# 1 Visual Event Perception

*Jeffrey Mark Siskind*
University of Toronto

## Abstract

*Humans can describe visually-observed events such as throwings and droppings. This capacity links visual perception with linguistic competence. The procedures for detecting visually-observed events can be thought of as semantic descriptions of the verbs used to describe those events. The aggregate events described by many simple spatial-motion verbs are best formulated as combinations of simpler events, namely the changing support, contact, and attachment relations between participant objects. I present a method for determining support relations by reducing a set of kinematic constraints to a linear-programming problem. I then present a method for determining the truth of aggregate event descriptions from the truth of lower-level primitive event descriptions.*

## 1. Introduction

The ability to describe visually observed events sits at the center of human cognition. It is the meeting point of perception, language, knowledge representation, and spatio-temporal reasoning. While there has been some prior work on visual event perception within the machine vision community (e.g. [6, 3, 24, 41, 44, 27, 42, 45, 50, 28, 30, 43, 46, 1, 21, 49, 4, 2, 22, 25, 32, 40, 26]) this work has by-and-large not addressed the problem of classifying occurrences of visually-observed events into classes that correspond to ordinary simple spatial-motion verbs. Similarly, while there has been substantial prior work on describing the lexical semantics of such verbs within the linguistics community, very little work has been done on precisely characterizing the truth conditions of verbs such as *pick up*, *put down*, *drop*, or *throw* and formulating such truth conditions in a way that can be grounded in perceptual input. This chapter attempts to bridge this gap by presenting a logical calculus for representing the truth conditions for events described by simple spatial-motion verbs along with the necessary perceptual mechanisms for detecting occurrences of such verbs in simulated visual input.

Numerous researchers (e.g. [20, 23, 33, 14, 15, 29]) have long realized the role played by the causal, aspectual, and directional qualities of motion in specifying the meanings of simple spatial-motion verbs. For example, part of what it means to *throw* something to someone is to cause an object to begin to move towards that person. Similarly, part of what it means to *pick* something *up* is to cause an object to begin upward motion. Some researchers (e.g. [13, 16]) have also realized the role played by notions such as support, contact, and attachment in specifying the meanings of spatial prepositions. For example, in some situations, part of what it means for one object to be *on* another object is for the former to be in contact with, and supported by, the latter. In other situations, something can be on something else by way of attachment, as in *the knob on the door*.

It is rarely noticed, however, that the notions of support, contact, and attachment also play a central role in specifying the truth conditions of many spatial-motion verbs. I have argued extensively elsewhere [35] that this is the case. For example, causing an object to move to someone is insufficient evidence for a throwing event. That definition admits rolling and sliding events as well. Part of what it means to *throw* something is for an object to be in unsupported motion after it leaves the thrower's hand. Similarly, not all causation of upward motion constitutes picking something up. One can cause a ball to move upward by kicking it without picking it up. Part of what it means for an agent to *pick* something *up* is to change its source of support. That object must have previously been supported by something other than the agent's hand. Now that object must be supported by virtue of being in contact with, and attached to, the agent's hand.

For several years I have been building a system called ABIGAIL[35, 36, 37, 38] that uses these notions of support, contact, and attachment as the basis for grounding language in perception. This system is similar in intention to work done by Badler[6], Okada[27], Borchardt[8, 9], Hays[12], Regier[31], and others. ABIGAIL watches movies constructed out of animated line drawings and produces semantic descriptions of the events that occur in those movies. Figure 1 illustrates the key frames of one such movie presented as input to ABIGAIL. This movie depicts a man picking up a ball from a table, bouncing it on the floor, catching it, and placing it back on the table. Given just the positions, orientations, and sizes of the line segments and circles constituting each frame of this movie as input, ABIGAIL can detect the lifting, throwing, dropping, falling, bouncing, and putting down events that occur.

Earlier publications have described much of the visual event perception mechanism used by ABIGAIL. The reader is referred to [35, 38] for comprehensive overviews of ABIGAIL. This chapter focuses instead on the details of two components of ABIGAIL that have not yet been described in depth: a method for determining the stability of a configuration of objects using a reduction to linear programming and a method for detecting the occurrence of aggregate events given occurrences of primitive events that comprise those aggregate events. Section 2 first gives a brief overview of ABIGAIL. Section 3 then discusses stability determination. Section 4 discusses the detection of aggregate events. Finally, section 5 concludes with some remarks about on-going work.

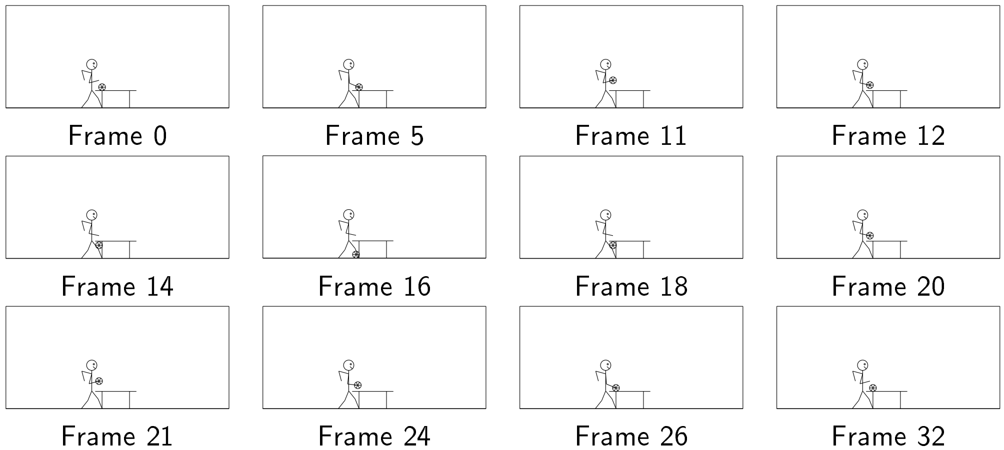


Figure 1: *Several key frames from a typical movie presented as input to* ABIGAIL.

## 2. THE OPERATION OF ABIGAIL

ABIGAIL processes the input movie frame by frame. For each frame, ABIGAIL receives as input only the positions, orientations, and sizes of the line segments and circles that constitute that frame. These line segments and circles are taken to be the atomic entities out of which aggregate objects are constructed. ABIGAIL is not told that certain collections of line segments and circles are to be interpreted as tables, balls, and people. She is not even told how to group the collection of line segments and circles in a given frame into distinct objects. She uses a novel *functional* approach to perform such grouping.

Part of the underlying world is observable, namely the position, orientation, and size of each input line segment and circle. Part of the underlying world, however, is hidden. Hidden information includes knowledge of how line segments and circles are attached to each other as well as knowledge of the relative depth of different objects in the nominally two-dimensional image. I refer to such hidden information as a *world model*. ABIGAIL must construct an initial world model when processing the first frame of the movie and update that world model to be consistent with each new frame. I call this process *model reconstruction*.

The world model consists of two components: a *joint model* and a *layer model*. The joint model represents attachment information while the layer model represents depth information. The nature of the information that can be included in a world model constitutes an ontology that ABIGAIL projects onto the world by her perceptual processes. This ontology allows her to recover information that is not directly perceivable. The general format of the world model, and the model reconstruction procedure itself, are part of ABIGAIL's innate endowment. The particular sequence of world models constructed will, of course, vary from movie to movie.

ABIGAIL's ontology allows pairs of figures to be connected by *joints*. Such joints may be independently rigid or flexible along each of the three relative degrees of freedom between the two joined figures. I refer to such degrees of freedom as *joint parameters*. The rigidity of joint parameters may change over time. For example, observing someone bend their elbow implies that the elbow joint has a flexible rotation parameter. A later observation of that same arm supporting some grasped object requires the adoption of the belief that the elbow-joint rotation parameter is now rigid in order to offer the necessary support to the grasped object. Similarly, the existence of joints may change over time. For example, the process of grasping an object when picking it up is modeled by the formation of a new joint between one's hand and the object. Likewise, the process of releasing an object when putting it down is modeled by the dissolution of that joint. The set of joints and their parameters collectively constitutes a joint model. Since joints are not directly perceivable, ABIGAIL must construct and maintain a joint model that is consistent with the observed world.

ABIGAIL's ontology projects a third dimension onto the two-dimensional observed world. This is necessary since most movies depict events that would require objects to pass through one another if the world were two dimensional. For example, in figure 1 the ball might appear to pass through the table as it bounces. Humans are strongly biased against event interpretations that require one object to pass through another object. Such interpretations constitute violations of the substantiality constraint. A human observer would conjecture instead that the ball passed either in front of, or behind, the table during its bounce. To model such phenomena, ABIGAIL does not need a full third dimension—an impoverished one will do. ABIGAIL's ontology assigns each figure to a *layer*. Layers are unordered. There is no notion of one layer being in front of, or behind, another. Furthermore, there is no notion of one layer being adjacent to another. ABIGAIL's ontology allows only for the knowledge that two figures lie on the same, or on different, layers. Such knowledge is represented by a layer model that specifies whether or not two figures are known to be on the same layer.

Just as the joint model might need to change over time, the layer model too might need to change to remain consistent with the movie. For example, in figure 1 the ball must initially be on the same layer as the table top to account for the fact that the table top supports the ball, preventing it from falling. Later, as the ball bounces, it must be on a different layer than the table top to avoid a substantiality violation. Finally, as the ball comes to rest again on the table top at the end of the movie, they must again be on the same layer.

ABIGAIL uses physical knowledge to construct and maintain the world model. The goal is to adopt a world model that can *explain* the movie. The world model must be *consistent*, that is, joints that are observed to be flexible must indeed be flexible in the model and objects that appear to overlap must indeed be on different layers in the model. Furthermore, the world model must explain how

```
Frame:  0
Joints added:
(MOUTH JOHN)<->(HEAD JOHN) with rigid delta-g theta
(HEAD JOHN)<->(TORSO JOHN) with rigid delta-g theta
(TORSO JOHN)<->(LEFT-THIGH JOHN) with rigid delta-g theta
(TORSO JOHN)<->(LEFT-FOREARM JOHN) with rigid delta-g theta
(RIGHT-THIGH JOHN)<->(RIGHT-CALF JOHN) with rigid delta-g theta
(LEFT-THIGH JOHN)<->(LEFT-CALF JOHN) with rigid delta-g theta
(RIGHT-CALF JOHN)<->(GROUND GROUND) with rigid delta-g theta
(LEFT-CALF JOHN)<->(GROUND GROUND) with rigid delta-g theta
(RIGHT-UPPER-ARM JOHN)<->(LEFT-UPPER-ARM JOHN) with rigid delta-g theta
(RIGHT-UPPER-ARM JOHN)<->(RIGHT-FOREARM JOHN) with rigid delta-g theta
(LEFT-UPPER-ARM JOHN)<->(LEFT-FOREARM JOHN) with rigid delta-g theta
(LINE-SEGMENT3 BALL)<->(LINE-SEGMENT1 BALL) with rigid delta-g
(LINE-SEGMENT2 BALL)<->(LINE-SEGMENT1 BALL) with rigid delta-g
(RIGHT-LEG TABLE)<->(GROUND GROUND) with rigid delta-g theta
(LEFT-LEG TABLE)<->(TOP TABLE) with rigid delta-g theta
(LEFT-LEG TABLE)<->(GROUND GROUND) with rigid delta-g theta
Layer assertions added:
(DIFFERENT-LAYER (TORSO JOHN) (LEFT-FOREARM JOHN))
(DIFFERENT-LAYER (LINE-SEGMENT3 BALL) (LINE-SEGMENT2 BALL))
(DIFFERENT-LAYER (LINE-SEGMENT3 BALL) (LINE-SEGMENT1 BALL))
(DIFFERENT-LAYER (LINE-SEGMENT2 BALL) (LINE-SEGMENT1 BALL))
(SAME-LAYER (SURFACE BALL) (LINE-SEGMENT1 BALL))
(SAME-LAYER (SURFACE BALL) (TOP TABLE))
```

Figure 2: *The world model constructed by* ABIGAIL *when processing frame 0 of the movie shown in figure 1.*

those objects that do not fall are supported. The model reconstruction procedure tries to find the simplest consistent world model that can explain the stability of each frame. Doing so requires a procedure for checking whether or not the objects in a given frame are stable when interpreted according to a given world model. Such a procedure is described in section 3.

The model reconstruction procedure performs *counterfactual* analysis. It decides that some joint or same-layer relation is necessary by asking whether or not the world could be stable without that joint or same-layer relation. For instance, when processing frame 0 of the movie shown in figure 1, ABIGAIL decides that the man's left thigh must be joined to his left calf by a joint that is rigid along two of the three degrees of freedom, since it is impossible to prevent the man from falling without such a joint. Similarly, in the same frame, ABIGAIL decides that the ball is on the same layer as the table top to prevent it from falling. Figure 2 illustrates the world model constructed by ABIGAIL when processing frame 0 of the movie shown in figure 1.

ABIGAIL can use attachment information in the world model to group line segments and circles into aggregate objects. Once she has done so, she must then determine which events occur between those objects. Event recognition is anal-

$$\textsc{Throw}(x,y) \triangleq \neg\Diamond\textsc{Part}(y,x) \wedge \exists z \left(\begin{array}{c} \textsc{Part}(z,x)\wedge \\ \left[\left(\begin{array}{c}\textsc{Translating}(z)\wedge\\ \textsc{Contacts}(z,y)\wedge\\ \textsc{Attached}(z,y)\end{array}\right);\left(\begin{array}{c}\neg\Diamond\textsc{Contacts}(z,y)\wedge\\ \neg\Diamond\textsc{Attached}(z,y)\wedge\\ \neg\Diamond\textsc{Supported}(y)\end{array}\right)\right]\wedge \\ \textsc{Translating}(y)\end{array}\right)$$

$$\textsc{Fall}(x) \triangleq \neg\Diamond\textsc{Supported}(x) \wedge \textsc{TranslatingDown}(x)$$

$$\textsc{Drop}(x,y) \triangleq \exists z \left(\begin{array}{c} \textsc{Part}(z,x)\wedge \\ \left[\left(\begin{array}{c}\textsc{Contacts}(z,y)\wedge\\ \textsc{Attached}(z,y)\wedge\\ \textsc{Supports}(x,y)\wedge\\ \textsc{Supported}(y)\end{array}\right);\left(\begin{array}{c}\neg\Diamond\textsc{Contacts}(z,y)\wedge\\ \neg\Diamond\textsc{Attached}(z,y)\wedge\\ \neg\Diamond\textsc{Supports}(x,y)\wedge\\ \neg\Diamond\textsc{Supported}(y)\end{array}\right)\right]\end{array}\right)$$

$$\textsc{Bounce}(x) \triangleq \left(\begin{array}{c}\textsc{Translating}(x)\wedge\\ \exists y[\neg\Diamond\textsc{Contacts}(x,y);\textsc{Contacts}(x,y);\neg\Diamond\textsc{Contacts}(x,y)]\end{array}\right)$$

$$\textsc{Jump}(x) \triangleq \textsc{Supported}(x);\left(\begin{array}{c}\neg\Diamond\textsc{Supported}(x)\wedge\\ \textsc{TranslatingUp}(x)\end{array}\right)$$

$$\textsc{Put}(x,y) \triangleq \exists w \left(\begin{array}{c} \textsc{Part}(w,x)\wedge \\ \left[\left(\begin{array}{c}\textsc{Translating}(w)\wedge\\ \textsc{Contacts}(w,y)\wedge\\ \textsc{Attached}(w,y)\wedge\\ \textsc{Supports}(x,y)\wedge\\ \textsc{Translating}(y)\end{array}\right);\exists z\left(\begin{array}{c}\textsc{Disjoint}(z,w)\wedge\\ \neg\Diamond\textsc{Translating}(y)\wedge\\ \textsc{Supported}(y)\wedge\\ \textsc{Supports}(z,y)\end{array}\right)\right]\end{array}\right)$$

$$\textsc{PickUp}(x,y) \triangleq \exists w \left(\begin{array}{c} \textsc{Part}(w,x)\wedge \\ \left[\left(\exists z\left(\begin{array}{c}\textsc{Disjoint}(z,w)\wedge\\ \textsc{Supported}(y)\wedge\\ \textsc{Supports}(z,y)\wedge\\ \textsc{Contacts}(z,y)\end{array}\right)\right);\left(\begin{array}{c}\textsc{Translating}(w)\wedge\\ \textsc{Contacts}(w,y)\wedge\\ \textsc{Attached}(w,y)\wedge\\ \textsc{Supports}(x,y)\wedge\\ \textsc{Translating}(y)\end{array}\right)\right]\end{array}\right)$$

Figure 3: *Some of the aggregate event types currently incorporated into* ABIGAIL.

ogous, in many ways, to object recognition. Just as one must *segment* an image into potentially overlapping objects and *classify* those objects into different object types, one must also segment an image *sequence* into potentially overlapping event occurrences and classify the event occurrences into different event types. ABIGAIL is given a model for each event type. These models are analogous to the object models used in model-based object recognition. Figure 3 illustrates some of the event models currently incorporated into ABIGAIL.

