved with a qualitative reasoning method which will be explained in detail in section 6.4.3. The contact graph of each single image will then be combined to produce a contact relation graph over all input images.

In the stability analysis module, we adopt the definition of the structural stability in Livesley, 1978 and analyze static equilibrium of the structure by representing reacting forces at each contact area as a system of equations. A structure is considered as stable if there is a solution to the equations. Given the input scene is static, several schemes will be used to adjust the unseen part of the structure to make the static equilibrium hold.

6.4 View Registration

In this section, we will introduce the algorithm for combining images from different views of the same static scene. We first apply a feature based object matching algorithm to find an initial match. Then a qualitative spatial reasoning approach will be used to detect incorrectly matched objects and resolve conflicts and ambiguities in the initial matching.

6.4.1 Initial Object Matching

The initial matching process is similar to the algorithm described in Panda et al., 2016. For a pair of segmented images, each region in one view will be matched to the whole image of the other view.

Formally, \( V_A \) and \( V_B \) denote two views of a static scene. \( \{ R_{A1}, R_{A2},...,R_{Am} \} \) denotes the set of regions in \( V_A \), and \( \{ R_{B1}, R_{B2},...,R_{Bn} \} \) denotes the set of regions in \( V_B \). \( \text{Match}(R_{Ai}, R_{Bj}) \) denotes the match between \( R_{Ai} \) and \( R_{Bj} \) after matching \( R_{Ai} \) with the regions in \( V_B \).

We first use SHOT Tombari et al., 2010 to calculate key point descriptors of both the candidate region and the image. Then we find the correspondences between the region and the scene by looking for the nearest neighbor of each descriptor in the region to the one in the scene represented as a k-d tree Bentley, 1975. Due to segmentation errors and the lack of information, there are mistakes in the initial matching results. We found three major types of problematic matches from the experiments:

- **Obvious match conflict**

  An obvious match conflict occurs when two matches \( \text{Match}(R_{Ai}, R_{Bm}) \) and \( \text{Match}(R_{Bm}, R_{Aj}) \) are detected, where \( i \neq j \).

- **Multiple-to-one match conflict**

  An multiple-to-one match conflict occurs when two matches \( \text{Match}(R_{Ai}, R_{Bm}) \) and \( \text{Match}(R_{Aj}, R_{Bm}) \) are detected, where \( i \neq j \).
• One-to-one match error

Assume $\text{Match}(R_{Ai}, R_{Bm})$ and $\text{Match}(R_{Bm}, R_{Ai})$ are ground truth matches. A one-to-one match error occurs when $\text{Match}(R_{Ai}, R_{Bn})$ and $\text{Match}(R_{Bn}, R_{Ai})$ are detected, where $m \neq n$.

Obvious match conflict is also mentioned in [Panda et al., 2016] and has been resolved by adjusting match to the one with more key point correspondences. This method works in most obvious match conflict cases, however, it produces one-to-one match conflicts or multiple-to-one matches occasionally.

Multiple-to-one match conflict often occurs when an object is segmented into multiple regions in one view. One-to-one match conflict usually happens when there is a relatively large angle difference between two views so that an object is incorrectly matched to one with similar shape.

6.4.2 Combination of ERA and Cardinal Relations

In order to resolve One-to-one match conflict and Multiple-to-one match conflict, we use spatial relations between the objects as constraints to detect and adjust the mismatches.

For example, in figure 6.2, the top two objects are correctly matched and the bottom object is mismatched. The mismatched object can be detected by qualitatively reasoning about the spatial relation (e.g., topological relation and directional relation) change comparing to the correctly assigned objects. This method is especially useful when the descriptor of the key points are not well distinguished.

In our previous work [Zhang and Renz, 2014], we have developed a spatial calculus named extended rectangle algebra (ERA). This calculus has been shown to be able to qualitatively represent spatial relations and reason about the structural stability of a group of connected 2D rectangular objects.

In this problem, ERA is not expressive enough as the objects are incomplete 3D entities with irregular shapes. In section 6.3, we mentioned that the ground plane has been detected under the Manhattan space assumption, thus it is possible to analyze spatial relations separately from vertical and horizontal directions. Although we do not assume all images are taken from the same height relative to the ground, it is reasonable to assume the images are taken from a human-eye view but not a bird-eye view. As the ground plane is detected, vertical spatial relations become stable to view changes. In contrast, horizontal spatial relations changes dramatically as the view point changes. In order to analyze the horizontal spatial relations independently, all regions are projected onto the ground plane that forms a two dimensional Euclidean space.

