

## **Part III.**

### **Loom Applied to Domain Reasoning**

The second part of the VEIL effort pioneered two thrusts within the field of image understanding that until now have received relatively little attention. First, VEIL reasons at a semantic level about scene information. A knowledge base augments an original image understanding (IU) model (representing the output of a lower level IU system) with structural and functional information. VEIL can apply both spatial and temporal reasoning to detect event sequences that span multiple images. Second, VEIL provides users with an intelligent interface that relies on a library of domain-specific terms and queries to provide a domain-specific browsing and query facility. The library is implemented as an extensible collection of domain specific definitions and queries in the Loom knowledge representation system. Users can easily customize these definitions and queries to facilitate analysis activities.

#### **10. Domain Reasoning Experiment Overview**

Human image analysts are able to carry out both spatial and temporal reasoning to detect event sequences that span multiple images. Such reasoning has received relatively little attention in the image understanding field. We have been developing the VEIL system for carrying out this type of reasoning. VEIL performs high level interpretation of scene images using the deductive capabilities of the Loom knowledge representation system [MacGregor and Bates 1987, Brill 1993], and provides users with semantically enriched browsing and editing capabilities. The VEIL experiments used a database of RADIUS Model Board 2 image site models stored in SRI's RCDE system. This database is augmented by a knowledge base stored in Loom that includes references to the underlying RCDE-object models. The Loom knowledge base also contains representations of functional and structural knowledge not contained in the RCDE model, and a library of high-level spatial reasoning functions. The Loom knowledge base also contains abstract definitions for objects and events. Using this architecture as a base, VEIL supports queries that search within an image to retrieve concrete or abstract objects; or that search across images to retrieve images that contain specific objects or events. An event in VEIL is composed of entities and/or subevents that collectively satisfy a set of temporal and spatial constraints. VEIL can scan a sequence of images and detect complex events. In the example presented in this paper, VEIL finds Field Training Exercise events consisting of four subevents occurring in distinct images.

VEIL is implemented as a modular architecture wherein all communication between Loom and the underlying IU system is mediated by RCDE protocols and data structures [RADIUS Manual 1993]. In the future, it would be practicable for

us to incorporate multiple IU systems into the VEIL architecture. Also, VEIL could be exported to other sites along with RCDE, allowing other IU researchers to connect their systems to VEIL. Thus, VEIL provides a generic means for extending a RCDE-based IU system to include semantic processing of image data. Use of VEIL also promotes the use of explicit, declarative domain models. We forecast that this approach will be a key enabling technology when it becomes time to interconnect image understanding systems with other knowledge-intensive systems and applications.

## 11. Underlying Technology

We are extending the semantics of the information that is captured by an image understanding program by associating domain-level information with images. We use the following terminology. The *image* means the digital input data. For our examples these are photographs. The *site model* is a geometric model of objects found in a particular image. Such objects can be roughly divided into objects representing terrain features, structures and vehicles. A *domain model* is a semantic model of items of interest in the domain. This includes buildings and vehicles as well as abstract notions such as the function of objects, groups of objects, convoys and field training exercise events.

### 11.1 RADIUS

Our experiments use the forty RADIUS Model Board 2 images of a hypothetical armored brigade garrison and exercise area. A site model common to all forty images was provided by the University of Maryland. This RCDE-object site model was used with only minor modifications in our work.<sup>1</sup> We augmented the common site model with vehicle models for a subset of ten images. Vehicles were identified by a graduate student and their location noted in a file. Vehicle model objects are needed for VEIL's event processing, but the source of the models is irrelevant. A suitable automatic vehicle detector could be substituted for our manual methods.

### 11.2 Loom

We use Loom, an AI knowledge representation language in the KL-ONE family, to provide the infrastructure for semantic reasoning. Loom provides the following benefits:

- Declarative language. Information is encoded in an easy-to-understand format. This makes it easy to comprehend and extend the model.

