THE COMPUTATIONAL PERCEPTION OF SCENE DYNAMICS

by

Richard Mann

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto

Copyright © 1998 by Richard Mann

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-35237-4

Abstract

The Computational Perception of Scene Dynamics

Richard Mann

Doctor of Philosophy

Graduate Department of Computer Science

University of Toronto

1998

Understanding observations of image sequences requires one to reason about *qualitative scene dynamics*. For example, on observing a hand lifting a cup, we may infer that an ‘active’ hand is applying an upwards force (by grasping) on a ‘passive’ cup. In order to perform such reasoning, we require an *ontology* that describes object properties and the generation and transfer of forces in the scene. Such an ontology should include, for example: the presence of gravity, the presence of a ground plane, whether objects are ‘active’ or ‘passive’, whether objects are contacting and/or attached to other objects, and so on. In this work we make these ideas precise by presenting an implemented computational system that derives symbolic force-dynamic descriptions from video sequences.

Our approach to scene dynamics is based on an analysis of the Newtonian mechanics of a simplified scene model. The critical requirement is that, given image sequences, we can obtain estimates for the shape and motion of the objects in the scene. To do this, we assume that the objects can be approximated by a two-dimensional ‘layered’ scene model. The input to our system consists of a set of polygonal outlines along with estimates for their velocities and accelerations, obtained from a view-based tracker. Given such input, we present a system that extracts force-dynamic descriptions for the image sequence. We provide computational examples to demonstrate that our ontology is sufficiently rich to describe a wide variety of image sequences.

This work makes three central contributions. First, we provide an ontology suitable for describing object properties and the generation and transfer of forces in the scene. Second, we provide a computational procedure to test the feasibility of such interpretations by reducing the problem to a feasibility test in linear programming. Finally, we provide a theory of preference ordering between multiple interpretations along with an efficient computational procedure to determine maximal elements in such orderings.

Acknowledgements

Studying in the Computer Science Department at the University of Toronto has been a very rewarding experience. I would like to thank the people who have supported or encouraged me in my work there.

First and foremost, I would like to thank my advisor, Allan Jepson. During the past six years, Allan has provided constant support (both academic and personal), meeting with me at least once a week, and often more. Right from the beginning, Allan treated me as a colleague, encouraging me to experiment and ask questions. In addition, Allan has always been ready to share ideas and to get directly involved in students' research. I can safely say that the work described in this thesis would not have been possible without Allan's help and encouragement.

Second, I would like to thank Jeff Siskind. Jeff spent two and a half years as a visiting professor at Toronto. During that time, he gave unselfishly of his time. In addition to providing a sounding board for many of the ideas in this thesis, Jeff taught me how to program in Lisp and Scheme, and helped me to organize my ideas. A large part of my motivation for building a computational implementation of the ideas described in this thesis comes from my interaction with Jeff.

Third, I would like to thank the members of my thesis committee: John Tsostos, Ray Reiter, and Geoff Hinton. In addition to serving on my committee, these professors were valuable intellectual resources throughout my studies. As head of the vision group, John Tsotsos encouraged interaction among the students. Ray Reiter welcomed me into the Knowledge Representation group, and encouraged me to get involved with the Cognitive Robotics project. Geoff Hinton welcomed me into the Neural Networks group. All of these exchanges were beneficial since they broadened my interests, particularly during the first few years of my studies.

I would also like to thank the members of my external committee: Patrick Bennett (Psychology) and Whitman Richards (MIT). In addition I would like to thank Whitman Richards for many helpful discussions, and for inviting me to present my work at MIT.

For many useful discussions (thesis related and otherwise), I would like to thank my fellow students, in particular James MacLean, Sean Culhane, Chakra Chennubhotla, and Tom El-Maraghi. I would also like to thank Michael Black (Xerox PARC) and Dimitri Metaxas (U Penn) for supporting my work.

Finally, I would like to thank my mother and father for their support, my brother, Steve, for many technical discussions, and my wife, Sarah, for many editorial suggestions.

Dedication

This thesis is dedicated to my wife, Sarah, for her love and encouragement.

Contents

1	Introduction	1
1.1	The Problem	1
1.1.1	Qualitative Scene Dynamics	3
1.2	Theoretical Framework	3
1.2.1	The Computational Perceiver	3
1.3	Implementation of the Theory	4
1.4	Central Claims	4
1.5	Outline of the Dissertation	5
2	Related Work	6
2.1	Psychological evidence for the perception of dynamics	6
2.2	Psychological and linguistic theories	9
2.3	Computational approaches	10
2.3.1	Template-based approaches	10
2.3.2	Knowledge-based systems	11
2.3.3	Qualitative physical models	11
2.3.4	Function-based recognition	12
2.3.5	Summary	12
	Do humans perceive real physics?	13
	Physics-based simulation or scene perception?	13
	How does this compare with other “physics-based” approaches?	13
	How does this relate to work in knowledge representation?	14
3	Overview of the System	15
3.1	An Outline of the Approach	16
3.2	Preferred Explanations	18
3.3	Implementation and Limitations	21
4	Representation	23
4.1	Knowledge Representation Framework	23

4.2	Representation for the Motion Domain	24
4.2.1	Objects	24
4.2.2	Kinematic Model	25
	Contact	25
	Attachment	30
4.2.3	Dynamic Model	31
	Mass and Mass Distribution	31
	Forces	31
	Ground Objects	31
	Force Generators (Motors)	31
4.3	Interpretations	32
5	Feasible Interpretations	35
5.1	Introduction	35
5.2	Testing Feasibility by “Force balancing”	35
5.2.1	Forces and Torques	36
5.2.2	Contact Conditions	36
5.2.3	Mass and Mass Distribution	37
5.3	Reduction to Linear Programming	39
5.3.1	Linear Programming Problem	41
5.3.2	Problem complexity	42
5.4	Examples of the force-balancing procedure	43
5.5	On the Relationship Between Assertions and Feasibility Constraints	46
5.6	Summary	48
6	Preferred Interpretations	50
6.1	Preferences	50
	Example	51
6.1.1	Aside: Qualitative Probabilities and Preferences	51
6.2	Formalism	53
6.2.1	Preference Ordering	53
6.2.2	Preferred Interpretations	54
6.2.3	Preferences for the Dynamics Domain	54
6.3	Computing Preferred Interpretations	55
6.3.1	Exploiting Domain Structure	56
6.3.2	Partially Minimized Interpretations	57
6.3.3	Algorithm	58
	Correctness of Algorithm	60
6.4	Summary	61

7	Implementation	62
7.1	Representation	62
7.1.1	Objects	63
7.1.2	Contacts	63
7.1.3	Attachments and Motors	64
7.1.4	Interpretations	65
7.2	Hypothesis Generation	65
7.2.1	Contacts	65
	Polygon Intersection	66
	Motion at Contact Regions	68
7.2.2	Attachments	69
7.2.3	Motors	71
7.2.4	Summary	72
7.3	Testing Feasibility of Interpretations	72
7.4	Search for Preferred Interpretations	72
7.4.1	Generalizing the search to deal with multiple contact sets	75
7.5	Summary	78
8	Experiments	79
8.1	Sequences	79
8.2	Processing	85
8.3	Results and Discussion	91
8.3.1	Anomalies	101
8.4	Summary	102
9	Integrating Information Over Time	103
9.1	Ascribing Properties to Objects	103
9.2	Sequence Processing	104
9.3	Preliminary results	105
9.4	Summary	106
10	Conclusion	111
10.1	Contributions of this work	111
10.2	Extensions	112
10.3	Implications and Future Work	113
10.3.1	The role of force-dynamic representations in vision	113
10.3.2	Designing computational perceivers: The link between knowledge representation and perception	113
10.3.3	Will the computational perceiver approach generalize to new domains? . .	114

10.4 Conclusion	115
A Object Tracking	116
A.1 View-based Tracking	116
A.2 Robust Object Tracking by Mixture Models	117
A.2.1 Imaging Model	117
A.2.2 Mixture Model	118
A.2.3 Implementation	119
Additional Details of the Implementation	120
A.3 Summary	121
B Motion Estimation	122
B.1 Robust Interpolation by Mixture Models	122
B.1.1 Additional Details	123
B.1.2 Examples	124
B.2 Motion Estimation	126
Bibliography	127
References	127

List of Figures

1.1	Example: processing of the coke image sequence.	2
2.1	Perception of the causal interaction between objects.	7
2.2	A frame from the animation prepared by Heider & Simmel.	8
3.1	Illustrative example: the coke image sequence.	15
3.2	Tracked polygons for the coke sequence.	17
3.3	Preference ordering for the interpretations of a single frame of the coke sequence.	19
4.1	Contact types in layered scene model consisting of 2D polygonal objects.	28
4.2	Contact conditions for 2D polygonal objects.	29
5.1	Bounds on the moment of inertia for a two-dimensional object.	38
5.2	Bounds on the magnitude of the frictional force at a contact point.	40
5.3	Force balancing for a sliding box.	44
5.4	Force balancing for a tipping box.	45
5.5	Constraints for resting block.	47
5.6	Constraints for a block sliding on a horizontal surface.	49
7.1	Contact types used in the implementation.	64
7.2	Polygon intersection process.	67
7.3	Abutments inconsistent with the layered scene model.	69
7.4	Removal of <i>vanishing</i> contacts.	70
7.5	Attachment relations for the layered scene model.	70
7.6	Motor assertions in the layered model	71
7.7	Illustration of the search process for a frame of the coke sequence.	74
7.8	Illustration of the search process for a frame of the cars sequence	76
7.9	Preference ordering during the “approach” phase of the coke sequence.	77
7.10	Cases where there could be multiple maximal contact sets	78
8.1	coke image sequence.	80
8.2	cars image sequence.	81

8.3	hit image sequence.	82
8.4	arch image sequence.	83
8.5	tip image sequence.	84
8.6	Tracking results for the coke image sequence.	86
8.7	Tracking results for the cars image sequence.	87
8.8	Tracking results for the hit image sequence.	88
8.9	Tracking results for the arch image sequence.	89
8.10	Tracking results for the tip image sequence.	90
8.11	Motion estimation results for the coke sequence.	92
8.12	Motion estimation results for the cars sequence.	93
8.13	Motion estimation results for the hit sequence.	94
8.14	Motion estimation results for the arch sequence.	95
8.15	Motion estimation results for the tip sequence.	96
8.16	Legend of symbols used to denote assertions.	97
8.17	Preferred interpretations for selected frames of the coke sequence.	97
8.18	Preferred interpretations for selected frames of the cars sequence.	98
8.19	Preferred interpretations for selected frames of the hit sequence.	99
8.20	Preferred interpretations for selected frames of the arch sequence.	100
8.21	Preferred interpretations for selected frames of the tip sequence.	100
8.22	Preferred interpretations for frame 30 of the tip sequence (all types of motors). .	101
9.1	Collision detection process for image sequences.	105
9.2	Starts and stops of sliding motion.	106
9.3	Temporal integration results for the coke sequence.	107
9.4	Temporal integration results for the cars sequence.	108
9.5	Temporal integration results for the hit sequence.	109
9.6	Temporal integration results for the tip sequence.	109
A.1	Illustration of the object tracking process.	117
B.1	Example of the polynomial fitting process	125
B.2	Another example of the polynomial fitting process	125

Chapter 1

Introduction

1.1 The Problem

In this work, we consider the problem of *motion understanding*. An example of the type of problem we are considering is shown in Figure 1.1(a). In this sequence, a hand reaches for, grasps, and lifts a coke can off a table top. Given this sequence, we form the natural inference that the hand is lifting the coke can. Even given a sparse representation, consisting of only the polygonal outlines of the objects (see Figure 1.1(b)), we still arrive at the same inference. While these interpretations may appear obvious, we are making a number of *implicit inferences* when watching this movie. First, although there is no direct perceptual evidence for it, we infer that the can as a ‘passive’ object that is supported by the table. Second, we infer that the hand is an ‘active’ object that is lifting the coke can. Finally, in order to lift the can, we infer that the hand must be attached to the can. In this thesis, we construct a computer system that can perform these types of inferences given this type of polygonal data.

In order to perform this kind of reasoning, we require some physical knowledge about the scene. In this work we propose a representation of the scene based on Newtonian mechanics of a simplified scene model. Such a representation includes, for example, the presence of gravity, the presence of a ground plane, whether objects are active or passive, whether objects are attached to each other, and so on. We make these ideas precise by presenting an implemented computational theory that extracts such symbolic descriptions from image sequences. Our system takes as input a polygonal representation of the objects along with estimates of their velocities and accelerations. (This input is provided by an object tracker which will be described later.) The system generates as output conceptual descriptions as shown in Figure 1.1(c). The large circle denotes the interpretation of the hand as an ‘active’ object, while the large dots denote attachment between the hand and the can.

Figure 1.1: Example: processing of the coke image sequence. (a) The sequence depicts a hand is reaching for, grasping, and lifting a coke can off a table top. (b) Sparse representation consisting of the polygonal outlines of the object parts. (c) Interpretation formed by the system. The open circle at the center of the hand denotes an 'active' object. The large dots denote attachment points between the hand and the can. The small dots denote contact points between the objects.

1.1.1 Qualitative Scene Dynamics

As illustrated by the above example, understanding observations of interacting objects requires one to reason about qualitative scene dynamics. In particular, the observer must postulate the existence of object masses, and forces that act on objects. The observer must then check that there exists at least one assignment of masses and forces that is consistent with the observations, the laws of Newtonian mechanics, and an interpretation of which objects are active and/or attached.

It is important to note that while the proposed system is based on an explicit quantitative theory of Newtonian mechanics, the interpretations produced by the system are *qualitative* since they refer to entire classes of solutions (e.g., whether objects are active or passive, whether objects are contacting, attached, or disjoint, etc). For a concrete example of the qualitative nature of the interpretations, consider the interpretations in Figure 1.1(c) again. While the system has inferred that the hand must be an active object, it has not reported the exact mass of the hand or the can. Nor has it reported the exact force between the hand and the can. It has merely reported that, subject to some general consistency constraints (such as the fact that objects have positive, nonzero mass), the objects in the scene obey the laws of Newtonian mechanics.

As we will argue in Chapter 3, the exact form of the representation is not critical as long as:

1. The representation is sufficiently rich to describe a wide variety of dynamic phenomena.
2. The components of our representation can be translated into constraints that can be tested by our dynamic theory.

1.2 Theoretical Framework

Psychology researchers have argued that perception is a process of inference involving knowledge and intelligence. For example, Gregory (1970) describes perception as “finding the most probable interpretation of sensory data, in terms of the world of objects”. Rock (1983) states: “Perception is the outcome of a process unconscious inference, problem solving, and the building of structural descriptions of the external world. Within this must be included the notion of description, solution finding, solution acceptance, and preference.” While computational models are not specified, the message is clear: to model perception we must have an internal representation of the external world and a way of matching this representation to our sensory data.

1.2.1 The Computational Perceiver

In this thesis, we provide a computational approach to perception motivated by these arguments. Our approach involves the specification of three general components considered essential for any

perceptual system. First, we require an *ontology* that specifies our representation for the domain. Second, we require a *domain theory* that specifies which interpretations are consistent with the system’s world knowledge. Third, since the perception problem is typically under-constrained (i.e., there are multiple interpretations consistent with the sensor data), we require some form of *preferences* to select plausible interpretations. Together these provide a computational definition of a *perceiver*, for which percepts are defined to be the maximally preferred interpretations consistent with both the sensor data and the system’s world knowledge (Jepson & Richards, 1993).

1.3 Implementation of the Theory

In this thesis we show how the computational perceiver framework can be applied to the perception of scene dynamics.

As described above, the first part of the proposed system, the *ontology*, is based on the qualitative dynamic properties of the scene objects. As we will detail in Chapter 4, our representation includes: the presence of gravity, the presence of a ground plane, whether objects are ‘active’ or ‘passive’, whether objects are contacting and/or attached to other objects, and so on. Collectively, we refer to a set of hypotheses that completely specify the properties of interest in the scene as an *interpretation*.

For the second component of our system we use a domain theory based on the Newtonian mechanics of a simplified scene model. Specifically, we say that an interpretation is *feasible* if there exists a set of forces and masses that explain the observed accelerations of the scene objects. As we will show in Chapter 5, if we consider scenes containing rigid bodies in continuous motion, such a feasibility test can be reduced to a linear programming problem.

In general, given feasibility conditions alone, there will be multiple interpretations consistent with the observed data. For example, a trivial interpretation can always be obtained by making every scene object active. In order to find informative interpretations we would like to choose interpretations that make the fewest assumptions about the participant objects. To do this, we use a set of *preferences* to choose among interpretations. As we will show in Chapter 6, we can use preference rules to choose interpretations that describe the “simplest” scenes—that is, scenes containing the fewest active objects and attachments between objects.

1.4 Central Claims

This thesis makes two central claims:

1. *Qualitative scene dynamics* provides a useful representation for describing a wide range of motion sequences.

2. The *computational perceiver* provides a useful framework for perception problems in computational vision.

In the remainder of the thesis these claims are substantiated by constructing a suitable dynamic representation for describing motion in image sequences and by constructing a computational perceiver for this domain. While this is not a guarantee of the validity of these claims in general, it does demonstrate the applicability of the approach to a sufficiently complex and nontrivial domain.

1.5 Outline of the Dissertation

An outline of the dissertation is as follows.

Chapter 2 provides a brief description of related work.

Chapter 3 provides a brief, self-contained description of the system, including a detailed description of the sample sequence shown in Figure 1.1.

Chapters 4–6 describe the three components of our computational perceiver for scene dynamics, namely the Representation, Feasibility Test, and Preference Theory.

Chapter 7 describes the implementation of the base system that considers the instantaneous motion of objects in each frame of the image sequence.

Chapter 8 describes experimental results for several image sequences.

Chapter 9 shows how the base system can be extended to process continuous sequences by integrating inferences over time.

Chapter 10 summarizes the results and concludes with a discussion of how scene dynamics, and the computational perceiver in general, can be used in computational vision.

Finally, the Appendices describe the image processing details. Appendix A describes the object tracker, while Appendix B describes the process used to estimate the object motions given the object positions from the tracker.

Chapter 2

Related Work

In this chapter, we provide a brief review of related work on motion understanding and scene dynamics. We begin with a review of psychological evidence for motion understanding and the perception of scene dynamics. Next, we consider various psychological and linguistic theories for event recognition and motion understanding. Finally, we summarize various computational approaches to the problem.

2.1 Psychological evidence for the perception of dynamics

Several psychology researchers have argued that real or imagined forces among objects provide a basis for our conceptual description of the world.

Michotte (1963) showed that people perceive a causal connection between the motions of objects. To demonstrate this, Michotte constructed animations of simple objects exhibiting linear motions. The first animation, called *launching* (Figure 2.1(a)), depicts a collision between two objects. The first object (object A) moves towards the second object (object B). When object A reaches object B, object A stops and object B starts moving. When shown this display, subjects reported that the first object “pushed” or “launched” the second object. A second animation, called *entraining* (Figure 2.1(b)) consists of the same two objects, except that on contact, the first and second object move together. When shown this display, subjects reported that the first object “carried” the second object. Michotte found that by introducing a delay between the time of contact and the time of the motion of the second object, the perception of causality was lost; instead subjects reported that the event consisted of two separate motions. Based on these experiments, Michotte argued that humans directly perceive causal events. That is, rather than than perceiving the motions of the individual objects, subjects perceive single causal events based on the objects’ combined motions. Furthermore, since we still perceive causality when abstract objects such as dots are used, our perception does not depend on the specific properties of the objects. Michotte refers to this direct perception of causality as *phenomenal causality*.

Figure 2.1: Perception of the causal interaction between objects. (After Michotte (1963).) (a) *Launching* event. (b) *Entraining* event. (The arrows below the objects denote their velocities.)

As well as perceiving concrete mechanical interactions, it appears that humans also perceive more abstract causal relations among objects. In a compelling experiment, Heider and Simmel (1944) showed subjects a simple animated movie consisting of the coordinated movements of simple geometric objects (a circle and two triangles) in a simple scene containing a “room” with a “door”. (A frame from the movie is shown in Figure 2.2.) Even though the objects convey little or no information in their static shapes, observers arrive at surprisingly similar interpretations about the properties of the objects, and about the story conveyed by the animation. Heider & Simmel argue that our perception is based on the *origin* (the object that is causing the motion) and the *roles* of the objects (their goals and their current state). For example, if an object (such as the small triangle) and the door move together, we may not know whether the door or the triangle is generating the motion. However, in the animation, we perceive that the triangle is *opening* the door (by pushing or pulling), because we attribute the origin (the cause of motion) to the triangle. Similarly, if we observe the large triangle following the small triangle, we may say that either the second object is chasing the first, or that the first object is leading the second, depending on the roles of the objects. However, in the animation, we perceive that the large triangle is *chasing* the small triangle, because we have previously assigned an “aggressor” role to the large triangle. In summary, it appears that our perception is based not only on *mechanical causality*, based on the physical interactions among objects, but also *psychological causality*, based on the roles of the objects.

In an extensive series of experiments Johansson (1975) showed that humans can perceive rich motion structure from so-called *moving light displays*. In these experiments, small lights (or reflectors) are attached to actors’ bodies in key positions, such as the joints. Actors are then filmed performing various actions such as walking, dancing, and so on. As well as perceiving kinematic structure (jointed objects) from moving light displays, subjects are also able to perceive subtle differences in motions, such as the identity or gender of the actor based on

Figure 2.2: A frame from the animation prepared by Heider and Simmel (1944). The animation consists of the coordinated motion of three objects (a large triangle, a small triangle and a circle) in a simple scene consisting of a rectangle (the “house”) and an opening (the “door”).

differences in their gait (Cutting, Proffitt, & Kozlowski, 1978). There is also evidence that subjects can perceive dynamic information (implied forces) in moving light displays. For example, Runeson and Frykholm (1981) showed that people perceive the relative mass of objects when shown moving light displays of people lifting objects.

Freyd, Pantzer, and Cheng (1988) provide evidence that humans perceive the static forces between objects in equilibrium. Specifically, when subjects are presented with an image of a supported object, followed by one in which the same object is unsupported, subjects tend to “imagine” that the object has moved down slightly, as if it were under the influence of gravity. In related work, Freyd (1987) showed that humans have a representation of *perceptual momentum*. Specifically, if shown a sequence of moving figures, subjects expect the objects to continue moving with similar velocities in subsequent frames.

Arnheim (1988) has argued that dynamic representations play a central role in our perception of art. In particular, Arnheim suggests that *visual dynamics* – the implied forces in static scenes – provides a useful description of pictures. For example, in describing a photograph of an apparently unsupported object, Arnheim suggests that we perceive dynamic “tension” because the object appears as if it is about to fall over.

Finally, there has also been considerable work in the development of sensitivity to so-called force-dynamic concepts. Spelke and Valle (1993) provide evidence that infants develop representations of qualitative kinematics such as spatiotemporal coherence, object substantiality, object persistence, and so on, at an early age. Leslie and Keeble (1987) provide evidence to suggest that infants develop a sensitivity to causal interactions of the type described by Michotte.

In summary, we feel that these experiments, especially those of Michotte (1963), Heider and Simmel (1944), and Johansson (1975), provide significant motivation for our current approach. In particular, since it is possible to convey rich dynamic structure using very simple geometric models, one would expect to be able to reverse this process to extract useful dynamic information

from simple geometric models of real scenes.

These examples also serve to show the wide range of representations possible with force-dynamic representations. We can view these as being on a continuum, going from *concrete* representations of real forces in scenes, to providing conceptual descriptions of *events* and *actions*, to describing abstract cognitive concepts such as *goals* and *intentions*. In this work, however, we focus on only the concrete representation of real forces in scenes. We will return to the issue of how to use these concrete force-dynamic descriptions in higher-level conceptual descriptions in the Conclusion.

2.2 Psychological and linguistic theories

There have been many attempts to define motion events in the linguistics literature. For example, Miller (1972) and Miller and Johnson-Laird (1976) define several English motion events in terms of a set of motion predicates. (See (Schank, 1973) for another proposal.) While it is tempting to try to use these verb definitions in an event recognition system, the definitions are not explicit enough to be implemented computationally. The main problem is that the definitions are not expressed in terms of *perceivable* features (i.e., features that can be extracted from sensory input). For example, in order to define simple verbs like *throw*, Miller relies on the use of abstract (and undefined) concepts like *move* and *cause*. (See (Siskind, 1992) for a discussion.)

Rather than defining individual motion events, Talmy (1988) proposes a general representation object interaction based on their “force-dynamic” properties. Specifically, to represent interactions, Talmy proposes primitive concepts such as: the exertion of a force, resistance to such an exertion and the overcoming of such a resistance, blockage of a force and the removal of such blockage, and so on. These concepts are used to describe a number of linguistic concepts such as *helping*, *hindering*, and so on. In addition to describing *physical* process, Talmy argues that such a representation could also be used to describe abstract *psycho-social* concepts such as *requiring* and *forbidding*.

Finally, Leslie (1993) describes a general theory in which he attempts to distinguish “agents” from other physical objects. Specifically, Leslie proposes that agents can have the following different types of properties:

- *Mechanical properties*. Agents have an internal and renewable source of energy.
- *Actional properties*. Agents act in the pursuit of goals.
- *Cognitive properties*. Agents hold certain attitudes or beliefs.

Leslie argues that the learning and perception of these properties pose distinct information processing problems, and that the inferences at each level are built upon those of the previous level.

While none of these works provides an explicit computational theory for perceiving or reasoning about dynamics, they certainly provide motivation for such a theory. We believe that the computational approach presented in this thesis provides a framework to explore some of the ideas put forth in the linguistics community. As mentioned in the previous section, however, we emphasize the fact that our work here will be restricted to inferring the *mechanical properties* of objects.

2.3 Computational approaches

2.3.1 Template-based approaches

Much of the early work in motion understanding focussed on extracting conceptual descriptions from image sequences based on the spatiotemporal features of the input. The objective of these systems was to describe the motions of objects by a small set of motion categories that are presumed to be useful for describing and/or recognizing events in scenes (see (Badler, 1975; Borchardt, 1985; Nagel, 1988; Neumann, 1989; Borchardt, 1994) for examples). We refer to these approaches as *template-based* since they attempt to describe events based on simple spatiotemporal features of the input rather than some underlying conceptual or physical knowledge about the domain.

Template-based approaches have recently seen a revival in the form of *hidden markov models* (*HMMs*). Like the template-based approaches, *HMMs* recognize events based on the time course of features such as the positions and motions of the participant objects (Siskind & Morris, 1996; Brand & Oliver, 1996). Unlike the previous approaches that had fixed motion templates, however, hidden markov models are not limited to a specific domain, and can be trained to recognize different motion templates.

In contrast to the physics-based approach presented in this thesis, these template-based approaches have no underlying representation for object properties (such as active or passive objects, contact, or attachment) that are not directly observable from the sensory input. While one could argue that a template-based method such as a hidden markov model could represent these properties, more observations of the events would be required to train the system since the model is not exploiting domain-specific knowledge about Newtonian mechanics. As a concrete example of the limitation of template-based approaches, consider the classification of *pulling* vs. *pushing* events on a horizontal surface. While there is no direct perceptual evidence for it, a physics-based approach is able to infer attachment between the two objects in the pulling event. For a template-based method to differentiate between these events, it would have to recognize that what is important is not the direction of the object motions themselves, but the direction of the acceleration of the objects *relative to the contact region between the two objects*. Furthermore, the relevant features would become even more complicated if different types of scenes were considered, such as scenes involving sloped surfaces, or multiple attached objects.

In summary, we argue that in order to understand the structure of motion sequences, we will require a significant amount of domain knowledge.