ERA relations can be expressed with extended interval algebra (EIA) (see table 5.1) in each dimension in a 2D Euclidean space based on Allen’s interval algebra [Allen and Koomen, 1983] by introducing center point of each interval. As a result, EIA has 27 basic relations (denoted by $B_{eint}$) which produce $27^2$ ERA relations (please refer to Chapter 5 for formal definitions of ERA). Then, the ERA relation for two regions $A$ and $B$ can be denoted by $\text{ERA}(A, B) = (\text{EIA}_x(A, B), \text{EIA}_y(A, B))$. We would like to reason about the change of directional relations between objects with respect to the
horizontal view change by applying ERA. This information will allow us to determine the order for better registration of the images.

**Definition 4** (region, region centroid, region radius). Let $PC$ denote the raw input point cloud. A region $a_l$ is a set of points $p_1, ..., p_n \in PC$ labeled by the same value $l$ by the segmentation algorithm. Let $c = (x_c, y_c, z_c)$ denote the region centroid, where

$$
x_c = \frac{\sum_{i=1}^{n} x_{p_i}}{n}, \quad y_c = \frac{\sum_{i=1}^{n} y_{p_i}}{n}, \quad z_c = \frac{\sum_{i=1}^{n} z_{p_i}}{n} 
$$

(6.1)

Let $r$ denote the region radius of a region $a_l$ and $\text{dist}(c, p_i)$ denote the Euclidean distance between region centroid $c$ and an arbitrary point $p_i \in a_l$.

$$
r = \max_{i \in \{1, ..., n\}} \text{dist}(c, p_i) \quad (6.2)
$$

Let $mbr$ denote the minimal bounding rectangle of a region. The $mbr$ will change with the change of views. As a result, the $EIA$ relation between two regions will change accordingly. By analyzing the change in $EIA$ relations, an approximate horizontal rotation level can be determined between two views. Before looking at the regions that are incomplete due to occlusion or noises during the sensing process,
we first research how the values of $r_x, r_x^+, r_y^-, r_y^+$ change assuming the regions are completely sensed. We identified a conceptual neighborhood graph of $EIA$ which includes all possible one-step changes with respect to horizontal rotation of the views (see figure 6.4).

**Definition 5** (view point, change of view). The view point $v$ is the position of the camera.

Let $v_1$ and $v_2$ denote two view points, and $c$ to be the region centroid of the sensed connected regions excluding the ground plane. Assuming the point cloud has been rotated such that the ground plane is parallel to the plane defined by x-axis and y-axis of a 3D coordination system. Let $v_{1xy}$, $v_{2xy}$ and $c_{xy}$ to be the vertical projection of $v_1$, $v_2$ and $c$ to the xy plane.

The change of view (denoted as $C$) is the angle difference between the line segments $c_{xy}v_{1xy}$ and $c_{xy}v_{2xy}$

**Definition 6** (symmetric EIA relation). Let $R \in B_{eint}$ to be an arbitrary EIA atomic relation. The symmetric EIA relation of $R$ (denoted by $symm(R)$) is defined as $R$’s axially symmetric atomic relation against the axis of symmetry formed by relations $\{cd, eq, cd\i\}$ in the conceptual neighborhood graph given in figure 6.4. For example, $symm(mol) = lomi$. Notably, the symmetric relation of $cd$, $eq$ and $cd\i$ are the same as themselves.

**Lemma 7.** Let $C_{cw\pi/2}$ denote the view change of $\pi/2$ clockwise from view point $v_1$ to $v_2$. Let $ERA_{ab_1} = (r_{x1}, r_{y1})$ and $ERA_{ab_2} = (r_{x2}, r_{y2})$ denote the ERA relations between region $a$ and $b$ at $v_1$ and $v_2$.

Assuming the connected regions are fully sensed, then $r_{x2} = symm(r_{y1}), r_{y2} = r_{x1}$ Similarly, if the view changes by $\pi/2$ anticlockwise, then $r_{x2} = r_{y1}, r_{y2} = symm(r_{x1})$

**Proof.** Lemma 7 can be simply proved by reconstructing a coordination system at each view point. □
Although the conceptual neighborhood graph indicates possible relation changing path for a pair of objects in one dimension, the way the changes happen depends on the rotation direction (clockwise or anti-clockwise) and their ERA relation before the rotation. For example, \( \text{ERA}(A, B) = (m, m) \) means \( \text{mbr}(A) \) connects \( \text{mbr}(B) \) at the bottom-left corner of \( \text{mbr}(B) \), thus if the view rotates anti-clockwise, \( \text{mbr}(A) \) tends to move upwards related to \( \text{MBR}(B) \) regardless of the real shape of \( A \) and \( B \), therefore \( EIA_y(A, B) \) will change from ‘\( m \)’ to ‘\( lol \)’ but not ‘\( b \)’. To determine the changing trend of ERA relations more efficiently, we combine \( \text{ERA} \) with cardinal direction relations (CDR) which describes how one region is relative to the other in terms of directional position.