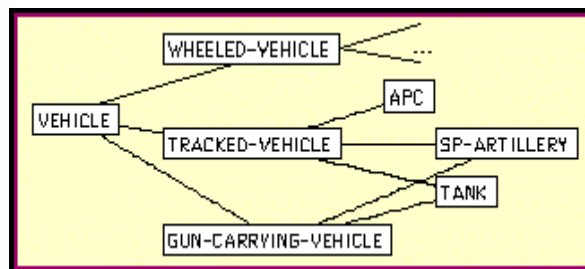
---

<sup>1</sup>The modifications were to ensure a consistent composite grouping of buildings which were represented in the site model as multiple cubes. Several of such complex-structure buildings were already present as composite objects. We manually rounded out the site model to assure consistency in the modeling.

- Well-defined semantics. The meaning of language constructs is well-defined. The meaning of the terminology is well established and validated by over 15 years of AI research into description logic [Brachman 1979, Brachman *et al.* 1983].
- Expressivity. Loom is one of the most expressive languages in its class.
- Contexts. Assertions (facts) about multiple images can be accessed at the same time. This is a key feature used in recognizing events.

### 11.2.1 Definitions

Loom reasons with *definitions*, which equate a term with a system-understood description in terms of necessary and sufficient conditions. This allows useful flexibility in reasoning for a recognition domain. Combined with a hierarchy of concepts one is able to make assertions that precisely capture the amount of information available. When details are not known, one is not forced to overcommit in making entries to the knowledge base. As more information becomes available it can be added incrementally, improving the picture of the world. If enough additional information is added, Loom's classifier automatically recognizes an instance as belonging to a more specific concept.



**Figure 12. Vehicle Hierarchy of the Domain Model**

We will illustrate how this works using the fragment of the domain model for vehicles shown in Figure 12. Suppose that the first pass of processing is able to identify some group of pixels in the image as a vehicle. Details about the type of vehicle are not yet known, so the information is entered as a “vehicle V<sub>1</sub> in location X.” With further processing, it may be determined that the vehicle has tracks. This information can be added, to the knowledge base, allowing the classification of the vehicle as a tracked vehicle. The classifier is able to perform this inference because the definition of a “tracked-vehicle” is a vehicle with a drive type of tracks. Since V<sub>1</sub> now satisfies the definition, Loom automatically concludes that it is of type Tracked-Vehicle. If an appropriate type of gun is detected, V<sub>1</sub> may finally be recognized as a tank.

By using definitions, Loom can make the inferences licensed by the definitions automatically. This service frees applications built on top of Loom from needing to implement their own inference mechanism.

Since Loom's definitions are logical equivalence statements, they can be used to reason in both directions. The example above illustrated using the components of a definition to perform a recognition task—synthetic reasoning. One can also assert the presence of higher level objects and then use the definitions to identify components that should be present.

For example, a particular SAM unit may be known to deploy with a radar vehicle and three launchers. If such a unit is asserted to exist in a scene, Loom concludes that there are three launchers present, even if they are not identified. The definition can then be used as a guide to what other objects should be present. It can be used to drive the reasoning. This type of reasoning was used in another part of the VEIL project that identified runways [Price *et al.* 1994].

### **11.2.2 Contexts**

Loom has a context mechanism that allows one to maintain distinct assertion sets. Loom's contexts are organized hierarchically, which allows inheritance. Siblings are separate, so this allows information about different scenes to be kept separate, but in the same lisp image. The query language (see below) is able to perform queries across contexts, so one can make comparisons and look for particular patterns.

Augmenting this flexibility is the fact that Loom contexts are themselves first-class objects. That means that assertions and annotations about the context themselves can be represented in the Loom formalism and be used to select appropriate contexts. This capability was added to Loom version 3.0 in response to the needs of the VEIL project.

For example, if one had a context associated with a particular image, one could annotate the context with information such as sun angle, time of day, camera location, etc. This information is available for image retrieval purposes. At the end of this paper, we will discuss the use of this context mechanism in event detection.. Event detection will involve searching for a sequence of images (contexts) that fulfill the criteria for a given event. This uses the ability of Loom to have several image descriptions in memory simultaneously as well as the ability to formulate and execute queries that cover several images.