2.3.2 Knowledge-based systems

Other systems have considered how motion categories could be used for event recognition in specific domains. Examples include the classification of motion in medical (heart) imagery (Tsotsos, Mylopoulos, Covvey, & Zucker, 1980), the recognition of events in traffic scenes (Neumann & Novak, 1983; Nagel, 1988), natural language description of events in soccer games (André, Herzog, & Rist, 1988), and the recognition of actions in robotics domains (Kuniyoshi & Inoue, 1993).

We refer to these approaches as *knowledge-based* since they rely on domain-specific knowledge to describe motion events. Because of the specialized knowledge, these systems are generally limited to very narrow domains, containing a fixed repertoire of events. In contrast to these approaches, our work attempts to form descriptions based on a more general physical model of the dynamics of the scene.

2.3.3 Qualitative physical models

Both AI and psychology researchers have proposed representations for the underlying causal structure of motion, usually in the form of linguistic or conceptual descriptions. Qualitative physical models can be viewed as an attempt to provide a computational theory for reasoning with such descriptions.

The analysis of static scenes presented by (Fahlman, 1974; Brand, Birnbaum, & Cooper, 1993) is based on evaluating a set of rules to determine whether objects in a scene have stable support. The output of the system can be provided either as a natural language description of the support relations in the scene, or as the input to another system such as a planner.

Other approaches to static scene description are based on *qualitative simulation*, which predicts how a scene will behave under the influence of gravity. (See (Funt, 1980).) Siskind (1992) used a qualitative simulation system along with *counterfactual* or *hypothetical* reasoning to determine the support, contact, and attachment relations among objects. In this system, object A is said to be *supporting* object B if the removal of object A means that B will fall. Once the support relations are determined, they are used for the recognition of simple motion verbs. While only a synthetic domain was used (movies containing 2D polygonal objects), this system showed the utility of qualitative simulation in recognizing motion verbs.

Finally, physics-based descriptions are also common in the qualitative reasoning community. For example, Forbus (1987) considers how simple spatial relations along with qualitative simulation can be used to describe motion events such as a bouncing ball. While not concerned with recognition or perception, other systems such as that presented by Joskowicz and Sacks (1991), have been used to provide qualitative descriptions and/or simulations of mechanisms.

While qualitative physical approaches are appealing, for our purposes here we will pursue an approach based on quantitative (exact) physics. The reason for this choice is twofold. First, in order to make sound inferences about the world we seek a good approximation to the Newtonian mechanics of the scene. Second, since an analysis of Newtonian mechanics can be achieved with relatively little computational effort, we see no reason to perform any approximate or qualitative reasoning. We note that there may still be a role for more approximate reasoning (e.g., to reason at a higher level of abstraction or to reason when there is incomplete sensor information), but we leave this as an issue for future work.

2.3.4 Function-based recognition

Several researchers in computational vision have argued that we perceive objects based on their *function* rather than their *form*. For example, objects like chairs or tools can better be described by how people interact with them than by their specific shapes.

Typically, most work in function-based recognition has focussed purely on the shape of static objects (such as chairs (Stark & Bowyer, 1992)) rather than the dynamics of moving scenes.

Ikeuchi and Suehiro (1994) and Kang and Ikeuchi (1993) describe how kinematic relations of objects (in static and moving scenes) can be used to classify various behaviors in both robotic and human manipulation of objects. Bogoni and Bajcsy (1995) describe how force and motion sensors can be used in an active robotics system to detect the occurrence of various events such as *piercing*. These examples illustrate that in order to describe object function we need to know about both kinematics (i.e., the allowable motion of objects) and dynamics (i.e., the implied forces among objects).

2.3.5 Summary

Unlike previous approaches to motion understanding, our approach is based on an analysis of the Newtonian mechanics of a simplified scene model.

Our work is closely related to that presented in (Blum, Griffith, & Neumann, 1970). In this work, a statics analysis is performed to determine if a pile of blocks is stably supported. A similar approach is used for the analysis of static scenes in (Mulgaonkar, Cowan, & DeCurtins, 1992). Another approach to determining the stability of static scenes is described in (Siskind, 1994). In this system, instead of analyzing the dynamics, an analysis of the kinematics is performed. In particular, a scene is said to be *unstable* if there exists some admissible motion of the objects that decreases the potential energy of the system. Our system extends these approaches to consider both kinematic and dynamic relations in time-varying scenes containing rigid objects.

A different approach to physics-based motion understanding is presented in (Shavit & Jepsen, 1993). Here, the dynamics of non rigid motions are analyzed based on phase portraits.

We conclude this chapter by answering some commonly posed questions about our approach.

Do humans perceive real physics?

It is worth noting that while there is significant evidence that humans generate qualitative physical descriptions of scenes, it also appears that humans have a limited understanding of some dynamic events.

Todd and Warren (1982) have shown that humans have incorrect notions of conservation of momentum when interpreting collisions of objects. In particular, if a light object hits a heavier object, our judgments of the relative mass of objects tends to be incorrect. (See also (Gilden & Proffitt, 1989).) McCloskey (1983) has shown that humans have common misconceptions concerning the path of moving objects. For example, if subjects are asked to describe (or draw) the path an object takes when exiting from a curved pipe, subjects will draw a curved rather than a straight path. Finally, Proffitt, Kaiser, and Whelan (1990) have shown that humans also lack consistent theories for angular momentum. In the experiments, subjects were shown animations of an object (a rotating satellite) that expands or contracts (by opening or closing its solar panels) as it rotates. Subjects did not have any preference for interpretations in which the rotation slows, speeds up, or stays the same as the object changes shape. This contradicts Newtonian mechanics (conservation of angular momentum) which says that rotating objects should slow down as they expand, and speed up as they contract.

While one might argue that we should use some form of *qualitative* or *approximate* reasoning to account for human perception, the approach taken here is to solve the dynamics inference problem exactly. We leave the issue of cognitive modeling for future work.

Physics-based simulation or scene perception?

While the problem is quite different, our representation of geometric and dynamic models borrows heavily from the physical simulation and graphics communities (Featherstone, 1987; Cremer, 1989; Baraff, 1995). The difference between our work and the physics-based simulation work is that rather than predicting the behavior of the scene objects, we are trying to infer the properties of objects given observations of their behavior. We refer to the first problem as the *scene simulation* problem, and the second problem as the *scene perception* problem. Note that, as mentioned above in the discussion of qualitative physics, a scene simulation component may be part of the scene perception component. Nonetheless, the objectives of the two systems are quite different.

How does this compare with other “physics-based” approaches?

A number of other approaches to computational vision are based on dynamic models. Most notably, so-called *deformable models* have been used to fit, segment, and track objects. Deformable models may be non-parametric (such as splines or snakes (Kass, Witkin, & Terzopoulos, 1987)), or may combine some parametric form (such as superquadrics) with a local

deformations (Metaxas & Terzopoulos, 1993). These models are dynamic in the sense that they are subject to “forces” from the image (and possibly internal forces) that are used to control their motion.

While these approaches may appear similar to our work, they are quite different. In particular, we are attempting to infer the dynamics in the scene rather than using forces as some method for shape fitting or object tracking.

How does this relate to work in knowledge representation?

As described in the Introduction, our approach is based on the *computational perceiver* framework, first proposed in (Jepson & Richards, 1993).

Our representation of an interpretation as a set of logical propositions about scene properties is similar to the approach presented by Reiter and Mackworth (1989). Unlike our system, however, that system considered a simple domain (maps) where feasibility could be expressed as a set of logical constraints. In addition, while that system allowed the representation of uncertainty via multiple interpretations, preferences were not used to choose among the feasible interpretations.

Finally, we note that the perception problem (finding plausible interpretations consistent with a domain theory and a set of observations) is similar to so called *diagnosis* problems considered in the AI literature (Reiter, 1987).

Chapter 3

Overview of the System

In this chapter we present a brief overview of our system. Here we emphasize the basic intuition behind each of the components and the prerequisites for their use. In the remaining chapters, we will describe each of the components in detail, and present experimental results.

We provide this overview in terms of a single illustrative example, the coke sequence shown in Figure 3.1. (This is one of several sequences used for the computational examples in Chapter 8.) In this sequence a hand is reaching for, grasping, and then lifting a coke can from a desk top. As mentioned in the Introduction, our eventual goal is to have a machine vision system that, when given image sequences such as this one, can understand the basic force generation and force transfer relationships of the various objects in the scene. For this example sequence, the system should understand that the can is initially supported by the table and that the hand (and arm) is possibly an ‘active object.’ Roughly speaking, an active object is something that can generate forces other than those due to gravity, friction, and acceleration. Objects that are not active, such as the table and the coke can, are said to be ‘passive objects.’ During the lifting phase of this sequence, if the system remembers that it has determined that the hand is an active object, it should conclude that the hand is attached to the coke can (i.e., grasping it) and applying an upward force on it.

Figure 3.1: An illustrative example: the coke image sequence. In this sequence a hand is reaching for, grasping, and lifting a coke can off a table top. Our subsequent analysis is based on the motion in the highlighted frame, marked “lifting”.

This goal of understanding forces and dynamics given image sequences is an ambitious one. Obviously we do not achieve it completely here. However, we do present an implementation that takes us much of the way to our goal, albeit in a simple domain. The critical simplification is that the objects and their motions can be approximated by a layered 2D representation. Such a representation provides a crude but useful approximation for a variety of situations, as is substantiated by the computational examples presented in this thesis. Moreover, the simplicity of the 2D domain serves to highlight the basic principles behind our approach.

3.1 An Outline of the Approach

In order to reason about dynamic properties of the objects in the scene we use the equations of Newtonian mechanics. Given a concrete hypothesis about the scene content, we simply express the corresponding equations of Newtonian mechanics in a suitable form and check to see if they have a feasible solution.

This physics-based approach places stringent requirements on the richness and the level of detail of the underlying scene model. Indeed, we have the basic requirement that the geometry of the scene must be completely specified in order to express these equations. That is, we need to know object shape and placement, along with the surfaces of contact between objects. In addition, we need estimates for the velocities and accelerations of the objects. We also need to assume something about the distribution of mass within each object. For our purposes here, we simply take an object's center of mass to be its centroid, as if it had a uniform mass distribution. The total mass and the inertial matrix are then treated as constrained unknowns when the system checks for a feasible solution. In summary, a strong prerequisite on the hypothesis-generation mechanism is that a detailed description of the scene is needed before the physics-based model can be applied.

For a concrete example, consider the lifting phase in the coke sequence (the rightmost frame in Figure 3.1). Through the use of a tracking algorithm, we obtain the 2D positions, velocities, and accelerations of polygons that roughly describe the shape of the scene objects. This information, along with a line denoting the table surface, constitutes the “configuration” of the scene. (See the rightmost image in Figure 3.2.) Here we define the *configuration* to be the set of scene properties that are necessarily present, given the image data and any restrictions inherent in the ontology. For example, in the current system, the positions, velocities, and accelerations of the objects are provided by the image observations, and the positions of the centers of mass are fixed, by our ontology, to be at the object centroids.

In order to supply the information missing from a configuration, we consider *assertions* taken from a limited set of possibilities. These assertions correspond to our hypotheses about the basic force generation and force transfer relationships between objects. For brevity, we constrain ourselves here to only those assertions that play a significant role in the coke sequence.

Figure 3.2: Tracked polygons for the coke image sequence. The polygons outline the objects while the vectors show the estimates for polygon accelerations. (The horizontal line denotes the table surface.)

The full set is provided in the next chapter. In particular, here we need the following three types of assertions:

- (i) $\text{CONTACT}(o_1, o_2, R_c)$ — objects o_1 and o_2 contact in the scene with the region of contact R_c ;
- (ii) $\text{ATTACH}(o_1, o_2, R_a)$ — objects o_1 and o_2 are attached at some set of points, $p \in R_a$, in the contact region;
- (iii) $\text{BODYMOTOR}(o)$ — object o has a ‘body motor.’

The intuitive meanings of these assertions are: (i) objects o_1 and o_2 are contacting, either in depth or abutting; (ii) objects are attached on some set of points, $p \in R_a$, in the contact region; these attachment points are functionally equivalent to rivets, fastening the objects together; (iii) the object can generate an arbitrary force and torque on itself, as if it had several thrusters. Note that the attachment assertion is properly understood to be a characterization of what sorts of forces are supported at the attachment points. Attached objects can be pulled, pushed, and sheared without coming apart while, without the attachment, the contacting objects may separate or slide on each other depending on the applied forces and the coefficient of friction. The precise choice of these kinematic and dynamic assertions is not critical for the purpose of this thesis. What is important is that they are sufficiently rich to describe a wide variety of phenomena.

Note that, given the hand and can polygons in Figure 3.2, there is no evidence that any of these assertions are individually true or false. It is for this reason that neither they, nor their negation, belong to the configuration. Furthermore, notice that some sets of assertions are not *admissible*, in that they violate basic constraints for their application. For example, two objects cannot be both attached and not contacting. Moreover, some sets of assertions are not *complete* in that they leave some properties unspecified.

In order to check for a feasible force balance, however, our physical model requires the set of assertions to be complete (i.e., it must fully specify each admissible assertion as either true or false). For any complete and admissible set of assertions, we can check the feasibility of those

assertions according to whether or not a force balance exists in our physics-based model. This feasibility test can, to a good approximation, be reduced to a linear programming problem (see Chapter 5). By definition, inadmissible assertion sets are also infeasible. Finally, we define a *feasible interpretation* to be a combination of a configuration and a feasible set of assertions which completely specify the scene.

For the lifting phase of the coke sequence, there turn out to be five feasible interpretations within our ontology, as displayed in Figure 3.3. Note that each of these interpretations can be seen to have a feasible force balance, given that we can adjust both the object masses and the forces generated by the body motors. In contrast, any interpretation without a body motor is infeasible since the vertical acceleration cannot be ‘explained’ (i.e., balanced with a corresponding force). Similarly, the active hand in the top left interpretation in Figure 3.3 must be attached to the can since, in our ontology, we do not model forces in depth. This means, in turn, that the physics-based model cannot generate a frictional force between the hand and the can. So, without attachment, the observed vertical acceleration of the can cannot be balanced with a vertical force. While this sort of ‘reasoning’ seems sophisticated, it is simply relying on the feasibility of linear programming problems that the system can pose for itself.

A naive system would generate all feasible interpretations thereby producing all (and just) the interpretations depicted in Figure 3.3. Our algorithm, however, does not generate all such feasible interpretations since, as we discuss next, many of these interpretations are not of interest. It turns out that the algorithm need not evaluate all interpretations in order to find the interesting ones.

3.2 Preferred Explanations

Thus far we have discussed the nature of an interpretation in terms of a scene configuration along with assertions that specify further dynamic and kinematic properties. We pointed out that the feasibility of such an interpretation can be checked using a physics-based model for the forces involved along with the admissibility of the assertions. However, given a fairly rich ontology, it is common for there to be multiple feasible interpretations for a given scene configuration (see Figure 3.3).

Indeed, given that a body motor is capable of generating an arbitrary force and torque on the object possessing it,¹ it follows that for any scene configuration there is always at least one trivial interpretation in which every object has a body motor. The bottom three interpretations in Figure 3.3 are examples of such trivial solutions. They are guaranteed to pass the force-balance feasibility criterion and, as such, are not very informative interpretations on their own. Rather, we seek interpretations that require, in some specified sense, the *weakest* properties of the various objects.

¹Note that this is an idealization, since real objects can only generate limited forces and torques.

Figure 3.3: Preference ordering for the interpretations of the rightmost frame of the image sequence shown in Figure 3.1. A large open circle at object center denotes a BODYMOTOR. The small black dots denote contact points between CONTACTING objects, while the larger dots denote attachment points between ATTACHED objects. A textual form of the assertions appears adjacent to each interpretation. The three levels of priority are represented by each line of text. Note that the absence of an assertion denotes its negation. See text for details.

Model preference relations, as discussed by Richards, Jepson, and Feldman (1996), can be used to express suitable preference orderings. The basic idea is simple, namely to compare two different interpretations in terms of a prioritized set of elementary preference relations. Our current ontology includes the elementary preference for the absence of the body motor assertion for each object. In other words, we prefer to see a particular object as a passive object, if feasible, given that other elementary preference relations do not contradict this preference. In addition, we prefer the absence of a kinematic constraint of attachment between two objects over the presence of such a constraint. Finally, we are indifferent as to whether we assert that two objects are contacting or not (always assuming admissibility and feasibility).

These three elementary preference relations are taken to be stratified in terms of priority. In particular, the preference for the absence of a body motor is taken to have a higher priority than the preference for no attachment. Finally, we have no elementary preference relation for contact over no contact—these two situations are considered to be equally preferable. Note that this priority ordering is indicated in Figure 3.3 by listing the assertions adjacent to each interpretation. Assertions are placed on different lines to indicate their priority. It is convenient to place contact assertions on the bottom row even though there is no preference between different contact assertions.

Given these three elementary preference relations along with the priority ordering, the induced ordering on our five feasible interpretations for the coke example is provided simply by a prioritized subset ordering. In particular, suppose one is given two feasible interpretations along with their corresponding assertion sets. If, at the highest level of priority, one interpretation's assertion set is a strict subset of the other's, then the first interpretation is preferred. For example, the top left interpretation in Figure 3.3 is preferred over the three lower interpretations because it does not assert a body motor on the can. In a sense, not needing a body motor is considered to be major simplification in what the various objects are asserted to be capable of doing, and thus the preference against body motors is placed at a high priority.

Alternatively, if the assertion sets at the highest priority do not satisfy a subset relation then the interpretations are considered to be unordered. For example, the top two interpretations in Figure 3.3 are considered unordered since the body motors are on different objects. If the sets at the highest priority are the same, then we check the assertions at the next lower priority. For example, the three lower interpretations in Figure 3.3 all contain the same motor assertions, but can be further ordered based on minimizing attachment. Finally, if interpretations agree on both motor and attachment assertions, we are indifferent as to the preference relation. This occurs in Figure 3.3 for the middle two interpretations, since these differ only in terms of their contact relations.

This approach to preference ordering will not always yield a unique preferred interpretation. Therefore, multiple percepts are possible. For example, in the coke sequence there are two maximally-preferred interpretations, namely either the hand is lifting the can, or vice versa.

Given that our ontology includes nothing about hands or cans, that is, we just have moving polygons and notions of mass and force, arriving at these two interpretations for this single frame is intuitively the right thing to do. Indeed, Jepson and Richards (1993) and Richards, Jepson, and Feldman (1996) propose that such maximal interpretations provide a computational model for a ‘percept.’ Moreover, Richards, Jepson, and Feldman (1996) explore the relationship between such preference orderings and qualitative probabilistic models (see also (Jepson, Richards, & Knill, 1996)).

3.3 Implementation and Limitations

Our current implementation has a number of limitations.

One limitation is that our system uses a 2D layered representation. Furthermore, our system does not represent the relative depth among objects. Given this representation we can process only fronto-parallel scenes and cannot reason about occlusion or motion in depth. This is a reasonable approximation if we assume that objects move in planes roughly fronto-parallel to the camera. Moreover, we believe that this limitation is not fundamental to our general approach. In particular, although it is not necessary for the domain considered in this thesis, we believe that our system could be extended to reason about 3D scenes if given suitable 3D input.

Another limitation is that our system does not integrate information over time or reason about object properties. For example, consider the coke sequence once again. During the reaching phase of this sequence, the hand is seen to be above the table and accelerating upwards. Our system concludes that the hand must be an active object while the coke can could be passive and at rest on the table. This is the unique maximally-preferred solution for this frame. As we have just shown, however, there are multiple interpretations of the scene during the subsequent lifting phase; either the hand or the coke can could be active, but not both. Since our system lacks prior knowledge of object capabilities (such as the fact that hands are typically active, and coke cans are typically passive), and does not integrate conclusions over time (such as noticing that it previously concluded the hand must be active), the system is left with multiple interpretations for the lifting phase.

Finally, our system is limited in that object velocities and accelerations are taken to be continuous functions of time. In particular, we do not consider force impulses that give rise to step discontinuities in the velocities.

Despite these limitations, we are left with a surprisingly rich domain, as will be demonstrated by the computational examples in Chapter 8. In addition, in Chapter 9 we will show some preliminary efforts to extend our system to handle continuous motion sequences, thereby addressing the issues of integrating information over time and the detection of motion discontinuities.

We return to these limitations, and ways to overcome them, in Chapter 10.

Chapter 4

Representation

In this chapter, we describe the form of the system's representation for the motion domain.

As described in the Introduction, our approach to scene dynamics is based on the analysis of the Newtonian mechanics of a simplified scene model. In order to perform such reasoning we require an *ontology* that describes the objects, their properties, and the constraints of the domain. In particular, this representation must be suitable for specifying the geometry of the scene and the types of forces that can be generated on the various objects. In addition, the representation must allow us to specify our hypotheses about the properties of objects and the relations among objects.

4.1 Knowledge Representation Framework

To describe our system precisely, we need to specify some representation for the domain. As described in (Genesereth & Nilsson, 1987), we provide a *conceptualization* of a domain by specifying:

- a set of *entities* that represent the objects and concepts that are presumed to exist in our domain,
- a *language* with which to express statements about the domain,
- and a set of statements or *axioms* that describe the constraints in the domain.

In other words, a conceptualization for a domain is an abstract, simplified view of the world that we wish to represent for some purpose (Davis, Shrobe, & Szlovits, 1993). It should be obvious that all systems are committed to some conceptualization, either implicitly or explicitly. The approach taken in this thesis is to provide an explicit representation that isolates the different components of the computational framework.

To provide a semantics, we need to specify a language in which to write our axioms. In the knowledge representation community, it is common to specify the axioms in some logic,

such as propositional or first-order logic. In this work, we find it convenient to adopt a more liberal approach: in addition to specifying logical axioms, we will also allow some axioms to be written in terms of algebraic constraints in some *state-space* used to describe the domain. For example, in order to specify that objects don't interpenetrate, we will specify constraints on the relative position and motion of objects. Similarly, in order to specify that objects obey the laws of Newtonian mechanics, we will write constraints on the object masses, accelerations, and the forces acting on the objects. Intuitively, our state space will provide a representation of all of the kinematic and dynamic entities of interest (e.g., object shapes and motions, object masses, the forces and torques acting on objects, etc.) in our domain. As we will show later, by reducing logical statements to state-space constraints, we can give a precise definition of an interpretation. In addition, in later chapters we will make use of particular properties of the constraints in order to argue for the correctness of our inference process.

We now describe the representation for the motion domain and motivate some of the choices made. We will then provide a precise specification for an *interpretation*.

4.2 Representation for the Motion Domain

As described in the Introduction, in this work we present a basic system that makes a number of simplifications:

- We assume a layered 2D scene model consisting of polygonal objects.
- We analyze the motion of objects at a single point in time (i.e., we analyze the *instantaneous* motion of objects).
- We assume the motion is *continuous* (i.e., the contact geometry is not changing and there are no discontinuous changes in velocity or acceleration due to collisions, force impulses, or the application or removal of forces).

Despite these simplifications, this representation still contains significant structure as will be illustrated by the computational examples in Chapter 8. Furthermore, we will show in Chapter 9 how a representation of the instantaneous mechanics can provide a basis for an enhanced system that considers continuous motion sequences.

The following three subsections describe the particular representation used. First we describe the representation for individual objects. In the second and third subsections, we describe the representation of the kinematic and dynamic properties of the scene.

4.2.1 Objects

Objects are represented by rigid two-dimensional convex polygons. Objects have positions and orientations in 2D. For simplicity, we assume that the scene is described using some global reference frame. For each object we represent the following information:

- The object *shape*. The shape is represented by the vertices of the two-dimensional bounding polygon.
- The *centroid* of the object, which is computed from the polygonal shape.
- The object *motion*. We represent the motion of each object by specifying the velocity $\mathbf{v}$ and acceleration $\dot{\mathbf{v}}$ of the centroid of the object. The velocity and acceleration are two-dimensional vectors. Moreover, the angular motion is specified by the angular velocity ω and the angular acceleration $\dot{\omega}$. For the two-dimensional case considered in this thesis, the angular velocity and angular acceleration are scalars.
- Each object's *mass*, m , and *moment of inertia*, I . For the 2D case considered here, I is a scalar.

The objects are denoted $o_1, o_2, \dots, o_n \in O$, where O is a finite set of unique objects. We assume the objects have unique names and that all scene objects are given to the system.¹ In addition, we assume that the object shapes and motions are determined from image measurements by an image preprocessing stage. (See Chapter 8 for details of the processing along with computational examples.)

4.2.2 Kinematic Model

In physics it is customary to divide kinetics, the study of motion, into two components: *kinematics*, which is concerned with object motions without regard for the influence of mass and force, and *dynamics* which concerns the relation between motion and the forces that produce it. In our system we also find it convenient to make this distinction. The reason is that, while the analysis of kinematics can be performed directly from the observation of the objects' shapes and motions, the analysis of dynamics requires additional knowledge about the object masses and the forces acting on objects. In this subsection we describe the kinematic model. In the next subsection we describe the corresponding dynamic model.

Contact

Layered Depth Model. We use a *layered depth model* to represent the spatial relationships among objects in the scene. There is no depth ordering in our layered depth model. Instead, we represent only the qualitative depth relations between pairs of objects. Specifically, pairs of objects can be in the same layer, in adjacent layers, or in layers separated in depth. We denote the qualitative depth relations between objects by the following mutually exclusive predicates defined on pairs of objects:

¹That is, we assume that all the scene objects are *unique* and *visible*. These correspond to the “unique names assumption” (UNA) and the “closed-world assumption” (CWA) in the knowledge representation literature (Reiter, 1978).

- $\text{SAMEDEPTH}(o_1, o_2)$ —Objects o_1 and o_2 are in the same depth layer.
- $\text{ADJACENTINDEPTH}(o_1, o_2)$ —Objects o_1 and o_2 are in adjacent depth layers.
- $\text{SEPARATEDINDEPTH}(o_1, o_2)$ —Objects o_1 and o_2 are in layers that are separated in depth.

The only constraint in the layered model is that pairs of objects in the same depth layer cannot overlap (i.e., objects cannot interpenetrate). Aside from this condition, pairs of objects can have any depth relationships. For simplicity, however, we adopt the convention that for non-overlapping pairs of objects, objects are taken to be separated in depth.

Note that because there is no ordering between the depth layers, we do not have a representation of the relative depth of objects. This means, for example, that we cannot model the relative depth constraints between objects indicated by the occlusion relationships observed in the image data. In addition, because the model considers only pairwise *non-transitive* depth relations between objects, we do not propagate depth constraints between objects. For example, if object o_1 and o_2 are in the same layer and object o_2 and o_3 are in the same layer, it is not necessary for objects o_1 and o_3 to be in the same layer. Finally, because we do not represent the depth relations between non-overlapping objects, we cannot reason about depth constraints between non-contacting objects.²

Intersection Conditions. Objects can contact either within the same layer or between adjacent layers. The first type of contact, called *abutment*, occurs when two objects in the same depth layer contact at a point or at an edge along their boundary. The second type of contact, called *overlap*, occurs when two objects in adjacent depth layers contact over part of their surfaces and the region of overlap has non-zero area. (See Figure 4.1.) The degenerate case where objects overlap at a boundary but objects are in adjacent depth layers is not considered to be a contact. As a consequence, when objects overlap in the image, contact in the scene implies overlapping contact, while when objects abut in the image, contact in the scene implies abutment. Finally, objects that don't intersect in the image cannot be contacting in the scene. We denote both types of contact by the predicate

- $\text{CONTACT}(o_1, o_2, R_c)$

where o_1 and o_2 are the objects, c denotes the contact, and R_c denotes the region of contact between the two objects. Since the intersection between convex polygonal objects is also a convex polygon, we represent the contact region R_c by a set of points at the vertices of the contact region.