The basic CDR contains nine cardinal tiles as shown in figure 6.5a. The symbol ‘\( B \)’ represents the relation ‘belong’, and the rest of the eight symbols represent eight cardinal directions, namely, north, northeast, east, southeast, south, southwest, west, northwest. Formally, the definition of CDR is given below based on [Skiadopoulos and Koubarakis, 2004]:

**Definition 8** (basic cardinal direction relation). A basic cardinal direction relation (basic CDR) is an expression \( R_1 : \ldots : R_k \) where:

1. \( 1 \leq k \leq 9 \)
2. \( R_1, \ldots, R_k \in \{ B, N, NE, E, SE, S, SW, W, NW \} \)
3. \( R_i \neq R_j, \forall 1 \leq i, j \leq k, i \neq j \)
4. \( \forall b \in \text{REG}, \exists a_1, \ldots, a_k \in \text{REG} \) and \( a_1 \cup \ldots \cup a_k \in \text{REG} \) (Regions that are homeomorphic to the closed unit disk \( (x, y) : x^2 + y^2 \leq 1 \) are denoted by REG).

If \( k = 1 \), the relation is called a single-tile relation and otherwise a multi-tile relation.

Similar to the extension from RA to ERA, we introduce center points to extend basic CDR to the extended CDR (detnoed as \( ECDR \), see figure 6.5b) in order to...
express inner relations and detailed outer relations between regions. Notably, a similar extension about inner relations of CDR was proposed in [Liu et al., 2005]. However, as we focus on \( \text{mbr} \) of regions in this problem, their notions for inner relations with trapezoids are not suitable for our representation.

![Figure 6.5: Cardinal direction relations](image)

In [Navarrete and Sciavicco, 2006], rectangular cardinal direction relations (RCDR) which combines RA and CDR was studied. As a subset of CDR, RCDR considers single-tile relations and a subset of multi-tile relations that represent relations between two rectangles whose edges are parallel to the two axes. We combine ERA and ECDR in similar way to produce the extended rectangular cardinal direction relations (ERCDR). Including all single-tile and multi-tile relations, there exists 100 valid relations (which are not all listed in this chapter) to represent the directional relation between two \( \text{mbrs} \).

### 6.4.3 Conflict Resolving and View Ordering

In this section, algorithms for identifying match errors and ordering the input view images will be introduced.

Basically, ‘one-to-one match error’ will lead to an inconsistency when estimating the view rotation angle using pairs of matched regions.

**Example 9.** Let \( \text{match}(a_1, b_1), \ldots, \text{match}(a_n, b_m) \) be a set of correct region matches between view \( a \) and view \( b \), and let \( \text{match}(a_{n}, b_{n}) \) be a wrong match, where \( n \notin \{1, \ldots, m\} \). Let \( \theta \) be the view rotation angle between \( a \) and \( b \). Let \( \{\hat{\theta}_{ij} \mid \forall i, j \in \{1, \ldots, m\}, i \neq j\} \) be the set of estimated rotation angles using the correctly matched pairs. And \( \{\hat{\theta}_{nk} \mid \forall k \in \{1, \ldots, m\}\} \) denotes the set of estimated rotation angles with incorrectly matched pairs involved.

In non-coincident cases, there will be a relatively large variance between the estimated angles in the two sets. The wrong matches can be detected by discovering different values.
Some relevant terms are defined first.

**Definition 10** (directional property of a single-tile relation). Each tile $t$ in ECDR has both a horizontal directional property $HDP(t) \in \{E,W\}$ and a vertical directional property $VDP(t) \in \{N,S\}$. The value is determined by the relative directional relation between the centroid of $t$ and the centroid of the reference region $b$. For example, $HDP(INE) = E$ and $VDP(INE) = N$.

**Definition 11** (directional property of multi-tile relation). The directional property of a multi-tile relation $mt$ is determined by majority single tile directional properties in the multi-tile relation, where $HDP(mt) \in \{E,M,W\}$ and $VDP(mt) \in \{N,M,S\}$, where $M$ represents ‘middle’ which appears when the counts of the single tile directional properties are equal. For example, $HDP(WMN: EMN: INW: INE) = M$ and $VDP(WMN: EMN: INW: INE) = N$.