The event models are formulated as expressions in a particular logic I call *event logic*. Event-logic expressions describe how aggregate compound events are composed out of simpler primitive events. Event-logic expressions describe constraints on the temporal ordering of the primitive events that comprise aggregate events. Section 4 gives the detailed semantics of event logic. Event-logic expressions can be formulated around any inventory of primitive event types. ABIGAIL currently incorporates the primitives illustrated in figure 4. The truth values of many of these primitives can be computed simply by tracking the changing positions and orientations of objects from frame to frame. Others, particularly the primitives in the left hand column, require hidden knowledge from the world model to determine their truth values. For instance, whether or not an object is supported can depend on whether or not it is attached to, or on the same layer as, a potential source of support. Similarly, whether one object is in contact with another object will depend on whether they are on the same layer. Likewise, whether one object

| | | |
|---|---|---|
| Exists($x$) | MovingPart($x$) | $x = y$ |
| Prominent($x$) | Rotating($x$) | Part($x, y$) |
| Supported($x$) | RotatingClockwise($x$) | Disjoint($x, y$) |
| Supports($x, y$) | RotatingCounterClockwise($x$) | |
| Contacts($x, y$) | Translating($x$) | |
| Attached($x, y$) | TranslatingUp($x$) | |
| At($x, y$) | TranslatingDown($x$) | |
| | TranslatingTowards($x, y$) | |
| | TranslatingAwayFrom($x, y$) | |
| | Flipping($x$) | |
| | SlidingAgainst($x, y$) | |

Figure 4: *The primitive event types currently incorporated into* Abigail.

```
[6:12](TRANSLATING-AWAY-FROM [(LEFT-UPPER-ARM JOHN)] [GROUND])
[11:12](TRANSLATING-DOWN [(LEFT-FOREARM JOHN)])
[6:12](TRANSLATING [(LEFT-FOREARM JOHN)])
[11:12](ROTATING-CLOCKWISE [(LEFT-FOREARM JOHN)])
[6:12](ROTATING [(LEFT-FOREARM JOHN)])
[11:12](TRANSLATING-AWAY-FROM [(LEFT-FOREARM JOHN)] [JOHN-part 6])
[7:12](SUPPORTS [(LEFT-FOREARM JOHN)] [(SURFACE BALL)])
[11:12](TRANSLATING-AWAY-FROM [(LEFT-FOREARM JOHN)] [(EYE JOHN)])
[7:12](ATTACHED [BALL-part 2] [(LEFT-FOREARM JOHN)])
[7:12](CONTACTS [BALL-part 2] [(LEFT-FOREARM JOHN)])
[7:12](SUPPORTS [(LEFT-FOREARM JOHN)] [BALL-part 2])
[11:12](TRANSLATING-TOWARDS [(LEFT-FOREARM JOHN)] [GROUND])
```

Figure 5: *A subset of the primitive event occurrences recognized by* Abigail *when processing the movie shown in figure 1.*

is attached to another will depend on whether there is a joint that connects a component of one to a component of the other. Determining source of support requires counterfactual analysis. An object $A$ supports another object $B$ if $B$ is supported but ceases to be supported when $A$ is removed.

Figure 5 illustrates some of the primitive event occurrences detected by Abigail when processing the movie shown in figure 1. Figure 6 illustrates some of the aggregate event occurrences inferred from all of the primitive events detected during the movie (not just those shown in figure 5). Section 4 discusses the algorithm used by Abigail for inferring aggregate event occurrence from primitive event occurrence.

### 2.1. The Input to Abigail

The input to Abigail consists of a sequence of *frames*, each being a collection of *figures*. The input specifies the position, orientation, shape, and size of each figure in each frame. Shape is specified explicitly by labeling each figure as either

```
[20:21](RAISE [(LEFT-FOREARM JOHN)] [BALL])
[7:11](RAISE [(LEFT-FOREARM JOHN)] [BALL])
[20:26,27](PUT [JOHN-part 3] [BALL])
[16,17:19](JUMP [BALL])
[6:15,17:26](BOUNCE [BALL])
[7:12,13:15](DROP [JOHN-part 3] [BALL])
[13:15](FALL [BALL])
[6:12,13:15](THROW [JOHN-part 3] [BALL])
```

Figure 6: *Some of the aggregate event occurrences recognized by* ABIGAIL *when processing the movie shown in figure 1.*

a line segment or a circle. Position, orientation, and size are specified implicitly: a line segment $f$ is specified by its two endpoints $p(f)$ and $q(f)$, while a circle $f$ is specified by its center $p(f)$ and some point $q(f)$ on its circumference. This method of specifying the position, orientation, and size of figures makes some unrealistic assumptions about the ability to perceive planar configurations of line segments and circles. First, it assumes that a line segment that points from $p(f)$ to $q(f)$ can be distinguished from one that points in the reverse direction from $q(f)$ to $p(f)$. Second, it assumes that circles have perceivable orientation. Third, it assumes that two collinear intersecting line segments can be perceived as distinct figures. Finally, it assumes that two concentric circles with the same radius can also be perceived as distinct figures. Furthermore, since the input consists solely of skeletal configurations of line segments and circles, no objects are ever occluded.

The input to ABIGAIL must satisfy some further restrictions. First, all frames in a given movie must contain the same number of figures. Second, the figures in each frame must be placed in a one-to-one correspondence with figures in adjacent frames. Third, ABIGAIL must be given this correspondence as input. The first two restrictions imply that neither figures, nor objects constructed from figures, can appear, disappear, or enter or leave the field of view. Furthermore, while objects constructed from figures can bend, break, or fuse, low-level figures themselves cannot undergo such transformations. The third restriction implies that an exogenous process must perform tracking. While it would be straightforward to compute the correspondence by finding a mapping that minimized some cost function of the distance between paired figures, I chose not to implement such correspondence computation as it is tangential to the primary focus of this work.

Some terminology is necessary before proceeding. Two figures *intersect* if they share a common point. This intersection may be empty, or it may consist of a single point, two points, or an infinite set of points. Each intersection point is classified as being either a *touching* point or a point of *overlap*. A touching point is one that either coincides with an endpoint of one of the two intersecting figures, or one that coincides with a point of tangency between two circles or a line segment and a circle. An intersection point is a point of overlap if it is not a touching point. Two figures *touch* if their intersection includes a touching point. Two figures *overlap* if their intersection includes a point of overlap. If two figures have a finite intersection, and both touch and overlap, then one must be a line segment and the other must be a circle.

## 2.2. Model Reconstruction

Before formally specifying the format of the world model, some ancillary definitions are needed. For a point $r$ on $f$, let $\rho(r, f)$ denote the *displacement* of $r$ along $f$. If $f$ is a line segment, then $0 \leq \rho(r, f) \leq 1$. The quantity $\rho(r, f)$ denotes the fraction of the way $r$ is along $f$ from $p(f)$ to $q(f)$:

$$\rho(r,f) = \begin{cases} \frac{x(r)-x(p(f))}{x(q(f))-x(p(f))} & x(q(f)) - x(p(f)) \neq 0 \\ \frac{y(r)-y(p(f))}{y(q(f))-y(p(f))} & \text{otherwise} \end{cases}$$

If $f$ is a circle, then $\rho(r, f)$ denotes the difference between the orientation of the vector from $p(f)$ to $r$ and the orientation of the vector from $p(f)$ to $q(f)$. We will not need to compute $\rho(r, f)$ when $f$ is a circle.

I represent the layer model as a binary relation $\bowtie$ between figures. The formula $f \bowtie g$ will denote the proposition that $f$ and $g$ are on the same layer. The layer model $\bowtie$ is required to be an equivalence relation, i.e. it must be reflexive, symmetric, and transitive. I represent the joint model as a set $J$ of joints. Each joint $j \in J$ contains six parameters: $f(j)$, $g(j)$, $r(j)$, $\delta_f(j)$, $\delta_g(j)$, and $\theta(j)$. The parameters $f(j)$ and $g(j)$ specify the two figures connected by $j$. The parameter $r(j)$ specifies the point of intersection. It is necessary to specify the point of intersection since the intersection of two figures may contain more than one point. The Boolean parameters $\delta_f(j)$, $\delta_g(j)$, and $\theta(j)$ collectively specify the degrees of freedom of $j$. If the parameter $\delta_f(j)$ is true, then $\rho(r(j), f(j))$, the displacement of the joint along $f(j)$, is required to be fixed. Similarly, if $\delta_g(j)$ is true, then $\rho(r(j), g(j))$, the displacement of the joint along $g(j)$, must be fixed. Finally, if $\theta(j)$ is true, then the relative orientation of $f(j)$ and $g(j)$ must be fixed. The world model consists of the pair $\langle \bowtie, J \rangle$.

The precise process by which Abigail constructs and maintains the world model is beyond the scope of this chapter. Detailed descriptions of this process are given in [35, 38]. Intuitively, however, for each frame Abigail tries to find the weakest world model that ensures the stability of the objects in that frame. One joint model is not weaker than another if every pair of figures attached in the latter is also attached in the former, and every parameter that is rigid in the latter is also rigid in the former. Likewise, one layer model is not weaker than another if every pair of figures that are on the same layer in the latter are also on the same layer in the former. Finally, one world model is not weaker than another if the joint model of the former is not weaker than that of the latter and the layer model of the former is also not weaker than that of the latter.

The process of finding the weakest world model is complicated by several issues. First, there may be multiple weakest world models. In fact, there may be exponentially many such weakest world models. ABIGAIL thus adopts only one weakest world model. Second, the problem of finding the weakest world model using a global metric of counting joints, rigid joint parameters, and same-layer relations is likely to be NP-hard. Thus ABIGAIL uses a greedy algorithm to find a world model that minimizes the above local relative notion of world model weakness. Third, it might not be possible to find *any* world model that can ensure the stability of *all* of the objects in some frame. Thus ABIGAIL first tries to find the maximal set of figures that *could* be supported by adopting *some* world model and then tries to find the weakest world model that can support *this* set of figures. Fourth, any time that it is possible to support an object with a same-layer relationship, it is also possible to support that same object with a joint. The converse is not true however. Thus while there are situations that would force ABIGAIL to adopt a joint-based explanation over a same-layer-based explanation, there are no situations that would force a same-layer-based explanation over a joint-based explanation. To remedy this, ABIGAIL adopts the further strength-ranking principle that states that one world model is not weaker than another if every pair of figures that are on the same layer in the latter are either on the same layer, or joined, in the former. Finally, the model reconstruction process described so far operates on each frame in isolation. The actual model reconstruction process used by ABIGAIL takes into account information from the world model constructed for the previous frame when constructing the world model for the current frame.

## 3. Stability Determination

As discussed previously, a central component of the event perception mechanism used by ABIGAIL is a procedure for determining whether or not a collection of objects in an image is stable. An earlier version of ABIGAIL, described in [35, 38], used a kinematic simulator for this purpose. In this section, I will describe a different, more static, method for determining stability. This approach is similar to that taken by Blum et al.[7] and Fahlman[11] in that it reduces a set of constraints to a linear-programming problem. The constraints used in these prior approaches, however, attempt to balance the forces applied to objects in the image. The constraints used in the approach described here are more kinematic in nature. They attempt to find a set of hypothetical velocities for objects in the image that uphold the substantiality and joint constraints while decreasing the potential energy.

Stability determination takes the form of a procedure $\textsc{Stable}(F, \langle \bowtie, J \rangle)$. The input to this procedure consists of a set $F$ of figures, along with a world model. I collectively refer to this as a *configuration*. The output is a single Boolean value expressing whether or not the given configuration is stable.

The input to the stability determination procedure contains only static information, namely the position, orientation, shape, and size of each figure in a single frame. Dynamic information, such as the current velocity or acceleration of each figure, is not given as input to the stability determination procedure. Stability is determined, however, by *hypothesizing* a velocity for each figure and asking whether or not it is possible to assign a velocity to each figure in such a way that the collection of velocity assignments results in a decrease in the potential energy of the configuration while satisfying the following constraints:

**rigidity:** Figures do not not shrink or stretch.

**gravity:** Potential energy cannot increase.

**ground plane:** Figures cannot pass through the ground.

**substantiality:** Figures on the same layer cannot penetrate each other.

**attachment:** Attached figures must overlap.

**displacement:** The rigidity of joints with fixed displacement must be upheld.

**rotation:** The rigidity of joints with fixed rotation must be upheld.

For each figure $f$, let $\dot{p}(f)$ and $\dot{q}(f)$ denote the hypothesized instantaneous velocities of $p(f)$ and $q(f)$ respectively. The quantities $\dot{p}(f)$ and $\dot{q}(f)$ constitute variables. Each of the above seven constraints can be expressed as equations and inequalities between these variables.

### 3.1. Stability Constraints

In the following formulation, if $(x, y)$ is a vector, then $\overline{(x, y)}$ denotes a vector of the same magnitude rotated counterclockwise 90°. $\overline{(x, y)} = (-y, x)$. Furthermore, I use $||\mathbf{x}||$ to denote the magnitude of $\mathbf{x}$, namely $\sqrt{\mathbf{x} \cdot \mathbf{x}}$.

#### 3.1.1. Rigidity

We require each individual line segment and circle to have fixed size when asking whether or not a configuration is stable. If we did not, then every configuration would be unstable, since it would always be possible for figures to shrink. This constraint can be expressed uniformly for both line segments and circles by requiring that the distance from $p(f)$ to $q(f)$ be constant. Let $\sigma$ denote a vector from $p(f)$ to $q(f)$. Thus $\sigma = q(f) - p(f)$. For the distance between $p(f)$ and $q(f)$ to remain constant, the component of $\dot{p}(f)$ along $\sigma$ must be equal to the component of $q(f)$ along $\sigma$. This yields the following constraint:

$$(\forall f \in F)\dot{p}(f) \cdot \sigma = \dot{q}(f) \cdot \sigma$$

This constraint is depicted graphically in figure 7.