Motion Conditions. In addition to position information, the relative motion of objects constrains their allowable contact and layer relations. In particular, in the case of abutting

²We could strengthen the layered model to enforce a total ordering on the depth relations between objects, but this would probably be too restrictive for general views of three-dimensional scenes. Instead, we probably require a depth model that enforces only *local* ordering constraints on the surfaces denoting objects. Such a model is proposed by Williams and Hanson (1996).

contacts, contact is admissible only when the relative velocity of the contacting objects is *tangential* to the contact region (i.e., objects can slide along their contact region, but cannot penetrate or separate).

To ensure that contacting objects do not separate or penetrate, we write constraints on their relative motions. Without loss of generality, we represent the contact between convex objects by the set of vertices of the convex hull of the contact region. Specifically, let $\mathbf{p}(t_0)$ denote a contact point (in the global reference frame) between objects a and b at time t_0 . Let $\mathbf{p}_o(t; \mathbf{p}, t_0)$ describe the position of the point $\mathbf{p}(t_0)$ on object $o \in (a, b)$ as a function of time t . For each contact point $\mathbf{p}$ we can write a *characteristic function* (Featherstone, 1987; Baraff, 1995) that gives the relative normal displacement of the contact point between the objects

$$\chi_{ba}(t; \mathbf{p}, t_0) = \mathbf{n}_a(t; \mathbf{p}, t_0) \cdot (\mathbf{p}_b(t; \mathbf{p}, t_0) - \mathbf{p}_a(t; \mathbf{p}, t_0)) \quad (4.1)$$

where $\mathbf{n}_a(t; \mathbf{p}, t_0)$ is the surface normal at point $\mathbf{p}$ on object a at time t_0 as a function of time t . (See Figure 4.2). By convention $\mathbf{n}_a(t; \mathbf{p}, t_0)$ points outward from object a . Note that $\mathbf{n}_a(t; \mathbf{p}, t_0)$ is defined only when: 1) $\mathbf{p}$ is on the boundary of object a at time t_0 , and 2) $\mathbf{p}$ is not on a vertex of object a at time t_0 . According to Equation 4.1 positive displacement corresponds to *separation* of the two objects at point $\mathbf{p}$, while negative displacement corresponds to *penetration*.

We compute the relative motion at the contact points by taking derivatives of the normal displacement. This gives the *relative normal velocity*

$$\begin{aligned} \dot{\chi}_{ba}(t; \mathbf{p}, t_0) &= \dot{\mathbf{n}}_a(t; \mathbf{p}, t_0) \cdot (\mathbf{p}_b(t; \mathbf{p}, t_0) - \mathbf{p}_a(t; \mathbf{p}, t_0)) \\ &\quad + \mathbf{n}_a(t; \mathbf{p}, t_0) \cdot (\dot{\mathbf{p}}_b(t; \mathbf{p}, t_0) - \dot{\mathbf{p}}_a(t; \mathbf{p}, t_0)) \end{aligned} \quad (4.2)$$

and the *relative normal acceleration*

$$\begin{aligned} \ddot{\chi}_{ba}(t; \mathbf{p}, t_0) &= \ddot{\mathbf{n}}_a(t; \mathbf{p}, t_0) \cdot (\mathbf{p}_b(t; \mathbf{p}, t_0) - \mathbf{p}_a(t; \mathbf{p}, t_0)) \\ &\quad + 2\dot{\mathbf{n}}_a(t; \mathbf{p}, t_0) \cdot (\dot{\mathbf{p}}_b(t; \mathbf{p}, t_0) - \dot{\mathbf{p}}_a(t; \mathbf{p}, t_0)) \\ &\quad + \mathbf{n}_a(t; \mathbf{p}, t_0) \cdot (\ddot{\mathbf{p}}_b(t; \mathbf{p}, t_0) - \ddot{\mathbf{p}}_a(t; \mathbf{p}, t_0)) \end{aligned} \quad (4.3)$$

Since our analysis is only concerned with the motion at time $t = t_0$ (i.e., the instantaneous motion), we have $\mathbf{p}_a(t) = \mathbf{p}_b(t)$ and the normal velocity simplifies to

$$\begin{aligned} v_{ba}^n(\mathbf{p}) &= \dot{\chi}_{ba}(t; \mathbf{p}, t_0)|_{t=t_0} \\ &= \mathbf{n}_a(\mathbf{p}) \cdot (\mathbf{v}_b(\mathbf{p}) - \mathbf{v}_a(\mathbf{p})) \end{aligned} \quad (4.4)$$

where we use $v_{ba}^n(\mathbf{p})$ to denote the relative normal velocity of the contact point $\mathbf{p}$ (at $t = t_0$), and $\mathbf{v}_a(\mathbf{p})$ and $\mathbf{v}_b(\mathbf{p})$ to denote the velocity (in the global reference frame) of point $\mathbf{p}$ on objects

Figure 4.1: Contact types in layered scene model consisting of 2D polygonal objects. (a) *Overlapping contact*: objects in adjacent depth layers overlap. (b) *Abutting contact*: objects in the same layer touch at their boundaries (contact can be either *edge-to-edge*, *point-to-edge*, or *point-to-point*). The dots depict the vertices of the contact region, R_c .

Figure 4.2: Contact conditions for 2D polygonal objects. The relative normal velocity (acceleration) at the contact point, $\dot{\chi}(t)$ ($\ddot{\chi}(t)$), is positive if objects are separating, negative if objects are penetrating, and zero if objects are moving tangentially relative to one another.

a and b , respectively. Similarly, the normal acceleration simplifies to

$$\begin{aligned} \dot{v}_{ba}^n(\mathbf{p}) &= \ddot{\chi}_{ba}(t; \mathbf{p}, t_0)|_{t=t_0} \\ &= \mathbf{n}_a(\mathbf{p}) \cdot (\dot{\mathbf{v}}_b(\mathbf{p}) - \dot{\mathbf{v}}_a(\mathbf{p})) + 2\dot{\mathbf{n}}_a(\mathbf{p}) \cdot (\mathbf{v}_b(\mathbf{p}) - \mathbf{v}_a(\mathbf{p})) \end{aligned} \quad (4.5)$$

where $\dot{\mathbf{v}}_a(\mathbf{p})$ and $\dot{\mathbf{v}}_b(\mathbf{p})$ denote the acceleration (in the global reference frame) of point $\mathbf{p}$ on objects a and b , respectively and $\dot{\mathbf{n}}_a(\mathbf{p})$ is the derivative of the normal vector. Given the object representation in §4.2.1 above, we compute the velocity and acceleration of a point $\mathbf{p}$ on object $o \in \{a, b\}$ by

$$\mathbf{v}_o(\mathbf{p}) = \mathbf{v}_o + \omega_o \times (\mathbf{p} - \mathbf{r}_o) \quad (4.6)$$

$$\dot{\mathbf{v}}_o(\mathbf{p}) = \dot{\mathbf{v}}_o + \dot{\omega}_o \times (\mathbf{p} - \mathbf{r}_o) + \omega_o \times \omega_o \times (\mathbf{p} - \mathbf{r}_o) \quad (4.7)$$

where $\mathbf{r}_o$ denotes the center of gravity of object o , while $\mathbf{v}_o$, ω_o , $\dot{\mathbf{v}}_o$, $\dot{\omega}_o$ denote the linear velocity, angular velocity, linear acceleration and angular acceleration, respectively.³ Finally, the derivative of the normal vector $\mathbf{n}_a(\mathbf{p})$ is given by $\dot{\mathbf{n}}_a(\mathbf{p}) = \omega_a \times \mathbf{n}_a(\mathbf{p})$.

Note that the condition on tangential motion can only be defined when objects contact along an edge (i.e., objects are in *edge-to-edge* or *point-to-edge* contact). Currently, we do not allow contact relations between objects in (degenerate) *point-to-point* contact.⁴

³Note that for the two-dimensional model considered here, ω_o and $\dot{\omega}_o$ are both scalars.

⁴We could extend our system to consider point contact by enforcing a *disjunction* of two tangential motion constraints of the form given in Equation 4.1, however, we do not deal with this case in this thesis.

In summary, we have the following constraints on contacts in the layered scene model:

- C1. (Object substantiality)** Objects in the same depth layer cannot interpenetrate.
- C2. (Depth constraints on Contacts)** Objects in *abutting* contact must be in the same depth layer, while objects in *overlapping* contact must be in adjacent depth layers. (Objects that don't contact are taken to be in separate depth layers.)
- C3. (Intersection)** Contacting objects must intersect in the image. In addition, objects in *overlapping* contact must intersect over a region on non-zero area.

The first two conditions enforce the constraints on contact relations described above. The third condition says that there can be “no action at a distance”—that is, contacting objects must actually intersect in the scene.

Attachment

Attachment provides additional kinematic constraints among objects. In general there can be numerous constraints representing various types of kinematic mechanisms (Kramer, 1990). In this work, we consider a subset of mechanisms generated by rigid attachment at a set of one or more points within the contact region. The intuitive meaning is that attachment points are functionally equivalent to ‘rivets’, fastening the objects together. Attached objects can be pulled, pushed, and sheared without coming apart while, without the attachment, the contacting objects may separate or slide on each other depending on the applied forces and the coefficient of friction. A single attachment corresponds to a *revolute joint*, while two or more attachments result in a rigid attachment between the objects.⁵ We denote attachment between objects by the following predicate

- $\text{ATTACH}(o_1, o_2, R_a)$

where o_1 and o_2 are the objects and R_a is set of attachment points.

We apply the following constraints to attachments:

- A1. (Attachment requires contact)** Objects o_1 and o_2 must contact and the attachment region R_a must be wholly contained within the contact region, R_c .
- A2. (Rigid attachment)** The relative velocity and relative acceleration of objects at all of the attachment points must be zero.

Condition A1 says that if we assert objects are attached, we must also assert that they contact. In addition, the region of attachment, R_a , must be wholly contained in the contact region R_c .

⁵Note that in the three-dimensional case, one attachment would correspond to a ‘ball-in-socket’ joint, while two attachments correspond to a revolute or ‘hinge’ joint. Three or more attachments produce rigid attachment. However, in this thesis we will consider only two-dimensional scenes.

Condition **A2** enforces rigidity at the attachment points. Note that for a single attachment point, **A2** allows rotation about the point, while for two or more attachment points, **A2** allows no relative motion between the objects. We enforce the rigidity constraint by extending Equations (4.1–4.3) above to enforce zero motion for the *tangential* as well as the normal component of the motion at the contact points.

4.2.3 Dynamic Model

To check the feasibility of an interpretation, we need to represent dynamic information about the scene. This involves specifying the acceleration of each object along with its mass, the distribution of mass (inertia), and the various forces acting on it. The complete dynamic model will be described in Chapter 5. In this section, we will just provide an overview of the components.

Mass and Mass Distribution

To determine mass and mass distribution from image data, we assume that the center of mass of each object is at that object's geometric center. Object masses and moments of inertia, however, are left as free parameters that can be adjusted by the physics-based model in order to find a feasible solution.

Forces

An object is subject to gravitational and inertial forces. In addition, we model forces and torques resulting from contact with other objects. This includes frictional as well as normal forces. Forces are represented quantitatively as vectors acting at the objects' centers and at the regions of contact. The dynamics of the object under these forces is obtained from the physics-based model described in Chapter 5.

Ground Objects

Particular objects are designated as *ground*. We typically use this to represent the table top. The motion (or absence of motion) for ground objects need not be explained by the dynamic model. We use the predicate

- $\text{GROUND}(o)$

to denote such objects.

Force Generators (Motors)

In addition to the forces listed above, we allow a number of force-generators in the scene. In the current system we consider the following types of force generators:

- **BODYMOTOR**(o)
- **LINEARMOTOR**(o_1, o_2, R_m)
- **ANGULARMOTOR**(o_1, o_2, p)

The intuitive meaning of **BODYMOTOR**(o) is that object o can generate an arbitrary force and torque on itself, as if it had several thrusters. **LINEARMOTOR**(o_1, o_2, R_m) means that a linear motor exists between the abutting objects o_1 and o_2 . This motor can generate an arbitrary tangential shear force across the motor region R_m . Linear motors are used to generate a shear force across an abutment (providing an abstraction for the tread on a bulldozer). Finally, **ANGULARMOTOR**(o_1, o_2, p) means that an angular motor exists at a single point p between objects o_1 and o_2 , and the motor can generate an arbitrary torque about that point. Angular motors are used to generate torques at joints.

We apply the following constraints to the motor assertions:

- M1. (Body motors)** Body motors are allowed on all objects.
- M2. (Linear motors)** Linear motors are admissible only at contacts where the direction for application of the tangential force is defined. Thus linear motors are only allowed when objects are in abutting contact, and the contact is either *edge-to-edge* or *point-to-edge*. Linear motors are not allowed at abutting contacts that are *point-to-point*, or at overlapping contacts.
- M3. (Angular motors)** Angular motors are admissible only at a single point within the contact region between two objects. The objects must be attached at this point.

4.3 Interpretations

Interpretations of the scene are constrained by the observed image data, our representation, and the Newtonian mechanics of the scene model. As described in Section 4.1, we take an interpretation to be the solution to a *constraint satisfaction problem* (CSP) — that is, the solution to a set of logical and algebraic constraints.

To describe these constraints, we find it convenient to decompose our representation into the following three components:

1. *Configuration.* The set of scene properties that are uniquely determined by the image observations;
2. *Assertions.* A set of atoms (positive literals) that specify all potential hypotheses about the scene content;
3. *Free Parameters.* Any remaining degrees of freedom in our representation.

We denote the configuration by the symbol C , the assertions by the symbol A , and the free parameters by the symbol Θ .

In the system considered here, the configuration, C , consists of the objects, their shapes, and their motions provided by the image observations. Since the positions of the centers of mass are fixed by our ontology to be at the object centroids, we take them to be part of the configuration as well. Finally, we assume that the system is provided with a set of **GROUND** objects, which we take to be part of the configuration as well.

The assertions, A , describe our potential hypotheses about the scene content. In the current system, the assertions correspond to assignments of the **CONTACT**, **ATTACH**, **BODYMOTOR**, **LINEARMOTOR**, and **ANGULARMOTOR** predicates that describe the kinematic and dynamic properties of the scene. We say that a set of assertions is *complete* iff every admissible assertion is specified as being either true or false.

Finally, the free parameters, Θ , describe parts of the representation that are not in the *configuration* or *assertions*. In the current system, the free parameters correspond to the masses and inertias of the objects and the forces acting on objects. These terms are varied by the physics-based model in order to find feasible interpretations.

An *interpretation* corresponds to a single hypothesis about the scene content.

Definition 4.1 (Interpretation) An *interpretation*, $I = (A, C)$, consists of a configuration C along with a complete assignment of assertions A in our representation.

In the current system, an interpretation includes the following items: a specification of the objects in the scene; the object positions, velocities and accelerations; the contact and attachment relations between all pairs of objects; all force-generators (motors) in the scene.

Only a subset of all possible interpretations are *admissible*.

Definition 4.2 (Admissible interpretation) An interpretation $I = (A, C)$ is *admissible* if the assertions A satisfy a set of admissibility constraints, Q .

We denote the admissibility constraints by the symbol Q . We write $Q(I)$ to mean that the interpretation I *satisfies* the constraints in Q . As mentioned in §4.1, these constraints may be written in terms of logical and/or algebraic constraints. Specifically, we say that I *satisfies* Q if there is some setting of the free parameters Θ that is consistent with the set of constraints specified in Q and the set of observations and assertions fixed by $I = (A, C)$. For the current domain, the admissibility constraints are specified by the conditions (C1–3), (A1–2), and (M1–3) in §4.2.2 and §4.2.3. Intuitively, we can view the admissibility constraints as specifying a set of “syntactic” or “coherence” constraints about the assertions in our domain.

Finally, given an admissible interpretation I , we must check whether it is consistent with our domain knowledge.

Definition 4.3 (Feasible interpretation) An interpretation I is *feasible* if it: (i) is admissible, (ii) satisfies a set of feasibility constraints, T .

We denote the feasibility constraints by the symbol T . We write $T(I)$ to mean that the interpretation I satisfies the constraints in T . Also, we adopt the convention that any feasible interpretation is also admissible. That is, $(\forall I).T(I) \subset Q(I)$.

For the current domain, the feasibility constraints are specified by the Newtonian mechanics of the simplified scene model. We present the theory of Newtonian mechanics and the feasibility test in the next chapter.

Chapter 5

Feasible Interpretations

5.1 Introduction

Given an interpretation, we use a theory of dynamics to determine if the interpretation is consistent with our domain knowledge. As described in the previous chapter, we say that an interpretation is *feasible*, if there exists a set of free parameters (forces, torques, masses, and inertias) that are consistent with the observations and with our dynamic model. In this chapter, we show that the test for consistency within the physical theory can be expressed as a set of algebraic constraints that, when provided with an admissible interpretation, can be tested with linear programming.

In this chapter, we present a theory for the general three-dimensional case. The experimental results we describe in Chapter 8 were produced using a two-dimensional variant of this theory.

5.2 Testing Feasibility by “Force balancing”

For rigid bodies under continuous motion, the dynamics are described by the Newton-Euler equations of motion (Goldstein, 1980). For rigid bodies of non-varying mass, the appropriate equations are:

$$\begin{aligned}\mathbf{F} &= \dot{\mathbf{p}} \\ &= M\dot{\mathbf{v}}\end{aligned}\tag{5.1}$$

$$\begin{aligned}\mathbf{N} &= \dot{\mathbf{L}} \\ &= \mathbf{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega}\end{aligned}\tag{5.2}$$

The first equation relates the total applied force $\mathbf{F}$ to the rate of change of linear momentum $\dot{\mathbf{p}}$. For bodies with non-varying mass, this reduces to $M\dot{\mathbf{v}}$, where M is the object mass and $\dot{\mathbf{v}}$ is the acceleration. The second equation relates the total applied torque $\mathbf{N}$ to the rate of change of angular momentum $\dot{\mathbf{L}}$. For rigid bodies, the rate of change of angular momentum is given

by $\mathbf{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega}$, where $\mathbf{I}$ is the inertial tensor, $\boldsymbol{\omega}$ is the angular velocity, and $\dot{\boldsymbol{\omega}}$ is the angular acceleration.¹

5.2.1 Forces and Torques

Given a scene with convex polygonal objects, without loss of generality, we can represent the forces between contacting objects by a set of forces acting on the vertices of the convex hull of their contact region (Featherstone, 1987). Using this representation, the equations of motion for each object can be written as

$$M\dot{\mathbf{v}} = M\mathbf{g} + \mathbf{F}_b + \sum_{c \in C} \delta_c \left(\mathbf{F}_m^c + \sum_{p \in R_c} \mathbf{F}_p^c \right) \quad (5.3)$$

$$\mathbf{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega} = \mathbf{N}_b + \sum_{c \in C} \delta_c \left(\mathbf{N}_m^c + (\mathbf{r}^c - \mathbf{r}) \times \mathbf{F}_m^c + \sum_{p \in R_c} (\mathbf{r}_p - \mathbf{r}) \times \mathbf{F}_p^c \right) \quad (5.4)$$

where $\mathbf{g}$ is the acceleration due to gravity, $\mathbf{r}$ is the center of gravity of the object, and $\mathbf{F}_b$ and $\mathbf{N}_b$ are optional body forces and torques that may act upon the object. We use the body forces and torques to implement the BODYMOTOR assertion in our ontology. C denotes the set of all contacts involving the object. For each contact $c \in C$, let R_c denote its contact region. As described in Chapter 4, the contact region is represented by a set of points, $\{p | p \in R_c\}$, at the vertices of the contact region. The vector $\mathbf{r}_p$ denotes the position of point p in the global reference frame. For each point p we apply the contact force vector $\mathbf{F}_p^c$ at the point $\mathbf{r}_p$. In addition, we allow optional force $\mathbf{F}_m^c$ and torque $\mathbf{N}_m^c$ at each contact c . These are used to implement the linear motor and angular motor assertions in the ontology. Note that the force $\mathbf{F}_m^c$ is applied at the *centroid* of the contact region R_c (denoted by $\mathbf{r}^c$), so we must add the resulting torque, $(\mathbf{r}^c - \mathbf{r}) \times \mathbf{F}_m^c$, to Equation 5.4. Finally, the terms $\delta_c \in \{-1, 1\}$ encode the direction of the normal vector between contacting objects. (The signs of δ_c are arbitrary so long as the contact forces between objects have opposite signs.)

5.2.2 Contact Conditions

Assuming that there are no degenerate contacts we can represent each contact region R_c by a set of one or more contact points and a vector $\mathbf{n}^c$ normal to the contact region.² Contact points that are not asserted to be ATTACHED must obey the normal force constraint

$$\mathbf{F}_p \cdot \mathbf{n}^c \geq 0 \quad (5.5)$$

¹Note that for the 2D case, the moment of inertia, $\mathbf{I}$ is a scalar, and the term $\boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega}$ vanishes.

²As described in Chapter 4, contacts are *degenerate* when the normal vector to the contact region is undefined. Our current system does not allow these contacts.

where $\mathbf{F}_p$ is the contact force at each point $p \in R_c$. This constraint enforces the condition that non-attached objects can “push” each other at contact points, but cannot “pull” each other.

In addition, contact points have tangential forces limited by friction. When the relative velocity at the contact point is zero we constrain the tangential forces by a *Coloumbic friction* model

$$\|\mathbf{F}_p^t\| \leq \mu_r \mathbf{F}_p^n \quad (5.6)$$

where $\mathbf{F}_p^t$ and $\mathbf{F}_p^n$ are the tangential and normal components of the contact force, $\mathbf{F}_p$, respectively. The coefficient μ_r is the *coefficient of resting friction*. In the case of objects with relative motion we limit the tangential forces as above, but use a different coefficient μ_s , the *coefficient of sliding friction*. In general we will have $\mu_s < \mu_r$. In addition to the magnitude constraint, we also need to limit the direction of sliding friction to be *opposing* the direction of motion. This is achieved by the additional constraint

$$\mathbf{F}_p \cdot \mathbf{v}_p \leq 0 \quad (5.7)$$

where $\mathbf{v}_p$ is the relative velocity of the contact point p on the two objects.

When there is a linear motor between two objects, there can be an arbitrary tangential force, $\mathbf{F}_m^c$, between them. This tangential force can exist irrespective of the relative motion of the two objects. (When the objects are not moving relative to one another, the tangential force acts like the resting friction force except that the magnitude of the force is not restricted; when there is relative motion, the force can act either in opposition or in support of the observed motion.) To ensure that $\mathbf{F}_m^c$ exerts only a tangential force between contacting objects, we need the additional constraint

$$\mathbf{F}_m^c \cdot \mathbf{n}^c = 0 \quad (5.8)$$

Where $\mathbf{n}^c$ is the normal to the contact region R_c . (When there is no linear motor between the objects, $\mathbf{F}_m^c$ is constrained to be 0.)

Finally, when there is an angular motor between two objects, we allow the torque $\mathbf{N}_m^c$ to be arbitrary. Otherwise, $\mathbf{N}_m^c$ is constrained to be 0.

5.2.3 Mass and Mass Distribution

To perform force balancing, we need each object’s center of mass, total mass, and moment of inertia. To determine these properties from image data, we assume that center of mass of each object is at that object’s geometric center, but allow the object to have a variable density and mass distribution.³

Object masses are left as free parameters (constrained to be positive) that can be adjusted

³The actual value used for the center of mass is not critical, provided it is uniquely determined from the sense data. If the center of mass is treated as a variable, however, the equations of motion (Equations 5.3 and 5.4) become nonlinear, and feasibility can no longer be tested with linear programming.

Figure 5.1: Bounds on the moment of inertia for a two-dimensional object. (a) Upper bound for I provided by considering an extreme case where all of the mass is placed at the furthest point(s) from the center (r_{max}). (b) Lower bound provided by considering an alternate case where all of the mass is distributed uniformly inside a disk inscribed in the object (r_{min}).

in order to find a feasible solution. In the current implementation, we constrain the object masses to be proportional to the area of the objects in the image

$$\frac{1}{\gamma}A \leq M \leq \gamma A \quad (5.9)$$

where A is the image area and γ is a tolerance factor (a value of 2 is used in the current system).

For the inertial tensor, I , we place bounds on the moments in particular directions. For the case of two-dimensional motion considered in this thesis, the inertial tensor is a scalar. In order to reflect the uncertainty of the actual mass distribution, we allow a range for I . An upper bound for I is provided by considering an extreme case where all of the mass is placed at the furthest point(s) from the center. A lower bound is provided by considering an alternate case where the mass is distributed uniformly inside a disk inscribed in the object. (See Figure 5.1.) Together, these provide the constraint

$$\frac{1}{2}Mr_{min}^2 \leq I \leq Mr_{max}^2 \quad (5.10)$$

where r_{min} and r_{max} are the minimum and maximum radii of the object, respectively.

An interpretation is feasible if the motion equations can be satisfied subject to the contact conditions and the bounds on the mass and inertia described above.

5.3 Reduction to Linear Programming

In this section we show how to reduce the constraints in the previous section to a set of linear equalities and inequalities. The existence of a solution can then be determined by linear programming.

The constraints given in the previous section can be approximated by a set of linear equations as follows. For each object o , we introduce variables, $\mathbf{F}_b^o = [F_{bx}^o, F_{by}^o, F_{bz}^o]^T$ and $\mathbf{N}_b^o = [N_{bx}^o, N_{by}^o, N_{bz}^o]^T$ corresponding to the body forces and torques acting on the object. In addition, we have a variable M^o , corresponding to the object mass, and a variable I^o , corresponding to the object inertia. For each contact c , we introduce variables, $\mathbf{F}_p^c = [F_{px}^c, F_{py}^c, F_{pz}^c]^T$ for each contact point $p \in R_c$, where R_c is the contact region for contact c . Finally, for each contact we introduce variables $\mathbf{F}_m^c = [F_{mx}^c, F_{my}^c, F_{mz}^c]^T$ and $\mathbf{N}_m^c = [N_{mx}^c, N_{my}^c, N_{mz}^c]^T$ for the forces and torques due to the linear motor and angular motors at contact c .

For each object, the motion equations (Equations 5.3 and 5.4) can be written as a set of linear equalities by taking the x , y , and z components separately. For each contact, the constraints on normal force (Equation 5.5) and sliding friction (Equation 5.7) are linear inequalities. To approximate the bounds on the magnitude of the tangential component of the frictional force (Equation 5.6), we restrict the magnitude in a set of two or more component directions. We use the following approximation (taken from (Blum et al., 1970))

$$\begin{aligned} |\mathbf{F}_p \cdot \mathbf{t}_1| &\leq K\mu|\mathbf{F}_p \cdot \mathbf{n}^c| \\ |\mathbf{F}_p \cdot \mathbf{t}_2| &\leq K\mu|\mathbf{F}_p \cdot \mathbf{n}^c| \end{aligned} \quad (5.11)$$

where $\mathbf{t}_1$ and $\mathbf{t}_2$ are orthonormal vectors tangent to the contact region. The constant K , $\frac{1}{\sqrt{2}} \leq K \leq 1$, controls the form of the approximation of the constraint (see Figure 5.2).