Directional property can be used to estimate the trend of the relation change. Let a view point rotate clockwise,

- if $HDP = E$ and $VDP \in \{S,M\}$, the change trend is considered to be from south to north. The trend direction is vertical.
- if $HDP \in \{E,M\}$ and $VDP = N$, the change trend is considered to be from east to west. The trend direction is horizontal.
- if $HDP = W$ and $VDP \in \{N,M\}$, the change trend is considered to be from north to south. The trend direction is vertical.
- if $HDP \in \{W,M\}$ and $VDP = S$, the change trend is considered to be from west to east. The trend direction is horizontal.

One ERCDR relation can be transformed to the other by lifting the four bounding lines of the single/multi-tile. The distance $d$ between two ERCDR relations is calculated from horizontal and vertical directions by counting how many grids each boundary line lifts over. The distance is related to the direction of the view point change as well as the directional properties of the region.

However, with the same angle difference, the inner tiles take much fewer changes than the outside tiles. We introduce the *quarter distance size* to represent the angle change of $\pi/2$ for normalizing the distance between ERCDR relation pairs corresponding to the angle difference.

Based on lemma 7, we can infer the ERA relation between two regions after rotating the view point by $\pi/2$, $\pi$ and $3\pi/2$ either clockwise or anticlockwise. The ERA relation can be then represented by an ERCDR tile.

In order to calculate the distance between ERCDR tiles, we label each corner of the single tiles with a 2D coordinate with bottom-left corner of tile SW to be the origin $(0,0)$ (see figure 6.5b).

**Definition 12** (distance between ERCDR tiles). Let $t_1$ and $t_2$ be two ERCDR tiles. $x_1^-$ and $x_1^+$ denote the left and right bounding lines of $t_1$. $y_1^-$ and $y_1^+$ denote the top and bottom bounding lines of $t_1$. $x_2^-$ and $x_2^+$ denote the left and right bounding lines of $t_2$. $y_2^-$ and $y_2^+$ denote the top and bottom bounding lines of $t_2$. 

The unsigned distance is
\[ |d(t_1, t_2)| = |(x_2^+ - x_1^+) + (x_2^- - x_1^-)| + |(y_2^+ - y_1^+) + (y_2^- - y_1^-)| \] (6.3)

The sign of \( d \) is determined by looking at whether the change trend suggested by directional property is followed. If so, \( d(t_1, t_2) = |d(t_1, t_2)| \), else \( d(t_1, t_2) = -|d(t_1, t_2)| \).

If there is no change or the change is symmetric on the trend direction, \( d(t_1, t_2) = 0 \).

If there is a significant angle difference between the two tiles, the distance may not be accurate due to multiple path for the transformation. Here we introduce three more reference tiles by rotating the original tile by \( \pi/2 \), \( \pi \) and \( 3\pi/2 \) in turn.

**Definition 13** (quarter distance). Let \( t_1 \) be an ERCDR tile and \( t_1' \) be the ERCDR tile produced by rotating \( t_1 \) by \( \pi/2 \) either clockwise or anticlockwise. By applying lemma 7, the reference tiles can be easily mapped to ERCDR tiles. The quarter distance of \( t_1 \) is defined as:
\[ d_{qt} = d(t_1, t_1') \] (6.4)

**Definition 14** (normalized distance). Let \( t_1 \) be an ERCDR tile. \( t_{\pi/2}, t_{\pi} \) and \( t_{3\pi/2} \) denote the three reference tiles for \( t_1 \). Let \( t_2 \) be another ERCDR tile. The normalized distance \( d_{norm}(t_1, t_2) \) will be calculated in two parts:

1. The base distance \( d_{base} \).
   
   Let \( T = \{t_1, t_{\pi/2}, t_{\pi}, t_{3\pi/2}\} \).
   
   \[ d_{base}(t_1, t_2) = \begin{cases} 
   0 & \text{if } \arg\min_{t \in T}(|d(t, t_2)|) = t_1 \\
   1 & \text{if } \arg\min_{t \in T}(|d(t, t_2)|) = t_{\pi/2} \\
   2 & \text{if } \arg\min_{t \in T}(|d(t, t_2)|) = t_{\pi} \\
   3 & \text{if } \arg\min_{t \in T}(|d(t, t_2)|) = t_{3\pi/2} 
   \end{cases} \] (6.5)

2. The normalized distance \( d_{norm} \).
   
   \[ d_{norm}(t_1, t_2) = d_{base}(t_1, t_2) + \frac{d(\arg\min_{t \in T}(|d(t, t_2)|), t_2) + 1}{d_{qt}(t_1) + 1} \] (6.6)

The view images will be registered with the order derived from Algorithm 7. Unknown voxels will be assigned by applying the scene completion method in Zheng et al., 2015.