### **11.2.3 Query Mechanism**

Loom includes a general query facility that is able to find objects based on their name, type or value of role (relation) fillers.

**Queries for Particular Objects:** Specific objects can be queried for in images. Examples include looking for all buildings, all headquarters, all tanks, etc. These queries allow a seamless use of collateral information in the RCDE system:

```
(retrieve ?bldg
 (headquarters ?bldg))
```

**Queries for Relationships:** In addition to queries that relate to single objects, one can also query about relationships. Examples include finding all buildings with an area of more than 5000 square feet, locating all tanks on roads, finding headquarters that are near barracks, etc.

```
(retrieve ?bldg
 (and (building ?bldg)
 (> (area ?bldg) 5000)))
```

Since Loom has the flexibility to allow new concepts to be defined dynamically, users can create queries and assign names to the resulting concepts. In future queries, this defined name can be used. This enriched vocabulary allows easier **customization** of the knowledge base as well as a more compact expression of the queries. For example, one could take the query about “buildings with an area of more than 5000 square feet” and introduce the named concept “large-building” to describe that query:

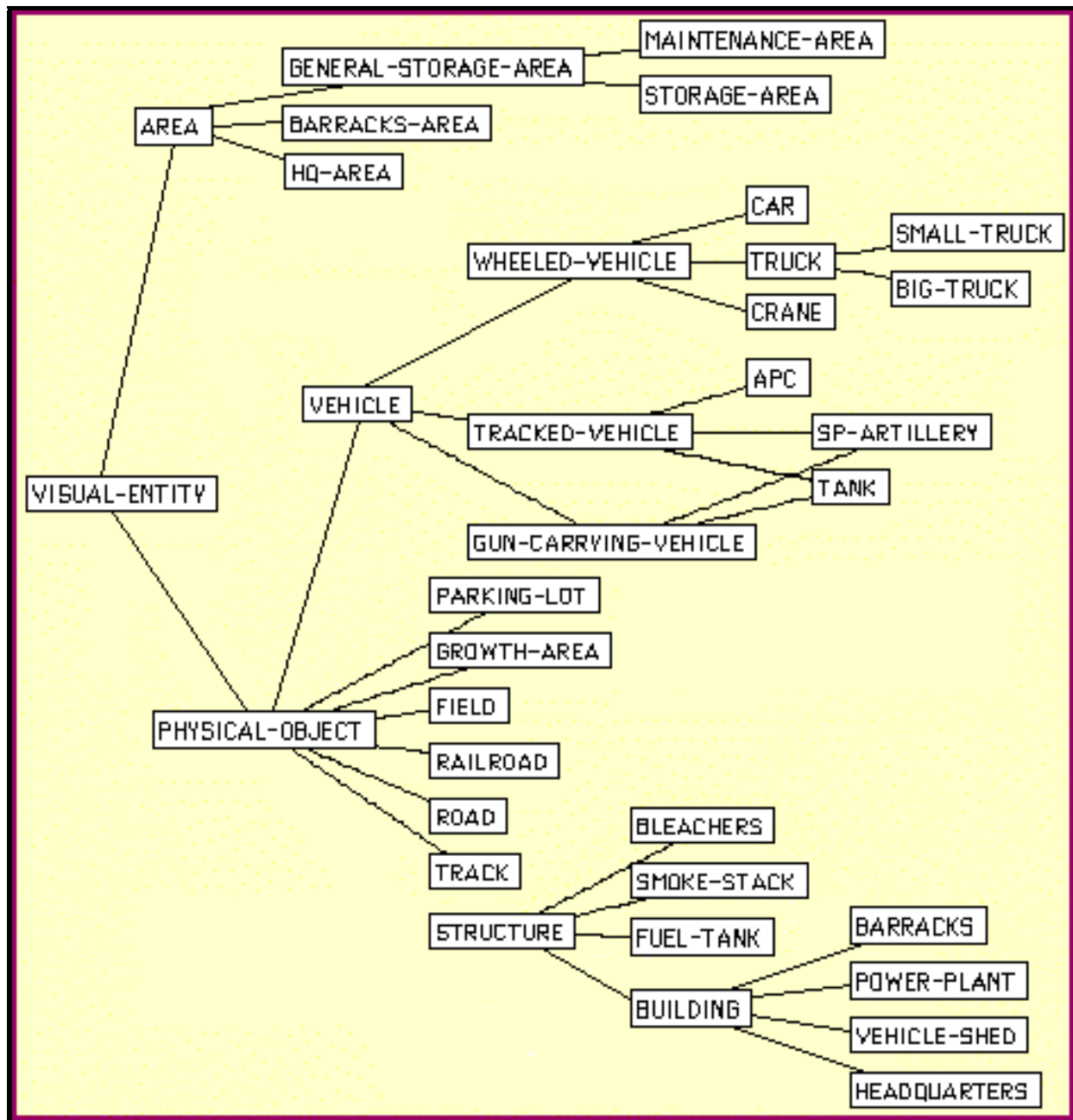
```
(defconcept large-building :is
 (satisfies (?bldg)
 (and (building ?bldg)
 (> (area ?bldg) 5000))))
```