The settings of the assertions determine which constraints are active. Specifically, when objects are contacting but not attached, the contact forces $\mathbf{F}_p^c$ are constrained by Equations (5.5), (5.6) and (5.7). Otherwise the contact forces are unconstrained. When objects are contacting but do not have a linear motor, the forces $\mathbf{F}_m^c$ are constrained to be zero. Otherwise these forces are unconstrained. Similarly, when objects are contacting but do not have an angular motor, the torques $\mathbf{N}_m^c$ are constrained to be zero. Otherwise these torques are unconstrained. Finally, when objects do not have a body motor, we must constrain the body forces and torques ($\mathbf{F}_b^o$ and $\mathbf{N}_b^o$). Since the measurement of the object motions will never be completely accurate, however, we need to allow a small tolerance for force and torque equations. To do this, rather than setting the forces and torques to be zero, we bound the magnitude of the body forces and torques by small values

$$\begin{aligned} \|\mathbf{F}_b^o\| &\leq M^o \dot{v}_0 \\ \|\mathbf{N}_b^o\| &\leq I^o \dot{\omega}_0 \end{aligned} \quad (5.12)$$

Figure 5.2: Bounds on the magnitude of the frictional force at a contact point by using two or more linear inequalities. (a) $\mathbf{F}_p$, the contact force at point p decomposed into the normal force $\mathbf{F}_n$ and the tangential force $\mathbf{F}_t$. $\mathbf{t}_1$ and $\mathbf{t}_2$ are two basis vectors in the contact plane. (b) Limiting the magnitude of the $\mathbf{F}_t$ by bounding the the projection in the component directions $\mathbf{t}_1$ and $\mathbf{t}_2$. The constant K determines the form of the approximation ($K = \frac{1}{\sqrt{2}}$ for the inner square, $K = 1$ for the outer square).

where M^o is the object mass, I^o is the object inertia, and $\dot{v}_0 > 0$ and $\dot{\omega}_0 > 0$ are small tolerances for the linear and angular acceleration, respectively. To enforce these conditions, we use an approximation like that in Equation (5.11) where we limit the magnitudes in each of the x , y , and z component directions.

5.3.1 Linear Programming Problem

Since all of the above equations and inequalities are linear, dynamic feasibility can be reduced to a feasibility test using linear programming. In this section we present a brief overview of linear programming and show how the feasibility test above can be translated into a linear programming problem.

A *Linear programming problem* is defined as follows (Luenberger, 1984).⁴ Given N independent variables $x_1, x_2, \dots, x_N$, maximize the function

$$z = c_1x_1 + c_2x_2 + \dots + c_Nx_N \quad (5.13)$$

subject to nonnegativity constraints

$$x_1 \geq 0, x_2 \geq 0, \dots, x_N \geq 0 \quad (5.14)$$

and simultaneously subject to M additional constraints, of the form

$$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{iN}x_N \leq b_i \quad (5.15)$$

or,

$$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{iN}x_N \geq b_i \quad (5.16)$$

or,

$$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{iN}x_N = b_i \quad (5.17)$$

where $b_i \geq 0$. The various a_{ij} 's can have either sign, or be zero. The fact that the b 's must all be nonnegative is a matter of convention only, since you can multiply any contrary inequality by -1 . Note that there is no particular significance to the number of constraints M being less than, equal to, or greater than the number of unknowns N .

To translate our constraints to the form of Equations 5.13–5.17, we need to employ two transformations (Luenberger, 1984). First, to allow for positive and negative values for the unknowns (e.g., for the components of the force and torque vectors in our system), we replace each variable, V , by the expression $V^+ - V^-$, where $V^+ \geq 0$ and $V^- \geq 0$ are enforced by the nonnegativity constraint in Equation 5.14. Second, to implement bounds on the absolute values

⁴An alternate form for linear programming problems is the so-called *restricted normal form* which contains only equality and nonnegativity constraints. Without loss of generality, any linear program can be translated into normal form by the introduction of 'slack variables' (Luenberger, 1984).

of quantities (e.g., Equations 5.11 and 5.12), we introduce a pair of constraints. Specifically, we replace each constraint $|V| \leq v$ by the conjunction of constraints $V \geq -v$ and $V \leq v$. Given these transformations, all the constraints can be written in the form given by Equations 5.13–5.17.

Linear programming is usually posed as an *optimization* problem. However, here we are concerned only with the existence of a feasible solution, and not with a particular optimization. To determine if a feasible solution exists, we need to perform another transformation. Specifically, we write an *auxiliary objective function*

$$z = -z_1 - z_2 - \cdots - z_M \quad (5.18)$$

where z_i are so-called ‘artificial’ variables. For each constraint of the form $\mathbf{a}_i^T \mathbf{x} \leq b_i$, we write $z_i = b_i - \mathbf{a}_i^T \mathbf{x} - y_i$. For each constraint $\mathbf{a}_i^T \mathbf{x} \geq b_i$, we write $z_i = b_i - \mathbf{a}_i^T \mathbf{x} + y_i$. And for each constraint $\mathbf{a}_i^T \mathbf{x} = b_i$, we write $z_i = b_i - \mathbf{a}_i^T \mathbf{x}$. The terms $y_i \geq 0$ are so-called ‘slack variables’, introduced to translate inequality constraints to equality constraints. Since the terms z_i are zero only when each type of constraint is satisfied, and since all y_i and z_i are nonnegative, a feasible solution exists only when the optimal value of z is zero. Therefore, to test feasibility of our constraints, we can set up an auxiliary objective function, optimize it, and test whether z reaches zero.

In practice, most optimization packages perform a feasibility test automatically as part of the optimization, without requiring the user to explicitly set up the auxiliary optimization function or slack variables described above. To test feasibility we used one such package, based on the *Simplex routine* (Press, Teukolsky, Vetterling, & Flannery, 1992).

5.3.2 Problem complexity

The size of the linear programming problem will depend on the number of objects, the number of contacts, and the number of individual points in each contact. To simplify the following analysis, we will count the number of *signed* variables. (The actual number of variables and constraints may increase by at most a factor of two by the introduction of signed variables.) Given a scene with $|O|$ objects, $|C|$ contacts, and a total of $|P|$ contact points (the total number of points in all contacts), the system will contain $N = 8|O| + 6|C| + 3|P|$ variables. That is, for each object there will be 8 variables (three degrees of freedom (dof) for the body force, three dof for the body torque, one dof for the object mass, and one dof for the object inertia). For each contact *region* there will be three dof corresponding to the forces from the linear motors and three dof corresponding to the torques from the angular motors. Finally, for each contact *point* there will be three dof for the contact force. The system will contain $6|O| \leq M \leq 12|O| + 6|C| + 4|P|$ constraints on the variables. For each object, there will be six constraints corresponding to the constraints on the observed linear and angular accelerations

in Equations 5.3 and 5.4. The exact number of other constraints will depend on the assertions: If objects have a body motor, there will be no additional constraints on objects, otherwise there will be six additional constraints corresponding to the limitations on the body forces and torques (Equation 5.12). If objects are attached there will be no constraints at the contact points. Otherwise there will be three constraints if the objects are in *resting* contact (one due to Equation 5.5 and two due to Equation 5.11), or four constraints if the objects are in *sliding* contact (Equations 5.5, 5.11 and 5.7). Finally, if contacts don't have a linear motor or angular motor, there will be six additional constraints ensuring that the forces and torques are zero at the contact.

5.4 Examples of the force-balancing procedure

To illustrate how the force balancing procedure works, we show two examples.⁵ In the first example (Figure 5.3) a box is sliding on a table top. (The other object in the figure depicts a hand that has previously hit the box.) Figure 5.3(a) shows the acceleration vectors for the box. Figure 5.3(b) shows the feasible solution found by the force balancing procedure. All of the force and torque vectors (including the effect of gravity) on the box are shown, as well as the values for the mass and inertia of the box. In order to balance, the sum of all the forces and torques in each of the component directions (x , y , and z) must be zero. In this case, the vertical force due to gravity is offset by the upward forces from the table. The leftward forces from the table represent the friction between the table and the box, accounting for the deceleration of the box. Note the presence of small body forces and torques corresponding to the tolerances allowed by Equation 5.12. The depiction of forces acting on the object in Figure 5.3(b) is similar to “free-body diagrams” used in physics. In particular, for each object in the scene (except the ground object), there will be a corresponding force balance condition (plus the condition that contact forces between objects are equal and opposite).

Figure 5.4 shows a second example of the force balancing procedure. In this case the box is tipping. (There is also a hand in the scene, but only the motion of the box is analyzed in this figure.) Figure 5.4(a) shows the accelerations while Figure 5.4(b) shows the hypothesized forces and torques acting on the box.⁶ Since the vertical and angular acceleration of the box can be explained by the torque generated around the contact point, the box motion can be explained without the presence of any motors.

Force balancing for other objects, such as the hand, proceeds in a similar fashion. Note that in Figure 5.3, since the hand is not supported by the ground plane, a feasible force balance for this object will require a body motor.

⁵Both of these examples depict 2D objects in a layered scene model. Recall that, in the current implementation, all the forces and torques are stored as 3D vectors, with the third (i.e., z) component of each vector set to zero.

⁶Note that since the box is accelerating away from the hand, the contact between the hand and the box is *vanishing* and therefore cannot exert any force on the box.

Figure 5.3: Force balancing for a sliding box. (A frame from the hit sequence in Chapter 8.) The box is sliding (decelerating) along the table top. (The other object, shown with a dotted line, depicts the hand.) (a) Accelerations for the box. (b) Feasible force balance (forces, torques, mass, and inertia) for the box.

Figure 5.4: Force balancing for a tipping box. (A frame from the tip sequence in Chapter 8.) The box is tipping to the right. (The other object, shown with a dotted line, depicts the hand.) (a) Accelerations for the box. (The arc denotes the angular acceleration of the box.) (b) Feasible force balance (forces, torques, mass, and inertia) for the box.

5.5 On the Relationship Between Assertions and Feasibility Constraints

Before continuing, we pause to highlight an important property of the assertions used in the dynamics domain considered in this thesis. We claim that adding assertions to an interpretation will always result in a *weakening* of the feasibility constraints for that interpretation. In particular, this means that, adding any set of contact, attachment, or motor assertions to a feasible interpretation can never make that interpretation infeasible. We will exploit this particular property when we describe the search for preferred interpretations in the next chapter.

To justify this claim, consider the following argument. It is easy to see that adding any type of motor assertions will weaken the feasibility constraints. Specifically, adding a BODYMOTOR assertion removes any constraints on the body forces and torques applied to objects. Similarly, adding a LINEARMOTOR (or ANGULARMOTOR) assertion at a contact removes any restrictions on forces (or torques) applied at the contact point(s). Finally, adding an ATTACH assertion removes constraints on the normal and tangential forces between objects at their contact points (Equations 5.5–5.7). Adding CONTACT assertions between objects also weakens the feasibility constraints. To see this, imagine introducing an extra, zero-valued, force between every two objects that do not contact. If a contact were added between any two objects, the extra force could take on other, nonzero, values, and hence results in a weakening of the constraints.

To illustrate the relationship among the assertions and the feasibility constraints, we present some simple examples.

Figure 5.5(a) shows the constraints for a block resting on a horizontal surface. Considering only the y component of the forces, the force-balance condition says that:

$$F_y^m + F_y^c + Mg_y = 0 \quad (5.19)$$

where F^c is the contact force between the block and the surface, F^m is the force acting on the block due to an (optional) BODYMOTOR, g is the acceleration due to gravity, and M is the mass of the block. Solutions to the force-balance must lie on this constraint line, as shown at the right in Figure 5.5(a). The constraint, $F_y^c \geq 0$, is derived from the contact condition between the block and the table (i.e., $F^c \cdot \mathbf{n} \geq 0$). The constraint $F_y^m = 0$ says that there is no motor on the block. When there are no attachment or motor assertions, there is a single feasible solution, shown by the dot in the figure. Note that the addition of ATTACH and BODYMOTOR assertions will remove both the contact and motor constraints, respectively. Figure 5.5(b) shows a similar situation, except that the block is suspended under the horizontal surface. We have the same force-balance condition as Figure 5.5(a), except now, there cannot be a feasible solution unless we have either an attachment between the block and the surface, or a motor on the block.

Figures 5.6(a) and 5.6(b) show a block decelerating and accelerating on a horizontal surface, respectively. Considering the x -component of the dynamics equations, the force-balance

Figure 5.5: Constraints for a resting block. F^c denotes the contact force between the surface and the block. F^m denotes the (optional) motor force acting on the block. Mg denotes the gravitational force. (Only the y -components of the forces are shown.) (a) A block resting on a horizontal surface. The dot shows the feasible solution. (b) A block suspended below a horizontal surface. There is no feasible solution (unless we allow motors or attachments).

condition gives the constraint:

$$F_x^m + F_x^c = Ma_x \quad (5.20)$$

where F^c is the contact force between the block and the surface, F^m is the force acting on the block due to an (optional) BODYMOTOR, a is the acceleration of the block, and M is the mass of the block. The additional constraints are given by: $F_x^m = 0$ (i.e., the block is a passive object), $F_x^c \leq 0$ (i.e., the frictional force opposed the direction of motion), and $F_x^c \geq -\mu_s |F_y^c|$ (the magnitude of frictional force), respectively. The constraints shown in Figure 5.6(a) have a feasible solution (provided the required contact force, F_x^c , is within the range allowed by friction). The constraints shown in Figure 5.6(b), however, do not have a feasible solution (unless we allow a motor on the block).

5.6 Summary

We have presented a test for feasibility based on the Newtonian mechanics of a simplified scene model. We have shown that the feasibility conditions can be written as a set of linear constraints that can be tested by linear programming. Finally, we have shown that, for the particular choice of assertions in this thesis, the addition of assertions to an interpretation results in a weakening of the feasibility constraints for that interpretation. The system described in Chapter 7 uses this domain theory during the search for preferred interpretations. Before describing the implementation, however, we will first describe the computation of preferred interpretations in the next chapter.

Figure 5.6: Constraints for a block sliding on a horizontal surface. The block is sliding to the right (i.e., $v_x > 0$). F^c denotes the contact force between the surface and the block. F^m denotes the (optional) motor force acting on the block. Ma denotes the force due to acceleration. (Only the x -components of the forces are shown.) (a) The block is decelerating (i.e., $a_x < 0$). The dot shows the feasible solution. (b) The block is accelerating (i.e., $a_x > 0$). There is no feasible solution (unless we allow a motor on the block).

Chapter 6

Preferred Interpretations

6.1 Preferences

Given a fairly rich ontology, it is common for there to exist multiple feasible interpretations for a given scene. Indeed, for any configuration there is always at least one trivial interpretation in which every object has a motor. Similarly, since it is always possible to attach objects that are contacting and not moving relative to one another, extra attachments are also possible. Thus, multiple interpretations can be expected. Rather than considering all interpretations, we seek interpretations that require, in some specified sense, *minimal* assumptions about the various objects.

In order to specify a preference ordering, we must specify what is minimized and under what conditions the minimization is to be applied. In the current system we use the following general principles to specify preferred interpretations:

1. Prefer interpretations that contain the smallest sets of motors and attachments. (i.e., the smallest set of BODYMOTOR, LINEARMOTOR, ANGULARMOTOR, and ATTACH assertions).
2. The minimization of motors is taken to occur at a *higher priority* than the minimization of attachments. That is, we are allowed to add any number of attachments in order to remove any of the motor assertions.
3. Finally, in selecting a minimal set of motors and attachments, the set of CONTACT assertions is allowed to vary.

Item 1 encodes the specification that it is preferable not to resort to the use of a motor or an attachment, all else being equal. That is, the absence of these assertions is considered to be a weaker assumption about the scene than the presence of these assertions. Such a rule

would make sense when these assertions are relatively unlikely to occur in the world.¹ Item 2 encodes the specification that motors are considered to be much stronger assumptions than attachments. Such a rule would make sense, for example, when the occurrence of motors is much less likely than the occurrence of attachments. Finally, Item 3 encodes the specification that we are indifferent to the various contact relations in the scene. That is, we are allowed to assert any contact relations that allow us to reduce the set of motors and/or attachments in the scene. In the work that follows, we will refer to the assertions to be minimized (the motor and attachment assertions) as *ordered assertions*, while we will refer to the assertions that are allowed to vary (the contact assertions) as *variable assertions*.

When the preferences can be written in this simple form, the induced preference relation on interpretations is given by a *prioritized subset ordering* on the sets of ordered assertions in the various feasible interpretations. We determine the preference order for any two interpretations by first comparing the assertions made at the highest priority. If the highest priority assertions in one interpretation are a subset of the highest priority assertions in a second interpretation, the first interpretation is preferred. If the two sets of assertions at this priority are not ordered by the subset relation, that is neither set contains the other, then the two interpretations are considered to be unordered. Finally, in the case that the assertions at the highest priority are the same in both interpretations, then we check the assertions at the next lower priority, and so on. Note that the prioritized subset ordering applies only to the ordered assertions; the variable assertions can assume any value in order to minimize the sets of ordered assertions.

Example

An example of such a preference ordering was given in Chapter 3. (See Figure 3.3.) For this example, there exist two preferred interpretations (at the top of the figure) consisting of either a body motor on the hand or a body motor on the can. Since these interpretations contain different sets of motor assertions, however, they are incomparable. The lower three interpretations are all less preferred since they contain both motor assertions. Finally, the bottom interpretation is the least preferred of the lower three interpretations since, in addition to containing both motors, it also contains an extra attachment assertion. Note that the two interpretations in the middle of the figure are *equally preferred* since they contain identical sets of ordered assertions (attachments and motors).

6.1.1 Aside: Qualitative Probabilities and Preferences

Before proceeding, it is worth pausing to consider the motivation for the preference theory used in this thesis. As described above, the general idea of preferences is to choose the interpretations that are *most plausible* or *most likely* given the sense data. In particular, we can view perception

¹Note that this is a very weak preference in that it does not include any information about the relative likelihood of the various assertions.

as a *Bayesian inference* problem where the objective is to find the most likely interpretation given the sense data, our prior assumptions, and the constraints in the world. (See (Knill & Richards, 1996) for a recent collection of papers advocating a Bayesian view of perception.) However, as argued by Richards, Jepson, and Feldman (1996), in the real world we will usually not have exact prior probabilities for all possible events. Instead, it is more appropriate to specify some *qualitative* probabilities that place an ordering on the interpretations, but do not require an exact computation of their probability. Note that since we are not specifying the exact probabilities, the resulting preference ordering need not be complete. In fact, in most cases preferences will only serve to provide a *partial ordering* on interpretations.

Several different preference theories are possible, depending how much we know about the prior probability of the interpretations. A very simple preference order is based on *subset ordering*. That is, an interpretation I_1 is preferred over I_2 iff the set of assertions A_1 in I_1 is a strict subset of the assertions A_2 in I_2 . These type of preference rules are very weak, since they do not exploit knowledge about the relative probability of the various assertions. In particular, such a theory will often result in multiple preferred interpretations, each containing incomparable subsets of the ordered assertions. Such a theory would be suitable, for example, when we know that the prior probability of the assertions is small, but we do not know their relative likelihoods.²

Another approach is to rank interpretations based on the *cardinality* of the assertion sets (i.e., the number of ordered assertions present in the interpretation). While this will result in a total ordering of the interpretations, this type of ordering would only be appropriate when the ordered assertions are known to occur with roughly equal probability.³ If we know the relative probability of the assertions, this approach can be extended by assigning different *weights* to each of the assertions, and ranking interpretations according to their total weight. This type of ordering is similar to *minimum principles* (Hochberg & McAlister, 1953; Leeuwenberg, 1969) and the so-called *workshop metaphor* proposed by Adelson and Pentland (1996).⁴

In this thesis, we use the former approach, based on subset inclusion, except that we also add a *priority ordering* between different assertion sets. That is, assertions at the highest priority (e.g., motors) are minimized before assertions at the lower priority (e.g., attachments).

⁵ Such theories provide a compromise between those requiring detailed information about the

²Note this form of preference is *sound* (i.e., will not result in orderings that are inconsistent with the prior probabilities of the interpretations) only when $Pr(I_1)$, the prior probability of an interpretation I_1 , is less than $Pr(I_2)$, whenever $A_1 \subset A_2$. We can consider this condition on the prior probability distribution to be a form of *weak independence*. See also (Richards et al., 1996) for a justification of such preference rules based on *conditional independence* of the assertions (e.g., probability distributions where the dependencies can be described by a directed acyclic graph (DAG)).

³Again, note that we are making implicit independence assumptions when using this form of preference theory.

⁴For arguments against these approaches see (Leeuwenberg & Boselie, 1988) and (Jepson, 1996).

⁵Note that, we are implicitly assuming the prior probability of every setting of the lower priority assertions is less than the prior probability of *any* interpretation with a larger set of higher priority assertions. Strictly speaking, this assumption is justified only when the ratio of the prior probability of the higher priority assertions to the lower priority assertions is infinitesimal. That is, no combination of lower priority assertions can have a

probability distribution and those based on simple subset inclusion. As described by Jepson and Richards (1993), Jepson and Richards (1992), and Richards, Jepson, and Feldman (1996), the introduction of priorities provides a convenient way to resolve conflicts among the preferences and thereby reduce the number of preferred interpretations. (See also (McCarthy, 1986) for a discussion of other applications of prioritized preference theories.)

With these points made, we now continue with a discussion of the theory and implementation of preferences based on prioritized subset inclusion.

6.2 Formalism

We introduce the following notation to describe preference ordering. Consider an interpretation $I = (A, C)$ with the configuration C and the set of assertions A .⁶ Let the assertions A be broken into disjoint sets P and V , where P are the *ordered assertions* – the terms to be minimized in our interpretation, and V are the *variable assertions* – the terms that are allowed to vary while minimizing the assertions in P . The ordered assertions P are divided into disjoint segments $P^1, P^2, \dots, P^L$ where P^1 denotes the set of assertions at the highest priority, P^2 denotes the set of assertions at the next highest priority level, and P^L denotes the set of assertions at the lowest priority. Finally, we will use the notation $P^{i,j}$, $1 \leq i \leq j \leq L$ to refer to the set of assertions in P from priority level i to j inclusive.

6.2.1 Preference Ordering

We would like to prefer interpretations that contain minimal sets of ordered assertions P . We want to express the idea that assertions in P^1 should be minimized at a higher priority than those in P^2 , P^2 should be minimized at a higher priority than P^3 , and so on, up to P^L . Let P_1 and P_2 be two sets of assertions. We write the prioritized subset ordering relation (Lifschitz, 1985): $P_1 \preceq^{1,L} P_2$ when

$$\bigwedge_{i=1}^L \left(\bigwedge_{j=1}^{i-1} (P_1^j = P_2^j) \Rightarrow (P_1^i \subseteq P_2^i) \right) \quad (6.1)$$

where $P_1^1, \dots, P_1^L$ are the parts of the tuple P_1 corresponding to the parts $P_2^1, \dots, P_2^L$ of tuple P_2 . This formula represents the lexicographic combination of L subset relations $P_1^i \subseteq P_2^i$. If $L = 1$, it becomes $P_1 \subseteq P_2$; if $L = 2$, it becomes

$$(P_1^1 \subseteq P_2^1) \wedge ((P_1^1 = P_2^1) \Rightarrow (P_1^2 \subseteq P_2^2))$$

We write $P_1 \prec^{1,L} P_2$ when $P_1 \preceq^{1,L} P_2$ and $P_1 \neq P_2$.

prior probability that is less than any single higher priority assertion.

⁶As described in Chapter 4, we represent an interpretation by the set of positive assertions. Assertions missing from A are taken to be false.

Given two assertion sets, we define a preference relation based on the sets of ordered assertions in each interpretation.

Definition 6.1 (Preference relation) Given two assertion sets $A_1 = P_1 \cup V_1$ and $A_2 = P_2 \cup V_2$ we write the *preference relation* $A_1 \prec A_2$ when $P_1 \prec^{1,L} P_2$.

The relation $\prec$ defines a *partial order* on assertion sets. Recall that a relation $\prec$ is a partial ordering if it has the following properties. Given assertion sets X , Y , and Z ,

1. The relation is *transitive*. That is, if $X \prec Y$ and $Y \prec Z$ then $X \prec Z$.
2. The relation is *antisymmetric*. That is, if $X \prec Y$ then $\neg(Y \prec X)$.
3. The relation is *antireflexive*. That is, $\neg(X \prec X)$.

Note that the since the order is *partial* the relation need not be defined between every pair of assertion sets. Note also that since the relation considers only the ordered assertions P , it does not depend on the variable assertions V .

6.2.2 Preferred Interpretations

Given a configuration C and a preference relation $\prec$, we define a preferred interpretation as follows.

Definition 6.2 (Preferred interpretation) An interpretation $I = (A, C)$ is *preferred* iff: (i) I is *feasible*. (ii) There exists no other set of assertions A' such $A' \prec A$ and $I' = (A', C)$ is feasible.

The goal of our system is to find feasible interpretations with minimal sets of assertions given by this ordering.

6.2.3 Preferences for the Dynamics Domain

Returning to the dynamics domain, we partition the assertions as follows:

- P^1 contains all BODYMOTOR, LINEARMOTOR and ANGULARMOTOR assertions.
- P^2 contains all ATTACH assertions.
- V contains all CONTACT assertions.

As described above, we find minimal sets of assertions P^1 and P^2 with V variable, and with P^1 minimized at a higher priority than P^2 .

6.3 Computing Preferred Interpretations

To find preferred interpretations, we must search the space of possible assertions. In particular, we have the following computational problem.

Given:

- Configuration C .
- A set of all possible assertions, $\mathcal{A}$, in the scene.
- A set of *admissibility constraints*, $Q(I)$, that hold iff the interpretation $I = (A, C)$ is admissible.
- A set of *feasibility constraints*, $T(I)$, that hold iff the interpretation $I = (A, C)$ is feasible.
- A preference relation $\prec$ that places an partial order on subsets of assertions $A \subseteq \mathcal{A}$.

Find:

- All assertion sets A such that $I = (A, C)$ are preferred according to the preference relation $\prec$.

The *admissibility* conditions Q were described in Chapter 4. The *feasibility* conditions T were described in Chapter 5.

The configuration C and the set of all possible assertions $\mathcal{A}$ are assumed to be provided from observations by a preprocessing stage. In particular, this stage must identify all scene objects, estimate their motions, and determine all possible assertions (contact, attachment and various types of motors) in the scene. Specifically, we have $\mathcal{A} = \mathcal{P} \cup \mathcal{V}$, where $\mathcal{P}$ are the ordered assertions and $\mathcal{V}$ are the variable assertions. Furthermore, we partition the ordered assertions into priority levels $\mathcal{P} = \mathcal{P}^1 \cup \mathcal{P}^2 \cup \dots \mathcal{P}^L$. This set of assertions forms the vocabulary for our system; Any interpretation will contain a subset of these assertions. The details of how the assertions are enumerated will be provided in Chapter 7.

A naive algorithm would simply perform an incremental search, beginning with the empty set of assertions and adding admissible assertions to this set. Each branch of the search would terminate upon finding a minimal set of assertions that satisfies the feasibility condition. Note that because of the prioritized ordering, while searching for minimal sets of a assertions at the highest priority, we would have to consider *all* settings of the lower priority assertions. Furthermore, because we are indifferent to contacts, we would have to explore every set of admissible contact assertions at each stage of the search.

When the domain has a particular structure, however, we can achieve significant computational savings by performing the search over each priority level separately. Specifically, for the dynamics domain considered here, we can search for minimal sets of motors using only the

interpretations that contain all admissible attachments. Similarly, when searching for minimal sets of attachments, we can consider only the interpretations that contain all admissible contacts. It is critical to note that this search is only correct because of the special structure of the assertions and the domain. The critical property is that if there is a feasible interpretation $I = (A, C)$ and if A' is the set obtained by adding all admissible assertions to A , then the interpretation $I' = (A', C)$ must also be feasible. This property justifies the approach where we set all of the lower priority assertions to their “most permissive” settings during each stage of the minimization. In general we refer to this property as *monotonicity*.

In the next subsection we provide a precise definition of the required structure of the domain and present the efficient search algorithm to find preferred interpretations.

6.3.1 Exploiting Domain Structure

To describe the structure of the domain, we introduce the following notation for the feasibility and admissibility conditions. Let $T(A, C)$ be a predicate that is true iff the interpretation $I = (A, C)$ is feasible. We also define $Q(A, C)$ to be a predicate that is true iff the interpretation $I = (A, C)$ is admissible. We can now describe the structure of the domain in terms of properties of the predicates T and Q .