### 6.5 Stability Analysis

We use a modified version of the structural analysis method in Ge et al., 2017. A structure is in static equilibrium when the net force and net torque of the structure...
Algorithm 5: Multiple-to-one Conflict Resolving

1 Function MULTIToOneAdjust(M)
2
3 Input: A set of matched regions from two images
4 \{Match(a_i, b_j)|i, j \in \{1 \ldots n\}\}
5
6 Output: A set of adjusted matches M_{adjust}
7
8 \(M_{adjust} \leftarrow \{\}, \ Map_a \leftarrow \{\}\)
9 for \(i \leftarrow 0; i < n; i++\) do
10 if \(Map_a\).keys.contain(a_i) then
11 \(List_b \leftarrow Map_a\).getValue(a_i)
12 \(List_{combine} \leftarrow \{\}\)
13 for \(b_j\) in \(List_b\) do
14 if \(b_j\) connects with \(b_i\) then
15 \(List_{combine}\).append(b_j)
16 \(List_b\).remove(b_j)
17 \(b_{combine} \leftarrow \emptyset\)
18 for \(b_k\) in \(List_{combine}\) do
19 \(b_{combine} \leftarrow b_{combine} \cup b_k\)
20 \(List_b\).append(b_{combine})
21 else
22 \(Map_a\).append((a_i, \{b_i\}))
23 for \((a_i, List_{b_i})\) do
24 for \(b_{i_m}\) in List_{b_i} do
25 \(M_{adjust}\).append((a_i, b_{i_m}))
26 return \(M_{adjust}\)

equal zero. The static equilibrium is expressed in a system of linear equations \cite{Whiting et al., 2009}:

\[ A_{eq} \cdot f + \omega = 0 \]
\[ \|f^n\| \geq 0 \quad \text{(1)} \]
\[ \|f^s\| \leq \mu \|f^n\| \quad \text{(2)} \]  

\(A_{eq}\) is the coefficient matrix with each column stores the unit direction vectors of the forces and the torque at a contact point. To identify the contact points between two contacting objects, we will first fit a plane to all the points of the connected regions between the objects. We then project all the points to the plane and obtain the minimum oriented bounding rectangle of the points. The resulting bounding rectangle approximates the region of contact, and the four corners of the rectangle will be used as contact points. \(f\) is a vector of unknowns representing the magnitude of each force at the corresponding contact vertex. The forces include contact forces \(f^n\) and friction forces \(f^s\) at the contact vertex. The constraint 1) requires the normal forces to be positive and constraint 2) requires the friction forces to comply with the Coulomb model where \(\mu\) is the coefficient of static friction. The structure is stable
Algorithm 6: One-to-one Matching Error Detection

Function ONE_TO_ONE_ADJUST(M)

Input: A set of matched regions from two images a and b,
\{Match(a_i, b_j)|i, j \in \{1 \cdots n\}\}

Output: A list of error matches \(M_{error}\)

1. \(M_{adjust} \leftarrow \text{Multi2OneAdjust}(M)\)
2. \(Array_{dist} \leftarrow \{\{}\}\)
3. \(k \leftarrow M_{adjust}.\text{size}\)
4. for \(i \leftarrow 0; i < k; i++\) do
5.  \hspace{1em} for \(j \leftarrow i + 1; j < k; j++\) do
6.     \hspace{2em} \(Array_{dist}[i][j] \leftarrow (d_{\text{norm}}(\text{tile}(a_i, a_j), \text{tile}(b_i, b_j)))\)
7.     \hspace{2em} Mean \leftarrow \text{Mean}(Array_{dist})
8.     \hspace{1em} for \(i \leftarrow 0; i < k; i++\) do
9.         \hspace{2em} for \(j \leftarrow i + 1; j < k; j++\) do
10.            \hspace{3em} if \(|Array_{dist}[i][j] - Mean| \geq \epsilon\) then
11.                \hspace{4em} \(M_{error}.\text{append}((a_i, b_j))\)
12. return \(M_{error}\)

when there is a solution to the equations.

Using the structural analysis method, we can identify support relations between objects. Specifically, we are interested in identifying the core supporters [Ge et al., 2016] of each object in the scene. An object \(o_1\) is a core supporter of another object \(o_2\) if \(o_2\) becomes unstable after removal of \(o_1\). Given a contact between \(o_1\) and \(o_2\), to test whether \(o_1\) is the core supporter of \(o_2\), we first identify the direction vectors of forces and torque given by the contact on \(o_2\), and set them to zero in Eq. 6.7. This is equivalent to removing all the forces that \(o_1\) imposes on \(o_2\). If the resulting Eq. 6.7 has no solution, then \(o_1\) is the core supporter. We test each pair of objects in the scene and obtain a support graph. The support graph is defined as a directed graph with each vertex representing an object. There is an edge from \(v_1\) to \(v_2\) if \(o_1\) is a core supporter of \(o_2\).