In subsequent queries, the term “large-building” itself can be used. This provides the ability to dynamically extend the vocabulary used in the domain. By packaging and naming these new concepts, it is easier to formulate complicated queries because the introduction of abstract terms hides the underlying complexity and makes it easier to manage.

The examples so far demonstrate how Loom queries can be used with a single image. But Loom queries are not restricted to single images, but can extend across images. This type of query is used in the event detection example below.

## 12. The Domain Model

A prototype knowledge base containing domain concepts was created for use in VEIL. The type of knowledge encoded in this domain model ranged from the concrete to the abstract. Loom models (instances) for concrete, visible objects such as roads, buildings and vehicles (see Figure 13) are linked to geometric objects in the RCDE site model. This linking is accomplished by making the RCDE the value of a Loom relation on the Loom instance.



**Figure 13. Domain Model for Visible Objects in VEIL**

Collateral information about objects in a scene, such as “building B44 is a brigade headquarters”, is associated with the Loom instance representing the building. Other RADIUS research [Burhans *et al.* 1994] underscores the usefulness of associating collateral information in image interpretation. A convenient method of adding such annotations to a knowledge base is described in [Srihari *et al.* 1996] (see also the next article in this book). The use of collateral information improves the match between the vocabulary of the domain experts and that of the computer support system.

Abstract concepts such as groups, functional roles and events are used to augment reasoning about the concrete objects. This abstraction process mirrors and extends the abstraction done in moving from geometric object to conceptually meaningful domain objects. The main examples that we use in VEIL are the concept of a group of vehicles<sup>2</sup> and the concept of high-level events. The events are abstractions composed of sub events.

Abstract entities can be specialized based on their characteristics. For example, VEIL defines a convoy as a group of vehicles with at least 65% of them on a road. Additional constraints can be added such as requiring a minimum number of vehicles (i.e., >4). Loom's flexible knowledge representation easily supports specialization of the general vehicle convoy such as defining a convoy of tanks.

The definition of a convoy combines information that is present in the Loom level (such as group membership) with information that is inherently geographic (such as the location of vehicles on roads). Loom's forte is symbolic reasoning. Determination of geographic location is geometric reasoning that is best handled using RCDE model structures. Accordingly, we have developed several representative and interesting geometric predicates and linked them to Loom relations. Reasoning is performed at the appropriate level and the results integrated by Loom.