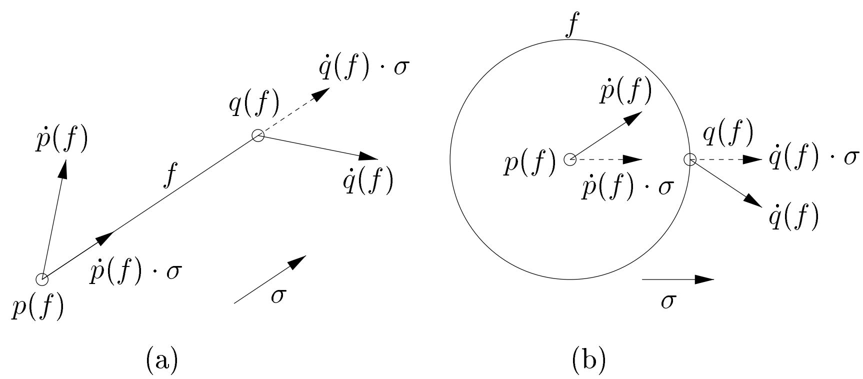


Figure 7: *Enforcing the rigidity constraint. Note that $\sigma$, $\dot{p}(f) \cdot \sigma$, and $\dot{q}(f) \cdot \sigma$ are not drawn to scale.*

#### 3.1.2. Gravity

Since we have no information about the actual velocities of the figures in the image, we will assume that they are all zero. Under this assumption, the kinetic energy of the configuration is zero and the potential energy of the configuration cannot increase. Thus the first derivative of the potential energy of the configuration must not be positive. The total potential energy of the configuration is the sum of the potential energies of the individual figures. The potential energy of each figure is, in turn, proportional to its mass times the height of its center of mass. Let us assume that mass is distributed evenly and proportional to the length of line segments and circumference of circles. Under this assumption, the center of mass of a line segment is located at its midpoint, while the center of mass of a circle is located at its center. This yields the following expression of the gravitational constraint:

$$\frac{dPE(f)}{dt} = \begin{cases} \frac{1}{2}||q(f)-p(f)||(y(\dot{p}(f))+y(\dot{q}(f))) & f \text{ is a line segment} \\ \pi||q(f)-p(f)||^2 y(\dot{p}(f)) & f \text{ is a circle} \end{cases}$$

$$\sum_{f \in F} \frac{dPE(f)}{dt} \leq 0 \qquad (1)$$

#### 3.1.3. Ground Plane

Since figures cannot pass through the ground, when a figure touches the ground, the vertical component of the velocity at the point of contact must not be negative. A line segment touches the ground when either of its endpoints touches the ground. A circle touches the ground when the height of its center equals its radius. Thus

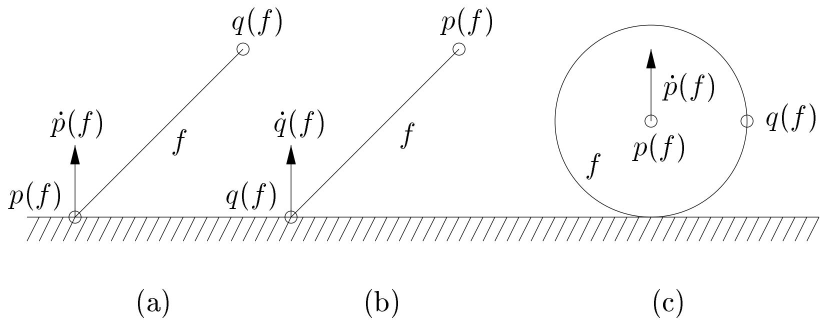


Figure 8: *Enforcing the ground-plane constraint.*

the ground-plane constraint can be formulated as follows:

$$\begin{gathered}(\forall f \in F)\text{LineSegment}(y) \wedge y(p(f)) = 0 \rightarrow y(\dot{p}(f)) \geq 0\\(\forall f \in F)\text{LineSegment}(y) \wedge y(q(f)) = 0 \rightarrow y(\dot{q}(f)) \geq 0\\(\forall f \in F)\text{Circle}(y) \wedge y(p(f)) = ||q(f) - p(f)|| \rightarrow y(\dot{p}(f)) \geq 0\end{gathered}$$

These constraints are depicted graphically in figure 8.

### 3.1.4. Substantiality

For all pairs of figures $f$ and $g$ we must enforce the constraint that $f$ and $g$ not penetrate each other if they are on the same layer. Since we are concerned with the instantaneous behavior of the configuration, a penetration can commence only when $f$ touches $g$. Let $r$ be the point of contact and let $\dot{r}(f)$ and $\dot{r}(g)$ be the velocities of $f$ and $g$ at $r$ respectively. Let $\sigma$ be a vector such that a penetration would occur if the component of $\dot{r}(f)$ along $\sigma$ were greater than the component of $\dot{r}(g)$ along $\sigma$. The substantiality constraint could then be stated simply as follows:

$$(\forall f \in F)(\forall g \in F) f \bowtie g \wedge \text{TouchAt}(f, g, r) \rightarrow \dot{r}(f) \cdot \sigma \leq \dot{r}(g) \cdot \sigma \tag{2}$$

To make this constraint concrete, we need to specify $\dot{r}(f)$, $\dot{r}(g)$, and $\sigma$.

There are several cases to consider, since $f$ and $g$ can each be either a line segment or a circle. First consider the case where both $f$ and $g$ are line segments. In this case there are four subcases to consider, since either endpoint of $f$ can touch $g$ and either endpoint of $g$ can touch $f$. Consider the subcase where $p(f)$ touches $g$. In this subcase, $\dot{r}(f) = \dot{p}(f)$ and $\dot{r}(g) = (1 - \rho(r, g))\dot{p}(g) + \rho(r, g)\dot{q}(g)$. The vector $\sigma$ is chosen so that it is perpendicular to $g$ in the direction away from $q(f)$:

$$\sigma = -[\overline{q(g) - p(g)} \cdot (q(f) - p(f))]\overline{q(g) - p(g)}$$

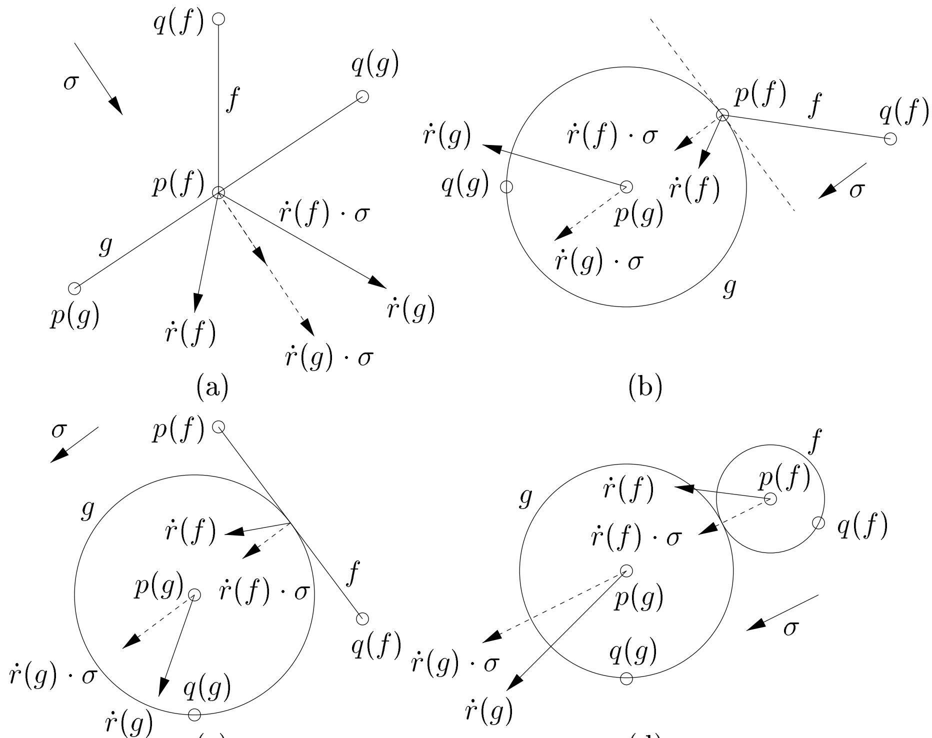


Figure 9: *Enforcing the substantiality constraint. Note that* $\sigma$, $\dot{r}(f)\cdot\sigma$, *and* $\dot{r}(g)\cdot\sigma$ *are not drawn to scale.*

This case is depicted graphically in figure 9(a). The remaining three subcases are analogous to this subcase.

Now consider the case where $f$ is a line segment and $g$ is a circle. In this case $f$ can touch $g$ either at an endpoint of $f$ or by being tangent to $g$. For the first subcase, either endpoint of $f$ can touch $g$. We will only consider the case where $p(f)$ touches $g$ since the case where $q(f)$ touches $g$ is analogous. In this case, it is easy to see that $\dot{r}(f) = \dot{p}(f)$, since $r = p(f)$. It is also easy to see that $\dot{r}(g) = \dot{p}(g)$. The vector $\sigma$ is chosen so that it is perpendicular to the line that is tangent to the circle at $r$ in the direction away from $q(f)$:

$$\sigma = -[(q(f) - p(f)) \cdot (p(f) - p(g))](p(f) - p(g))$$

This subcase is depicted graphically in figure 9(b).

Now consider the subcase where $f$ is tangent to $g$. In this subcase, $\dot{r}(f) = (1 - \rho(r, f))\dot{p}(f) + \rho(r, f)\dot{q}(f)$ and again $\dot{r}(g) = \dot{p}(g)$. The vector $\sigma$ is chosen to point from $r$ to the center of the circle. Thus $\sigma = p(g) - r$. This subcase is depicted graphically in figure 9(c).

Finally, consider the case where $f$ and $g$ are both circles. In this case $\dot{r}(f) = \dot{p}(f)$ and $\dot{r}(g) = \dot{p}(g)$. There are two subcases to consider when choosing $\sigma$. If the two circles are outside each other, $\sigma$ is chosen to point from the center of $f$ to the center of $g$. Thus $\sigma = p(g) - p(f)$. This case is depicted graphically in figure 9(d). Otherwise, $\sigma$ is chosen to point in the reverse direction from the center of $g$ to the center of $f$. Thus $\sigma = p(f) - p(g)$. This later case handles both the situation where $f$ is larger than $g$, as well as the situation where $f$ is smaller than $g$.

#### 3.1.5. Attachment

For all pairs of figures $f$ and $g$ we must enforce the constraint that $f$ and $g$ intersect if they are attached. Since again, we are concerned only with the instantaneous behavior of the configuration, two figures can cease to intersect only when they touch. The attachment constraint can be viewed as the opposite of the substantiality constraint. If $r$, $\dot{r}(f)$, $\dot{r}(g)$, and $\sigma$ are defined the same as for the substantiality constraint, then an attachment violation would occur if the component of $\dot{r}(f)$ along $\sigma$ were *less* than the component of $\dot{r}(g)$ along $\sigma$. Thus the attachment constraint can be stated simply as follows:

$$(\forall j \in J) f(j) \in F \wedge g(j) \in F \wedge \textsc{TouchAt}(f(j), g(j), r(j)) \rightarrow \dot{r}(f) \cdot \sigma \geq \dot{r}(g) \cdot \sigma$$

where $\dot{r}(f)$, $\dot{r}(g)$, and $\sigma$ are the same as for the substantiality constraint.

#### 3.1.6. Displacement

When $f$ and $g$ are connected by a joint $j$, and $\delta_f(j)$ is true, the displacement of $r(j)$ along $f$ must be fixed. Similarly, when $\delta_g(j)$ is true, the displacement of $r(j)$ along $g$ must be fixed. I will only illustrate the former constraint, since the latter is handled analogously. Let $r = r(j)$ and let $\dot{r}(f)$ and $\dot{r}(g)$ be the velocities

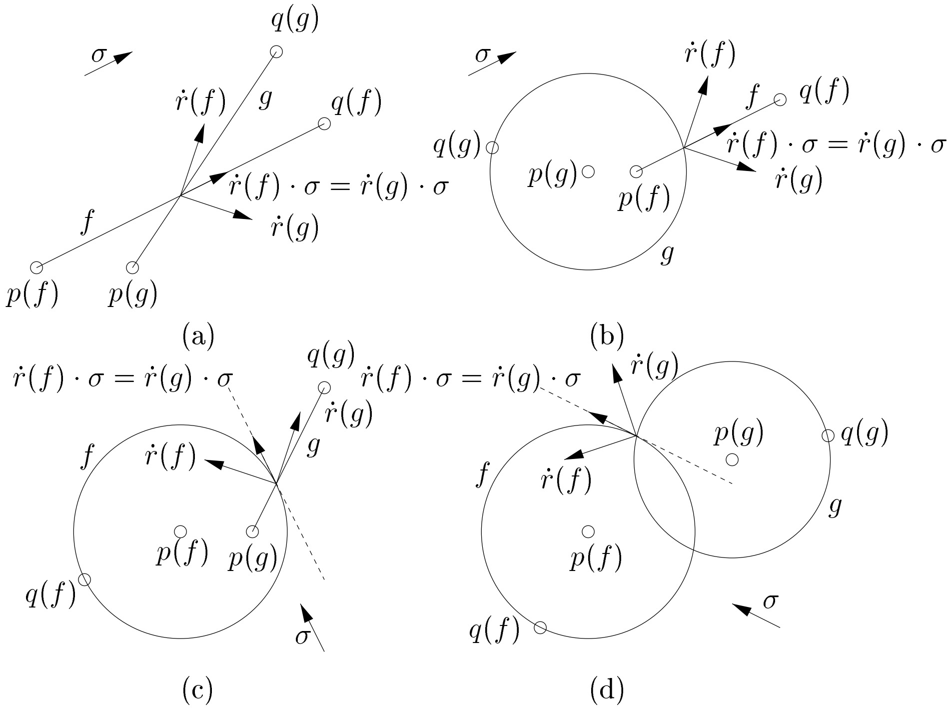


Figure 10: *Enforcing the displacement constraint. Note that* $\sigma$*,* $\dot{r}(f)\cdot\sigma$*, and* $\dot{r}(g)\cdot\sigma$ *are not drawn to scale.*

of $f(j)$ and $g(j)$ at $r$ respectively. If $f(j)$ is a line segment, then $\dot{r}(f)$ can be expressed as follows:

$$\dot{r}(f) = (1-\rho(r,f))\dot{p}(f) + \rho(r,f)\dot{q}(f)$$

Similarly, if $f$ is a circle, then $\dot{r}(f)$ can be expressed as follows:

$$\dot{r}(f) = \dot{p}(f) + [(\dot{q}(f)-\dot{p}(f))\cdot\overline{q(f)-p(f)}]\overline{r-p(f)}$$

One can express $\dot{r}(g)$ in an analogous fashion. Furthermore, let $\sigma$ be a vector that indicates the direction of potential displacement along $f$. If $f$ is a line segment, then $\sigma = q(f)-p(f)$. If $f$ is a circle, then $\sigma = \overline{r-p(f)}$, a vector tangent to $f$ at $r$. The displacement constraint can now be stated simply as follows:

$$(\forall j\in J) f(j)\in F \wedge g(j)\in F \wedge \textsc{TouchAt}(f(j),g(j),r(j)) \wedge \delta_f(j) \rightarrow \dot{r}(f)\cdot\sigma = \dot{r}(g)\cdot\sigma$$

The four cases of the displacement constraint, where $f$ and $g$ are each line segments and circles, are depicted graphically in figure 10.