The monotonicity property is ensured by specifying the following constraint on the predicates T and Q .

Property 6.1 The predicate $T(A, C)$ is *conditionally monotonic* given predicate $Q(A, C)$ iff:

1. If $T(A, C)$ is true, then $Q(A, C)$ is true,
2. For all assertion sets A_1, A_2 and configurations C such that: $A_1 \subseteq A_2$, $Q(A_1, C)$, and $Q(A_2, C)$ it follows that: $T(A_1, C) \Rightarrow T(A_2, C)$.

In words, this says that given $T(A_1, C)$ and a superset of the assertions A_1 , say A_2 , then from monotonicity it follows that $T(A_2, C)$ must hold so long as $Q(A_2, C)$ does. If Q is computationally tractable, then this Q can provide a useful necessary condition for T to be true.

It should be clear that the feasibility and admissibility predicates for the dynamics domain obey the conditional monotonicity property. Condition 1 holds since, as described in Section 4.3, an interpretation that is feasible is assumed to be admissible. Condition 2 says that adding admissible assertions (either ordered or variable assertions) to a feasible interpretation will never make that interpretation infeasible. This condition holds, since as described in Section 5.5, adding assertions always results in a *weakening* of the algebraic constraints of the feasibility test.

In addition to the monotonicity property, we also need the following property for the admissibility predicate Q .

Property 6.2 The predicate $Q(A, C)$ is *telescopic* iff:

1. (maximal set) For every set of variable assertions V and configuration C there exists a (unique) set of ordered assertions $\hat{P}(V, C)$ such that $Q(\hat{P}(V, C) \cup V, C)$ and $Q(P \cup V, C) \Rightarrow P \subseteq \hat{P}(V, C)$. (i.e., $\hat{P}(V, C)$ is maximal given V, C)
2. (stratified) For each assertion set $A = P \cup V$ and each priority level l : $Q(P \cup V, C) \Rightarrow Q(P^{1,l} \cup \hat{P}^{l+1,L}(V, C) \cup V, C)$
3. If $V_1 \subseteq V_2$ then $\hat{P}(V_1, C) \subseteq \hat{P}(V_2, C)$.

Condition 1 (maximal set) ensures that given the configuration C and the variable assertions V , there is a unique maximal set of ordered assertions, $\hat{P}(V, C)$. Condition 2 (stratification) ensures that at any priority level l , it will always be admissible to set all of the lower priority assertions to their maximal settings given by $\hat{P}(V, C)$. Finally, condition 3 ensures that adding variable assertions V can never remove assertions from the maximal set $\hat{P}(V, C)$. This last condition allows us to consider only the interpretations containing maximal settings of the variable assertions V . We will return to these conditions when we discuss the algorithm to find preferred interpretations.

It should be clear that admissibility predicate for the dynamics domain satisfies the telescopic property. In particular, as described in Chapter 4, since the admissibility of each ATTACH assertion depends only on the configuration C and the individual CONTACT assertions in V , there will always be a unique maximal setting of the ATTACH assertions. Similarly, since the admissibility of each BODYMOTOR, LINEARMOTOR, or ANGULARMOTOR assertion in P^1 depends only on the configuration C , the individual ATTACH assertions in P^2 , and CONTACT assertions in V , there will always be a unique maximal setting for the MOTOR assertions. Since the admissibility of the individual assertions at a given priority level depends only on the presence of lower priority assertions, the presence of variable assertions V , and the configuration C , conditions 1–3 must hold.⁷

6.3.2 Partially Minimized Interpretations

To describe the search algorithm, we need the notion of a *partially minimized* interpretation. We define I^{l-1} to be the set of all feasible interpretations of the form

$$I^{l-1} \equiv (P^{1,l-1} \cup \hat{P}^{l,L}(V, C) \cup V, C) \quad (6.2)$$

⁷Note that for the dynamics domain considered in this thesis, the admissible assertions at a given priority are also *independent*—that is, any subset of the admissible assertions is also admissible. This is not strictly necessary to satisfy the telescopic property of Q . Nor is it necessary for the algorithms described in this thesis.

where $P^{1,l-1}$ is a minimal setting⁸ of the assertions from priority level 1 to $l-1$ (inclusive), V is a setting of the variable assertions, and $\hat{P}^{l,L}(V, C)$ is the maximal admissible set of assertions from priority level l to L , given variable assertions V and configuration C .

Given $\mathbf{I}^{l-1}$, the set of all partially minimized interpretations up to priority level $l-1$, we can compute $\mathbf{I}^l$ by searching sets of assertions at priority level l .

For each setting of the assertions $P^{1,l-1}$ we define the search space

$$\begin{aligned} \mathbf{D}^l(P^{1,l-1}) \equiv \{ & I^l = (P^{1,l-1} \cup S \cup \hat{P}^{l+1,L}(V, C) \cup V, C) | \\ & (P^{1,l-1} \cup \hat{P}^{l,L}(V, C) \cup V, C) \in \mathbf{I}^{l-1}, S \subseteq \mathcal{P}^l, V \subseteq \mathcal{V} \} \end{aligned} \quad (6.3)$$

$\mathbf{D}^l(P^{1,l-1})$ contains all possible assignments of the priority l assertions for each of the interpretations in $\mathbf{I}^{l-1}$. The variable S ranges over all subsets $\mathcal{P}^l$ (all possible assertions at priority level l). Note that because the assertions from priority level 1 to $l-1$ are fixed, the only variable terms are the sets $S \subseteq \mathcal{P}^l$ and the variable assertions, $V \subseteq \mathcal{V}$.

Let $\mathbf{I}(P^{1,l-1})$ denote the set of all interpretations in $\mathbf{I}^{l-1}$ with the assertions at priority levels 1 to $l-1$ fixed at $P^{1,l-1}$. For each assertion set $P^{1,l-1}$ we compute

$$\mathbf{I}^l(P^{1,l-1}) = \min_{S \subseteq \mathcal{P}^l} \mathbf{D}^l(P^{1,l-1}) \quad (6.4)$$

where $\min_{S \subseteq \mathcal{P}^l}$ denotes *all* feasible interpretations in $\mathbf{D}^l(P^{1,l-1})$ that contain minimal subsets of $S \subseteq \mathcal{P}^l$. (Note that the sets $V \subseteq \mathcal{V}$ are allowed to vary during the minimization.) For convenience, if $\mathbf{D}^l(P^{1,l-1})$ is empty, we take $\mathbf{I}^l(P^{1,l-1})$ to be the empty set. Finally, performing the minimization over each minimal setting of assertions $P^{1,l-1}$ we have

$$\mathbf{I}^l = \bigcup_{P^{1,l-1}} \mathbf{I}^l(P^{1,l-1}) \quad (6.5)$$

Taken together, Equations (6.2–6.5) provide a recursive definition for preferred interpretations. We begin with $\mathbf{I}^0$ as a set of tuples of the form, $I^0 = (\hat{P}^{1,L}(V, C) \cup V, C)$ for every subset $V \subseteq \mathcal{V}$. For each priority level, $l = 1, \dots, L$, we compute $\mathbf{I}^l$ from $\mathbf{I}^{l-1}$ using Equations (6.3–6.5). At each stage of the computation (Equation 6.4) we perform a search for minimal subsets of assertions $S \subseteq \mathcal{P}^l$ for each partially minimized interpretation $I^{1,l-1} \in \mathbf{I}^{l-1}$.

6.3.3 Algorithm

To search for preferred interpretations we need a representation for the partially minimized interpretations $\mathbf{I}^l$. In the algorithm, we group interpretations based only on the ordered assertions $P^{1,l}$. Specifically, we represent each partially minimized interpretation by a tuple

⁸Recall that according to the prioritized subset ordering (Equation 6.1), the assertions $P^{1,l-1}$ are minimal when there is no other feasible interpretation $I' = (P' \cup V', C)$ with $P' \prec^{1,l-1} P$.

of the form $M^l = (P^{1,l}, V)$, where $P^{1,l}$ is a minimal setting of the assertions from priority 1 to l and V contains *all* assignments V of the variable assertions where the interpretation $I = (P^{1,l} \cup \hat{P}^{l+1,L}(V, C) \cup V, C)$ is admissible and feasible. Finally, we use the symbol $\mathbf{M}^l$ to represent the set of all partially minimized interpretations up to priority level l .

We use Algorithm 6.1 to find preferred interpretations according to the prioritized ordering. The algorithm begins at the highest priority (level 1) and finds minimal sets of assertions with all lower priority assertions (levels 2 to L) set to their maximal settings. For each minimal set of highest-priority assertions, the algorithm finds minimal set(s) of assertions at the next priority level, and so on, until the lowest priority (level L) is reached. At each step of the algorithm a standard *breadth-first* search is performed to find the minimal sets of assertions that are admissible and feasible. At each priority level, $\mathbf{M}^l$ holds all partially minimized interpretations up to and including priority level l . Finally, at the end of the search the sets of variable assertions associated with each M^L are adjoined to yield preferred interpretations of the form $I = (P^{1,L} \cup V, C)$.

Algorithm 6.1 Preferred Interpretations

0 Given:

- Configuration C .
- $V(C)$, the set of all admissible subsets of the variable assertions $\mathcal{V}$ given configuration C .
- Admissibility predicate $Q(A, C)$.
- Feasibility predicate $T(A, C)$.
- $\hat{P}(V, C)$, a function that returns the maximal admissible set of ordered assertions $\mathcal{P}$ given the variable assertions V and the configuration C .

1 Initialize $\mathbf{M}^0 = \{(\phi, V(C))\}$.

2 For each priority level $l = 1, \dots, L$,

2.1 For each $M^{l-1} = (P^{1,l-1}, V) \in \mathbf{M}^{l-1}$,

2.1.1 Let

$$\begin{aligned} \mathbf{M}^l(P^{1,l-1}) = \min_{\substack{S \subseteq \mathcal{P}^l \\ V \in \mathcal{V}}} \{ & (P^{1,l-1} \cup S, V) | \\ & Q(P^{1,l-1} \cup S \cup \hat{P}^{l+1,L}(V, C) \cup V, C) \\ & T(P^{1,l-1} \cup S \cup \hat{P}^{l+1,L}(V, C) \cup V, C) \} \end{aligned}$$

2.2 Let $\mathbf{M}^l = \bigcup \mathbf{M}^l(P^{1,l-1})$

3 Return the interpretations:

$$\mathbf{I} = \{(P \cup V, C) | (P, V) \in \mathbf{M}^L, V \in \mathcal{V}\}$$

While the algorithm presented here is quite straightforward, there are some subtle details of the algorithm worth noting. First, instead of recomputing the maximal admissible sets $\hat{P}^{l+1,L}(V, C)$ at each stage of the minimization, considerable savings can be achieved by pre-computing the set of all maximal assertions $\hat{P}^{1,L}(V, C)$ for each set of variable terms $V \in \mathcal{V}(C)$. In fact, the implementation described in Chapter 7 uses this strategy.

The second issue concerns need to check the admissibility predicate Q at step 2.1.1 of the algorithm. This check is required since it is possible for the admissibility of some assertions at higher priority levels $1 \dots l-1$ to depend on the presence of assertions at priority level l .⁹

Finally, the third issue concerns the treatment of variable assertions, V . While the algorithm above (and the implementation described in Chapter 7) can handle multiple sets of variable assertions, in most cases only a single set of variable assertions will be used. In the dynamics domain, this corresponds to the maximal set of admissible contacts in the scene. In a few cases, however, we may wish to allow multiple contact sets. We will return to this issue in the implementation in Chapter 7 and the experiments in Chapter 8.

Correctness of Algorithm

In this section we prove that Algorithm 6.1 finds all and only the minimal models according to the prioritized subset ordering.

Theorem 6.1 *Algorithm 6.1 finds all and only the preferred interpretations given by the prioritized subset ordering.*

Soundness (every interpretation found by the algorithm is preferred). Since the algorithm only returns interpretations I that are admissible ($Q(I)$) and feasible ($T(I)$), the algorithm cannot return an interpretation that is “more preferred” than any preferred interpretation. Suppose that the algorithm returns an interpretation $I' = (P' \cup V', C)$ that is not preferred. Since I' is not preferred, there must exist an interpretation $I = (P \cup V, C)$ such that $P \prec^{1,L} P'$. Let l , $1 \leq l \leq L$, be the highest priority level where $P'^l > P^l$ (i.e., $P'^i = P^i$ for $1 \leq i < l$). Since in step 2.1.1 of the algorithm, the search finds minimal sets of assertions $S \in \mathcal{P}^l$ at each priority level l such an interpretation I' cannot be found by the algorithm.

Completeness (the algorithm finds all preferred interpretations). We show that the algorithm finds every preferred interpretation $I = (P \cup V, C)$. Proof is by induction on priority level l .

Base case. By the *monotonicity* and *telescopic* properties, since $I = (P \cup V, C)$ is an admissible and feasible interpretation, we have $Q(P^1 \cup \hat{P}^{2,L}(V, C) \cup V, C)$ and $T(P^1 \cup \hat{P}^{2,L}(V, C) \cup V, C)$ where $\hat{P}^{2,L}(V, C) \supseteq P^{2,L}$ is the maximal admissible set of assertions, V are the variable assertions, and C is the configuration. Since step 2.1.1 of the algorithm finds all minimal subsets $S \subseteq \mathcal{P}^1$ where $\{S \cup \hat{P}^{2,L}(V, C) \cup V\}$ is admissible and feasible, the algorithm will build a partially minimized interpretation $M^1 = (P^1, V)$ with $V \in \mathcal{V}$.

⁹In fact, such a dependency occurs in our dynamics domain, where an ANGULARMOTOR assertion is admissible only when there is an corresponding, lower-priority, ATTACH assertion.

Induction Step. Suppose the algorithm has a partially minimized interpretation $M = (P^{1,l}, V)$ with $V \in \mathbf{V}$. By the *monotonicity* and *telescopic* properties, since $I = (P \cup V, C)$ is an admissible and feasible interpretation, we have $Q(P^{1,l} \cup \hat{P}^{l+1,L}(V, C) \cup V, C)$ and $T(P^{1,l} \cup \hat{P}^{l+1,L}(V, C) \cup V, C)$ where $\hat{P}^{l+1,L}(V, C) \supseteq P^{l+1,L}$ is the maximal admissible set of assertions, V are the variable assertions, and C is the configuration. Since step 2.1.1 of the algorithm finds all minimal subsets $S \subseteq P^{l+1}$ where $\{P^{1,l} \cup S \cup \hat{P}^{l+1,L}(V, C) \cup V\}$ is admissible and feasible, and adjoins the set S to $P^{1,l}$, the algorithm will build a partially minimized interpretation $M^{l+1} = (P^{1,l+1}, V)$ with $V \in \mathbf{V}$.

Conclusion. Since the algorithm builds a partially minimized interpretation $M^L = (P^{1,L}, V)$ with $V \in \mathbf{V}$ for each preferred interpretation $I = (P \cup V, C)$ and since the algorithm adjoins C and $V \in \mathbf{V}$ to each set of ordered assertions P , the algorithm is guaranteed to find all preferred interpretations $I = (P \cup V, C)$.

6.4 Summary

In this chapter, we provided a preference theory and an algorithm to find plausible interpretations. Our theory is based on prioritized minimization of the assertions in the interpretations. In particular, motor assertions are minimized at the highest priority, attachment assertions are minimized at a lower priority, and contact assertions are allowed to vary freely in the interpretations.

We showed that when the domain obeys certain properties, called *monotonicity*, the search for preferred interpretations can be performed efficiently by performing the search over each priority level separately.

Chapter 7

Implementation

The purpose of this chapter is to describe our implementation. Specifically, we present a computational system that generates all and only the preferred interpretations for the instantaneous motion of each frame in the input sequence. In this chapter we provide the implementation details for the system described in Chapter 3.

As described in Chapter 3, our implementation requires the following components:

- *Hypothesis generation.* Given image data, we need a way to generate interpretations that are consistent with the observed image data and admissible according to the layered scene model described in Chapter 4.
- *Feasibility test.* Given a candidate interpretation, we need an implementation of the feasibility test described in Chapter 5.
- *Search for preferred interpretations.* Finally, given a way to generate interpretations and check their feasibility, we need a search procedure that finds all feasible interpretations that are maximally preferred according to the preference theory described in Chapter 6.

In this chapter, we show how these individual components are integrated into a complete system. In the next chapter we will show examples of the system in operation.

Before describing these components of the system, however, we will first describe the representation for the interpretations themselves.

7.1 Representation

In this section, we describe the underlying representation used by the system. The following sections describe how this representation is used for the generation and evaluation of interpretations.

As described in Chapter 4, an interpretation consists of *configuration* information that describes the scene objects and their motions plus a set of logical propositions, called *assertions*,

that describe the various properties of objects and the relationships among objects in the scene.

7.1.1 Objects

Specifically, the configuration is represented by a list of object records, containing the following information:

- Object name
- Object shape and pose
- Object mass, center of gravity, and moment of inertia
- Object motion

The object name provides an identifier for each scene object. Implicitly, we assume that each object has a unique identifier and that all and only the objects in the scene are represented in the list.¹ The remaining parts of the object record describe the configuration information – that is, the scene information uniquely determined from the image observations and the constraints in our representation. (A discussion of the actual image preprocessing steps – how objects are tracked and how their motions are estimated – is provided in Appendices A and B, respectively.)

As described in Chapter 4, the object shape and pose are described by a single convex polygon in the global reference frame. In our system, we simply represent the polygon by an ordered list of image points corresponding to the object vertices, plus a point, r , the center of gravity of the object polygon.² The object mass and moment of inertia (a scalar for the 2D case) are unknowns that are varied (but constrained to be positive) during the feasibility test. Finally, the object motion fields record our estimates of object motion observed in the image. The linear motion for each object is represented by velocity and acceleration vectors (the motion of the center of gravity of the object in the global reference frame). The angular motion of each object is represented by angular velocity and angular acceleration vectors (both scalars for the 2D implementation).

7.1.2 Contacts

As described in Chapter 4, contacting objects can either be abutting or overlapping (see Figure 7.1). Furthermore, since objects are restricted to be convex polygons, the contact region between objects must be either a (nondegenerate) convex region (for overlapping contacts), or a point or edge (for abutting contacts). For each pair of contacting objects, we represent the contact by a record containing the following information:

¹These correspond to the unique names and closed world assumptions described in Chapter 4.

²As described in Chapter 5, in the current implementation, all points are three-dimensional; for the 2D implementation described here the third component is merely set to zero.

Figure 7.1: Contact types used in the implementation. (a) Abutting contact (*point-to-edge*), (b) Abutting contact (*edge-to-edge*), (c) Overlapping contact. For abutting contacts (cases (a) and (b)), we define a *normal vector*, $\mathbf{n}$, that points away from the first object, o_1 .

- Contact identifier
- Participant objects (o_1 , o_2)
- Contact type
- Contact geometry

The contact identifier is a unique name (formed by the concatenation of both object names), while the participant objects are pointers to the first and second objects in the contact. The contact type is one of: *point-to-edge*, *edge-to-edge*, or *overlap*³ (see Figure 7.1). The contact geometry describes the shape of the contact region (a convex polygon, represented by an ordered list of vertices), and, for abutting contacts, a unit vector, $\mathbf{n}$, normal to the contact region.⁴ The set of all scene contacts is represented by a list of contact records. In §7.2 we describe how this list is computed given image data.

7.1.3 Attachments and Motors

The remaining assertions (attachment and various types of motors) are defined in terms of the existing sets of objects and contacts. Specifically, an ATTACH assertion is represented by an attachment record that contains a pointer to the corresponding contact record. (Note that because attached objects must also contact, there will always be a corresponding contact for each attachment assertion.) Similarly, a BODYMOTOR assertion is represented by a body motor record containing a pointer to the corresponding active object. Finally, the LINEARMOTOR and ANGULARMOTOR assertions are each represented by a record containing a pointer to the

³As described in Chapter 4, we do not allow *point-to-point* contacts in our implementation.

⁴We adopt the convention that the normal vector points away from the first object, o_1 .

corresponding contact between the two participant objects. (Again, to assert a linear or angular motor between two objects requires that those two objects be contacting.)

7.1.4 Interpretations

An interpretation consists of a configuration along with a set of assertions. In our implementation, these sets are represented by lists of records as described above. Furthermore, instead of using separate lists for each type of record, a single, composite list is used. As described above, all scene objects and assertions are assumed to be in the list; any assertion not in the list is taken to be false. As we will show in the next two sections, this “flat” representation – a set of propositions represented by a list – considerably simplifies the generation of admissible assertions and the search for preferred interpretations.

7.2 Hypothesis Generation

As described in Chapter 3, the input to our system consists of a set of two-dimensional objects plus estimates for their motions in each frame of the image sequence. Specifically, for each frame of the sequence we are given a set of object outlines, described by convex polygons, plus estimates for the linear and angular velocity and the linear and angular acceleration of each polygon. This information is provided by an image preprocessing stage. Examples will be given in Chapter 8, when the experiments are described.

In order to explore the space of possible interpretations, we must first construct a set of admissible assertions. Specifically, given the configuration information from a single frame of the sequence, we generate all admissible contacts, attachments, and motors for that frame of the sequence. Our implementation begins by determining all possible contact relations between pairs of the scene objects. Once the set of possible contacts is determined the admissible sets of the remaining assertions can be determined.

7.2.1 Contacts

Given that the allowable forces between objects depends on the contact geometry and the relative motion of the objects, an analysis of the scene kinematics is necessary. Given the object list, we perform kinematic reasoning to find and classify the contact relations among the objects. In general, to determine all possible contact relations we must consider interactions among all objects. For the purposes of this implementation, however, we employ a partial test in which we consider only the pairwise constraints between objects. (We will see some limitations of this test in the experiments in the next chapter.)

Our implementation consists of two steps. First, we intersect all object polygons to determine potential contacts. Next, we perform an analysis of the motion at the contact regions to remove contacts that are inconsistent with the layered model.

Polygon Intersection

Since the object polygons are convex, when they intersect the result will be either a nondegenerate convex region for overlapping contacts, or a point or edge for abutting contacts (see Figure 7.1). In practice, however, the object polygons will have some noise. To deal with this, we apply a simple heuristic procedure consisting of three steps: noisy polygon intersection, averaging of nearby intersection points, and classification of resulting intersection regions. (Examples of the polygon intersection process are shown in Figure 7.2.)

To intersect pairs of polygons, we find all vertices of the first polygon that are inside the second polygon and vice versa. In addition, we find all points where an edge from the first polygon intersects an edge from the second polygon. Finally, if a vertex is just outside a polygon – as defined by the normal distance to an edge (possibly extended) – a single point is returned. This point is at the midpoint between the vertex and the closest point on the edge.⁵ Because there are often multiple nearby intersection points (see Figure 7.2(a)), we average nearby points to reduce the number of vertices. The convex hull of the resulting points forms the intersection region (see Figure 7.2(b)).

We expect that ideal contact regions between polygons will be either points, line segments, or nondegenerate polygons. To classify the intersection regions, we remove all vertices that are either near neighboring vertices or near the edges between neighboring vertices.⁶ The result of this process is a point, line segment, or a nondegenerate region which we take to be the intersection between the two objects (see Figure 7.2(c)). Given the intersection region between two objects, we classify the contact as either an overlap (three or more vertices), an abutment at an edge (two vertices), or an abutment at a point (one vertex). A point contact is further classified as either a *point-to-point* contact if the intersection point is near a vertex on both polygons, or a *point-to-edge* contact otherwise.

Finally, for abutting contacts, we must determine the vector normal to the contact region. For *edge-to-edge* contacts, we compute the unit vector normal to the edge defined by the two vertices of the intersection region. For *point-to-edge* contacts, we take a unit vector normal to the polygon edge involved in the contact.

While this simple polygon intersection procedure serves our purposes, it could clearly be improved by incorporating a noise model for the polygons and the contact regions. In addition, one would expect that, assuming smooth motion, we should be able to exploit motion information from neighboring frames to infer the shape and contact geometry between objects. We return to these issues in the Conclusion.

⁵Note that using the midpoint effectively loosens the constraints on abutment. The selection of this particular intersection point is not central to the approach; it is merely an attempt to get a better estimate when the objects are abutting.

⁶Note that this process will only produce ideal contact regions when the input polygons are “well conditioned”, in the sense that they contain no triplets of nearly collinear vertices.

Figure 7.2: Overview of polygon intersection process. (a) “Noisy” intersection of polygons to determine all potential contact points (all points on or near the boundary of the intersection region). (b) Averaging of nearby intersection points. (c) Classification of contact regions by removal of “degenerate” vertices. The rows depict *point-to-point*, *point-to-edge*, *edge-to-edge*, and *overlap* contact regions, respectively. (See text for details.)

Motion at Contact Regions

According to the layered scene model, pairs of objects in abutting contact cannot interpenetrate. (This corresponds to the object substantiality condition described in Chapter 4.) To check this condition, we need to consider the relative motion of the objects. As stated in Chapter 4, since the contact regions are described by convex polygons, it is sufficient to consider the relative motion of the vertices of the convex hull of the intersection region between the object polygons. Specifically, let $\mathbf{p}$ be a contact point between objects a and b , and let $\mathbf{n}_a$ be the normal vector (pointing outwards from object a) at the contact region. As described in Chapter 4, we compute the *relative normal velocity* of the objects at point $\mathbf{p}$ by

$$v_{ba}^n(\mathbf{p}) = \mathbf{n}_a \cdot (\mathbf{v}_b(\mathbf{p}) - \mathbf{v}_a(\mathbf{p})) \quad (7.1)$$

where $\mathbf{v}_a(\mathbf{p})$ and $\mathbf{v}_b(\mathbf{p})$ denote the velocity (in the global reference frame) of point $\mathbf{p}$ on objects a and b , respectively. Similarly, we compute the *relative normal acceleration* of the objects at point $\mathbf{p}$ by

$$\dot{v}_{ba}^n(\mathbf{p}) = \mathbf{n}_a \cdot (\dot{\mathbf{v}}_b(\mathbf{p}) - \dot{\mathbf{v}}_a(\mathbf{p})) + 2\dot{\mathbf{n}}_a \cdot (\mathbf{v}_b(\mathbf{p}) - \mathbf{v}_a(\mathbf{p})) \quad (7.2)$$

where $\dot{\mathbf{v}}_a(\mathbf{p})$ and $\dot{\mathbf{v}}_b(\mathbf{p})$ denote the acceleration (in the global reference frame) of point $\mathbf{p}$ on objects a and b , respectively and $\dot{\mathbf{n}}_a$ is the derivative of the normal vector.

As shown in Figure 7.3, assuming continuous motion, abutments are inconsistent with the layered model when there is an impending penetration (Figure 7.3(a,b)). In particular, an impending penetration occurs when objects abut at a point $\mathbf{p}$ and either: 1) $v_{ba}^n(\mathbf{p}) < 0$, or 2) $v_{ba}^n(\mathbf{p}) = 0$ and $\dot{v}_{ba}^n(\mathbf{p}) < 0$. In addition, abutments are inconsistent with the layered model if there is a *preceding* penetration (see Figure 7.3(c)). This occurs when objects abut at a point $\mathbf{p}$ and $v_{ba}^n(\mathbf{p}) > 0$.

In addition to the above conditions, we enforce a condition that contacts persist over time. This is to avoid spurious interpretations in which forces are applied between objects that are moving apart. As described by Baraff (1995), we say that a contact is *vanishing* if objects contact at time t_0 , but do not contact at time $t > t_0$. Our system must remove vanishing contact points so that forces cannot be applied at these points. Examples of vanishing contacts are shown in Figure 7.4. Note that all contact points may vanish, or only a subset of the contact points may vanish. While it is possible to determine which contact points vanish from the motion information and update the contact region accordingly, our implementation simply removes the entire contact if any of the points within it are vanishing.