## 12.1 Linking the Domain and Site Models

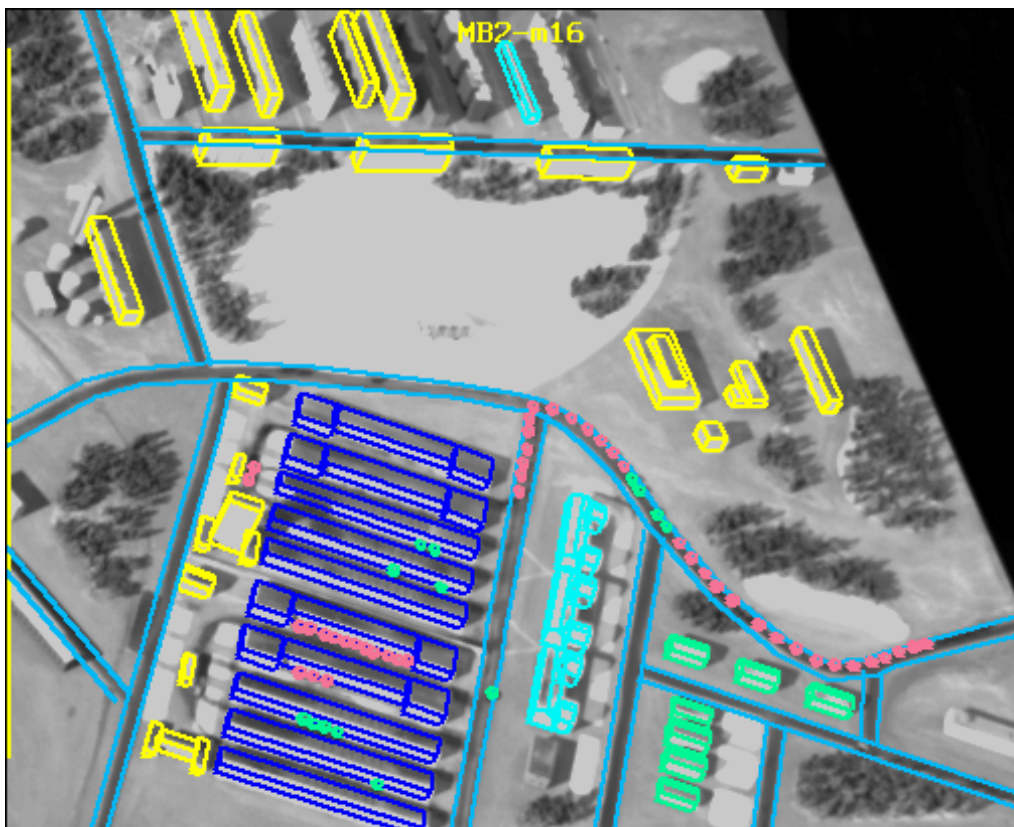
At the domain model level, the geometric information about the objects is not directly available. Instead, reasoning is focused on the function and wider role of the objects. At the geometric level, information about the location and size of objects is either directly available or computable from directly available information. For example, the location of a particular cube object is readily available and its volume can be easily computed using information stored about the length of the sides of the cube.

Figure 14 shows the original Image 16 from Model Board 2 and the view with the Loom geometric object level overlaid. The geometric objects are standard RCDE objects. Buildings are from the shared site model and vehicles are from the image-specific model.

Each Loom model object that has a geometric representation is linked to an underlying RCDE object. This allows the specification of queries that link Loom model concepts with geometric information. An example would be "Find all headquarters buildings (Loom level) with a size greater than 5000 square feet (geometric level).

---

<sup>2</sup>In the current implementation, groups are created by humans. Future work extending our ideas would involve providing tools for moving this into a semi-automated task.



**Figure 14. Image M16 – Plain and with Model Overlay**

**Table 1. Geometric Functions and RCDE Objects**

RCDE Object Type	Geometric Function				
	Location	Contains-point-p	Is-Near	Area	Volume
CUBE-OBJECT	X	X	X	X	X
CYLINDER	X	X	X	X	X
HOUSE-OBJECT	X	X	X	X	X
3D-CLOSED-CURVE	X	X	X	X	
3D-RIBBON-CURVE	X	X	X	X	
COMPOSITE-OBJECT	X	X	X	X	X
Others	X				

## 12.2 Geometric Relations

We have implemented several functions at the geometric level which are linked to Loom relations. Table 1 summarizes the basic relations. The most fundamental predicate is the one that returns locations. Given the three-space location of objects we implemented directional relations (north, northeast, etc.). We have also implemented computations for the area and volume of the most common geometric objects used in the site models.