### 3.1.7. Rotation

When $f$ and $g$ are connected by a joint $j$, and $\theta(j)$ is true, then the relative orientation of $f$ and $g$ must be fixed. This can be stated alternatively as the constraint that the instantaneous angular velocities of $f$ and $g$ must be equal. Since the angular velocity of $f$ is $[\dot{q}(f) - \dot{p}(f)] \cdot \overline{q(f) - p(f)}$, independent of whether $f$ is a line segment or a circle, the rotation constraint can be formulated simply as follows:

$$
\begin{array}{l}
(\forall j \in J) f(j) \in F \wedge g(j) \in F \wedge \theta(j) \rightarrow \\
\quad [\dot{q}(f(j)) - \dot{p}(f(j))] \cdot \overline{q(f(j)) - p(f(j))} = [\dot{q}(g(j)) - \dot{p}(g(j))] \cdot \overline{q(g(j)) - p(g(j))}
\end{array}
$$

## 3.2. Solving the Stability Constraints

A configuration is unstable if it is possible to find an assignment of velocities to each of the figures in the configuration that satisfies the aforementioned constraints and decreases the potential energy. Otherwise, the configuration is stable. Note that it is always possible to satisfy the constraints by assigning a zero velocity to each figure. If we let $\mathbf{x}$ be a vector consisting of the concatenation of the variables $x(\dot{p}(f))$, $y(\dot{p}(f))$, $x(\dot{q}(f))$, and $y(\dot{q}(f))$, then all of the aforementioned constraints can be expressed either as equations of the form $A_1\mathbf{x} = 0$, or inequalities of the form $A_2\mathbf{x} \leq 0$, where $A_1$ and $A_2$ are matrices of constant coefficients that depend only on the observed positions, orientations, shapes, and sizes of the figures in the configuration. Furthermore, the equations $A_1\mathbf{x} = 0$ can be converted to pairs of inequalities $A_1\mathbf{x} \leq 0$ and $-A_1\mathbf{x} \leq 0$. Alternatively, the inequalities can be converted to equations by adding slack variables. Thus one can formulate the stability determination problem as a problem of the following form:

$$
\begin{array}{ll}
\min & \mathbf{c} \cdot \mathbf{x} \\
 & A\mathbf{x} \leq 0
\end{array}
$$

Here, $\mathbf{c} \cdot \mathbf{x}$ denotes the instantaneous increase in potential energy of the configuration. It is the same as the left hand side of constraint (1). The configuration is stable if and only if the minimum value of the objective function is zero.

In the above problem the variables can take on negative values. One can reformulate this problem as a linear-programming problem by replacing each variable $x$ with the difference $x^+ - x^-$ of two nonnegative variables. One can thus reduce stability determination to a problem of the following form:

$$
\begin{array}{ll}
\min & \mathbf{c} \cdot \mathbf{x} \\
 & A\mathbf{x} = 0 \\
 & \mathbf{x} \geq 0
\end{array}
$$

Note that $\mathbf{x} = 0$ will always satisfy $A\mathbf{x} = 0$ for any $A$. Thus the above linear-programming problem will always have a feasible basic vector and zero is an upper bound on the minimum value of the objective function. Furthermore, if some $\mathbf{x} \neq 0$ satisfies $A\mathbf{x} = 0$, and has the property that $\mathbf{c} \cdot \mathbf{x} < 0$, then for any $k > 0$, it is also true that $Ak\mathbf{x} = 0$ and $\mathbf{c} \cdot k\mathbf{x} < \mathbf{c} \cdot \mathbf{x}$. Thus it is possible to scale any feasible basic vector by an arbitrary positive constant and if the objective function can take on some negative value, then it can take on any negative value. This has the physical interpretation that the stability of a configuration is independent of the gravitational acceleration constant. Thus the particular linear-programming problems constructed by the reduction discussed here have either a zero minimum, in which case the configuration is stable, or an unbounded minimum, in which case the configuration is unstable. Accordingly, one can alternatively formulate the stability determination problem as asking whether or not the following set of constraints is feasible:

$$\begin{aligned} \mathbf{c} \cdot \mathbf{x} &= -1 \\ A\mathbf{x} &= 0 \\ \mathbf{x} &\geq 0 \end{aligned}$$

### 3.3. Anomalies

The above procedure for determining stability has a number of limitations. The first arises either when two collinear line segments overlap and are on the same layer or when two concentric circles have the same radius and are on the same layer. In this situation, $\sigma$ will be zero when enforcing the substantiality constraint, (i.e. formula 2). Thus the substantiality constraint will be trivially satisfied independent of the velocities $\dot{r}(f)$ and $\dot{r}(g)$. This is symptomatic of a deeper problem. When two collinear line segments overlap, or when two concentric circles have the same radius, it is ambiguous in the current ontology as to which side of one the other resides. The current stability determination procedure errs toward leniency in this situation, never enforcing substantiality in such a situation. The kinematic simulator described in [35] errs in the opposite direction, *always* enforcing substantiality in such a situation. This has the consequence of permanently bonding together two such overlapping figures. It is not clear which is the lesser of two evils.

A similar problem occurs with the attachment constraint. In the above stability determination procedure, the attachment constraint will not be enforced between two attached collinear line segments that overlap or between two attached concentric circles that have the same radius. This is not a problem in the case of two circles since they will continue to overlap even when they move away from each other by an infinitesimal amount. I have not yet solved this problem in the case of two line segments but believe that it requires a disjunctive linear-programming problem.

The second problem occurs when two line segments on the same layer touch at both of their endpoints. This is illustrated in figure 11(a). If $g$ were stationary, then when enforcing the substantiality constraint, the current stability determination procedure would require the direction of $\dot{p}(f)$ to lie within the bounds of the inner arc. This is overly restrictive since, in reality, a substantiality violation would not occur so long as the direction of $\dot{p}(f)$ were to lie within the bounds of

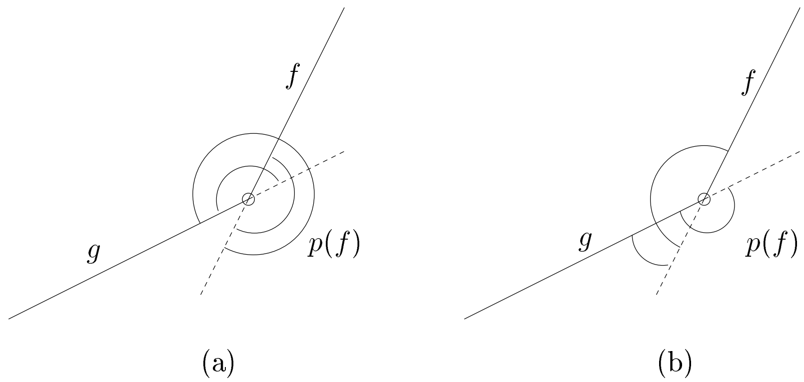


Figure 11: *Anomalies that arise when enforcing the substantiality (a) and attachment (b) constraints between line segments that touch at both of their endpoints.*

the outer arc. This outer arc is the *union* of the inner and central arcs. Unfortunately, one cannot handle such a union with linear constraints. One can, however, formulate this as a disjunctive linear-programming problem.

A similar problem occurs with the attachment constraint. This is illustrated in figure 11(b). If $f$ and $g$ were attached, and $g$ were stationary, then when enforcing the attachment constraint, the current stability determination procedure would require the direction of $\dot{p}(f)$ to lie within the bounds of the inner arc. This is not sufficiently restrictive since, in reality, the two line segments will cease to overlap unless the direction of $\dot{p}(f)$ were to lie within the bounds of the outer arc. This outer arc is the *intersection* of the inner and central arcs. Fortunately, in this case one can handle such an intersection simply as a conjunction of linear constraints.

A third problem arises with joints that have flexible rotation. This is illustrated in figure 12. In figure 12(a), joints $j_1$ and $j_2$ are rigid along all three degrees of freedom, $j_3$ is rigid along both displacement degrees of freedom but has flexible rotation, and $j_4$ is flexible along all three degrees of freedom. If it were not for $j_4$, $p(f)$ could move along the indicated circular path. In reality, this configuration is stable, since $p(f)$ cannot move along the circular path without causing $f$ to cease to overlap with $g$. But the current stability determination procedure can assign $\dot{p}(f)$ a nonzero downward velocity that is simultaneously parallel to $g$ and tangential to the circle. This will allow an instantaneous decrease in potential energy and cause the stability determination procedure to decide that the configuration is unstable. A similar situation is illustrated in figure 12(b). Here, joints $j_1$ and $j_2$ are rigid along all three degrees of freedom, $j_3$ and $j_4$ are rigid along both displacement degrees of freedom but flexible along the rotational degrees of freedom, and $j_5$ is flexible along all three degrees of freedom. If it were not for $j_5$, $p(f)$ and $p(g)$ could move along the indicated circular paths. Again, in reality, this configuration is stable, since neither $p(f)$ nor $p(g)$ could move along

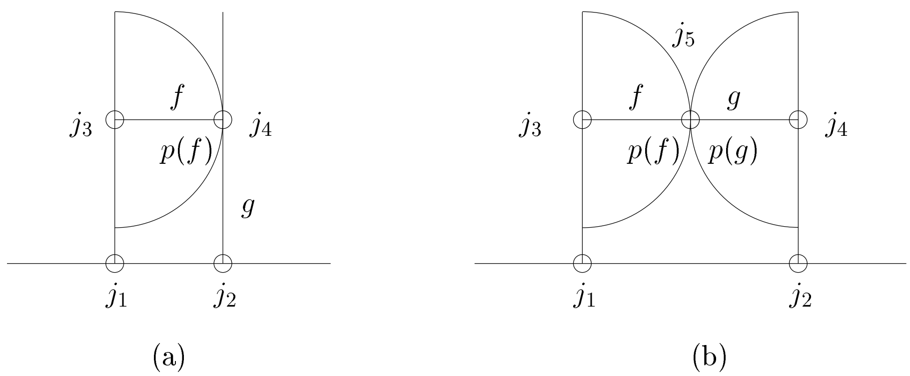


Figure 12: *Anomalies that arise when enforcing the attachment constraint in the presence of closed-loop kinematic chains.*

their circular paths without causing $f$ to cease to overlap with $g$. But again, the current stability determination procedure can assign parallel nonzero downward velocities of equal magnitude to $\dot{p}(f)$ and $\dot{p}(g)$. This will allow an instantaneous decrease in potential energy and cause the stability determination procedure to decide that the configuration is unstable.

These anomalies are symptomatic of a much deeper problem that arises whenever the configuration has a closed-loop kinematic chain. A closed-loop kinematic chain can constrain the motion of two points on that chain to follow complex curves. While the two points may coincide at a point where the two curves are tangential, the current stability determination procedure has no way to enforce constraints between higher order derivatives of point motion.

People also have difficulty predicting the stability of configurations with closed-loop kinematic chains. While they clearly can understand the mechanisms from figure 12, the human capacity breaks down when presented with mechanisms that are substantially more complex. More research needs to be done to determine precisely when such breakdown occurs and what stability determination procedure can deal with precisely those cases that humans can handle.

### 3.4. Variations

The procedure described above can determine whether or not an entire configuration is stable. Often one needs to answer similar, but slightly different, questions. For example, earlier I described the model reconstruction process. This process requires computing 'the maximal set of figures that could be supported by adopting *some* world model.' Performing this process requires, as a subroutine, the ability to determine the set of figures that are unsupported in a given configuration. This can be computed simply by looping over each figure and constructing for that figure, a linear-programming problem with the same constraints as the original reduction, including the constraint that the potential energy of the entire configuration not increase, but where the objective function simply tries to reduce the potential energy of that lone figure. The set of unsupported figures is the set of figures for which this linear-programming problem has an unbounded minimum.

Similarly, the event perception process used by ABIGAIL requires her to determine the stability of aggregate objects on a frame-by-frame basis. Each object is a collection of figures. The stability of an object can be determined by forming a linear-programming problem with the same constraints as the original reduction, including the constraint that the potential energy of the entire configuration not increase, but where the objective function simply tries to reduce the potential energy of the figures that comprise the object in question.

## 4. INFERRING AGGREGATE EVENTS FROM PRIMITIVE EVENTS

The stability determination procedure described in the previous section allows ABIGAIL to construct and maintain a world model containing unobservable joint and layer information. This information, along with further application of the stability determination procedure, allows ABIGAIL to determine the truth value of each of the primitive event types illustrated in figure 4 for each tuple of segmented objects in each frame of the movie. Having determined the truth values of these primitive event types, ABIGAIL must then determine the truth values of compound event types such as those illustrated in figure 3. In this section, I will describe the algorithms used by ABIGAIL to perform this task.

The requirements of this task are illustrated by the following simple example. Suppose that one observes occurrences of the primitive event types

$\textsc{TranslatingTowards}(\mathbf{ball}, \mathbf{ground})$,
$\textsc{Contacts}(\mathbf{ball}, \mathbf{ground})$, and
$\textsc{TranslatingAwayFrom}(\mathbf{ball}, \mathbf{ground})$

during the intervals $[1, 5]$, $[5, 6]$, and $[6, 10]$ respectively. Now further suppose that the compound event type $\textsc{BounceAgainst}(x, y)$ is defined as the sequence

$$\textsc{TranslatingTowards}(x, y); \textsc{Contacts}(x, y); \textsc{TranslatingAwayFrom}(x, y).$$

Under this definition, given the above primitive event occurrences, an aggregate $\textsc{BounceAgainst}(\mathbf{ball}, \mathbf{ground})$ event occurrence coincides with the interval $[1, 10]$. Given a library of such aggregate event type definitions, such as those illustrated in figure 3, a natural question arises: How can one determine which aggregate event types occur, and when they occur? This section gives an efficient algorithm for answering this question. Earlier efforts to address this question include the work of Badler[6] and Borchardt[8, 9].

### 4.1. Event Logic

I represent event types as expressions in a particular temporal logic that I term *event logic*. Event-logic expressions denote event types, not event occurrences. Thus, event-logic expressions are not formulas in the logical sense; they do not have truth values. The proposition that an occurrence of some event type $\Phi$ coincided with some interval $\mathbf{i}$ does have a truth value however. I represent such a proposition with the atomic formula $\Phi@\mathbf{i}$.

There are thirteen possible relations between two intervals. Following Allen[5] I refer to these relations as $=$, $<$, $>$, $\mathsf{m}$, $\mathsf{mi}$, $\mathsf{o}$, $\mathsf{oi}$, $\mathsf{s}$, $\mathsf{si}$, $\mathsf{f}$, $\mathsf{fi}$, $\mathsf{d}$, and $\mathsf{di}$. If $b(\mathbf{i})$ and $e(\mathbf{i})$ denote the beginning and end of $\mathbf{i}$ respectively, these relations can be defined as follows:

$$
\begin{aligned}
\mathbf{i} = \mathbf{j} &\triangleq b(\mathbf{i}) = b(\mathbf{j}) \wedge e(\mathbf{i}) = e(\mathbf{j})\\
\mathbf{i} < \mathbf{j} &\triangleq e(\mathbf{i}) < b(\mathbf{j})\\
\mathbf{i}\,\mathsf{m}\,\mathbf{j} &\triangleq e(\mathbf{i}) = b(\mathbf{j})\\
\mathbf{i}\,\mathsf{o}\,\mathbf{j} &\triangleq b(\mathbf{i}) < b(\mathbf{j}) < e(\mathbf{i}) < e(\mathbf{j})\\
\mathbf{i}\,\mathsf{s}\,\mathbf{j} &\triangleq b(\mathbf{i}) = b(\mathbf{j}) \wedge e(\mathbf{i}) < e(\mathbf{j})\\
\mathbf{i}\,\mathsf{f}\,\mathbf{j} &\triangleq e(\mathbf{i}) = e(\mathbf{j}) \wedge b(\mathbf{i}) > b(\mathbf{j})\\
\mathbf{i}\,\mathsf{d}\,\mathbf{j} &\triangleq b(\mathbf{i}) > b(\mathbf{j}) \wedge e(\mathbf{i}) < e(\mathbf{j})
\end{aligned}
$$

The relations $>$, $\mathsf{mi}$, $\mathsf{oi}$, $\mathsf{si}$, $\mathsf{fi}$, and $\mathsf{di}$ denote variants of the relations $<$, $\mathsf{m}$, $\mathsf{o}$, $\mathsf{s}$, $\mathsf{f}$, and $\mathsf{d}$ with reversed argument order. Exactly one of the above thirteen relations will hold between any two noninstantaneous intervals. If $\mathbf{i}$ is instantaneous but $\mathbf{j}$ is not, then $\mathbf{i}\,\mathsf{m}\,\mathbf{j}$ will coincide with $\mathbf{i}\,\mathsf{s}\,\mathbf{j}$ and $\mathbf{i}\,\mathsf{mi}\,\mathbf{j}$ will coincide with $\mathbf{i}\,\mathsf{f}\,\mathbf{j}$. Likewise, if $\mathbf{j}$ is instantaneous but $\mathbf{i}$ is not, then $\mathbf{i}\,\mathsf{m}\,\mathbf{j}$ will coincide with $\mathbf{i}\,\mathsf{fi}\,\mathbf{j}$ and $\mathbf{i}\,\mathsf{mi}\,\mathbf{j}$ will coincide with $\mathbf{i}\,\mathsf{si}\,\mathbf{j}$. Similarly, if both $\mathbf{i}$ and $\mathbf{j}$ are instantaneous, then $\mathbf{i} = \mathbf{j}$, $\mathbf{i}\,\mathsf{m}\,\mathbf{j}$, and $\mathbf{i}\,\mathsf{mi}\,\mathbf{j}$ will coincide.