6.6 Experiments

We first tested our method on the accuracy of ordering views by view angle changes. Then we show the method’s ability of identifying core supporters of an object in a structure.

6.6.1 Results on View Ordering

For view ordering, we use the data set from [Panda et al., 2016] which contains seven different scenes, and each scene contains 4-18 images from different views. In this dataset, the images for each view have a ground truth order which allows us to analyse the accuracy of our order prediction. Table 6.1 shows the results of view
§6.6 Experiments

Algorithm 7: View Ordering

Function ViewOrder($V$)

Input: A set of RGB-D pictures of the same static scene from different views, $V \leftarrow \{v_0, v_1, \ldots, v_n\}$

Output: Ordered list of views, $V_{ordered}$

1. $M \leftarrow \{\}$
2. for $i \leftarrow 1; i < n; i += 1$
   3. $M_{0i} \leftarrow \text{initialmatch}(v_0, v_i)$
   4. $M_{0i}.\text{remove}(\text{One2OneErrorDetection}(M_{0i}))$
   5. $M.\text{append}(M_{0i})$

List $dist \leftarrow$

6. for $i \leftarrow 1; i < n; i += 1$
   7. $l = M_{0i}.\text{size}$
   8. for $j \leftarrow 0; j < l; i += 1$
      9. for $k \leftarrow j + 1; k < l; k += 1$
         10. $\text{Array}_{dist}[j][k] \leftarrow (d_{\text{norm}}(\text{tile}(a_j, a_k), \text{tile}(b_j, b_k)))$
   11. List $dist.\text{append}(\text{Mean(Array}_{dist}))$

12. $V_{ordered}$ is gathered by ordering $V$ with ascending order of List $dist$

13. return $V_{ordered}$

<table>
<thead>
<tr>
<th>Scene [Panda et al., 2016]</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td># Images</td>
<td>5</td>
<td>11</td>
<td>6</td>
<td>3</td>
<td>5</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td># Correct Ordered</td>
<td>2</td>
<td>7</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>12</td>
<td>16</td>
</tr>
</tbody>
</table>

Table 6.1: View order results

ordering. Our algorithm works better when there are more objects in the scene, as the spatial relations tend to be unique in those cases. Also, if the scene is symmetric in terms of the object structure, i.e., the relations between objects in the structure are the same from two opposite views, the mismatching was not correctly detected sometimes because spatial relations are not distinguishable for both correct and incorrect cases.

6.6.2 Results on Core Supporter Identifying

For core supporter detection, we use our own data set which contains objects with special supporting relations (e.g. the top object supports the bottom object) as well as the data from [Panda et al., 2016]. Table 6.2 shows the core supporter detection results compared with the methods in [Panda et al., 2016]. Our algorithm is able to

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Method</td>
<td>72.5</td>
</tr>
<tr>
<td>Agnostic</td>
<td>65.0</td>
</tr>
<tr>
<td>Aware</td>
<td>59.5</td>
</tr>
</tbody>
</table>

Table 6.2: Support relation results
find most of the true support relations. In addition, our method is able to detect some special core supporter objects such as A in Figure 6.6 which is difficult to be detected by statistical methods.

In Figure 6.6, results on core supporter detection are presented. Notably, in the second row, we detected that even though object C is on top of object B, it contributes to the stability of B. Thus, C is a core supporter of B as well as A.

6.7 Conclusion and Future Work

In this chapter, we proposed a framework on identifying core supporting relations among a group of connected objects taking a set of RGB-D images about the same static scene from different view angles as input. We assume no knowledge about the objects and the environment beforehand. By qualitatively reasoning about the angle change between each pair of input images, we successfully identified the mismatches of the objects and sort the images by clockwise order. We use static equilibrium to analyse the stability of the whole structure and extract the core support relation between objects in the structure. We can successfully detect most of the support relations.

Core supporting relation is one major physical relation between connected objects. With the capability of analysing core supporting relations, the perception system is able to assist the AI agent to perform causal reasoning about consequences of an action applied on an object in a structure. Apparently this is only one aspect of physical relations that can be derived. In the future, more object features (e.g. solidity, density distribution, etc.) and relations between objects (e.g. containment, relative position, etc.) can be studied. On the other hand, the stability considered in this chapter is binary, i.e., it only cares about if the structure is stable or not (termed as
absolute stability), but does not model if the structure is easy to collapse (termed as relative stability). However, relative stability also plays an important role on safe manipulation. Relative stability is discussed in [Zheng et al., 2015], where the stability is modeled with gravitational potential energy. This method works fine with objects contacting each other via surfaces, but may miss important information when objects are placed in arbitrary poses (e.g. edge to surface contact). A more detailed relative stability analysis could be performed by combining symbolic reasoning which models configurations that cannot be easily reflected from quantitative methods. For example, informally, with the same height of mass centres, a cuboid placed flat should be more stable than an inclined cuboid which is supported from two edges.
Conclusion