Loom relations were linked to these functions. This enables Loom queries to seamlessly exploit both the semantic information contained in Loom's domain model and the geometric information from the underlying site model. An example of such a composite query is to find "all vehicle storage sheds with a floor area greater than 5,000 square feet":

```
(retrieve ?shed
  (and (vehicle-shed ?shed)
    (> (area ?shed) 5000)))
```

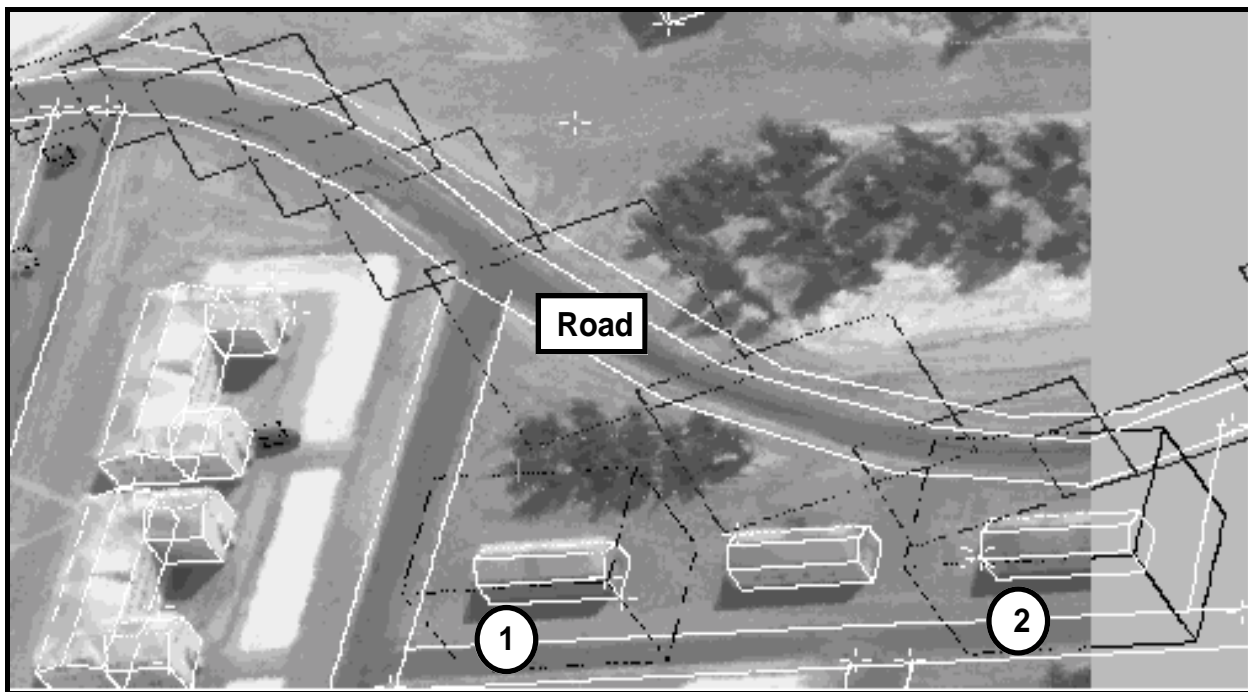
The concept *vehicle-shed* and the relation *>* are domain level operators. The relation *area* is a domain level relation that is linked to a **site-model-level function**. Loom's allows a computed relation to be defined by a Lisp function. [Haarslev *et al.* 1994] describes a similar method for linking Loom reasoning to an underlying spatial reasoning system.

Geometric relations can also be computed between objects. We implemented a containment test (contains-point-p), which tests to see if a given three-space point is contained in a 3 dimensional object (or located over a 2 dimensional object). This predicate is used in queries and concept definitions to locate vehicles that are on roads – for example in the concept of vehicles in a convoy.