By definition, the formula $\Phi@\mathbf{i}$ states that an occurrence of $\Phi$ *coincided* with the interval $\mathbf{i}$. In other words, an occurrence of $\Phi$ started at the beginning of $\mathbf{i}$ and finished at the end of $\mathbf{i}$. The formula $\Phi@\mathbf{i}$ would not be true if an occurrence of $\Phi$ did not precisely coincide with $\mathbf{i}$, but instead overlapped, partially or totally, with $\mathbf{i}$. In this case, I say that an occurrence of $\Phi$ *overlapped*[1] with $\mathbf{i}$. Events may have internal structure that render such a distinction important. Overlapped occurrence is a derived notion and can be expressed in terms of coincidental occurrence using event-logic primitives to be described shortly.

I take an *interval* to be an ordered pair of real numbers $[i, j]$ such that $i \leq j$. An interval $[i, j]$ is a *subinterval* of $[k, l]$ if $i \geq k$ and $j \leq l$. An interval $\mathbf{i}$ is a *super-interval* of $\mathbf{j}$ if $\mathbf{j}$ is a subinterval of $\mathbf{i}$. I will denote the set of subintervals

[1] I use the term 'overlapped' here in a different sense than Allen. Here, 'overlapped' connotes the union of the relations $\mathsf{o}$, $\mathsf{oi}$, $\mathsf{s}$, $\mathsf{si}$, $\mathsf{f}$, $\mathsf{fi}$, $\mathsf{d}$, and $\mathsf{di}$.

of an interval $\mathbf{i}$ as $\textsc{Sub}(\mathbf{i})$. An interval is *integral* if both endpoints are integers. Integral intervals have finite sets of integral subintervals. This fact is crucial for some, though not all, of the analyses later in this section. Finally, I define the *span* of two intervals $\mathbf{i}$ and $\mathbf{j}$, denoted $\textsc{Span}(\mathbf{i}, \mathbf{j})$, as the smallest super-interval of both $\mathbf{i}$ and $\mathbf{j}$.

When classifying events, prior researchers (e.g. [47, 10, 48, 19, 34]) have noted that some event types have the following two properties. First, if they are true during an interval $\mathbf{i}$, then they are also true during any subinterval of $\mathbf{i}$. Second, if they are true during two intervals $\mathbf{i}$ and $\mathbf{j}$ such that $\mathbf{i}\mathsf{m}\mathbf{j}$, then they are also true during $\textsc{Span}(\mathbf{i}, \mathbf{j})$. Events with these properties are termed *liquid*, following Shoham[34]. Some event types are liquid while others are not. The fact that some event types are liquid plays a crucial role in the algorithms developed in this section.

The syntax of event-logic expressions is defined as follows. We are given finite disjoint sets of constant and variable symbols along with a finite set of predicate symbols, each of a specified arity. An atomic event-logic expression is a predicate symbol of arity $n$ applied to a sequence of $n$ constants or variables. An event-logic expression is either an atomic event-logic expression, or one of the compound event-logic expressions $\neg\Phi$, $\Phi \vee \Psi$, $\forall x \Phi$, $\exists x \Phi$, $\Phi \wedge_R \Psi$, $\Diamond_R \Phi$, or $\Phi^+$, where $\Phi$ and $\Psi$ are event-logic expressions, $x$ is a variable, and $R \subseteq \{=, <, >, \mathsf{m}, \mathsf{mi}, \mathsf{o}, \mathsf{oi}, \mathsf{s}, \mathsf{si}, \mathsf{f}, \mathsf{fi}, \mathsf{d}, \mathsf{di}\}$. A *ground* event-logic expression is one that has no free variables.

Informally, the semantics of compound event-logic expressions is defined as follows:

- $\neg\Phi$ denotes the nonoccurrence of $\Phi$. An occurrence of $\neg\Phi$ coincides with $\mathbf{i}$ if no occurrence of $\Phi$ coincides with $\mathbf{i}$. Note that $(\neg\Phi)@\mathbf{i}$ could be true, even if an occurrence of $\Phi$ *overlapped* with $\mathbf{i}$, so long as no occurrence of $\Phi$ *coincided* with $\mathbf{i}$.

- $\Phi \vee \Psi$ denotes the occurrence of either $\Phi$ or $\Psi$.

- $\forall x \Phi$ denotes the simultaneous occurrence of $\Phi$ for all objects.

- $\exists x \Phi$ denotes the occurrence of $\Phi$ for some object.

- $\Phi \wedge_R \Psi$ denotes the occurrence of both $\Phi$ and $\Psi$. The occurrences of $\Phi$ and $\Psi$ need not be simultaneous. The subscript $R$ specifies a set of allowed Allen relations between the occurrences of $\Phi$ and $\Psi$. If occurrences of $\Phi$ and $\Psi$ coincide with $\mathbf{i}$ and $\mathbf{j}$ respectively, and $\mathbf{i}r\mathbf{j}$ for some $r \in R$, then an occurrence of $\Phi \wedge_R \Psi$ coincides with the span of $\mathbf{i}$ and $\mathbf{j}$. I abbreviate the special case $\Phi \wedge_{\{=\}} \Psi$ simply as $\Phi \wedge \Psi$ without any subscript. $\Phi \wedge \Psi$ describes an aggregate event where both $\Phi$ and $\Psi$ occur simultaneously. I also abbreviate the special case $\Phi \wedge_{\{\mathsf{m}\}} \Psi$ as $\Phi; \Psi$. $\Phi; \Psi$ describes an aggregate event where an occurrence of $\Phi$ is immediately followed by an occurrence of $\Psi$.

- An occurrence of $\Diamond_R \Phi$ coinciding with $\mathbf{i}$ denotes an occurrence of $\Phi$ at some other interval $\mathbf{j}$ such that $\mathbf{j} r \mathbf{i}$ for some $r \in R$. $\Diamond_R$ can act as a tense operator. Expressions such as $\Diamond_{\{<\}}\Phi$, $\Diamond_{\{>\}}\Phi$, $\Diamond_{\{\mathsf{m}\}}\Phi$, and $\Diamond_{\{\mathsf{mi}\}}\Phi$ specify that $\Phi$ happened in the noncontiguous past, noncontiguous future, contiguous past, or contiguous future respectively. The $\Diamond_R$ operator can also be used to derive overlapped occurrence from coincidental occurrence. An occurrence of $\Diamond_{\{=,\mathsf{o},\mathsf{oi},\mathsf{s},\mathsf{si},\mathsf{f},\mathsf{fi},\mathsf{d},\mathsf{di}\}}\Phi$ coincides with $\mathbf{i}$ if an occurrence of $\Phi$ overlaps with $\mathbf{i}$. I abbreviate $\Diamond_{\{=,\mathsf{o},\mathsf{oi},\mathsf{s},\mathsf{si},\mathsf{f},\mathsf{fi},\mathsf{d},\mathsf{di}\}}\Phi$ simply as $\Diamond\Phi$ without any subscript. Note that while $(\neg\Phi)@\mathbf{i}$ indicates that no occurrence of $\Phi$ *coincided* with $\mathbf{i}$, $(\neg\Diamond\Phi)@\mathbf{i}$ indicates that no occurrence of $\Phi$ *overlapped* with $\mathbf{i}$.

- $\Phi^+$ denotes repeated contiguous occurrence of $\Phi$.

Formally, the truth of an event-logic proposition $\Phi@\mathbf{i}$ is defined relative to a model. Let $I$ be the set of all intervals. A model $M$ is a triple $\langle O, T, P\rangle$ where $O$ is a set of objects, $T$ is a map from constants and variables to objects from $O$, and $P$ is map from predicate symbols of arity $n$ to subsets of $I \times \underbrace{O \times \cdots \times O}_{n}$. ($M$ can be viewed as either a model or as a movie.) $P$ thus maps predicate symbols to relations that take an interval as their first argument, in addition to the remaining object parameters. $T[x := o]$ denotes a map that is identical to $T$ except that it maps the variable $x$ to the object $o$. A model is *finite* if $O$ is finite, $P$ is finite, and each relation in the range of $P$ is finite. The *span* of a model is the smallest super-interval of all of the intervals that appear in any element of any relation in the range of $P$. The span of a model may be undefined if the model is infinite, but will always be defined if the model is finite. Finally, a model is *integral* if all of the intervals that appear in any element of any relation in the range of $P$ are integral.

The semantics of event logic is formally defined by specifying an entailment relation $M \models \Phi@\mathbf{i}$ as follows:

- $\langle O, T, P\rangle \models p(t_1, \ldots, t_n)@\mathbf{i}$ if and only if $\langle \mathbf{i}, T(t_1), \ldots, T(t_n)\rangle \in P(p)$.

- $M \models (\neg\Phi)@\mathbf{i}$ if and only if $M \not\models \Phi@\mathbf{i}$.

- $M \models (\Phi \vee \Psi)@\mathbf{i}$ if and only if $M \models \Phi@\mathbf{i}$ or $M \models \Psi@\mathbf{i}$.

- $\langle O, T, P\rangle \models (\forall x\Phi)@\mathbf{i}$ if and only if $\langle O, T[x := o], P\rangle \models \Phi@\mathbf{i}$ for every $o \in O$.

- $\langle O, T, P\rangle \models (\exists x\Phi)@\mathbf{i}$ if and only if $\langle O, T[x := o], P\rangle \models \Phi@\mathbf{i}$ for some $o \in O$.

- $M \models (\Phi \wedge_R \Psi)@\mathbf{i}$ if and only if there exist two intervals $\mathbf{j}$ and $\mathbf{k}$ such that $\mathbf{i} = \textsc{Span}(\mathbf{j}, \mathbf{k})$, $\mathbf{j} r \mathbf{k}$ for some $r \in R$, $M \models \Phi@\mathbf{j}$, and $M \models \Psi@\mathbf{k}$.

- $M \models (\Diamond_R\Phi)@\mathbf{i}$ if and only if there exists some interval $\mathbf{j}$ such that $\mathbf{j} r \mathbf{i}$ for some $r \in R$, and $M \models \Phi@\mathbf{j}$.

- $M \models (\Phi^+)@\mathbf{i}$ if and only if either $M \models \Phi@\mathbf{i}$, or there exist two intervals $\mathbf{j}$ and $\mathbf{k}$ such that $\mathbf{i} = \textsc{Span}(\mathbf{j}, \mathbf{k})$, $\mathbf{j}\mathsf{m}\mathbf{k}$, $M \models \Phi@\mathbf{j}$, and $M \models (\Phi^+)@\mathbf{k}$.

### 4.2. The Problem

In Abigail, the lower-level process described in the earlier sections of this chapter produces a finite model $M$ delineating the observed objects and the observed primitive event occurrences. If $M$ is constructed in this fashion from a movie with discrete frames, $M$ will also be integral. Given a particular ground event-logic expression $\Phi$, a particular integral interval $\mathbf{i}$, and a particular finite integral model $M$, one can efficiently compute whether or not $M \models \Phi@\mathbf{i}$ by straightforward application of the semantic definition. One can represent a finite integral model $M$ as a finite set of propositions $\Phi@\mathbf{i}$, where $\Phi$ is a ground atomic event-logic expression. The model thus delineates all primitive event occurrences. Let $C(M)$ be the set of all constants that appear in $M$. Furthermore, let $\Phi[x := c]$ denote the formula that results by replacing all free occurrences of the variable $x$ in $\Phi$ with the constant $c$. With such a representation, one can define the following procedure $\mathcal{V}(M, \Phi, \mathbf{i})$ to determine whether or not $M \models \Phi@\mathbf{i}$:

$$
\begin{aligned}
\mathcal{V}(M, p(c_1, \ldots, c_n), \mathbf{i}) &\triangleq p(c_1, \ldots, c_n)@\mathbf{i} \in M \\
\mathcal{V}(M, \neg\Phi, \mathbf{i}) &\triangleq \neg\mathcal{V}(M, \Phi, \mathbf{i}) \\
\mathcal{V}(M, \Phi \vee \Psi, \mathbf{i}) &\triangleq \mathcal{V}(M, \Phi, \mathbf{i}) \vee \mathcal{V}(M, \Psi, \mathbf{i}) \\
\mathcal{V}(M, \forall x \Phi, \mathbf{i}) &\triangleq \bigwedge_{c \in C(M)} \mathcal{V}(M, \Phi[x := c], \mathbf{i}) \\
\mathcal{V}(M, \exists x \Phi, \mathbf{i}) &\triangleq \bigvee_{c \in C(M)} \mathcal{V}(M, \Phi[x := c], \mathbf{i}) \\
\mathcal{V}(M, \Phi \wedge_R \Psi, \mathbf{i}) &\triangleq \bigvee_{\substack{\mathbf{j} \in \textsc{Sub}(\mathbf{i}) \\ \mathbf{j} \text{ integral}}} \bigvee_{\substack{\mathbf{k} \in \textsc{Sub}(\mathbf{i}) \\ \mathbf{k} \text{ integral}}} \left[\mathbf{i} = \textsc{Span}(\mathbf{j}, \mathbf{k}) \wedge \mathcal{V}(M, \Phi, \mathbf{j}) \wedge \mathcal{V}(M, \Psi, \mathbf{k}) \wedge \bigvee\nolimits_{r \in R} \mathbf{j} r \mathbf{k}\right] \\
\mathcal{V}(M, \Diamond_R \Phi, \mathbf{i}) &\triangleq \bigvee_{\substack{\mathbf{j} \in \textsc{Ext}([-\infty : \infty]) \\ \mathbf{j} \text{ integral}}} \left[\mathcal{V}(M, \Phi, \mathbf{j}) \wedge \bigvee_{r \in R} \mathbf{j} r \mathbf{i}\right] \\
\mathcal{V}(M, \Phi^+, \mathbf{i}) &\triangleq \mathcal{V}(M, \Phi, \mathbf{i}) \vee \bigvee_{\substack{\mathbf{j} \in \textsc{Sub}(\mathbf{i}) \\ \mathbf{j} \text{ integral}}} \bigvee_{\substack{\mathbf{k} \in \textsc{Sub}(\mathbf{i}) \\ \mathbf{k} \text{ integral}}} \left[\mathbf{i} = \textsc{Span}(\mathbf{j}, \mathbf{k}) \wedge \mathbf{j}\mathsf{m}\mathbf{k} \wedge \mathcal{V}(M, \Phi, \mathbf{j}) \wedge \mathcal{V}(M, \Phi^+, \mathbf{k})\right]
\end{aligned}
$$

**Claim 1** *For any finite integral model $M$ and any integral interval* $\mathbf{i}$*:* $\mathcal{V}(M,\Phi,\mathbf{i})$ *yields true if and only if* $M \models \Phi@\mathbf{i}$.