In this thesis, a set of algorithms that attempt to complement high level perception capabilities of AI agents is demonstrated. A high level perception system is expected to have the capability of abstracting concepts and relations from sensory input from a variety of sources. In this thesis, we demonstrate the feasibility of such high level perception systems by focusing on spatial and physical perspectives which are two major aspects to be considered in the real world. Between the two aspects, spatial representation of entities and their relations is addressed first. This is mainly because spatial representation implies constraints for modelling and reasoning about physical process [Raper, 1999], such as connectivity between objects and the freedom of movement.

Besides the implicit physical constraints, a proper spatial representation of sensory data helps AI agents to communicate with each other in an abstraction level [Freksa, 1991], which is essential for tasks required cooperation among multiple agents in the same environment. For example, in a warehouse scenario, there is a large object which needs to be moved by at least two robots. Using a qualitative spatial description of the object may help the agents to locate the target at an entity level, whereas a coordination level identification of the object may failed due to noise and incomplete information.

In Chapter 3, we proposed a method to integrate measurements of complex spatial regions from different sensor networks to obtain more accurate information about the regions by reasoning with a combination of size and directional information and local optimisation. Specifically, the reasoning approach provides an initial match of two measurements in order to omit local minima. Then local optimisation is performed without concerning of being stuck at local minima. Spatial information from the integrated sensor measurements in the form of a symbolic representation is also extracted after a proper match is found. At this stage, our method deals with binary sensory input, i.e., if the sensed point is inside of a region. In the future, the input format can be extended to more general types such as continuous value by introducing additional (domain-specific) constraints for matching the regions.

In Chapter 4, under the assumption of accurate representation of regions is achieved, we further developed an algorithm to analyse and predict the trend of spatial change of detected regions at different states. This problem is solved with a hybrid manner which qualitatively matches components of the same region at different state and then predict the changing trend with Kalman Filter. An outer approximation is also proposed to guarantee the coverage of the predicted changes, so that the
method can be applied in situations that require high precision of prediction such as wildfire. Having the algorithms from the above two chapters, both static properties and dynamic changes of the entities detected from raw sensory can be analysed. The proposed method does not depend on any domain knowledge of the regions. However, domain knowledge may help improve the prediction result. Therefore, an interface could be designed to accept models of different domain knowledge to adjust the prediction. On the other hand, the format of the model of domain knowledge should also be unified to guarantee the compatibility with the prediction module.

Spatial knowledge provides the AI agents with analytical capabilities of the properties of spatial entities and their relations. However, if the agent would like to interact with entities in the environment, physical properties of entities and their relations must be represented as well. Based on constraints from qualitative spatial representation, we use stability analysis as an example to demonstrate how physical reasoning can be performed in a high level perception system in the next two chapters.

In Chapter 5, in addition to spatial reasoning, we proposed a qualitative calculus to model support relations between connected objects and perform rule-based qualitative reasoning to determine local stability in 2D scenarios. The usefulness of the proposed reasoning model is evaluated in Angry Birds to detect the weak point of a structure. A weak point is the point which, when hit, will cause maximal components in the structure to change their static state. An extension version of qualitative local stability analysis, i.e., qualitative global stability analysis which evaluate the stability of a group of connected objects is discussed in [Ge et al., 2016].

In Chapter 6, an end-to-end perception system that takes multi-view RGB-D images as input and produces a support graph with information about core supporters is proposed. Assuming the RGB-D images are sparsely taken and with an arbitrary order, traditional SLAM or image registration algorithms may not work due to the lack of the corresponding descriptors. We hereby introduce an extended rectangle cardinal direction relation to qualitatively measure the view angle difference of each pair of images. Our method is able to detect mismatches of segmented objects between two images and sort the images clockwise to gather a better registration result. We then use static equilibrium to detect the core support relations between connected objects and generate an overall support graph of the structure. We currently use LCCP for segmentation of objects. A more sophisticated algorithm could be applied to improve the segmentation accuracy. For example, semantic segmentation such as [Shelhamer et al., 2017; Chen et al., 2017] could be adopted. In addition, the proposed method is able to reason about the relative solidity of the objects, i.e., the density of the objects is inferred from the mechanical analysis. In the future, the solidity could be further determined by having a robot to interact with the objects.