One of the more interesting relations that we have investigated is the "is-near" relation. This is a subjective relation adopted from the nearness predicate in Abella's Ph.D. thesis [Abella 95]. Her studies found that a psychologically valid

implementation of nearness was influenced by the size of the objects in question. In other words, the larger the object, the farther away one could be in absolute distance while still being considered near. She developed a function that computes an “extended bounding box” for each object, based on the object’s dimensions. When two bounding boxes intersect, the objects are “near”.

We extended her formula to three dimensions. For buildings or vehicles, this yields appropriate results. The approach breaks down when the aspect ratio becomes very large. Extremely long, thin objects end up with very large bounding boxes because of the effect of their length on the size of the nearness boundary. Roads are prime examples from the site model that we use. The length of a road influences how far away one can be from a road and still be considered *near*. This produces counterintuitive results.



**Figure 15. Extended Bounding Boxes for Computing Nearness**

Site model objects have white boundaries. Extended bounding boxes for selected objects are black.

We therefore modified the algorithm for the case of long, thin objects. Objects with high aspect ratio disregard the long dimension when computing the nearness boundary. This modification produces appropriate results for our purposes. Figure 15 shows an image and the associated extended bounding boxes of a curved road and two buildings. Building 2 (on the right) satisfies the “near-to” relation with respect to the curved road, but Building 1 (on the left) does not.<sup>3</sup>

<sup>3</sup>The road is shown divided into bounding boxes segment-wise. The rectangular boxes are used for a rough test of nearness. A more sophisticated test which implements a smooth envelope is used for the final comparison.

```

(make-event
  :name 'field-training-exercise
  :case-roles '((armored-unit ?y))
  :components
    '(:scene ?s1 ?y (in-garrison ?y))
      (:scene ?s2 ?y (convoy ?y))
      (:scene ?s3 ?y (deployed-unit ?y))
      (:scene ?s4 ?y (convoy ?y)))
  :constraints '((before+ ?s1 ?s2)
                (before+ ?s2 ?s3)
                (before+ ?s3 ?s4))))

(retrieve (?Y ?S1 ?S2 ?S3 ?S4)
  (and (within-world ?S1
        (In-Garrison ?Y))
        (within-world ?S2
        (Convoy ?Y))
        (within-world ?S3
        (Deployed-Unit ?Y))
        (within-world ?S4
        (Convoy ?Y))
        (before+ ?S1 ?S2)
        (before+ ?S2 ?S3)
        (before+ ?S3 ?S4)))

```

**Figure 16. Event Definition and Corresponding Loom Query**

## 13. Event Detection

In this section we describe how we define events – objects that satisfy constraints both within and across images. We also outline how VEIL is able to locate such events in its database.

### 13.1 A Definition Language

An event is a sequence of *scenes* that satisfy certain criteria. Some of the criteria apply within scenes whereas other criteria describe the relationship between different scenes. Accordingly, we defined a language that allows these constraints to be specified in a natural way. The scenes in an event are described separately, specifying any criteria that apply within a single scene. A set of global constraints is then used to specify the conditions that must hold between scenes. The most common cross-scene constraint is that of order. A sequence of scenes implies that there is an ordering to the scenes.

### 13.2 Sample Event Definitions

Figure 16 shows an event definition named “Field-Training-Exercise” and its associated Loom query. The event consists of four scenes involving an armored unit “?y”. The scenes must include one with ?y “in-garrison”, two scenes with ?y in convoy and one with ?y deployed. In addition, the scenes are constrained temporally by the :constraints field. Translating this into English, we are looking for a sequence of scenes showing an armored unit in a garrison, then moving in convoy, then deployed in a training area and finally in convoy again. A set of images showing this evolution is shown in the example below.



**Figure 17. Field Training Exercise Event Found in an Image Sequence**

### 13.3 Example of Event Detection

Figure 17 shows a master view of the ten images we used in our experiments. An example of a field training exercise event is highlighted. Figure 18 shows a close-up of the field training exercise with the objects participating in the event highlighted. A colored box is drawn around the group of vehicles in each image. (In these figures, the group bounding box has been enhanced for better black-and-white printing.)