Often however, we are given neither a particular ground event-logic expression $\Phi$ nor a particular interval $\mathbf{i}$. Instead, we are given a set $E$ of event-logic expressions constituting a lexicon of event types whose occurrences we wish to detect. Figure 3 is an example of such a set. The expressions in $E$ potentially contain free variables. Given some finite integral model $M$, we wish to find all ground instantiations $\Phi$, of all expressions $e$ in $E$ (by consistently replacing all free variables in $e$ with constants), along with all integral subintervals $\mathbf{i}$ of the span of $M$, such that $M \models \Phi@\mathbf{i}$. The naive way to do this would be via the following algorithm:

ALGORITHM 1:

> **for** $e \in E$
> **do for** $\Phi$ being some ground instantiation of $e$
>     **do for** $\mathbf{i} \in \textsc{Ext}([-\infty : \infty]), \mathbf{i}$ integral
>         **do if** $\mathcal{V}(M,\Phi,\mathbf{i})$
>             **then print** $M \models \Phi@\mathbf{i}$ **fi od od od**

While this method is straightforward and correct, it is very inefficient in practise. The remainder of this section addresses improved methods for computing this same information.

### 4.3. The Algorithm

One source of inefficiency in the naive algorithm lies in the representation of primitive event occurrences in the model. In many applications, among them ABIGAIL, the primitive event types are liquid. That means that if $M$ contains some primitive event occurrence $\Phi@\mathbf{i}$, then it also contains all primitive event occurrences $\Phi@\mathbf{j}$ where $\mathbf{j}$ is a subinterval of $\mathbf{i}$.[2] For finite integral models, there are quadratically-many such subintervals. Such information is essentially redundant. Instead of storing all such primitive event occurrences in the form $\Phi@[i,j]$, one can store a single denotation $\Phi@[i:j]$ to indicate that an occurrence of $\Phi$ coincided with the interval $[i,j]$ and all of its subintervals. In general, one can extend the notation for event occurrence to include propositions of the form $\Phi@I$, where $I$ is a set of intervals, to indicate that occurrences of $\Phi$ coincided with each interval in $I$. Adopting this extension, $[i:j]$ then becomes nothing but shorthand for the set of all subintervals of $[i,j]$. I refer to such a concise representation of a set of intervals as a *spanning interval*.

More specifically, a spanning interval is a pair of real numbers $[i:j]$ such that $i \le j$. A spanning interval $[i:j]$ denotes a set of intervals, namely the set of all of the subintervals of $[i,j]$. A set $I$ of spanning intervals denotes the union of the

[2] This is true even if $M$ is not integral or not finite.

sets of spanning intervals denoted by the members of $I$. Note that the spanning interval $[i:i]$ denotes a set containing a single interval, namely $\{[i,i]\}$, and not the empty set of intervals. Thus I define $\emptyset$, the *empty spanning interval*, to denote the empty set of intervals. Finally, I define the *expansion* of an interval $[i,j]$, denoted $\textsc{Expand}([i,j])$, to be $[i:j]$.

The set of intervals denoted by a spanning interval, a set of spanning intervals, or the empty spanning interval, is its *extension*:

$$\begin{array}{rcl}\textsc{Ext}([i:j]) & \triangleq & \textsc{Sub}([i,j])\\ \textsc{Ext}(I) & \triangleq & \bigcup_{\mathbf{i}\in I}\textsc{Ext}(\mathbf{i})\\ \textsc{Ext}(\emptyset) & \triangleq & \{\}\end{array}$$

Furthermore, a spanning interval $\mathbf{i}$ *subsumes* a spanning interval $\mathbf{j}$ if $\textsc{Ext}(\mathbf{j}) \subseteq \textsc{Ext}(\mathbf{i})$. Finally, a set $I$ of spanning intervals is *redundant* if it contains either the empty spanning interval $\emptyset$, or two spanning intervals $\mathbf{i}$ and $\mathbf{j}$ such that $\mathbf{i}$ subsumes $\mathbf{j}$. A set of spanning intervals can be made irredundant, without changing its extension, simply by removing empty and subsumed spanning intervals.

**Claim 2** *For all real numbers $i$, $j$, $k$, and $l$ such that $i \le j$ and $k \le l$:* $\textsc{Ext}([i:j]) \subseteq \textsc{Ext}([k:l])$ *if and only if $i \le k$ and $j \le l$.*

Thus one can efficiently compute the subsumption relation between two spanning intervals.

Instead of storing in $M$, propositions of the form $\Phi@\mathbf{i}$ where $\mathbf{i}$ is an interval, propositions of the form $\Phi@I$ where $I$ is a set of intervals, or even propositions of the form $\Phi@\mathbf{i}$ where $\mathbf{i}$ is a spanning interval, one can store propositions of the form $\Phi@I$ where $I$ is a set of spanning intervals. By merging propositions $\Phi@I$ and $\Phi@J$ into $\Phi@(I \cup J)$ in the model, and by storing in the model, only propositions $\Phi@I$ where $I$ is irredundant, the size of a finite integral model can be reduced quadratically.

One can compute the *intersection* $\mathbf{i} \cap \mathbf{j}$ of two spanning intervals $\mathbf{i}$ and $\mathbf{j}$ as follows:

$$[i:j]\cap[k:l] \triangleq \left\{\begin{array}{ll}[\max(i,k):\min(j,l)] & \max(i,k) \le \min(j,l)\\ \emptyset & \text{otherwise}\end{array}\right.$$

**Claim 3** *For all spanning intervals $\mathbf{i}$ and $\mathbf{j}$:* $\textsc{Ext}(\mathbf{i}\cap\mathbf{j}) = \textsc{Ext}(\mathbf{i}) \cap \textsc{Ext}(\mathbf{j})$.

Spanning intervals thus have the property that the intersection of the extensions of two spanning intervals can be represented as a *single* spanning interval. This property makes them useful for event perception and temporal inference.

Many aggregate event types are also liquid. Consider, for example, a simplified variant of the falling event defined as follows:

$$\textsc{Fall}(x) \triangleq \textsc{Unsupported}(x) \wedge \textsc{TranslatingDown}(x)$$

If the model contained

$\textsc{Unsupported}(\mathbf{ball})@[1:10]$ and $\textsc{TranslatingDown}(\mathbf{ball})@[1:10]$,

an occurrence of $\textsc{Fall}(\mathbf{ball})$ would coincide with every subinterval of $[1, 10]$. The naive algorithm would explicitly compute all such event occurrences independently. This seems redundant. It would be desirable to use spanning intervals to represent aggregate event occurrences, just as for primitive event occurrences. Doing so could reduce both the space and time requirements of the event perception task. For example, if the model instead contained $\textsc{Unsupported}(\mathbf{ball})@[1:10]$ and $\textsc{TranslatingDown}(\mathbf{ball})@[5:15]$, an appropriate algorithm could intersect the spanning intervals and compute $\textsc{Fall}(\mathbf{ball})@[5:10]$ in a single analytical operation.

Accordingly, we desire a procedure $\mathcal{E}(M, \Phi)$ that computes the set of all intervals $\mathbf{i}$ such that $M \models \Phi@\mathbf{i}$. Ideally, $\mathcal{E}(M, \Phi)$ would return such a set represented concisely as a small irredundant set of spanning intervals. Furthermore, $\mathcal{E}(M, \Phi)$ should compute internally with such a concise representation. With such a procedure, one can rewrite the event perception procedure as follows:

Algorithm 2:

**for** $e \in E$
**do for** $\Phi$ being some ground instantiation of $e$
  **do if** $\mathcal{E}(M, \Phi) \neq \{\}$
    **then print** $M \models \Phi@\mathcal{E}(M, \Phi)$ **fi od od**

An attempt to define $\mathcal{E}(M, \Phi)$ yields the following:

$$\mathcal{E}(M, p(c_1, \ldots, c_n)) \triangleq \begin{cases} I & p(c_1, \ldots, c_n)@I \in M \\ \{\} & \text{otherwise} \end{cases}$$

$$\mathcal{E}(M, \Phi \vee \Psi) \triangleq \mathcal{E}(M, \Phi) \cup \mathcal{E}(M, \Psi)$$

$$\mathcal{E}(M, \forall x \Phi) \triangleq \bigcup_{\mathbf{i}_1 \in \mathcal{E}(M, \Phi[x:=c_1])} \cdots \bigcup_{\mathbf{i}_n \in \mathcal{E}(M, \Phi[x:=c_n])} \{\mathbf{i}_1 \cap \cdots \cap \mathbf{i}_n\} \quad \text{where } C(M) = \{c_1, \ldots, c_n\}$$

$$\mathcal{E}(M, \exists x \Phi) \triangleq \bigcup_{c \in C(M)} \mathcal{E}(M, \Phi[x := c])$$

This definition has the property that each recursive call to $\mathcal{E}(M, \Phi)$ yields a set of spanning intervals. This set might be redundant. It can be made irredundant simply by removing all empty and subsumed spanning intervals.

Unfortunately, it is not possible to define $\mathcal{E}(M, \Phi)$ for all event-logic combining forms and maintain the property that $\mathcal{E}(M, \Phi)$ always yield a set of spanning intervals. In particular, $\mathcal{E}(M, \neg\Phi)$, $\mathcal{E}(M, \Phi \wedge_R \Psi)$, $\mathcal{E}(M, \Diamond_R \Phi)$, and $\mathcal{E}(M, \Phi^+)$ might not be representable as a set of spanning intervals, even when $\mathcal{E}(M, \Phi)$ and $\mathcal{E}(M, \Psi)$ are representable as such a set. This can be demonstrated by the following simple examples. Suppose that $\Phi@[1:10]$ were true. Then $(\neg\Phi)@[1,15]$ could be true even though $(\neg\Phi)@[5,10]$ would not be true. Similarly, suppose that $\Phi@[1:10]$ and $\Psi@[5:15]$ were true. Then $(\Phi \wedge_{\mathsf{m}} \Psi)@[1,15]$ would be true even though $(\Phi \wedge_{\mathsf{m}} \Psi)@[1,4]$ would not be. Likewise, suppose that $\Phi@[1:10]$ were true. Then $(\Diamond_{\mathsf{m}}\Phi)@[5,15]$ would be true even though $(\Diamond_{\mathsf{m}}\Phi)@[6,15]$ would not be. Finally, suppose that $\Phi@[1,10]$ and $\Phi@[10,20]$ were true, but that no occurrence of $\Phi$ coincided with any subintervals of $[1,10]$ or $[10,20]$. Then $\Phi^{+}@[1,20]$ would be true even though $\Phi^{+}@[5,15]$ would not be. It is easy to show that $\Phi \wedge_{\{r\}} \Psi$ and $\Diamond_{\{r\}}\Phi$ can yield sets of intervals that are not representable as sets of spanning intervals for all individual Allen relations $r$ except for $=$.

The above problems can be alleviated by defining the notion of an *extended spanning interval*. Before doing so, I will need to define the auxiliary notion of extended lower and upper bounds. An *extended lower bound* is either $x$, $x+\epsilon$, or $-\infty$, where $x$ is a real number. Similarly, an *extended upper bound* is either $x$, $x-\epsilon$, or $\infty$, where $x$ is a real number. The unary functions $+\epsilon$ and $-\epsilon$ are defined on lower and upper bounds respectively, as follows:

$$\begin{array}{rcl@{\qquad\qquad}rcl} x+\epsilon & \Rightarrow & x+\epsilon & x-\epsilon & \Rightarrow & x-\epsilon \\ x+\epsilon+\epsilon & \Rightarrow & x+\epsilon & x-\epsilon-\epsilon & \Rightarrow & x-\epsilon \\ -\infty+\epsilon & \Rightarrow & -\infty & \infty-\epsilon & \Rightarrow & \infty \end{array}$$

The results of applying $+\epsilon$ to an extended upper bound, or $-\epsilon$ to an extended lower bound, are undefined. The ordering relations are then defined so that $x \geq y+\epsilon$ implies $x > y$ and $x \leq y-\epsilon$ implies $x < y$. The $=$ relation and the functions min and max are defined appropriately for extended lower and upper bounds. Finally, the unary functions $\neg_{+}$ and $\neg_{-}$ are defined on lower and upper bounds respectively as follows:

$$\begin{array}{rcl@{\qquad\qquad}rcl} \neg_{+}x & \Rightarrow & x-\epsilon & \neg_{-}x & \Rightarrow & x+\epsilon \\ \neg_{+}x+\epsilon & \Rightarrow & x & \neg_{-}x-\epsilon & \Rightarrow & x \end{array}$$

The results of applying $\neg_{+}$ to $-\infty$ or an extended upper bound, or $\neg_{-}$ to $\infty$ or an extended lower bound, are undefined.

An extended spanning interval is a 6-tuple $[i:j,k:l]^{\alpha}_{\beta}$, where $i$ and $k$ are extended lower bounds, $j$ and $l$ are extended upper bounds, and $\alpha$ and $\beta$ are Boolean values. An extended spanning interval $[i:j,k:l]^{\alpha}_{\beta}$ denotes the set of all intervals $[m,n]$ such that $i \leq m \leq j$, $k \leq n \leq l$, $m \leq n$, $m \neq n$ if $\alpha$ is false, and $m \not< n$ if $\beta$ is false. In other words, $\alpha$ being false excludes instantaneous intervals while $\beta$ being false excludes noninstantaneous intervals. The notions of extension, subsumption, and irredundancy are suitably generalized both to extended spanning intervals and to sets of extended spanning intervals. Spanning intervals are special cases of extended spanning intervals. The spanning interval $[i:j]$ has the same extension as the extended spanning interval $[i:j,i:j]^{\mathbf{T}}_{i\neq j}$. Also note that while an interval $[i,j]$ cannot be represented as a spanning interval when $i \neq j$, it can be represented as the extended spanning interval $[i:i,j:j]^{i=j}_{i\neq j}$.

An extended spanning interval $[i:j,k:l]^{\alpha}_{\beta}$ is *normalized* if $i \leq j \leq l$, $i \leq k \leq l$, $\alpha \leftrightarrow \max(k,i) \leq \min(j,l)$, and $\beta \leftrightarrow i \neq l$. Being normalized implies that it is nonempty, and furthermore, $\alpha$ and $\beta$ are true if and only if it has instants and noninstants in its extension respectively. Let us define the following normalization operator:

$$\langle i:j,k:l\rangle^{\alpha}_{\beta} \triangleq \begin{cases} [i:\min(j,l),\max(k,i):l]^{\max(k,i)\leq\min(j,l)}_{i\neq l} \\ \quad \alpha \wedge \beta \wedge i \leq \min(j,l) \wedge \max(k,i) \leq l \\ [\max(k,i):\min(j,l),\max(k,i):\min(j,l)]^{\mathbf{T}}_{\mathbf{F}} \\ \quad \alpha \wedge \neg\beta \wedge \max(k,i) \leq \min(j,l) \\ [i:\min(j,l-\epsilon),\max(k,i+\epsilon):l]^{\mathbf{F}}_{\mathbf{T}} \\ \quad \neg\alpha \wedge \beta \wedge i \leq \min(j,l-\epsilon) \wedge \max(k,i+\epsilon) \leq l \\ \emptyset \quad \text{otherwise} \end{cases}$$

**Claim 4** *For all extended lower bounds $i$ and $k$, all extended upper bounds $j$ and $l$, and all Boolean values $\alpha$ and $\beta$:* $\textsc{Ext}(\langle i:j,k:l\rangle^{\alpha}_{\beta}) = \textsc{Ext}([i:j,k:l]^{\alpha}_{\beta})$ *and $\langle i:j,k:l\rangle^{\alpha}_{\beta}$ is normalized if it is nonempty.*

Thus extended spanning intervals can be normalized in constant time without any increase in storage requirements. Furthermore, it is possible to determine by inspection whether or not a normalized extended spanning interval has any instants or noninstants in its extension, and it is possible to discard those extended spanning intervals that have no extension. The algorithms discussed in this section always construct and maintain normalized extended spanning intervals.

Normalization also allows one to efficiently compute the subsumption relation between two extended spanning intervals.