It is important to note that, as described in Chapter 4, because our ontology only represents objects with continuous velocity and acceleration, it will be unable to represent the dynamics of systems with changing contact geometries. Such cases occur, for example, when there are collisions, or with the application or removal of contact forces. We will return to some of these issues when we consider the processing of entire sequences in Chapter 9. For the current

Figure 7.3: Abutments that are inconsistent with the layered scene model (assuming continuous motion). (a) Impending overlap: normal component of relative velocity of contact point is negative. (b) Impending overlap: normal component of relative velocity is zero and relative acceleration is negative. (c) Preceding overlap: normal component of the relative velocity is positive. (Note: we use a single arrowhead to denote the relative velocity at the contact point and a double arrowhead to denote the relative acceleration.)

implementation (and the experiments shown in Chapter 8), however, we assume that all objects are in continuous motion.

As with the polygon intersection process, the motion analysis must deal with noisy shape and motion information. To do this, tolerances are used for the velocity and acceleration conditions described above.⁷

In summary, the result of the polygon intersection and motion analysis is a set of idealized contact regions which are used in the generation of the remaining assertions.

7.2.2 Attachments

Attachments provide additional constraints on the object motions. There are numerous possible kinematic constraints between objects. As described in Chapter 4, we consider only the class of kinematic constraints formed by rigid attachment at one or more points between the objects. (This is analogous to fastening objects with one or more rivets). Examples of attachments are shown in Figure 7.5. Note that, in our two-dimensional layered model, objects may be attached at an abutment, or at an overlap.

To be consistent with the kinematic model, objects that are attached must contact (and the attachment points must be within the contact region). In addition, the relative velocity and relative acceleration of objects must be zero at the attachment points. In our implementation we make a number of simplifications to the attachment scheme described in Chapter 4. First, rather than allowing arbitrary attachment points, we require that attachment points occur only at the points on the convex hull of the contact region. Second, we require that if objects are

⁷For each of the experiments described in Chapter 8 a single value was used for the normal velocity and normal acceleration tolerances. The values ranged from 1.0–2.0 *pixels/frame* for velocity and 0.5–1.0 *pixels/frame*² for acceleration, depending on the sequence.

Figure 7.4: Removal of *vanishing* contacts. Abutments that are consistent with the layered scene model, but are vanishing. The relative normal velocity at vanishing contact points is zero, while the relative normal acceleration is positive. (a) Completely vanishing contact (both p_1 and p_2 vanish). (b) Partially vanishing contact (pivots on point p_2 ; only p_1 vanishes). (Note: we use a single arrowhead to denote the relative velocity at the contact point and a double arrowhead to denote the relative acceleration.)

Figure 7.5: Attachment relations for the layered model. (a) Attachment at a *point-to-edge* contact. (b) Attachment at an *edge-to-edge* contact. (c) Attachment at an *overlapping* contact. The filled circles denote the attachment points. In our implementation, attachments must occur at all and only the vertices of the convex hull of the contact region.

Figure 7.6: Motors in the layered scene model. (a) $\text{BODYMOTOR}(o_1)$. (b) $\text{LINEARMOTOR}(o_1, o_2, R_m)$ (along edge). (c) $\text{LINEARMOTOR}(o_1, o_2, R_m)$ (at point). (d) $\text{ANGULARMOTOR}(o_1, o_2, p)$. Note that for an ANGULARMOTOR to be admissible between two objects, the objects must also be attached.

attached, they must be attached at *all* of their contact points. (Otherwise we would have to consider all subsets of the contact points as potential attachment points.) Finally, since our implementation does not allow *point-to-point* contacts, we do not allow objects to be attached in this way either. Examples of attachments allowed by our implementation are shown in Figure 7.5.

7.2.3 Motors

Given the contact and attachment relations, the admissibility of the dynamic assertions can be determined directly as follows.

A body motor, $\text{BODYMOTOR}(o)$, is admissible for any object o (except for objects designated as **GROUND** in our ontology).

A linear motor, $\text{LINEARMOTOR}(o_1, o_2, R_m)$, is admissible for any pair of abutting objects o_1 and o_2 , provided the motor region R_m is contained within the contact region R_c between the two objects. As with attachments, in order to limit the range of hypotheses that can be generated, our current implementation makes the additional restriction that the motor region R_m must correspond to the entire contact region R_c between the two objects. Therefore, given a LINEARMOTOR assertion, the motor region R_m is uniquely determined.

Finally, an angular motor, $\text{ANGULARMOTOR}(o_1, o_2, p)$, is admissible at any pair of objects o_1 and o_2 , provided p is an attachment point between the objects. Our current implementation allows angular motors only when the attachment region is a single point. In this case the point p is uniquely determined. Note that since we do not allow *point-to-point* contacts, and since attachments must occur over the entire contact region, in our current implementation, ANGULARMOTORS will occur only at *point-to-edge* contacts.

7.2.4 Summary

In summary, the result of this process is that all admissible contacts and other assertions are determined from the configuration information. Any given interpretation will contain a subset of these assertions— that is, no other assertions are admissible in any interpretation.

7.3 Testing Feasibility of Interpretations

Given a hypothesis about scene content, we use the feasibility test described in Chapter 5 to check whether the interpretation is consistent with Newtonian mechanics. Specifically, we translate the configuration information (the list of objects plus their motion estimates) plus the set of assertions into a set of linear equations, which are tested with the linear programming procedure.

Since each object, contact, and point described in Chapter 5 is given a unique name, this process can be performed automatically by building a lookup table of variables, and then translating each constraint equation into a form suitable for the linear programming procedure.

7.4 Search for Preferred Interpretations

In this section, we show how to find preferred interpretations given the set of all admissible assertions from the previous section.

As described in Chapter 6, the preferred interpretations are the feasible interpretations that contain minimal sets of motor and attachment assertions according to a prioritized ordering. Specifically, in the ordering considered here, motors are minimized at the highest priority, while attachments are minimized at the next priority. The contacts are allowed to vary freely during the minimization.

Based on the algorithm described in Chapter 6, our implementation of the prioritized minimization has two stages: We first minimize the set of motors, then the set of attachments.

To find minimal sets of motors, we perform a breadth-first search, starting with the empty set of motors, incrementally adding new motors to the scene. A set of motors is sufficient for force balancing if there exists some feasible interpretation that contains all and only the motor assertions in this set. Each branch of the search terminates upon finding a minimal set of motor assertions required for feasible force balancing.

It may appear that to evaluate each set of motors, we have to consider every possible set of attachments and contacts. However, as described in Chapter 6, because of the special structure of the domain, it is sufficient to consider only a single interpretation containing the maximal sets of contact and attachment assertions.⁸ Specifically, to evaluate each set of motors, we test

⁸The critical structure is that: 1. There exists a unique maximal admissible set of contact and attachment assertions, and 2. Adding contact and/or attachment assertions corresponds to a weakening of the constraints of

the feasibility of one interpretation containing the set of motors plus the maximal admissible set of contacts and attachments. The feasibility test is performed as described in Chapter 5.

Figure 7.7 shows an example of the search process for a frame during the “lifting” phase of the coke sequence. (During this sequence, the hand is lifting the can above the table top.) Each ellipse denotes one of the eight admissible settings of the motor and attachment assertions. (Note that the contact assertions are fixed at their maximal admissible settings so the hand and the can are contacting in all interpretations.) The shaded ellipses denote interpretations that are infeasible.

The search begins with the empty set of motor assertions, but with hand and the can attached (Interpretation 1 in the figure). Since this interpretation is infeasible, the search continues by adding motor assertions. Note that the system need not check the other interpretation(s) having no motors, as they must also be infeasible. Adding a motor to either the hand (Interpretation 2) or to the can (Interpretation 3) is sufficient for feasible force balancing. The search terminates with a list of all minimal sets of motors sufficient for force balancing (Interpretations 2 and 3 in this case). Note that because the search terminates early, the search never considers interpretations containing more motors than necessary (interpretations containing motors on both the hand and the can in this example).

In the next stage of the search, for each minimal setting of motors, we search for minimal sets of attachment assertions. As described in Chapter 6, this is done by fixing the set of motor assertions and performing a breadth-first search over the attachments.⁹ (As with the search for motors, during the search, the maximal admissible set of contacts is used.) Referring again to the example in Figure 7.7, there are two searches for minimal sets of attachments, one for each minimal setting of motors. The first search begins with a motor on the hand (Interpretation 4), while the second search begins with a motor on the can (Interpretation 5). In both cases, the interpretations without attachments turn out to be infeasible, so the maximal sets of attachments are required.

Note that, as illustrated by the example in Figure 7.7, our algorithm achieves significant computational savings compared to an exhaustive search. The first savings occurs because, as the search terminates early, the search never considers interpretations containing more motors than necessary (interpretations containing motors on both the hand and the can in this example). The second savings occurs because, as the search for motors is performed first, when a particular set of motors is insufficient for force balancing (such as Interpretation 1 in the figure), we do not have to search for varying sets of attachments at these nodes. The result of these

the feasibility test. These conditions are ensured by the properties of the admissibility and feasibility predicates described in Chapter 6.

⁹During the breadth-first search for attachments, there is one additional issue that needs to be considered. As described in Chapter 4, in order for an angular motor to be admissible in an interpretation, the participant objects must be attached. This condition needs to be checked during the minimization of attachments. To handle this case, we modify the feasibility test to include a precondition that for any angular motor assertion, there must be a corresponding attachment assertion between the same two objects.

Figure 7.7: Illustration of the search for preferred interpretations. A frame during the “lifting” phase of the coke sequence. Each ellipse shows an admissible setting of motors and attachments. Infeasible interpretations are shaded. Interpretations not visited by the search are shown with a dotted boundary. The numbers in the ellipses denote the order that feasibility tests are executed during the search. The arrows show the preference ordering on interpretations (more preferred interpretations are at the top of the figure). The preferred interpretation(s) are outlined in bold. See text for details.

two savings is that of the eight admissible interpretations in Figure 7.7, only five are tested for feasibility.

Figure 7.8 shows a more complex search example. This corresponds to a single frame of the cars sequence (see Chapter 8). At this frame, two cars are accelerating to the right on a table top. The front and rear cars contact each other as well as the table top. Linear motors are admissible between the the cars and the table, and also between the two cars.¹⁰ Since both cars are sliding along the table top, attachment is only admissible between the two cars. The search begins by finding minimal sets of motors. Of the first four interpretations searched, only Interpretations 2 and 4 are feasible. In the next stage of the search, Interpretation 5 was feasible, while Interpretation 6 was infeasible. Note that of the 16 admissible interpretations, only six were tested during the search.

7.4.1 Generalizing the search to deal with multiple contact sets

As described above, our search implementation exploits the fact there is a unique maximal set of contact and attachment assertions for a given configuration. Such an assumption is valid for many problems, including the model considered here where we have a layered depth model and where we are indifferent to the different contact sets. More generally, however, we may wish to consider multiple contact sets in the search.

One reason for allowing multiple contact sets is that we may wish to infer what contacts are *necessary* in a particular scene. For example, during the “approach” phase of the coke sequence, there turn out to be five feasible interpretations as shown in Figure 7.9. While the current search algorithm discovers that a BODYMOTOR on the hand is sufficient to explain the dynamics, it does not determine whether the contact between the can and the table is necessary or not. The reason is that, while the search algorithm determines the smallest set of attachments and motors, it does vary the contacts from their maximal settings.

A second reason for variable contacts is if there are multiple (incompatible) maximal contact sets. Such a case occurs if we try to generalize the layered scene model. In particular, as described in Chapter 4, since we consider only *pairwise* kinematic constraints in the layered model, there was always a unique, maximal admissible set of contacts. If we attempt to generalize the model, however, by adding algebraic depth constraints between the layers, there may no longer be a single maximal contact set. (See Figure 7.10(a).) A similar problem arises if we attempt to consider three-dimensional scenes (with corresponding algebraic depth constraints) such as that shown in Figure 7.10(b).

The algorithm presented in Chapter 6 handles both of these problems (with a corresponding increase in computational cost), if we provide all possible contact sets in $\mathbf{V}$ at the start of the search. The search operates as described above, except that we must consider every possible set of contacts at each stage of the search. We return to the issue of inference of the contact set

¹⁰For simplicity body motors are not considered in this example.

Figure 7.8: Illustration of the search for preferred interpretations. A frame during the “pushing” phase of the cars sequence. See text for details.

Figure 7.9: Preference ordering for a frame during the "approach" phase of the coke sequence. There are five feasible interpretations. See text for details.

Figure 7.10: Cases where there could be multiple maximal contact sets. (a) The addition of depth constraints between layers. For example, suppose the layers have variable thickness and layer b were thicker than layer d . In this case, layer d could not contact layer a and layer c simultaneously. (b) Analogous depth constraints can occur in 3D scenes. Object x cannot touch t (the table) and the rear face of object b simultaneously. (The example shown in (b) is adapted from (Jepson & Richards, 1993).)

in the experiments in Chapter 8. We will return to issue of the limitation of pairwise kinematic constraints in Chapter 8 and Chapter 10.

7.5 Summary

In this chapter, we have described our implementation; namely, a completely automatic, vertically integrated system that goes from image observations (tracked polygons) to inferences (preferred interpretations).

Examples of the system in operation will be shown in next chapter.

Chapter 8

Experiments

8.1 Sequences

We have applied our system to several image sequences taken by a desktop environment (see Figures 8.1 to 8.5). The sequences were taken from a video camera attached to a SunVideo imaging system. MPEG image sequences were acquired at a rate of thirty frames per second and a resolution of 320×240 pixels. The 24-bit color image sequences were converted to 8-bit grey-scale images used by the tracker.

The image sequences depict events involving a variety of different types of objects (hands, toy cars, boxes, etc), and various types of interactions (grasping, hitting, sliding, etc). In particular, the sequences depict the following events:

- coke. A coke can is resting on the table top. A hand approaches, grasps the can, and lifts it off the table.
- cars. A hand (not modeled) releases a wind-up car. The car accelerates and hits a second car. After the collision, both cars remain in contact and continue to accelerate.
- hit. A box is resting on the table. A hand hits the box. The box slides, decelerates, and comes to a stop.
- arch. Three boxes are arranged in the form of an arch (i.e., the top box is supported by the left and right boxes). A hand approaches and pulls the left box out from the arch. When the left box is removed, the top block tips, falls, slides on the table top, and comes to a stop.
- tip. A hand raises a box onto its corner and allows it to tip.

Figure 8.1: The coke image sequence. Fig. 8.1 to 8.5 are the image sequences used for the experiments. The frame numbers are given below each image. Only even frames of the sequences are shown. The first frame of the sequence is shown in the upper left of each figure.

Figure 8.2: cars image sequence.

Figure 8.3: hit image sequence.

Figure 8.4: arch image sequence.

Figure 8.5: tip image sequence.

8.2 Processing

In this work, we do not attempt to solve the object recognition or scene segmentation problem; instead, we assume that the user has identified the participant objects in advance. To identify an object, the user outlines it in the first frame of the sequence. In addition, the user inputs the *ground plane*, which is taken to be a GROUND object in our ontology.¹

We use a view-based tracker to obtain the pose (position and orientation) of the objects in each frame. The motion (velocity, acceleration, angular velocity, and angular acceleration) of each object is then estimated by interpolating the pose information and differentiating. Since the tracking and motion estimation procedures are not critical to the research presented in this thesis, the details are described in Appendix A and Appendix B, respectively.

Once the sequences are processed, the input to our system consists of a set of object polygons along with estimates of their motions. In the next section we will show the interpretations for selected frames of the sequences. Before showing the interpretations, however, in the remainder of this section we will show the processing results for the image sequences.

Figures 8.6 to 8.10 show the pose estimates provided by the tracker for each of the sequences in Figures 8.1 to 8.5. The overlaid polygons show the pose of the objects in each frame. Except for a few frames where there are abrupt changes in the motion (such as the collision of the block and the table at frames 57–58 of the arch sequence), the tracker is able to follow the objects in each sequence. Note that, because we allow outliers in the matching process, the tracker is able to follow the objects as the occlusion relationships change. (Examples of occluded objects are the can in the coke sequence and the left block in the arch sequence.)

Figures 8.11 to 8.15 show the velocity and acceleration estimates for each of the image sequences. In each figure, the upper left graph shows the estimates for the x and y component velocities of each of the objects, while the upper right graph shows the estimates for the angular velocities. The lower graphs show the corresponding estimates for the linear and angular accelerations. While these estimates are somewhat noisy, we can clearly interpret the event structure from the graphs. For example, we can distinguish two distinct phases of the coke sequence (the approach and the lift) by examining the velocities in Figure 8.11. During the approach (frames 30–50), the x and y velocities of the can are zero, while the velocities of the hand are nonzero. During the lifting phase (frames 50–70), the velocities of both the hand and the can are approximately equal. The acceleration data also mirrors this trend, although the data is more noisy. Similarly, we can see clear evidence of the event structure (the collision) in the velocity and acceleration plots for the cars and hit sequences (Figures 8.12 and 8.13, respectively). In each sequence, there is a sudden increase in the velocity of the patient object after the collision. Note that since our motion interpolation algorithm allows multiple interpolants

¹The exact shape of the ground plane is not critical, provided it can be represented by a polygon and provided it is the same in each frame. Since all our experiments involve fronto-parallel motion on a table top, we simply input the ground plane as a horizontal line.

Figure 8.6: Tracking results for the coke image sequence. In Fig. 8.6 to 8.10, the object polygons were outlined (by hand) in the first frame (upper left frame in the figure). In each subsequent frames the position and orientation of the object polygons is estimated by the tracker and overlaid on the image.

Figure 8.7: Tracking results for the cars image sequence.

Figure 8.8: Tracking results for the hit image sequence.

Figure 8.9: Tracking results for the arch image sequence. Note the gap between the first frame (frame 15, where the objects are outlined), and the rest of the sequence (frames 32 to 68).

Figure 8.10: Tracking results for the tip image sequence.

at a given frame (see Appendix B), our system is able to obtain reasonable velocity estimates near step discontinuities. However, the acceleration estimates, determined by differentiating the velocities, are generally not reliable near collisions. We will return to the issue of how to deal with collisions in the next chapter. Finally, Figures 8.14 and 8.15 show the velocities and accelerations for sequences that contain significant rotational motion. The angular velocity and angular acceleration plots clearly show the onset of tipping of the top block in the arch sequence, and the forward and backward tipping motion of the box in the tip sequence.

8.3 Results and Discussion

Figures 8.17 to 8.21 show some of the preferred interpretations found for selected frames from each sequence. For each sequence, we show frames ordered from left to right. When there are multiple interpretations for a frame, the frames are organized vertically. A legend of symbols used to denote assertions is shown in Figure 8.16. The symbols are the same as those introduced in Chapter 3, except for the addition of the `LINEARMOTOR` and `ANGULARMOTOR` assertions.

Figure 8.17 shows the preferred interpretations for selected frames of the coke sequence. During the approach phase (frames 32 and 37), the system has inferred that, since the hand is moving above the table top (and not falling), the hand must have a body motor. (The `BODYMOTOR` assertion is denoted by the open circle at the center of the hand). During the lifting phase (frame 60) there are two preferred interpretations. In the first interpretation, there is a body motor on the hand, while in the second interpretation, there is a body motor on the can. In both interpretations, there is an attachment between the hand and the can. As described in Chapter 3, both of these interpretations are reasonable given that we analyze the instantaneous motion (i.e., single frames) and that we do not have any knowledge about the participant objects. We will return to this example in the next chapter, when we consider integration over time.

In addition to determining the motor and attachment assertions, our system also determines the possible contact relationships among objects. For example, in frames 32 and 37 of the coke sequence, the system has determined that in order for the can to be *supported*, the can must contact the table top. (The contact is denoted by the small dots between the can and the table.) In frame 37, there are two preferred interpretations. In the first interpretation, the hand is not contacting the can while in the second interpretation, the hand is contacting (and sliding against) the can. (The sliding contact is denoted by the small open circles at the contact points between the hand and the can.) Note that both of these interpretations are reasonable given our layered depth model.² In order to resolve this ambiguity, we would need to know

²As described in Chapter 4, the only constraint on contact relationships is that objects in abutting contact must be the same depth layer and objects that overlap in the image must be in different depth layers. In addition, the system does not use knowledge of the occlusion relationships between objects to determine the relative depth of objects in the scene.

Figure 8.11: Motion estimation results for the coke sequence. Fig. 8.11 to 8.15 show the results given by the view-based tracker followed by the robust interpolation algorithm. In each figure, the top row shows the estimates for velocity and angular velocity. The bottom row shows the estimates for acceleration and angular acceleration.

Figure 8.12: Motion estimation results for the cars sequence.

Figure 8.13: Motion estimation results for the hit sequence.

Figure 8.14: Motion estimation results for the arch sequence.

Figure 8.15: Motion estimation results for the tip sequence.

Figure 8.16: In the presentation of results to follow, we use: (a) small open circles to depict sliding contact; (b) small filled circles for resting contact; (c) large filled circles for attachment; while (d), (e), and (f) depict linear, angular, and body motors, respectively. For the first two motors, the closed curve surrounds the contact region over which the motors operate, while for body motors the large circle is placed at the object center.

Figure 8.17: Preferred interpretations for selected frames of the coke sequence. In Fig. 8.17 to 8.21, the top row shows the tracked polygons and acceleration estimates, while the frames below show the corresponding interpretation(s) found by the system. In Fig. 8.17 there are multiple interpretations for frames 37 and 60. The interpretations for frame 37 differ in whether there is a contact between the hand and the can or not. The interpretations for frame 60 differ in whether the body motor is on the hand or the can.

Figure 8.18: Preferred interpretations for selected frames of the cars sequence.

something about the three-dimensional position of the objects.

As described in §7.4.1, to infer what contacts exist in the preferred interpretations, the system must consider *all* possible contact relations between the objects. To reduce the number of interpretations found, the remaining sequences were processed using a single, maximal set of contact assertions.

Figure 8.18 shows the preferred interpretations for some frames in the cars sequence. For simplicity of presentation, we have shown only the preferred interpretations that contain linear motors. (For each interpretation shown, there is a corresponding interpretation with a body motor replacing each linear motor.) In frame 23, the acceleration of the left car is explained by a linear motor between the car and the table top. (The right car is at rest before the collision.) After the collision (frame 34), both cars are accelerating to the right. As in the coke sequence, there is ambiguity as to whether the first car is ‘pushing’ the second or whether the second car is ‘pulling’ the first. Note that in the case when the second car has the motor, the system correctly hypothesizes the attachment necessary to explain this motion. Finally, at the point of collision (frame 28) an anomalous interpretation is found. We will return to this issue in §8.3.1.

Figure 8.19 shows the preferred interpretations for the hit sequence. Before the collision (frame 28), the box is at rest (supported by the table), while the hand is moving above the table top. The system finds a single preferred interpretation with a body motor on the hand. After the collision (frame 36), the box is sliding (decelerating) and the hand is hovering above

Figure 8.19: Preferred interpretations for selected frames of the hit sequence.

the table top. Again the system finds a single preferred interpretation with a body motor on the hand. (The deceleration of the box can be explained by a frictional force between the box and the table top.) As with the cars sequence, at the point of collision (frames 31 and 33), anomalous interpretations are found. Again, we will return to this issue in §8.3.1.

Figure 8.20 shows the results for selected frames of the arch sequence. In the sequence, the top block of the arch changes from being *supported* in frame 45 to *tipping* in frame 52 when the supporting block is removed. Although the left block is moving in frame 45, it is still supporting the top block. As with the previous examples, the interpretations are ambiguous as to whether the hand is ‘pulling’ the left block or whether the left block is ‘carrying’ the hand.

The results for the tip sequence in Figure 8.21 highlight the richness of our dynamic domain. In this sequence, a hand raises a box onto its corner and allows it to tip to an upright position. Precisely at frame 30, the box changes from a state of being supported by the hand to a state of tipping. There are two preferred interpretations for frame 30 that involve only body motors: one where the hand ‘tips’ the box and the other where the box ‘tips’ itself and ‘drags’ the hand. Note that the latter interpretation requires an attachment between the box and the hand while the former does not. Frame 34 shows the situation shortly after the box is tipped. Since the center of gravity of the box is ahead of the support point, there is significant angular acceleration. Our system correctly infers that this angular acceleration is due to gravity rather than some type of motor.

Figure 8.22 shows a more detailed analysis of frame 30 of the tip sequence, where we allow all types of motors. The two images show the sequence frame with the object polygons and their velocities and accelerations. Note that the center of gravity of the box is directly above the pivot point, yet there is significant angular acceleration (caused by the hand). Below the images we show all of the preferred interpretations found by our system. The first two interpretations

Figure 8.20: Preferred interpretations for selected frames of the arch sequence.

Figure 8.21: Preferred interpretations for selected frames of the tip sequence.

Figure 8.22: How many ways are there to tip a box? The velocity and acceleration estimates for frame 30 of the tip sequence are depicted on the top row. (The lines at the object centers show the estimates for the linear velocities and accelerations, while the arcs show the estimates for angular velocities and accelerations.) All five maximally-preferred interpretations are shown below. Note that only the first two interpretations are plausible—the last three have a force balance, but are not consistent with rigid body motion. See the text for a discussion.

correspond to an active hand ‘pushing’ the box while the last three show the box ‘pulling’ the hand. Note that while all interpretations Figure 8.22 have a force balance, the last three are not consistent with rigid body motion. In particular, it is not *kinematically* feasible for the hand to be both attached to the box and in edge-to-edge sliding contact with the table. We will return to this issue in the next subsection.

8.3.1 Anomalies

While the results of our system are encouraging, the system does exhibit a number of anomalies. These anomalies generally fall into two classes. In the first type of anomaly, slowly-accelerating active objects are sometimes mistaken for passive objects. Examples of this are shown in frame 45 of the coke sequence, and in frame 33 of the hit sequence. Note that in both of these cases, the set of active objects is reduced by instantaneously attaching the hand to a neighboring object. This ignores the distinct motion of the hand in adjacent frames of the sequence. Such anomalies are to be expected given that our system only examines single frames in isolation.

A second problem concerns the detection of changing contact relations between objects. In particular, when objects collide, the estimates for relative velocity and acceleration at their contact points differ, resulting in the contact relation being deemed inadmissible. An example of this is shown in frame 28 of the cars sequence where, during a brief interval, the first car is

decelerating while the second car is accelerating. While the two cars are actually in contact, this abutment is deemed to be inadmissible due to a large difference in the acceleration estimates.³ Further examples of missed collisions are shown in frame 31 of the hit sequence and frame 36 of the tip sequence. Again, these anomalies are to be expected since our ontology is restricted to continuous velocity and acceleration, and is not designed to handle force impulses and abrupt changes in contact.

Finally, as indicated by the tip example, a more complete kinematic model is needed. As discussed in Chapter 7, since our system considers only pairwise constraints between objects, it does not check for global kinematic consistency. Further tests would be required to rule out interpretations that are not kinematically feasible. We return to this issue in the Conclusion.

8.4 Summary

In this chapter, we have presented results for the basic system described in Chapters 4–6. As described in Chapter 7, once provided with the image sequence and outlines of the participant objects, the system operates without user intervention to generate hypotheses, test their feasibility, and find preferred interpretations.

While this system has captured our intuitive knowledge of dynamics, the examples presented in this chapter show that there is often significant ambiguity in the interpretations found. In particular, when there are multiple interacting objects, there is often ambiguity as to which objects are active. In addition, there are often several possible contact relationships among objects, each of which is consistent with the observed motion. Since our system does not have any prior information about the object properties and since it considers only the instantaneous motion of objects, multiple interpretations should be expected. Nonetheless, such a representation provides a framework for an enhanced system that integrates knowledge about object properties and information observed over multiple frames. We will present preliminary attempts to integrate temporal information in the next chapter.