This thesis proposed some key capabilities of a high level perception system including spatial knowledge representation from multi-source sensory data from single state, spatio-temporal reasoning over a sequence of input states and reasoning under spatial and physical constraints. Notably, integrating high level perception algorithms into a standalone sensor is trending in many areas of the industry. For example, a SmartEye camera which accurately detects traffic-related objects (e.g. turning sign, speed limit sign) with semantic descriptions is applied in self-driving cars by Mercedes, BMW, etc [1]. The high level perception algorithms in this thesis

---

1 goo.gl/LWGngt
have potential applications for GIS systems with evolving regions, warehouse robots, rescue robots, etc.

7.1 Future Work

The research work demonstrated in this thesis focuses on adopting QR methods to enable high-level machine perception. We have developed a set of reasoning approaches to infer hidden commonsense information based on the superficial relations between entities gathered directly from sensors. For example, some general physical concepts, such as gravity and inertia, have been modelled and used for understanding physical relations between the entities in a scenario as well as the causal relation between two states of the same scenario. Although the effectiveness of our method has been shown in some real world and simulation experiments, there are some possible directions to extend this work.

7.1.1 The Integration of Domain Knowledge

A major assumption of this work is that little domain knowledge is involved in the reasoning process, i.e., only general knowledge has been used. Intuitively, properly including domain knowledge will improve the quality of the perception results. In different scenarios, the representation of the domain knowledge could be varied.

In the cases where gravity plays an important role, the physical properties of the objects, such as density and friction factor, will affect the reasoning system directly. Therefore, in these scenarios, physical properties of the objects should be tuned gradually by observing the difference between the reasoning result and the ground truth. There are existing methods about learning physical properties of the objects. For example, Wu et al. [2015] combine an internal physical simulator with deep learning to learn object properties from video observations; in robotics, the properties of objects can be learnt interactively by allowing the robot to actually play with the objects [Sinapov et al. 2009; Guler 2017]. Notably, Ge et al. [2018] propose a method that combines QR and simulation to estimate the properties of the objects in a stack-like structure.

There are also cases where the models about the domain have been well established, such as wildfire propagation [Clarke et al. 1994]. Different from physical properties of the objects which may affect the correctness of the reasoning results, these models can be treated as an additional layer of the reasoning system to improve its accuracy. A common problem for these models are that they usually only work under certain constraints or in certain areas. Thus, representing the applicability of the models is a challenge of automatic integration.

7.1.2 The Combination with Deep Reinforcement Learning

In a greater vision, one of the ultimate goals of QR systems or even AI systems is to have an agent to behave rationally as humans do [Russell and Norvig 2003b]. Despite the successful examples of QR systems discussed in Chapter 1, the main problem for QR methods is the lack of capability of modelling processes and actions in an environment with high freedom and uncertainty [Forbus et al. 1991]. And
in general, a common drawback of symbolic methods is the symbol grounding problem [Harnad, 1990], which indicated the fact that the hand-crafted representations are biased because of the designers’ diffident cognition of the real world. From the other perspective, reinforcement learning (RL) [Sutton and Barto, 2017] attempts to mimic the behaviour optimisation procedure of animals through a action-reward feedback loop. The limitation of RL is similar to QR. The features and rewards are usually manually designed and the learnt policies can only be applied to a restricted environment (mostly with low-dimensional discrete states). To overcome these limitations, deep reinforcement learning (DRL), which adopts the capabilities of deep neural network (DNN) [LeCun et al., 2015] to learn policies directly from high dimensional input, has demonstrated its usefulness in areas such as gaming [Mnih et al., 2015; Silver et al., 2016] and robotic manipulation [Yu et al., 2018].

On the other hand, DRL has the common shortcomings of deep learning methods, including long training time, lack of model transparency, lack of capability of high level reasoning [Garnelo et al., 2016]. To overcome these shortcomings, Garnelo et al. [2016] proposed a framework termed deep symbolic reinforcement learning (DSRL) which uses DNN to map raw input to symbolic representation (probabilistic first-order logic in this work) as RL input. Therefore, a compositional structure is enabled in this framework for causal reasoning in the later step. As a similar idea, auxiliary rewards are introduced [Mirowski et al., 2016; Riedmiller et al., 2018] to solve the problem of sparse reward in DRL with high dimensional input. Specifically, some auxiliary rewards are designed from human experience including interesting processes and configurations of object relations. The above mentioned work demonstrates a promising research direction of combining model-based method with DRL to let the agent perform rationally through “thinking rationally”. Our work on high level perception may serve an important role in the hybrid framework, especially when there is a demand of more sophisticated spatial and physical representation of the world.
Bibliography


CRAIK, K. J. W., 1943. The nature of explanation. Students Quarterly Journal, 14, 55 (1943), 44. (cited on page 14)


Murphy, R., 2000. *Introduction to AI robotics*. MIT press. (cited on page 1)