**Claim 5** *For all normalized extended spanning intervals $[i_1:j_1,k_1:l_1]^{\alpha_1}_{\beta_1}$ and $[i_2:j_2,k_2:l_2]^{\alpha_2}_{\beta_2}$:* $\textsc{Ext}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\beta_1}) \subseteq \textsc{Ext}([i_2:j_2,k_2:l_2]^{\alpha_2}_{\beta_2})$ *if and only if $i_1 \geq i_2$, $j_1 \leq j_2$, $k_1 \geq k_2$, $l_1 \leq l_2$, $\alpha_1 \rightarrow \alpha_2$, and $\beta_1 \rightarrow \beta_2$.*

Furthermore, one can compute the *intersection* $\mathbf{i} \cap \mathbf{j}$ of two normalized extended spanning intervals $\mathbf{i}$ and $\mathbf{j}$ as follows:

$$[i_1:j_1,k_1:l_1]^{\alpha_1}_{\beta_1} \cap [i_2:j_2,k_2:l_2]^{\alpha_2}_{\beta_2} \triangleq \langle \max(i_1,i_2):\min(j_1,j_2),\max(k_1,k_2):\min(l_1,l_2)\rangle^{\alpha_1\wedge\alpha_2}_{\beta_1\wedge\beta_2}$$

**Claim 6** *For all normalized extended spanning intervals $\mathbf{i}$ and $\mathbf{j}$:* $\textsc{Ext}(\mathbf{i} \cap \mathbf{j}) = \textsc{Ext}(\mathbf{i}) \cap \textsc{Ext}(\mathbf{j})$.

Extended spanning intervals thus maintain the property that the intersection of the extensions of two extended spanning intervals can be represented as a *single* extended spanning interval. Thus the existing cases in the definition of $\mathcal{E}(M,\Phi)$ can be reinterpreted without change as computing extended spanning intervals instead of spanning intervals.

Extended spanning intervals, however, allow filling in the remaining cases of $\mathcal{E}(M,\Phi)$. Before filling in these remaining cases, we need four more definitions. First, one can compute the *complement* $\neg\mathbf{i}$ of a normalized extended spanning interval $\mathbf{i}$ as follows:

$$\neg[i:j,k:l]^{\alpha}_{\beta} \triangleq \left\{\begin{array}{ll} \langle -\infty:\infty,-\infty:\neg_{+}k\rangle^{\mathbf{T}}_{\mathbf{T}}, & \text{include when } k\neq-\infty \\ \langle -\infty:\infty,\neg_{-}l:\infty\rangle^{\mathbf{T}}_{\mathbf{T}}, & \text{include when } l\neq\infty \\ \langle -\infty:\neg_{+}i,-\infty:\infty\rangle^{\mathbf{T}}_{\mathbf{T}}, & \text{include when } i\neq-\infty \\ \langle \neg_{-}j:\infty,-\infty:\infty\rangle^{\mathbf{T}}_{\mathbf{T}}, & \text{include when } j\neq\infty \\ \langle -\infty:\infty,-\infty:\infty\rangle^{\neg\alpha}_{\neg\beta} & \text{always include} \end{array}\right\}$$

**Claim 7** *For any normalized extended spanning interval* $\mathbf{i}$ *and for any interval* $\mathbf{j}$*:* $\mathbf{j}\in\textsc{Ext}(\neg\mathbf{i})$ *if and only if* $\mathbf{j}\notin\textsc{Ext}(\mathbf{i})$.

Extended spanning intervals have the property that the complement of their extension can be represented as a set containing at most *five* extended spanning intervals.

Second, one can generalize the notion of span to two normalized extended spanning intervals as follows:

$$\begin{array}{l} \textsc{Span}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\mathbf{T}},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq \\ \quad \{\langle \min(i_1,i_2):\min(j_1,j_2),\max(k_1,k_2):\max(l_1,l_2)\rangle^{\alpha_1\wedge\alpha_2}_{\mathbf{T}}\} \\ \textsc{Span}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\mathbf{T}},[i_2:j_2,i_2:j_2]^{\mathbf{T}}_{\mathbf{F}}) \triangleq \\ \quad \left\{\begin{array}{l} \langle \min(i_1,i_2):\min(j_1,j_2),\max(k_1,i_2):l_1\rangle^{\alpha_1}_{\mathbf{T}}, \\ \langle i_1:\min(j_1,j_2),\max(k_1,i_2):\max(l_1,j_2)\rangle^{\alpha_1}_{\mathbf{T}} \end{array}\right\} \\ \textsc{Span}([i_1:j_1,i_1:j_1]^{\mathbf{T}}_{\mathbf{F}},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq \\ \quad \left\{\begin{array}{l} \langle \min(i_1,i_2):\min(j_1,j_2),\max(i_1,k_2):l_2\rangle^{\alpha_2}_{\mathbf{T}}, \\ \langle i_2:\min(j_1,j_2),\max(i_1,k_2):\max(j_1,l_2)\rangle^{\alpha_2}_{\mathbf{T}} \end{array}\right\} \\ \textsc{Span}([i_1:j_1,i_1:j_1]^{\mathbf{T}}_{\mathbf{F}},[i_2:j_2,i_2:j_2]^{\mathbf{T}}_{\mathbf{F}}) \triangleq \\ \quad \left\{\begin{array}{l} \langle i_1:j_1,i_2:j_2\rangle^{\mathbf{T}}_{\mathbf{T}}, \\ \langle i_2:j_2,i_1:j_1\rangle^{\mathbf{T}}_{\mathbf{T}} \end{array}\right\} \end{array}$$

**Claim 8** *For all normalized extended spanning intervals* $\mathbf{i}$ *and* $\mathbf{j}$*:*

$$\textsc{Ext}(\textsc{Span}(\mathbf{i},\mathbf{j})) = \bigcup_{\mathbf{i}'\in\textsc{Ext}(\mathbf{i})}\ \bigcup_{\mathbf{j}'\in\textsc{Ext}(\mathbf{j})} \{\textsc{Span}(\mathbf{i}',\mathbf{j}')\}.$$

Extended spanning intervals have the property that the set of all intervals that span two intervals in the respective extensions of two extended spanning intervals can be represented as a set containing at most *two* extended spanning intervals.

Third, we need to define the function $\mathcal{I}(\mathbf{j},r,\mathbf{k})$. This function takes, as input, two normalized extended spanning intervals $\mathbf{j}$ and $\mathbf{k}$, along with an Allen relation $r$. It produces, as output, a set $I$ of extended spanning intervals, such that for all intervals $\mathbf{i}$ in the extension of $I$, there exist some intervals $\mathbf{j}'$ and $\mathbf{k}'$ in the extensions of $\mathbf{j}$ and $\mathbf{k}$ respectively, such that $\mathbf{j}'r\mathbf{k}'$ and $\mathbf{i}=\textsc{Span}(\mathbf{j}',\mathbf{k}')$:

$$\mathcal{I}(\mathbf{j},=,\mathbf{k}) \triangleq \{\mathbf{j}\cap\mathbf{k}\}$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\beta_1},<,[i_2:j_2,k_2:l_2]^{\alpha_2}_{\beta_2}) \triangleq$$
$$\textsc{ExcludeInstants}\left(\textsc{Span}(\langle i_1:j_1,k_1:\min(l_1,j_2-\epsilon)\rangle^{\mathbf{T}}_{\mathbf{T}},\langle\max(k_1+\epsilon,i_2):j_2,k_2:l_2\rangle^{\mathbf{T}}_{\mathbf{T}})\right)$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\mathbf{T}},\mathsf{m},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq$$
$$\textsc{Span}(\langle i_1:j_1,\max(k_1,i_2):\min(l_1,j_2)\rangle^{\mathbf{T}}_{\mathbf{T}},\langle\max(k_1,i_2):\min(l_1,j_2),k_2:l_2\rangle^{\mathbf{T}}_{\mathbf{T}})$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\mathbf{T}},\mathsf{m},[i_2:j_2,i_2:j_2]^{\mathbf{T}}_{\mathbf{F}}) \triangleq \{\langle i_1:j_1,\max(k_1,i_2):\min(l_1,j_2)\rangle^{\mathbf{T}}_{\mathbf{T}}\}$$

$$\mathcal{I}([i_1:j_1,i_1:j_1]^{\mathbf{T}}_{\mathbf{F}},\mathsf{m},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq \{\langle\max(i_1,i_2):\min(j_1,j_2),k_2:l_2\rangle^{\mathbf{T}}_{\mathbf{T}}\}$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\mathbf{F}}_{\mathbf{F}},\mathsf{m},[i_2:j_2,k_2:l_2]^{\mathbf{F}}_{\mathbf{F}}) \triangleq$$
$$\{\langle\max(i_1,i_2):\min(j_1,j_2),\max(k_1,k_2):\min(l_1,l_2)\rangle^{\mathbf{T}}_{\mathbf{T}}\}$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\mathbf{T}},\mathsf{o},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq$$
$$\textsc{Span}\left(\begin{array}{l}\langle i_1:\min(j_1,j_2-\epsilon),\max(k_1,i_2+\epsilon):\min(l_1,l_2-\epsilon)\rangle^{\mathbf{F}}_{\mathbf{T}},\\ \langle\max(i_1+\epsilon,i_2):\min(l_1-\epsilon,j_2),\max(k_1+\epsilon,k_2):l_2\rangle^{\mathbf{F}}_{\mathbf{T}}\end{array}\right)$$

$$\mathcal{I}([i_1:j_1,i_1:j_1]^{\mathbf{T}}_{\mathbf{F}},\mathsf{o},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\beta_2}) \triangleq \emptyset$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\beta_1},\mathsf{o},[i_2:j_2,i_2:j_2]^{\mathbf{T}}_{\mathbf{F}}) \triangleq \emptyset$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\mathbf{T}},\mathsf{s},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq$$
$$\textsc{Span}\left(\begin{array}{l}\langle\max(i_1,i_2):\min(j_1,j_2),k_1:\min(l_1,l_2-\epsilon)\rangle^{\mathbf{T}}_{\mathbf{T}},\\ \langle\max(i_1,i_2):\min(j_1,j_2),\max(k_1+\epsilon,k_2):l_2\rangle^{\mathbf{F}}_{\mathbf{T}}\end{array}\right)$$

$$\mathcal{I}([i_1:j_1,i_1:j_1]^{\mathbf{T}}_{\mathbf{F}},\mathsf{s},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq \{\langle\max(i_1,i_2):\min(j_1,j_2),k_2:l_2\rangle^{\mathbf{F}}_{\mathbf{T}}\}$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\beta_1},\mathsf{s},[i_2:j_2,i_2:j_2]^{\mathbf{T}}_{\mathbf{F}}) \triangleq \emptyset$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\mathbf{T}},\mathsf{f},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq$$
$$\textsc{Span}\left(\begin{array}{l}\langle\max(i_1,i_2+\epsilon):j_1,\max(k_1,k_2):\min(l_1,l_2)\rangle^{\mathbf{T}}_{\mathbf{T}},\\ \langle i_2:\min(j_1-\epsilon,j_2),\max(k_1,k_2):\min(l_1,l_2)\rangle^{\mathbf{F}}_{\mathbf{T}}\end{array}\right)$$

$$\mathcal{I}([i_1:j_1,i_1:j_1]^{\mathbf{T}}_{\mathbf{F}},\mathsf{f},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq \{\langle i_2:j_2,\max(i_1,k_2):\min(j_1,l_2)\rangle^{\mathbf{F}}_{\mathbf{T}}\}$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\beta_1},\mathsf{f},[i_2:j_2,i_2:j_2]^{\mathbf{T}}_{\mathbf{F}}) \triangleq \emptyset$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\mathbf{T}},\mathsf{d},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq$$
$$\textsc{Span}\left(\begin{array}{l}\langle\max(i_1,i_2+\epsilon):j_1,k_1:\min(l_1,l_2-\epsilon)\rangle^{\mathbf{T}}_{\mathbf{T}},\\ \langle i_2:\min(j_1-\epsilon,j_2),\max(k_1+\epsilon,k_2):l_2\rangle^{\mathbf{F}}_{\mathbf{T}}\end{array}\right)$$

$$\mathcal{I}([i_1:j_1,i_1:j_1]^{\mathbf{T}}_{\mathbf{F}},\mathsf{d},[i_2:j_2,k_2:l_2]^{\alpha_2}_{\mathbf{T}}) \triangleq$$
$$\begin{cases}\{\langle i_2:\min(j_1,j_2),\max(i_1,k_2):l_2\rangle^{\mathbf{F}}_{\mathbf{T}}\} & i_2<i_1\wedge j_1<l_2\\ \emptyset & \text{otherwise}\end{cases}$$

$$\mathcal{I}([i_1:j_1,k_1:l_1]^{\alpha_1}_{\beta_1},\mathsf{d},[i_2:j_2,i_2:j_2]^{\mathbf{T}}_{\mathbf{F}}) \triangleq \emptyset$$

The definition for $\mathcal{I}(\mathbf{j},r,\mathbf{k})$, when $r$ is one of $>$, mi, oi, si, fi, or di, is simply the corresponding definition for $<$, m, o, s, f, or d, with the arguments **j** and **k** reversed.

**Claim 9** *For all normalized extended spanning intervals* **j** *and* **k**, *and for* $r$ *being one of Allen's relations:*

$$\textsc{Ext}(\mathcal{I}(\mathbf{j},r,\mathbf{k})) = \bigcup_{\mathbf{j}'\in\textsc{Ext}(\mathbf{j})}\ \bigcup_{\substack{\mathbf{k}'\in\textsc{Ext}(\mathbf{k})\\ \mathbf{j}'r\mathbf{k}'}}\{\textsc{Span}(\mathbf{j}',\mathbf{k}')\}.$$

Thus the set of all intervals that span two intervals in the respective extensions of two extended spanning intervals that are related by one of Allen's relations can be represented as a set containing at most *two* extended spanning intervals.