³Note that the acceleration of the cars should be equal since they remain in contact after the collision. In our system, however, the interpolator has smoothed over this discontinuity and given unreliable estimates of the acceleration.

Chapter 9

Integrating Information Over Time

While promising, the current system suffers from a number of fundamental limitations. One problem is that, because we consider single frames in isolation, when objects interact there is often ambiguity as to which object (if any) is responsible for generating the forces (‘motors’) we infer in the scene. For example, during the lifting phase of the coke sequence we cannot determine which object is doing the lifting. Similarly during the common motion phase of the cars sequence we cannot determine which car is generating the force. Also, since linear and angular motors are allowed to operate at the contact regions between objects, the situation is further complicated, since we do not know which object (if any) is responsible for generating these forces. For example, when putting a linear motor between the car and the table in the cars sequence, we do not know whether to attribute the force to the car or the table. A second problem occurs because we analyze the instantaneous motion at single frames. Anomalous interpretations will occur when there are motion discontinuities due to collisions, changes in contact between objects, and starts or stops of motion.

To address these problems, we propose an extended representation in which we attribute force-dynamic properties to objects, and integrate these inferences over time.

9.1 Ascribing Properties to Objects

Our approach to integrating information is to ascribe to objects a set of *properties* or *abilities* that are assumed to persist over time. For example, given successive observations of a hand moving freely or imparting forces on other objects in the scene, we would like to infer that the hand has the ability to move autonomously. Similarly, given successive observations of a hand which is lifting or pulling other objects, we may infer that the hand has an ability to grasp other objects.

Specifically, we extend our representation to include the following *object properties*:

- FLYER(*o*) — object *o* can generate an arbitrary force and torque on itself;

- DRIVER(o) — object o can exert an arbitrary tangential force along any of its edges;
- ROTOR(o) — object o can exert an arbitrary torque at any of its vertices;
- GRASPER(o) — object o can attach to any object that contacts it in the scene;

where o refers to any object in the scene. The FLYER property is similar to the BODYMOTOR assertion, except that it means the object has a BODYMOTOR in all frames of the sequence. The DRIVER, ROTOR, and GRASPER properties are similar to the LINEARMOTOR, ANGULARMOTOR, and ATTACH assertions, except that instead of treating these assertions as attributes of the contact regions between pairs of objects, the assertions are ascribed to one of the objects involved in the contact. Given this extended representation, we seek interpretations that require the smallest set(s) of properties *over the whole sequence*.

As with the previous work, we assign priorities to minimization of the various properties. Specifically, we minimize flyer properties at the highest priority, followed by driver and rotor properties at the next priority, followed by grasper properties at the lowest priority.

9.2 Sequence Processing

To integrate the inferences over all the frames of the sequence, we need a way to detect anomalous interpretations due to collisions and changes in contact relations. As described in (Mann, Jepson, & Siskind, 1997), our current system assumes that objects are in continuous motion, and does not model collisions or force impulses. Therefore, in addition to integrating information over time, we need a way to select only those frames in which the motion is continuous.

In this work, we assume that all force impulses (step changes in velocities) are due to collisions. Furthermore, we assume that all such collisions occur among visible (non-occluded) objects. To detect collisions, we implement the simple test shown in Figure 9.1. Specifically, our system looks for any impending *penetration* (negative relative velocity normal to the contact region) between two abutting objects. As shown in Figure 9.1, the collisions can be of two sorts. The first collision, a *push*, occurs when the objects remain in contact after the collision. The second collision, a *bounce*, occurs when the objects separate immediately after collision. The tests are implemented by comparing contact regions in adjacent frames of the sequence. Upon detection of a collision, the two adjacent frames are removed. In addition, because the motion estimates are unreliable in the neighborhood of a collision, we removed one preceding and two subsequent frames. (The effect of this process was to remove frames 26 to 30 of the cars sequence. None of the coke frames were removed.)

As well as removing discontinuities due to collisions, we need to remove frames in which there is a significant tangential acceleration, but the sign of the tangential velocity cannot be resolved. This occurs at sliding starts and stops. When dealing with just a single frame, the relative signs of the tangential velocity and acceleration dictate whether the motion corresponds

Figure 9.1: Collision detection process. Collisions are detected when an impending penetration is followed by either an abutting contact (a), or a separation (b). The arrows depict the object velocities. The dashed box denotes the contact tolerance between the objects.

to a stop or a start (see Figure 9.2). In situations where the velocity is too small to be able to reliably infer the sign, we can easily confuse stops and starts. Confusing a stop with a start may cause our system to infer the anomalous presence of a motor. To avoid this, we simply remove all frames containing such starts and stops. (The effect of this process was to remove frames 20 and 21 from the cars sequence. None of the coke frames were removed.)

Once the offending frames are removed from the sequence, we attempt to find minimal sets of object properties sufficient to explain the motion in each frame. Intuitively, we can view the temporal integration process as an incremental search that starts with the assumption that none of the objects have any special properties, and gradually increases the set of object properties as more behaviors are observed. Unlike the earlier system, which performed the search on single frames, for each set of properties chosen, all valid frames of the sequence are tested.

9.3 Preliminary results

In this section, we show some preliminary results for ascribing properties to objects and for integrating this information over time.

Figure 9.3 shows the results for the coke sequence. Below each frame of the sequence we show the minimal set(s) of object properties (FLYERS, DRIVERS, ROTORS, and GRASPERs) sufficient to explain the instantaneous motion in that frame. At the bottom of the figure we show the minimal set(s) of object properties sufficient to explain motion in the *entire sequence*. Note that while the system has inferred that the hand is a flyer, there are multiple interpretations since it cannot determine whether the hand or the can is a grasper object.

Figure 9.4 shows the results for the cars sequence. In this case, while there is still ambiguity at some individual frames (e.g., frame 36), temporal integration results in a single interpretation in which the left car is a driver object.¹ As described in §9.2, our system removed frame 28 (the

¹Note that we have disallowed DRIVERS (or any other properties) for the table top.

Figure 9.2: Starts and stops of sliding motion. (a) depicts an object that is decelerating to rest (stopping), while (b) depicts an object that is accelerating from rest (starting). Stopping can usually be explained by sliding friction (a tangential force opposing the direction of motion), while starting requires the presence of a motor to generate a forward force. When the velocity becomes so small that its sign is uncertain, we cannot distinguish between cases (a) and (b) given data at a single frame.

collision) and frame 20 (start of the left car) from this sequence before integrating the results over time. Note that since frame 28 (incorrectly) provides evidence for a motor on the right car, removal of this frame is essential in order to obtain a unique interpretation for the sequence.

Figure 9.5 shows results for the hit sequence. Again, a unique interpretation is found since the motion can be explained by an active hand. As expected, the deceleration of the box is explained by sliding friction on the table top. To arrive at a unique interpretation, it was necessary to remove both the collision (frame 30) and the stop (frame 40) before temporal integration.

Finally, figure 9.6 shows the results for the tip sequence. The system removed frame 30 (hand starting to slide against the table) and frame 38 (collision between the box and table). In this case the temporal integration process is only partially correct. While the dynamics analysis has recognized the passive motion of the tipping box, it has not been able to detect the 'active' hand. The problem is that while the displacement of the hand changes over time, the magnitude of the velocity is too small to reliably detect the direction of motion in frame 30.

9.4 Summary

We have shown some preliminary experiments indicating that the integration of information over time can be used to reduce some ambiguity in dynamic interpretations found by our current system. Our approach is based on ascribing force-dynamic properties to objects and finding interpretations that require the smallest set(s) of properties throughout the entire sequence.

Sequence Interpretation(s): $[\text{FLYER}(\text{hand}), \text{GRASPER}(\text{hand})]$, $[\text{FLYER}(\text{hand}), \text{GRASPER}(\text{can})]$.

Figure 9.3: Object properties and temporal integration results for the coke sequence. Minimal set(s) of *object properties* are shown for each frame of the sequence. A large open circle at the object center denotes a FLYER object while a small open circle denotes a GRASPER object. The minimal set(s) of object properties that explain the entire sequence are listed at the bottom of the figure. In this case, temporal integration is sufficient to identify the hand as a FLYER object. However, there is still ambiguity as to whether the hand or the can is a GRASPER object.

Sequence Interpretation(s): [DRIVER(*left*)].

Figure 9.4: Object properties and temporal integration results for the cars sequence. A left-right arrow is used to denote a DRIVER object while a small open circle is used to denote a GRASPER object. The minimal set(s) of object properties that explain the entire sequence are listed at the bottom of the figure. For this sequence all motion is explained by identifying the left car as a DRIVER object. Note that frame 20 (start of the left car) and frame 28 (the collision) have been removed by the temporal integration procedure.

Sequence Interpretation(s): [FLYER(*hand*)].

Figure 9.5: Temporal integration results for the hit sequence. A single interpretation is found: all motion can be explained by identifying the hand as a FLYER object. Frames 30 (a collision) and 40 (a stop) were removed by the temporal integration procedure.

Sequence Interpretation(s): [].

Figure 9.6: Temporal integration results for the tip sequence. In this case a single (incorrect) interpretation is found for the sequence: all objects are passive. See text for a discussion. Frame 30 was removed due to the start of motion between the hand and the table. Frame 38 was removed due to the collision between the box and the table.

While this approach seems promising, our representation of object properties is still very simplistic. First, we require *correspondence* between frames. That is, we assume we can identify and track each object through all the frames (and that no objects appear or disappear in the sequence). Second, we assume that object properties *persist* throughout the entire sequence. Third, we make no attempt to enforce constant masses and coefficients of friction throughout the sequence.² Finally, by neglecting frames which may contain discontinuities (or ambiguities in the case of start/stops), we are throwing away potentially valuable information. We will return to some of these issues in the Conclusion.

²This motivates a more elaborate temporal model in which we try to model the dynamics of continuous sequences. Such an approach may be possible by modeling the scene as a linear dynamical system with a set of driving inputs that are allowed to vary over time (Geoff Hinton, *personal communication*).

Chapter 10

Conclusion

In this thesis, we have presented an implemented computational system that derives force-dynamic representations from image sequences. Our system embodies a rich ontology that includes both kinematic and dynamic properties of the observed objects. Our system evaluates the feasibility of interpretations relative to the Newtonian mechanics of a simplified scene model. Finally, the system provides a representation of uncertainty along with a theory of preferences that allows the selection of plausible interpretations.

10.1 Contributions of this work

As described in the Introduction, this thesis brings together two threads of research. First, we showed how *force-dynamic* concepts can provide useful descriptions for image sequences containing interacting objects. Second, we showed what sort of machinery is necessary to build a *computational perceiver* that can extract dynamic descriptions from images.

Specifically, this thesis makes the following contributions:

1. We proposed a physics-based representation suitable for describing scene dynamics. The representation is based on the kinematic and dynamic properties of objects.
2. We presented an explicit computational procedure to test feasibility of interpretations based on the Newtonian mechanics of a simplified scene model.
3. We showed that, for scenes containing polygonal objects in non-degenerate configurations, the feasibility test can be well-approximated by a linear programming problem.
4. We presented a theory of preference ordering that allows: (a) the selection of plausible interpretations, and (b) the representation of ambiguity in the form of multiple interpretations.
5. We showed that, when the preferences are stratified according to priority level and when the domain has a particular structure, called *monotonicity*, there exists an efficient compu-

tational procedure to search for preferred interpretations. We provided a precise definition of the required structure of the domain, and presented an efficient search algorithm that exploits this structure.

6. We showed how to integrate information over time by finding the minimal sets of object properties over the entire sequence.

10.2 Extensions

While the results of the system presented in this thesis are encouraging, they could be extended in several ways.

- As described in the Introduction, a major limitation of this work is that the objects must be identified and segmented (by hand) in the first frame before processing can proceed. A natural extension is to enable automatic extraction of the 2D polygons that are used to represent the moving objects. We note that any automatic procedure will introduce error into the shapes of the polygons, so an analysis of the allowable noise in object shapes will be necessary.
- In order to work in a general environment, 3D representations are required. As described in Chapter 5, our system should be able to reason about 3D scenes when provided with suitable input about 3D shape and motion. Further work will be required to determine what type of 3D representation is suitable and how accurate the shape and motion information will have to be. In addition, we will need to provide bounds for the object inertias, which become matrices for the full 3D case.
- As described in Chapter 7, since our system only considers *pairwise* constraints between objects, it does not check for global kinematic consistency. When more than two objects interact, this may lead to anomalous interpretations. This was demonstrated by the tip example, where the hand, box, and the table form a *kinematic chain*. To rule out these interpretations we will need an analysis of higher-order kinematic constraints (See (Kramer, 1990).)
- A major simplification of our implementation is that the hypothesis generation process is “bottom up”. That is, given the image data, the configuration information is determined entirely from the instantaneous motion of the objects at a single frame of the image sequence. Presumably, we could use our knowledge about the kinematics and dynamics in adjacent frames to help improve our estimates of the shapes and motions of the objects.
- As described in Chapter 9, in order to deal with continuous sequences, we need a way to detect and classify motion events (e.g., collisions, starts and stops of motion, etc.).

A preliminary set of event categories for point-mass objects is presented in (Jepson & Feldman, 1996). (See also (Rubin & Richards, 1985).) Further work is needed to extend these categories to scenes containing polygonal objects. In addition, further work is needed to determine the feasibility conditions for motion events involving velocity discontinuities. Such a theory would require a treatment of the transfer of *momentum* as well as forces, among objects.

- Finally, to represent the causal structure of time-varying scenes, we require a representation of object capabilities and how they are expected to change over time. (Chapter 9 provides preliminary work on time-varying scenes.)

10.3 Implications and Future Work

10.3.1 The role of force-dynamic representations in vision

So far, we have described a system that infers force-dynamic descriptions from observations of image sequences. To understand sequences, however, we need a way to relate natural categories of events such as *lift*, *drop*, *throw*, and so on, to the underlying force-dynamic representation. To address this problem, we could describe events as sequences of qualitative dynamic states involving the participant objects. For example, as described in (Siskind, 1992), we may describe an event such as “A drops B” as a concatenation of two adjacent time intervals: an interval where the object B is at rest and stably supported by object A, followed by an interval where object B is falling.¹

A more subtle question, however, concerns how to define *action*. In particular, what differentiates “passive” or “natural” events such as a rolling or bouncing ball, from “willful” actions such as a person hitting, dropping, or throwing an object? To address these issues, we believe that a more elaborate ontology will be required, perhaps including a representation of objects as *agents* with internal states, goals, and intentions.

10.3.2 Designing computational perceivers: The link between knowledge representation and perception

Regularities and structure in the world serve to constrain the forms of knowledge representation schemes we need to consider (Witkin & Tenenbaum, 1983; Bobick & Richards, 1986). In particular, as suggested in (Feldman, 1992; Richards & Jepson, 1995; Richards et al., 1996), natural categories of world structure can be described by placing qualitative probabilities over a discrete set of “modal” processes operating at different spatiotemporal scales. For example,

¹As argued in (Siskind, 1992), to determine causality in event recognition, we will probably require some additional *hypothetical* or *counterfactual* reasoning about the scene. For example, we can infer that “A supports B”, when: 1) B is supported (i.e., not falling), and 2) when A is removed, B is not supported (i.e., B will fall).

in the motion domain, we may define various qualitative categories such as: resting (stably supported by a ground plane), rolling, falling, and projectile motion (Jepson et al., 1996). (See also (Jepson & Feldman, 1996; Rubin & Richards, 1985).) These categories, based on world regularities, provide a restricted set of possible concepts our system needs to consider.

In this thesis, we considered a representation based on a simple set of preference rules. As described in (Richards et al., 1996), several alternate representations are possible, including a representation of qualitative probabilities in the configuration space of objects, or a representation of dependencies in a directed graph structure (i.e., a Bayes net (Pearl, 1988)). The choice of the appropriate representation is an important area for future research.

Furthermore, throughout this work we have assumed that the categories used to describe the scene are given in advance. An important research question concerns how these categories are initially determined. In particular, in order to adapt to new domains, a system should be able to learn which regularities exist in the world. This remains an issue for future investigation (although (Mann & Jepson, 1993) describes a preliminary attempt to address these issues for a simplified motion domain).

10.3.3 Will the computational perceiver approach generalize to new domains?

A crucial unanswered question concerns how the approach presented in this thesis can be extended to deal with complex scenes, changing domain structure, and changing tasks. To deal with complex scenes, the system will probably require some method of *abstraction*. That is, the system should be able to group the scene into larger pieces that can be described at a coarser resolution or “granularity” (Hobbs, 1985). For example, when shown a scene containing many blocks on a table, the system might simply represent the blocks as an amorphous “heap” rather than a collection of individual objects.

In addition, when dealing with unstructured input, one would expect to encounter several different forms of world structure. For example, a scene may contain both polygonal and curved objects, rigid and non-rigid objects, animate and inanimate objects, etc. A major challenge, then, is to know how and when to apply different domain theories. In addition to forming interpretations of the scene, we should be able to tell when the scene does not fit the prior models we have built into our representation. In general, we may view the choice of appropriate domain as a *model comparison* task, where Bayesian methods might be appropriate (MacKay, 1992).

Finally, the level of processing will depend on the task the visual system is expected to perform. For example, if the system is simply required to detect active objects (e.g., for focusing attention in an active vision system), a much simpler form of processing could be used. Conversely, if the visual system is used as input to a more complex system such as a planner, a richer output description will be required.

In summary, the key observation is that the system should be capable of changing the representation and the reasoning performed depending on the sensor input and the task at hand. We believe our current work provides the building blocks for such a system, but additional work will be required to show how this approach can be employed in a more complex environment.

10.4 Conclusion

In this thesis, we presented a proposal for designing computational perceivers. Our approach involves the specification of three general components considered essential for any perceptual system: an ontology, a domain theory, and a preference policy. We demonstrated the utility of the approach by showing that it provides a practical and elegant way to infer dynamic properties of objects in a simple domain. We believe that this framework provides a promising concrete proposal for building perceptual systems.

Appendix A

Object Tracking

As described in Chapter 7, at each frame of the sequence the tracker provides estimates for the position and orientation of each of the participant objects. In this Appendix we describe the procedure used to track objects in the image sequences. In Appendix B we describe how estimates for the object velocities and accelerations are obtained from the tracking data. Both the tracking and motion estimation work described here were implemented by Allan Jepson. However, since the work is necessary for the implementation of the system described in this thesis, a brief summary will be presented here.

A.1 View-based Tracking

To acquire the position and orientation of the object polygons for each frame we use a *view-based* tracking algorithm. An illustration of the tracking process is shown in Figure A.1. A *template* for each object is formed by taking a rectangular subimage from the first frame of the sequence (Figure A.1(a)). The shape of the object in the template image is input by selecting a set of points (in counterclockwise order) on the convex hull of the object, where each point describes the position relative to the upper left corner of the template (Figure A.1(b)). The tracker estimates the pose (position (a_x, a_y) , and orientation a_θ) of the template in each frame. The position of each of the object polygons is then obtained by mapping the original object outlines (in the template) according to the pose estimates of the templates (Figure A.1(c)).

Rather than using a global search for the object template in each frame, the tracker uses an *iterative* algorithm to find the template given an initial guess of the template pose. In the current system, the tracker uses an estimate of the object displacement in the previous two frames to predict the pose of the template in the current frame.

In addition, the process of matching the template to each frame is *robust*. That is, poorly matching pixels called *outliers*, are rejected so they do not affect the estimation. For this reason, the shape of the template is not critical, so long as it contains a significant portion of the object to be tracked. Also, since only a partial match is required, the tracker is able to deal with

Figure A.1: Illustration of the object tracking process. (a) One or more object templates are selected from the first frame of the sequence. (b) The object shape (a convex polygon) is outlined within each template. (c) The tracker computes the pose (i.e., position (a_x, a_y) and orientation a_θ) of the template in each frame. Given the pose of the template, the point p^0 in the template is mapped to the point p in frame t .

partial occlusion of the objects, so long as a portion of the object remains visible.

In the following sections, we will describe how an object template is tracked throughout the image sequence. Before proceeding, it is important to highlight the fact each template is tracked *independently*. That is, for each object, a separate template is chosen, and this template is tracked by an independent process. While one might imagine a way to exploit the joint motion of objects (for example, by noticing occlusion or disocclusion among objects), this information was not used in the current implementation.

A.2 Robust Object Tracking by Mixture Models

To implement a robust object tracker we use a *mixture model* framework. Specifically, the probability of observing an image pixel with a particular intensity is modeled by a mixture (i.e., weighted sum) of probability distributions, based on the various predicted locations of the object template. In addition, to account for noise, there is a background or ‘outlier’ process.

In the following subsection, we first describe the probabilistic model for the image pixels from an individual template. Next we describe the mixture model that allows multiple competing hypotheses for the pose of the object template.

A.2.1 Imaging Model

Each object is described by an *image template* denoted by I^0 .¹ We describe each frame of the sequence by an image I^t , $t \geq 0$ (I^0 corresponds to the image template). In the following presentation we will use I^t to refer to the set of all points $\mathbf{p} = [p_x, p_y]^T$ in the image I^t , while

¹In the tracker described here, the image templates are rectangular subimages taken from the first frame of the image sequence. In general, the exact source and form of the image templates is not critical to the approach.

we will use $I^t(\mathbf{p})$ to refer the pixel intensity at a particular point $\mathbf{p}$ in the image I^t .

To describe the motion of the template images, we use a *warping function* $\mathbf{p} = \mathbf{w}(\mathbf{p}^0; \mathbf{a})$ where $\mathbf{p} \in I^t$ is the predicted position of the point $\mathbf{p}^0 \in I^0$ and $\mathbf{a}$ is some set of parameters describing the warp between I^0 and I^t . In this work, we used a planar translation and rotation model given by

$$\mathbf{p} = \mathbf{w}(\mathbf{p}^0; \mathbf{a}) = \begin{bmatrix} p_x^0 \\ p_y^0 \end{bmatrix} + \begin{bmatrix} a_x + p_x^0 \cos a_\theta - p_y^0 \sin a_\theta \\ a_y + p_x^0 \sin a_\theta + p_y^0 \cos a_\theta \end{bmatrix} \quad (\text{A.1})$$

where $\mathbf{a} = [a_x, a_y, a_\theta]^T$ are the parameters of the warp with translation $[a_x, a_y]^T$ and rotation a_θ .

We describe the probability of observing a pixel with intensity $I^t(\mathbf{p})$ in frame t by a Gaussian noise model

$$p(I^t(\mathbf{p})|\mathbf{a}, I^0(\mathbf{p}^0)) = \frac{1}{\sqrt{2\pi}\sigma} \exp -\frac{1}{2\sigma^2} \left(I^t(\mathbf{p}) - I^0(\mathbf{p}^0) \right)^2 \quad (\text{A.2})$$

where $\mathbf{p}$ is the predicted location of pixel $\mathbf{p}^0$ in the original template image I^0 and σ is the standard deviation for the noise model.

A.2.2 Mixture Model

We consider a mixture model, where there can be one or more templates described by warp parameters $\mathbf{a}_1, \dots, \mathbf{a}_N$ plus an ‘outlier’ process. (For the experiments described in Chapter 8, the mixture model contained two patches, plus an outlier process.) The probability of each pixel is

$$p(I^t(\mathbf{p})|\mathbf{m}, \mathbf{a}_1, \dots, \mathbf{a}_N, I^0(\mathbf{p}^0)) = \sum_{n=1}^N m_n p(I^t(\mathbf{p})|\mathbf{a}_n, I^0(\mathbf{p}^0)) + m_o p_o(I^t(\mathbf{p})) \quad (\text{A.3})$$

where $\sum_{n=1}^N m_n = 1$ and $p_o(I^t(\mathbf{p})) = p_0$, with p_0 set to some constant outlier probability.² Finally, assuming independent pixel error, the log likelihood for the entire mixture model is given by

$$\log p(I^t|\mathbf{m}, \mathbf{a}_1, \dots, \mathbf{a}_N, I^0) = \sum_{\mathbf{p}^0} \log p(I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a}))|\mathbf{m}, \mathbf{a}_1, \dots, \mathbf{a}_N, I^0(\mathbf{p}^0)) \quad (\text{A.4})$$

where the sum is over the set of pixels $\mathbf{p}^0$ in the template image I^0 . Given the template image I^0 we attempt to find the set of parameters $\mathbf{m}, \mathbf{a}_1, \dots, \mathbf{a}_N$ that maximizes the log likelihood of observing the pixels in frame t .

²We chose $p_0 = \frac{1}{L}$ where L is the range for the values pixel intensities $I^t(\mathbf{p})$.

A.2.3 Implementation

As described in (McLachlan & Basford, 1988), it can be shown that at the local extrema of A.4, the parameters must satisfy

$$\sum_{\mathbf{p}^0} q_{np} = \lambda m_n \quad (\text{A.5})$$

$$\sum_{\mathbf{p}^0} q_{np} \frac{\partial}{\partial \mathbf{a}_n} \log p(I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a}_n)) | \mathbf{a}_n) = 0 \quad (\text{A.6})$$

where

$$q_{np} = \frac{m_n p(I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a}_n)) | \mathbf{a}_n)}{\sum_j m_j p(I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a}_n)) | \mathbf{a}_j)} \quad (\text{A.7})$$

are the ownership probabilities, that is, the probability that pixel $I^t(\mathbf{p})$ was generated by mixture distribution n .

As described in (McLachlan & Basford, 1988), we can use the EM algorithm to find the maximum likelihood estimate for the set of parameters. In particular, given an initial set of parameters, we use Equation A.7 to estimate the ownerships (the estimation step), followed by Equations A.5 and A.6 to maximize the parameters given the ownerships are fixed (the maximization step). Successive application of these steps is guaranteed not to decrease the likelihood at each step, and typically converges to a local maximum.

To solve Equation A.5 we observe that since $\sum_{n=0}^N m_n = 1$ we set

$$m_n = \frac{1}{|I^0|} \sum_{\mathbf{p}^0} q_{np} \quad (\text{A.8})$$

for $0 \leq n \leq N$ where $|I^0|$ is the number of pixels in the template.

While we could attempt to solve A.6 directly to find the $\mathbf{a}_n$, this is a nonlinear problem. Instead, we linearize $\log p(I^t(\mathbf{p}) | \mathbf{a})$ and perform a (closed form) solution of the linear system to obtain the $\mathbf{a}_n$ at each step.

The following approach is similar to that presented in (Black & Jepson, 1997). Let the position of a point $\mathbf{p}^0 \in I^0$ in frame t be

$$\mathbf{p} = \mathbf{w}(\mathbf{p}^0; \mathbf{a} + \mathbf{b}) \quad (\text{A.9})$$

where $\mathbf{a}$ is some initial guess of the warp and $\mathbf{b}$ is a correction for the transformation.³ The initial guess $\mathbf{a}$ comes from a (linear) predictive model based on the difference in the previous two frames of the sequence. Performing a Taylor expansion about $\mathbf{w}(\mathbf{p}^0; \mathbf{a})$ we get

$$I^t(\mathbf{p}) = I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a} + \mathbf{b})) = I^t + \nabla I^t \frac{\partial \mathbf{w}(\mathbf{p}^0; \mathbf{a})}{\partial \mathbf{a}} \mathbf{b} + O(\|\mathbf{b}\|^2) \quad (\text{A.10})$$

³In the following we will omit the subscript corresponding to the model number; The evaluations will be performed separately for each model n .

where I^t (the image intensity) and $\nabla I^t = [I_x^t, I_y^t]$ (the image gradient) are evaluated at the point $\mathbf{p} = \mathbf{w}(\mathbf{p}^0; \mathbf{a})$ in I^t and $\mathbf{b} = [b_x, b_y, b_\theta]^T$. Taking the derivative of A.10 wrt $\mathbf{b}$, we get

$$\frac{\partial I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a} + \mathbf{b}))}{\partial \mathbf{b}} \approx [I_x^t, I_y^t, I_\theta^t] \quad (\text{A.11})$$

where I_x^t and I_y^t are the image gradients from above and $I_\theta^t = I_x^t(-p_x \sin a_\theta - p_y \cos a_\theta) + I_y^t(p_x \cos a_\theta - p_y \sin a_\theta)$ and the $\mathbf{p} = [p_x, p_y]^T$ is given by the warp $\mathbf{w}(\mathbf{p}^0; \mathbf{a})$.