Finally, we need to define the function $\mathcal{D}(r,\mathbf{j})$. This function takes, as input, an Allen relation $r$ and a normalized extended spanning interval $\mathbf{j}$. It produces, as output, an extended spanning interval $\mathbf{i}$, such that for all intervals $\mathbf{i}'$ in the extension of $\mathbf{i}$, there exists some interval $\mathbf{j}'$ in the extension of $\mathbf{j}$, such that $\mathbf{j}'r\mathbf{i}'$:

$$
\begin{array}{rcl}
\mathcal{D}(=,\mathbf{j}) & \triangleq & \{\mathbf{j}\}\\
\mathcal{D}(<,[i:j,k:l]^{\alpha}_{\beta}) & \triangleq & \{\langle k+\epsilon:\infty,-\infty:\infty\rangle^{\mathbf{T}}_{\mathbf{T}}\}\\
\mathcal{D}(>,[i:j,k:l]^{\alpha}_{\beta}) & \triangleq & \{\langle -\infty:\infty,-\infty:j-\epsilon\rangle^{\mathbf{T}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{m},[i:j,k:l]^{\alpha}_{\beta}) & \triangleq & \{\langle k:l,-\infty:\infty\rangle^{\mathbf{T}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{mi},[i:j,k:l]^{\alpha}_{\beta}) & \triangleq & \{\langle -\infty:\infty,i:j\rangle^{\mathbf{T}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{o},[i:j,k:l]^{\alpha}_{\mathbf{T}}) & \triangleq & \{\langle i+\epsilon:l-\epsilon,k+\epsilon:\infty\rangle^{\mathbf{F}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{o},[i:j,i:j]^{\mathbf{T}}_{\mathbf{F}}) & \triangleq & \emptyset\\
\mathcal{D}(\mathsf{oi},[i:j,k:l]^{\alpha}_{\mathbf{T}}) & \triangleq & \{\langle -\infty:j-\epsilon,i+\epsilon:l-\epsilon\rangle^{\mathbf{F}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{oi},[i:j,i:j]^{\mathbf{T}}_{\mathbf{F}}) & \triangleq & \emptyset\\
\mathcal{D}(\mathsf{s},[i:j,k:l]^{\alpha}_{\beta}) & \triangleq & \{\langle i:j,k+\epsilon:\infty\rangle^{\mathbf{F}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{si},[i:j,k:l]^{\alpha}_{\mathbf{T}}) & \triangleq & \{\langle i:j,-\infty:l-\epsilon\rangle^{\mathbf{T}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{si},[i:j,i:j]^{\mathbf{T}}_{\mathbf{F}}) & \triangleq & \emptyset\\
\mathcal{D}(\mathsf{f},[i:j,k:l]^{\alpha}_{\beta}) & \triangleq & \{\langle -\infty:j-\epsilon,k:l\rangle^{\mathbf{F}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{fi},[i:j,k:l]^{\alpha}_{\mathbf{T}}) & \triangleq & \{\langle i+\epsilon:\infty,k:l\rangle^{\mathbf{T}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{fi},[i:j,i:j]^{\mathbf{T}}_{\mathbf{F}}) & \triangleq & \emptyset\\
\mathcal{D}(\mathsf{d},[i:j,k:l]^{\alpha}_{\beta}) & \triangleq & \{\langle -\infty:j-\epsilon,k+\epsilon:\infty\rangle^{\mathbf{F}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{di},[i:j,k:l]^{\alpha}_{\mathbf{T}}) & \triangleq & \{\langle i+\epsilon:\infty,-\infty:l-\epsilon\rangle^{\mathbf{T}}_{\mathbf{T}}\}\\
\mathcal{D}(\mathsf{di},[i:j,i:j]^{\mathbf{T}}_{\mathbf{F}}) & \triangleq & \emptyset
\end{array}
$$

**Claim 10** *For all extended spanning intervals* $\mathbf{j}$ *and for* $r$ *being one of Allen's relations:*

$$
\textsc{Ext}(\mathcal{D}(r,\mathbf{j})) = \bigcup_{\mathbf{j}'\in\textsc{Ext}(\mathbf{j})} \; \bigcup_{\substack{\mathbf{i}\in\textsc{Ext}([-\infty:\infty])\\ \mathbf{j}'r\mathbf{i}}} \{\mathbf{i}\}.
$$

Thus the set of all intervals related to some interval in the extension of an extended spanning interval by one of Allen's relations can be represented as a *single* extended spanning interval.

We can now fill in the remaining cases of $\mathcal{E}(M,\Phi)$:

$$\begin{aligned}
\mathcal{E}(M,\neg\Phi) &\triangleq \bigcup_{\mathbf{i}'_1\in\neg\mathbf{i}_1}\cdots\bigcup_{\mathbf{i}'_n\in\neg\mathbf{i}_n}\{\mathbf{i}'_1\cap\cdots\cap\mathbf{i}'_n\}\\
&\quad\text{where } \mathcal{E}(M,\Phi)=\{\mathbf{i}_1,\ldots,\mathbf{i}_n\}\\
\mathcal{E}(M,\Phi\wedge_R\Psi) &\triangleq \bigcup_{\mathbf{j}\in\mathcal{E}(M,\Phi)}\bigcup_{\mathbf{k}\in\mathcal{E}(M,\Psi)}\bigcup_{r\in R}\mathcal{I}(\mathbf{j},r,\mathbf{k})\\
\mathcal{E}(M,\Diamond_R\Phi) &\triangleq \bigcup_{\mathbf{j}\in\mathcal{E}(M,\Phi)}\bigcup_{r\in R}\mathcal{D}(r,\mathbf{j})\\
\mathcal{E}(M,\Phi^+) &\triangleq \mathcal{E}(M,\Phi)\cup\bigcup_{\mathbf{j}\in\mathcal{E}(M,\Phi)}\bigcup_{\mathbf{k}\in\mathcal{E}(M,\Phi^+)}\mathcal{I}(\mathbf{j},\mathsf{m},\mathbf{k})
\end{aligned}$$

**Claim 11** *For any event-logic expression $\Phi$, any finite model $M$, and any interval $\mathbf{i}$: $M\models\Phi@\mathbf{i}$ if and only if $\mathbf{i}\in\textsc{Ext}(\mathcal{E}(M,\Phi))$.*

Thus algorithms 1 and 2 produce the same result. Algorithm 2 is, in fact, more powerful than algorithm 1. While algorithm 1 works only with integral models $M$ and integral intervals $\mathbf{i}$, algorithm 2 has no such restriction.

The key idea of this section is that extended spanning intervals are concise representations of quadratically-many integral intervals. Claims 4 through 11 demonstrate that the entailment relation for event logic is closed over the space of extended spanning intervals (though not for spanning intervals). Spanning intervals can only represent liquid events. In contrast, extended spanning intervals can represent nonliquid events in a way that preserves the indeterminacy of their start- and end- points. This section advances the empirical claim that all primitive event types, and many aggregate event types, are liquid. Occurrences of such liquid event types can be concisely represented as a *single* (extended) spanning interval. Furthermore, occurrences of all remaining nonliquid event types will be representable as small sets of extended spanning intervals.

## 5. Conclusion

I have described a comprehensive theory for representing the truth conditions for events described by ordinary simple spatial-motion verbs in a way that can be grounded in simulated visual input. These truth conditions are expressed as compound expressions in a particular event logic. The truth values of such compound events can be determined from the truth values of the primitive events used to construct the compound event expressions. Primitive events describing the changing support, contact, and attachment relations between participant objects are crucial for specifying the truth conditions of simple spatial-motion verbs. Such changing support, contact, and attachment relations are not directly perceivable quantities. They require an observer to project an ontology onto the world and use that ontology to construct a world model containing the unobservable information. This model reconstruction process uses counterfactual analysis to hypothesize unobservable information and adopt those hypotheses that are necessary to explain the stability of objects in the input movie. Stability can be determined by formulating the kinematic constraints between objects in the movie and reducing these constraints to a linear-programming problem.

The ideas in this chapter have been implemented and tested, with the exception of the $\Phi^+$ event-logic combining form. The code is available by anonymous FTP from the file `ftp.cs.toronto.edu:/pub/qobi/abigail.tar.Z`. At the time of this writing, the stability determination procedure described here has not been fully integrated into ABIGAIL. While the code to reduce a stability determination problem to a linear-programming problem is operational and has been tested using Maple, all of the publicly-available linear-programming packages that I have tried exhibit buggy behavior when presented with problems constructed by this reduction. The example output given in figures 2, 5, and 6 was generated using a version of ABIGAIL that incorporates the kinematic-simulation-based stability determination procedure described in [35].

I am currently trying to extend the ideas presented in this chapter to process real camera input instead of animated line drawings. This is a difficult task, primarily because current model-based pose estimation techniques cannot provide pose with sufficient accuracy to derive reliable stability judgments. My more recent work [39] has adopted a maximum-likelihood framework to deal with such inaccuracies. The Hidden Markov Models used within this framework can be viewed as stochastic variants of the event-logic expressions described in this paper. This new framework, however, does not use the notions of support, contact, and attachment to classify event types. Motivation for this new framework comes from experiments, along the lines of those conducted by Johansson[17, 18], that indicate that people can robustly classify simple spatial-motion events even when the input stimulus lacks sufficient information to determine support, contact, and attachment relations. While in many cases, coarse event classification might be successful without such information, it is my belief nonetheless, that such relations are central to specifying the underlying truth conditions for most simple spatial-motion verbs. Future work is planned to reconcile these two different approaches to visual event perception.

## REFERENCES

[1] Norihiro Abe, Itsuya Soga, and Saburo Tsuji. A plot understanding system on reference to both image and language. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pages 77–84, Vancouver, BC, August 1981.

[2] Norihiro Abe and Saburo Tsuji. A learning of object structures by verbalism. In J. Horecky, editor, *COLING 92*, pages 1–6. Elsevier North-Holland, 1982.

[3] Mark R. Adler. Computer interpretation of Peanuts cartoons. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, page 608, Cambridge, MA, August 1977.

[4] G. Adorni, A. Boccalatte, and M. Di Manzo. Cognitive models for computer vision. In J. Horecky, editor, *COLING 92*, pages 7–12. Elsevier North-Holland, 1982.

[5] James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, November 1983.

[6] Norman I. Badler. Temporal scene analysis: Conceptual descriptions of object movements. Technical Report 80, University of Toronto Department of Computer Science, February 1975.

[7] M. Blum, A. K. Griffith, and B. Neumann. A stability test for configurations of blocks. A. I. Memo 188, M. I. T. Artificial Intelligence Laboratory, February 1970.

[8] Gary C. Borchardt. A computer model for the representation and identification of physical events. Technical Report T-142, Coordinated Sciences Laboratory, University of Illinois at Urbana-Champaign, May 1984.

[9] Gary C. Borchardt. Event calculus. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 524–527, Los Angeles, CA, August 1985.

[10] David R. Dowty. *Word Meaning and Montague Grammar*. D. Reidel Publishing Company, Boston, MA, 1979.

[11] Scott Elliott Fahlman. A planning system for robot construction tasks. *Artificial Intelligence*, 5(1):1–49, 1974.

[12] Ellen M. Hays. On defining motion verbs and spatial prepositions. In Ch. Freksa and Ch. Habel, editors, *Repräsentation und Verarbeitung räumlichen Wissens*, pages 192–206. Springer-Verlag, 1989.

[13] Annette Herskovits. *Language and Spatial Cognition: An Interdisciplinary Study of the Prepositions in English*. Cambridge University Press, New York, NY, 1986.

[14] Ray Jackendoff. *Semantics and Cognition*. The MIT Press, Cambridge, MA, 1983.

[15] Ray Jackendoff. *Semantic Structures*. The MIT Press, Cambridge, MA, 1990.

[16] Ray Jackendoff and Barbara Landau. Spatial language and spatial cognition. In Donna Jo Napoli and Judy Anne Kegl, editors, *Bridges Between Psychology and Linguistics: A Swarthmore Festschrift for Lila Gleitman*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1991.

[17] G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14(2):201–211, 1973.

[18] G. Johansson. Visual motion perception. *Scientific American*, pages 76–88, November 1976.

[19] Manfred Krifka. Thematic relations as links between nominal reference and temporal constitution. In Ivan A. Sag and Anna Szabolcsi, editors, *Lexical Matters*. CSLI, 1992.

[20] Geoffrey N. Leech. *Towards a Semantic Description of English.* Indiana University Press, 1969.

[21] Heinz Marburger, Bernd Neumann, and Hans-Joachim Novak. Natural language dialogue about moving objects in an automatically analyzed traffic scene. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pages 49–51, Vancouver, BC, August 1981.

[22] David Marr and Lucia Vaina. Representation and recognition of the movements of shapes. *Proc. R. Soc. Lond. B*, 214:501–524, 1982.

[23] George A. Miller. English verbs of motion: A case study in semantics and lexical memory. In Arthur W. Melton and Edwin Martin, editors, *Coding Processes in Human Memory*, chapter 14, pages 335–372. V. H. Winston and Sons, Inc., Washington, DC, 1972.

[24] H.-H. Nagel. Analysing sequences of TV-frames. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, page 626, Cambridge, MA, August 1977.

[25] Bernd Neumann and Hans-Joachim Novak. 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*, pages 724–726, Karlsruhe, August 1983.

[26] Gordon S. Novak Jr. and William C. Bulko. Understanding natural language with diagrams. In *Proceedings of the Eighth National Conference on Artifical Intelligence*, Boston, MA, July 1990.

[27] Naoyuki Okada. SUPP: Understanding moving picture patterns based on linguistic knowledge. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 690–692, Tokyo, August 1979.

[28] Joseph O'Rourke and Norman I. Badler. Model-based image analysis of human motion using constraint propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(6):522–536, November 1980.

[29] Steven Pinker. *Learnability and Cognition*. The MIT Press, Cambridge, MA, 1989.

[30] Richard F. Rashid. Towards a system for the interpretation of moving light displays. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(6):574–581, November 1980.

[31] Terrance Philip Regier. *The Acquisition of Lexical Semantics for Spatial Terms: A Connectionist Model of Perceptual Categorization*. PhD thesis, University of California at Berkeley, 1992.

[32] John M. Rubin and W. A. Richards. Boundaries of visual motion. A. I. Memo 835, M. I. T. Artificial Intelligence Laboratory, April 1985.

[33] Roger C. Schank. The fourteen primitive actions and their inferences. Memo AIM-183, Stanford Artificial Intelligence Laboratory, March 1973.

[34] Yoav Shoham. Temporal logics in AI: Semantical and ontological considerations. *Artificial Intelligence*, 33(1):89–104, September 1987.

[35] Jeffrey Mark Siskind. *Naive Physics, Event Perception, Lexical Semantics, and Language Acquisition*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, January 1992.

[36] Jeffrey Mark Siskind. Grounding language in perception. In *Proceedings of the Annual Conference of the Society of Photo-Optical Instrumentation Engineers*, Boston, MA, September 1993.

[37] Jeffrey Mark Siskind. Axiomatic support for event perception. In Paul McKevitt, editor, *Proceedings of the AAAI–94 Workshop on the Integration of Natural Language and Vision Processing*, pages 153–160, Seattle, WA, August 1994.

[38] Jeffrey Mark Siskind. Grounding language in perception. *Artificial Intelligence Review*, 1995. To appear.

[39] Jeffrey Mark Siskind and Quaid Morris. A maximum-likelihood approach to visual event perception. m. s.

[40] Robert Thibadeau. Artificial perception of actions. *Cognitive Science*, 10(2):117–149, 1986.

[41] John K. Tsotsos. Some notes on motion understanding. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, page 611, Cambridge, MA, August 1977.

[42] John K. Tsotsos and John Mylopoulos. ALVEN: A study on motion understanding by computer. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 890–892, Tokyo, August 1979.

[43] John K. Tsotsos, John Mylopoulos, H. Dominic Covvey, and Steven W. Zucker. A framework for visual motion understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(6):563–573, November 1980.

[44] Saburo Tsuji, Akira Morizono, and Shinichi Kuroda. Understanding a simple cartoon film by a computer vision system. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, pages 609–610, Cambridge, MA, August 1977.

[45] Saburo Tsuji, Michiharu Osada, and Masahiko Yachida. Three dimensional movement analysis of dynamic line images. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 896–901, Tokyo, August 1979.

[46] Saburo Tsuji, Michiharu Osada, and Masahiko Yachida. Tracking and segmentation of moving objects in dynamic line images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(6):516–522, November 1980.

[47] Zeno Vendler. *Linguistics in Philosophy*. Cornell University Press, Ithaca, NY, 1967.

[48] H. J. Verkuyl. Aspectual classes and aspectual composition. *Linguistics and Philosophy*, 12(1):39–94, February 1989.

[49] David L. Waltz. Toward a detailed model of processing for language describing the physical world. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pages 1–6, Vancouver, BC, August 1981.

[50] David L. Waltz and Lois Boggess. Visual analog representations for natural language understanding. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 926–934, Tokyo, August 1979.

## Acknowledgements

This research was supported in part by an AT&T Bell Laboratories Ph.D. scholarship to the author, by a Presidential Young Investigator Award to Robert C. Berwick under National Science Foundation Grant DCR–85552543, by a grant from the Siemens Corporation, and by the Kapor Family Foundation. This research was also supported in part by ARO grant DAAL 03–89–C–0031, by DARPA grant N00014–90–J–1863, by NSF grant IRI 90–16592, by Ben Franklin grant 91S.3078C–1, and by the Canadian Natural Sciences and Engineering Research Council. Part of this research was performed while the author was visiting Xerox

PARC as a research intern and as a consultant. This work has benefited from numerous discussions with Peter Dayan, Richard Mann, and Michael Moore. Any errors, of course, remain the sole responsibility of the author. Small excerpts of this chapter, including figures 1, 3, 4, and 6, were previously published in [36, 37, 38].