Taking the derivative of the probabilistic model (equation A.2) we obtain

$$\frac{\partial}{\partial \mathbf{b}} \log p(I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a} + \mathbf{b}))) = -\frac{1}{\sigma^2} (I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a} + \mathbf{b})) - I^0(\mathbf{p}^0)) \frac{\partial I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a} + \mathbf{b}))}{\partial \mathbf{b}} \quad (\text{A.12})$$

Using A.11 in A.12 we can solve a linear system to find the $\mathbf{b}$ that solves A.6. Specifically, we get a 3-by-3 system of equations for each set of parameters $\mathbf{b}$

$$\begin{bmatrix} \sum_p q_{np} (I_x^t)^2 & \sum_p q_{np} I_x^t I_y^t & \sum_p q_{np} I_x^t I_\theta^t \\ \sum_p q_{np} I_y^t I_x^t & \sum_p q_{np} (I_y^t)^2 & \sum_p q_{np} I_y^t I_\theta^t \\ \sum_p q_{np} I_\theta^t I_x^t & \sum_p q_{np} I_\theta^t I_y^t & \sum_p q_{np} (I_\theta^t)^2 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_\theta \end{bmatrix} = \begin{bmatrix} \sum_p q_{np} I_x^t (I^0(\mathbf{p}^0) - I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a}))) \\ \sum_p q_{np} I_y^t (I^0(\mathbf{p}^0) - I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a}))) \\ \sum_p q_{np} I_\theta^t (I^0(\mathbf{p}^0) - I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a}))) \end{bmatrix} \quad (\text{A.13})$$

where the sum is over all pixels $\mathbf{p}^0$ in the template I^0 .

Once the algorithm finds an optimal transformation adjustment, $\mathbf{b}$, this value is added to the existing value of $\mathbf{a}$ and another EM step is performed. Eventually the parameters $\mathbf{a}$ will converge (and the adjustments $\mathbf{b}$ will become zero).

Additional Details of the Implementation

To make the above algorithm work effectively and efficiently, a number of additional enhancements were used.

To reduce computation, rather than warping and resampling each frame t to compute $I^t(\mathbf{w}(\mathbf{p}^0; \mathbf{a}))$ at each step, the templates I^0 were warped and resampled instead. That is, instead of adding the correction $\mathbf{b}$ to the warp $\mathbf{a}$ at each step, the original template images I^0 were replaced by warped versions, where $\hat{I}^0(\mathbf{p}) = I^0(\mathbf{w}(\mathbf{p}, \mathbf{a} + \mathbf{b}))$.

Instead of using image intensities, I , local phase information was used. While the experiments in this thesis could probably have been performed using image intensities alone, as described in (Fleet, 1992), phase information is much more robust to image variations, such as changes in illumination.

In order to avoid local minima in the optimization, several enhancements were made to the optimization procedure.

First, to get good estimates for the parameters, a multiscale (coarse-to-fine) optimization strategy was used. Specifically, for each frame of the sequence, the image was first filtered using a bandpass filter tuned to a large wavelength. Several steps of the optimization were

performed, then once a stable estimate was obtained, this was used as a starting point for another optimization using a higher frequency bandpass filter was used, and so on. For the experiments performed here, a three level pyramid structure was used. (See (Black & Jepson, 1997) for details of the multiscale optimization strategy.)

Second, during the optimization, the value of the noise term, σ was slowly decreased, starting at a large value, and ending at a small value, consisting of the estimated value of the pixel noise in the image.

Finally, to avoid local minima due to vanishing mixture components, the mixing proportions in Equation B.7 were constrained to be above some small value, m_{min} . Specifically, at each iteration we compute $c = \min_i(m_{min} - m_i)$, where m_i is the estimate for mixing proportion i . If $c > 0$ then all mixing proportions are adjusted as follows:

$$m_i^{(new)} = m_i(1 - c) + c \frac{1}{M} \quad (\text{A.14})$$

for $i = 0, \dots, M$ where M is the number of mixing proportions. (Note that after A.14, the mixing proportions will still sum to one.)

A.3 Summary

In summary, the method described allows the *robust* tracking of objects. As shown by the results in Chapter 7, this approach is suitable, provided a portion of each object is visible in each frame of the sequence.

Appendix B

Motion Estimation

Given the pose (i.e., the position (x, y) and orientation θ) of the objects in each frame, we estimate the motion by a robust interpolation procedure. Specifically, at each point in time, we robustly fit a cubic polynomial to each of the $x(t)$, $y(t)$, and $\theta(t)$ components over a sliding temporal window. We then differentiate the polynomials to obtain estimates for the velocity, acceleration, angular velocity, and angular accelerations for each object. In the current implementation, the polynomial fitting is performed within a sliding window centered at each time point in the sequence (i.e., there is no attempt fit polynomials to the entire sequence or to enforce any constraints between the fits at adjacent windows).

Often, a single polynomial will not fit the pose data. For example, if there is a step change in velocity or acceleration (e.g., due to a collision or a change in contact between objects), a single polynomial will provide inaccurate estimates near the discontinuity. To avoid this problem, our system performs a *robust* interpolation that allows two or more polynomials to fit the data within each temporal window. This is achieved by modeling the data within each temporal window by a *mixture model* consisting of one or more polynomials, plus an outlier process.

B.1 Robust Interpolation by Mixture Models

The data within a temporal window is modeled by one or more cubic polynomials of the form

$$\hat{y}_n(t; \mathbf{a}_n) = a_{n0} + a_{n1}(t - t_0) + a_{n2}(t - t_0)^2 + a_{n3}(t - t_0)^3 \quad (\text{B.1})$$

where $t = 1, \dots, T$, $t_0 = \frac{T}{2} + 1$ is the center of the temporal window and $\mathbf{a}_n = [a_{n0}, a_{n1}, a_{n2}, a_{n3}]^T$ is the set of parameters for the n th polynomial.

The probability of generating a measurement $y(t)$ is taken to be a normal distribution about the predicted point $\hat{y}(t)$

$$p(y(t)|\mathbf{a}_n) = \frac{1}{\sqrt{2\pi}\sigma} \exp -\frac{1}{2\sigma^2} (\hat{y}_n(t; \mathbf{a}_n) - y(t))^2 \quad (\text{B.2})$$

We model the probability of each point $y(t)$ as a mixture of one or more polynomials, plus an ‘outlier’ process

$$p(y(t)) = \sum_{n=1}^N m_n p(y(t)|\mathbf{a}_n) + m_0 p_0 \quad (\text{B.3})$$

where $\sum_{n=0}^N m_n = 1$ are the mixing proportions and p_0 is the probability for the outlier process.¹ Treating each observation $y(t)$ independently, the log likelihood of the data is $\mathbf{y} = (y(1), \dots, y(T))$ is

$$\log p(\mathbf{y}) = \sum_{t=1}^T \log p(y(t)|\mathbf{m}, \mathbf{a}_1, \dots, \mathbf{a}_N) \quad (\text{B.4})$$

Finally, we have

$$q_{nt} = \frac{m_n p(y(t)|\mathbf{a}_n)}{\sum_j m_j p(y(t)|\mathbf{a}_j)} \quad (\text{B.5})$$

which are the ownership probabilities, that is, the probability that observation $y(t)$ was generated by mixture distribution n .

As described in Appendix A, we can solve these equations using the EM algorithm by noting that the derivative $\frac{\partial}{\partial \mathbf{a}_n} \log p(y(t)|\mathbf{a}_n)$ is linear in $\mathbf{a}_n$. Specifically, perform the maximization step for each for each set of coefficients $\mathbf{a}_n$ by solving the following set of linear equations

$$\begin{bmatrix} \sum_t q_{nt} & \sum_t q_{nt}(t-t_0) & \sum_t q_{nt}(t-t_0)^2 & \sum_t q_{nt}(t-t_0)^3 \\ \sum_t q_{nt}(t-t_0) & \sum_t q_{nt}(t-t_0)^2 & \sum_t q_{nt}(t-t_0)^3 & \sum_t q_{nt}(t-t_0)^4 \\ \sum_t q_{nt}(t-t_0)^2 & \sum_t q_{nt}(t-t_0)^3 & \sum_t q_{nt}(t-t_0)^4 & \sum_t q_{nt}(t-t_0)^5 \\ \sum_t q_{nt}(t-t_0)^3 & \sum_t q_{nt}(t-t_0)^4 & \sum_t q_{nt}(t-t_0)^5 & \sum_t q_{nt}(t-t_0)^6 \end{bmatrix} \begin{bmatrix} a_{n0} \\ a_{n1} \\ a_{n2} \\ a_{n3} \end{bmatrix} = \begin{bmatrix} \sum_t q_{nt} y(t) \\ \sum_t q_{nt}(t-t_0) y(t) \\ \sum_t q_{nt}(t-t_0)^2 y(t) \\ \sum_t q_{nt}(t-t_0)^3 y(t) \end{bmatrix} \quad (\text{B.6})$$

Finally, the mixing proportions m_n are reestimated

$$m_n = \frac{1}{T} \sum_{t=1}^T q_{nt} \quad (\text{B.7})$$

for $n = 0, \dots, N$.

B.1.1 Additional Details

In the model described above we took the standard deviation of the noise, σ , to be a constant, equal for all mixture components. In the current implementation, we allow a separate noise term, σ_n , for each polynomial, and learn these terms from the data. In particular, we can estimate the noise terms at each M-step as follows:

$$\sigma_n^2 = \frac{\sum q_{nt} (y(t) - \hat{y}_n(t; \mathbf{a}_n))^2}{\sum q_{nt}} \quad (\text{B.8})$$

¹As with the tracking problem in Appendix A, we set p_0 to be some constant $\frac{1}{L}$, where L represents the range of possible data values. For the experiments here we set $L = 320$, the width of the image window.

We observed that the estimation of noise terms tends to increase the chance of numerical instability and getting stuck in local minima. To combat these problems, we adopted two simple heuristics. First, in order to prevent numerical instabilities in the learning algorithm, we placed a lower bound on the σ at each iteration. (A value of $\sigma \geq \sigma_{min} = 0.2$ was used in the experiments reported here.) Second, in order to prevent local minima due to overly large values of σ , we placed an upper bound on σ . Specifically, we set an upper bound, $s^{(i)}$, on the σ at each iteration i . The upper bound begins at an initial value s_{max} and decays (by multiplication with by a factor ρ) until some minimum s_{min} is reached. (In the experiments described here, we set $s_{max} = 100$, $\rho = 0.95$, and $s_{min} = 0.5$.)

B.1.2 Examples

Figures B.1 and B.2 show some examples of the polynomial fitting process. In our implementation, we fit two polynomials plus an outlier process within a window that is nine samples wide (i.e., there are four samples on either side of the center frame).

Figure B.1 shows the fit to horizontal position ($x(t)$) of the right car in the cars sequence, centered around frame 28. (This is at the frame where the collision occurs.) Figure B.1(a) shows the result for fitting two cubic polynomials (plus an outlier process), while Figure B.1(b) compares this to the fit obtained with a single cubic polynomial. Note that while fitting multiple polynomials allows a better fit of the data within the window, there is still uncertainty at some time points. In the mixture model this uncertainty corresponds to the decision of which polynomial to use for the interpolation (i.e., for the estimation of velocity and acceleration). Note also, that because there is no temporal constraint on the samples that each polynomial fits, there may be overlap (or gaps) in the samples fit by each polynomial. For example, in Figure B.1(a) the data at frame 25 is fit by both the ‘left’ and the ‘right’ polynomial.

Figure B.2 shows another example, fitting the horizontal position ($x(t)$) of the box in the hit sequence, centered around frame 32. (This is at the frame where the hand hits the box.) Note that the result shown in Figure B.2(a) describes the discontinuity almost exactly, while another run of the same algorithm shown in Figure B.2(b) fits the data very poorly. (It has also rejected a significant portion of the data as outliers.) It appears that the algorithm sometimes gets stuck in *local minima* that may not describe the data correctly.

While this interpolation process has been sufficient for our purposes here, clearly more work needs to be done to detect and classify motion discontinuities. These problems can probably be solved by adding some form of temporal constraints to the polynomial fitting process, however, we leave this for future research.

Figure B.1: Example of the polynomial fitting process. Fitting the horizontal position ($x(t)$) of the right car in the cars sequence. The window is centered around frame 28, near the collision of the two cars. (a) Two cubic polynomials (plus an outlier process) fit to the data. (b) Fit of a single cubic polynomial to the same data as in (a). The position data, $x(t)$, is depicted by the circles. The vertical bars on each polynomial depict the *ownership probabilities* at each point (longer bars indicate higher ownership probability).

Figure B.2: Another example of the polynomial fitting process. Fitting the horizontal position ($x(t)$) of the box in the hit sequence. The window is centered around frame 31, near the collision between the hand and the box. (a) Two cubic polynomials (plus an outlier process) fit to the x -component of the hit sequence. (b) Another run of the same algorithm that resulted in a local minima. Note the presence of *outliers*, that are not fit by either polynomial. (The ownership probabilities for the outliers are depicted by the vertical lines at the bottom of the plot, below each point.)

B.2 Motion Estimation

Given a set of polynomials, we estimate the motion by choosing the *best-fit* polynomial at the center of the window. That is, we choose the polynomial with the highest ownership probability, q_{nt} at the center of the window. To estimate the motion, we take the first and second derivatives of each polynomial (i.e., by differentiating Equation B.1). Recall that, as described above, to estimate the motion, we differentiate each component $(x(t), y(t), \theta(t))$ separately.

References

- Adelson, E. H., & Pentland, A. P. (1996). The perception of shading and reflectance. In D. Knill & W. Richards (Eds.), *Perception as bayesian inference* (pp. 409–423). Cambridge University Press.
- André, E., Herzog, G., & Rist, T. (1988). On the simultaneous interpretation of real world sequences and their natural language description: The system soccer. In *Proceedings of the eighth european conference on artificial intelligence* (pp. 449–454).
- Arnheim, R. A. (1988). Visual dynamics. *American Scientist*, 76, 585–591.
- Badler, N. I. (1975). *Temporal scene analysis: Conceptual descriptions of object movements* (Tech. Rep. No. 80). University of Toronto Department of Computer Science.
- Baraff, D. (1995). Interactive simulation of solid rigid bodies. *IEEE Computer Graphics and Applications*, 15(3), 63–75.
- Black, M. J., & Jepson, A. D. (1997). Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal for Computer Vision*. (To appear)
- Blum, M., Griffith, A. K., & Neumann, B. (1970). *A stability test for configurations of blocks* (A. I. Memo No. 188). MIT Artificial Intelligence Laboratory.
- Bobick, A., & Richards, W. (1986). *Classifying objects from visual information* (Tech. Rep. No. AI Memo 879). Massachusetts Institute of Technology.
- Bogoni, L., & Bajcsy, R. (1995). Interactive recognition and representation of functionality. *Computer Vision and Image Understanding*, 62(2), 194–214.
- Borchardt, G. C. (1985). Event calculus. In *Proceedings of the ninth international joint conference on artificial intelligence* (pp. 524–527). Los Angeles, CA.
- Borchardt, G. C. (1994). *Thinking between the lines: computers and the comprehension of causal descriptions*. Cambridge, Mass.: MIT Press.

- Brand, M., Birnbaum, L., & Cooper, P. (1993). Sensible scenes: Visual understanding of complex scenes through causal analysis. In *Proceedings of the eleventh national conference on artificial intelligence* (pp. 588–593). Washington, DC.
- Brand, M., & Oliver, N. (1996). Coupled hidden markov models for complex action recognition. In *Computer vision and pattern recognition (CVPR 96)*.
- Cremer, J. F. (1989). *An architecture for general purpose physical system stimulation—integrating geometry, dynamics, and control*. Unpublished doctoral dissertation, Department of Computer Science, Cornell University, Ithaca, NY 14853-7501.
- Cutting, J. E., Proffitt, D. R., & Kozlowski, L. T. (1978). A biomechanical invariant for gait perception. *Journal of Experimental Psychology: Human Perception and Performance*, 4(3), 357–372.
- Davis, R., Shrobe, H. E., & Szlovits, P. (1993). What is a knowledge representation? *AI Magazine, Spring 1993*, 14(1), 17–33.
- Fahlman, S. E. (1974). A planning system for robot construction tasks. *Artificial Intelligence*, 5(1), 1–49.
- Featherstone, R. (1987). *Robot dynamics algorithms*. Boston: Kluwer.
- Feldman, J. (1992). *Perceptual categories and world regularities*. Unpublished doctoral dissertation, Department of Brain and Cognitive Science. (Also TR-6, Rutgers Center for Cognitive Science)
- Fleet, D. (1992). *Measurement of image velocity*. Kluwer Academic Publishers.
- Forbus, K. D. (1987). Interpreting observations of physical systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 13, 350–359.
- Freyd, J. J. (1987). Dynamic mental representations. *Psychological Review*, 94(4), 427–438.
- Freyd, J. J., Pantzer, T. M., & Cheng, J. L. (1988). Representing statics as forces in equilibrium. *Journal of Experimental Psychology: General*, 117(4), 395–407.
- Funt, B. V. (1980). Problem-solving with diagrammatic representations. *Artificial Intelligence*, 13(3), 201–230.
- Genesereth, M. R., & Nilsson, N. J. (1987). *Logical foundations of artificial intelligence*. Los Altos: Morgan Kaufmann.
- Gilden, D. L., & Proffitt, D. R. (1989). Understanding collision dynamics. *Journal of Experimental Psychology: Human Perception and Performance*, 15(2), 372–383.

- Goldstein, H. (1980). *Classical mechanics* (second ed.). Addison-Wesley.
- Gregory, R. L. (1970). *The intelligent eye*. New Jersey: McGraw Hill.
- Heider, F., & Simmel, M. (1944). An experimental study of apparent behaviour. *American Journal of Psychology*, 57, 243–259.
- Hobbs, J. R. (1985). Granularity. In A. Joshi (Ed.), *Proceedings of the ninth international joint conference on artificial intelligence* (pp. 432–435). San Mateo, CA: Morgan Kaufmann.
- Hochberg, J., & McAlister, E. (1953). A quantitative approach to figural “goodness”. *Journal of Experimental Psychology*, 46, 361–364.
- Ikeuchi, K., & Suehiro, T. (1994). Towards an assembly plan from observation, part i: Task recognition with polyhedral objects. *IEEE Transactions on Robotics and Automation*, 10(3), 368–385.
- Jepson, A., & Richards, W. (1992). A lattice framework for integrating visual modules. *IEEE Transactions on Systems, Man & Cybernetics*, 22, 1087–1096.
- Jepson, A. D. (1996). Commentary on “the perception of shading and reflectance” by adelson and pentland. In D. Knill & W. Richards (Eds.), *Perception as bayesian inference* (pp. 488–495). Cambridge University Press.
- Jepson, A. D., & Feldman, J. (1996). A biased view of perceivers. In D. Knill & W. Richards (Eds.), *Perception as bayesian inference* (pp. 229–235). Cambridge University Press.
- Jepson, A. D., & Richards, W. (1993). *What is a percept?* (Tech. Rep. No. RBCV-TR-93-43). Department of Computer Science, University of Toronto. (Also MIT Cognitive Science Memo 43, 1991)
- Jepson, A. D., Richards, W., & Knill, D. (1996). Modal structure and reliable inference. In D. Knill & W. Richards (Eds.), *Perception as bayesian inference* (pp. 63–92). Cambridge University Press.
- Johansson, G. (1975). Visual motion perception. *Scientific American*, 232(6), 76–89.
- Joskowicz, L., & Sacks, E. P. (1991). Computational kinematics. *Artificial Intelligence*, 51(1–3), 381–416.
- Kang, S. B., & Ikeuchi, K. (1993). Toward automatic robot instruction from perception–recognizing a grasp from observation. *IEEE Transactions on Robotics and Automation*, 9(4), 432–443.

- Kass, M., Witkin, A., & Terzopoulos, D. (1987). Snakes: Active contour models. In *First international conference on computer vision, (london, england, june 8-11, 1987)* (pp. 259-268). Washington, DC.: IEEE Computer Society Press.
- Knill, D., & Richards, W. (1996). *Perception as bayesian inference*. Cambridge University Press.
- Kramer, G. (1990). Solving geometric constraint systems. In *Proceedings of the eighth national conference on artificial intelligence* (pp. 708-714). Boston, MA.
- Kuniyoshi, Y., & Inoue, H. (1993). Qualitative recognition of ongoing human action sequences. In *Proceedings of the thirteenth international joint conference on artificial intelligence* (pp. 1600-1609). Chambéry, France.
- Leeuwenberg, E. (1969). Quantitative specification of information in sequential patterns. *Psychological Review*, 76, 216-220.
- Leeuwenberg, E., & Boselie, F. (1988). Against the likelihood principle in visual form perception. *Psychological Review*, 95(4), 485-491.
- Leslie, A. M. (1993). *A theory of agency* (Tech. Rep. No. RuCCS TR-12). Center for Cognitive Science, Rutgers University.
- Leslie, A. M., & Keeble, S. (1987). Do six-month-old infants perceive causality? *Cognition*, 25, 265-288.
- Lifschitz, V. (1985). Computing circumscription. In *Proceedings of the ninth international joint conference on artificial intelligence* (pp. 121-127). Los Angeles, CA.
- Luenberger, D. G. (1984). *Linear and nonlinear programming*. Addison Wesley.
- MacKay, D. J. C. (1992). Bayesian interpolation. *Neural Computation*, 4, 415-447.
- Mann, R., & Jepson, A. (1993). Non-accidental features in learning. In *AAAI fall symposium on machine learning in computer vision*. Raleigh, NC.
- Mann, R., Jepson, A., & Siskind, J. M. (1997). The computational perception of scene dynamics. *Computer Vision and Image Understanding*, 65(2).
- McCarthy, J. (1986). Applications of circumscription to formalizing common sense reasoning. *Artificial Intelligence*, 28, 89-116.
- McCloskey, M. (1983). Naive theories of motion. In D. Gentner & A. L. Stevens (Eds.), *Mental models* (pp. 299-324). Hillsdale, NJ: Lawrence Erlbaum Associates.

- McLachlan, G. J., & Basford, K. E. (1988). *Mixture models: Inference and applications to clustering*. New York: Marcel Dekker.
- Metaxas, D., & Terzopoulos, D. (1993). Shape and nonrigid motion estimation through physics-based synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6), 580-591.
- Michotte, A. (1963). *The perception of causality*. Andover: Methuen.
- Miller, G. A. (1972). English verbs of motion: A case study in semantics and lexical memory. In A. W. Melton & E. Martin (Eds.), *Coding processes in human memory* (pp. 335-372). Washington, DC: V. H. Winston and Sons, Inc.
- Miller, G. A., & Johnson-Laird, P. N. (1976). *Language and perception*. Cambridge MA: Harvard University Press.
- Mulgaonkar, Cowan, & DeCurtins. (1992). Understanding object configurations using range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2), 303-307.
- Nagel, H.-H. (1988). From image sequences to conceptual descriptions. *Image and Vision Computing*, 6(2), 59-79.
- Neumann, B. (1989). Natural language description of time-varying scenes. In D. L. Waltz (Ed.), *Semantic structures* (pp. 167-207). Lawrence Erlbaum.
- Neumann, B., & Novak, H. (1983). Event models for recognition and natural language description of events in real-world image sequences. In *Proceedings of the eighth international joint conference on artificial intelligence* (pp. 724-726). Karlsruhe.
- Pearl, J. (1988). *Probabilistic reasoning in intelligent systems*. Morgan Kaufmann Publishers, Inc.
- Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1992). *Numerical recipes in C: The art of scientific computing* (2 ed.). Cambridge: Cambridge University Press.
- Proffitt, D. R., Kaiser, M. K., & Whelan, S. M. (1990). Understanding wheel dynamics. *Cognitive Psychology*, 22, 342-373.
- Reiter, R. (1978). On closed world data bases. In H. Gallaire & J. Minker (Eds.), *Logic and data bases*. New York: Plenum Press.
- Reiter, R. (1987). A theory of diagnosis from first principles. *Artificial Intelligence*, 32, 57-95.
- Reiter, R., & Mackworth, A. (1989). A logical framework for depiction and image interpretation. *Artificial Intelligence*, 41, 125-155.

- Richards, W., & Jepson, A. (1995). What makes a good feature? In L. Harris & M. Jenkin (Eds.), *Spatial vision in humans and robots* (pp. 89–126). Cambridge University Press. (Also MIT AI Memo 1356 (1992))
- Richards, W., Jepson, A. D., & Feldman, J. (1996). Priors, preferences and categorical percepts. In D. Knill & W. Richards (Eds.), *Perception as bayesian inference* (pp. 93–122). Cambridge University Press.
- Rock, I. (1983). *The logic of perception*. Cambridge MA: MIT Press.
- Rubin, J. M., & Richards, W. A. (1985). *Boundaries of visual motion* (Tech. Rep. No. AIM-835). Massachusetts Institute of Technology, Artificial Intelligence Laboratory.
- Runeson, S., & Frykholm, G. (1981). Visual perception of a lifted weight. *Journal of Experimental Psychology: Human Perception and Performance*, 7, 733–740.
- Schank, R. C. (1973). *The fourteen primitive actions and their inferences* (Memo No. AIM-183). Stanford Artificial Intelligence Laboratory.
- Shavit, E., & Jepson, A. D. (1993). Qualitative motion from visual dynamics. In *IEEE Workshop on Qualitative Vision*. New York.
- Siskind, J. M. (1992). *Naive physics, event perception, lexical semantics, and language acquisition*. Unpublished doctoral dissertation, Massachusetts Institute of Technology, Cambridge, MA.
- Siskind, J. M. (1994). Axiomatic support for event perception. In P. McKevitt (Ed.), *Proceedings of the AAAI-94 workshop on the integration of natural language and vision processing* (pp. 153–160). Seattle, WA.
- Siskind, J. M., & Morris, Q. (1996). A maximum-likelihood approach to visual event classification. In *Computer vision—ECCV 96* (pp. II:347–360).
- Spelke, E. S., & Valde, G. V. de. (1993). Perceiving and reasoning about objects: Insights from infants. In N. Eilan, R. McCarthy, & W. Brewer (Eds.), *Spatial representation*. New York, NY: Oxford: Basil Blackwell.
- Stark, L., & Bowyer, K. (1992). Indexing function-based categories for generic object recognition. In *Computer vision and pattern recognition (CVPR 92)* (pp. 795–797). Champaign, IL.
- Talmy, L. (1988). Force dynamics in language and cognition. *Cognitive Science*, 12(1), 49–100.
- Todd, J., & Warren, W. (1982). Visual perception of relative mass in dynamic events. *Perception*, 11, 325–335.

- Tsotsos, J. K., Mylopoulos, J., Covvey, H. D., & Zucker, S. W. (1980). A framework for visual motion understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(6), 563-573.
- Williams, L. R., & Hanson, A. R. (1996). Perceptual completion of occluded surfaces. *Computer Vision and Image Understanding*, 64(1), 1-20.
- Witkin, A., & Tenenbaum, J. (1983). On the role of structure in vision. In J. Beck, B. Hope, & A. Rosenfeld (Eds.), *Human and machine vision*. New York: Academic Press.

IMAGE EVALUATION TEST TARGET (QA-3)

APPLIED IMAGE, Inc.
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989

© 1993, Applied Image, Inc., All Rights Reserved