### 13.4 How It's Done

The Loom query in Figure 16 (right) is used to extract those scenes that meet the event criteria. This involves satisfying the conditions for each individual scene (such as finding a group that is in a garrison area in a scene) and also satisfying the cross-scene constraints (such as being in a particular temporal order). Loom concepts define the terms such as “in-garrison” (a group of vehicles in a maintenance or storage area) that are in turn used to define the event.

The result of this query will be a set of tuples. Each tuple consists of the group (?Y) and the four scenes (?S1-?S4: contexts associated with images) that satisfies the query. The link to the RADIUS allowed the visual displays in the examples of Figure 17 and Figure 18 to be created automatically from one such event match.



**Figure 18. Close-Up View of Field Training Exercise**

Because of Loom's named definitions, the query for finding events is quite compact and reasonably readable. This shows the power of having an extensible domain-specific language: even complex criteria can be expressed in a concise and natural manner.

## 14. Current Status

The current VEIL model has been tested using ten images from the RADIUS Model Board 2 image set (See Appendix B). It is integrated with the RCDE code and uses the RCDE graphics interface for user interaction and display purposes. The figures in this report are screen shots from the VEIL-RCDE system.

## 15. Future Work

There are several directions for extending our research. One major direction would be to improve the matching algorithm used to find events. The current match relies on using Loom's general query mechanism. While this provides flexibility, the logic-based query language does not take advantage of special features of the problem of event matching that can increase efficiency. For example, there is no direct exploitation of the fact that the scenes being looked for form an ordered sequence. Additional enhancements would be to modify the event matching language to allow inexact matches. This can take the form of partial matches, matches to key features but with missing elements, or a more general probabilistic matching scheme.

A sub-problem of the general matching task is associating groups from one image with those from a different image. (In the current work, such matching is done by hand.) An interim position would be to use a credulous matcher,<sup>4</sup> although that would need to be refined in order to scale well. The preferred approach would be to develop a compatibility score for matches between groups in one image and groups in another image. This score would be based on factors such as the size of the group, the composition of the group (i.e., with or without tanks), as well as heuristic reasoning based on other elements that are visible in an image. With a more sophisticated matcher, a list of candidate image sequences can be identified and ranked as to the closeness of the match.

Computer support for assigning individual vehicles to groups is another area for further investigation. The group assignment problem involves identifying a collection of vehicle that are related in some interesting way. Geometric proximity is one important consideration, but it is not always the most important. Consider a convoy driving by a parking lot. Some vehicles in the convoy will be closer to parked vehicles than to other convoy vehicles, but the importance of being on the road should be given more weight in the group assignment process. A semi-automated grouping tool would be a useful addition to RCDE.

---

<sup>4</sup> A *credulous matcher* is one that would return all potential matches. This guarantees that no matches will be missed, but is likely to return a lot of false positive matches.

## **Part IV.**

### **Conclusion.**

Vision programming tools are adequate for low-level vision processing, but less appropriate for high-level vision tasks. This proposal offered a remedy for this situation, namely, to use the Loom system to represent high-level visual knowledge. We employed a layered architecture wherein specialized data structures were used to represent lower-level knowledge, while symbolic structures were used to represent higher level knowledge.

VEIL was an experiment both to see how traditional knowledge representation technology can be applied to computer vision programs and to explore extensions to knowledge representation systems to directly aid computer vision research (especially in the area of spatial reasoning and event detection.)

### **16. Representation in Programs**

A key benefit of applying Loom knowledge representation technology to vision processing was that the VEIL system was able to use the semantic information expressed in Loom models to form expectations about the objects in the visual field, and thereby guide the low-level vision routines. This improved the overall accuracy and robustness of the vision system, in addition to easing system development, and improving its maintainability and extensibility.

High level reasoning provided by Loom was integrated with low- and mid-level processing techniques, such as line finding, segmentation, perceptual grouping and shape descriptions commonly used in vision systems. The goal of VEIL development is to allow vision application systems to be written more easily, by enabling the construction of explicit declarative vision models, and by exploiting the knowledge representation and reasoning capabilities provided by Loom. The experience with the airport example shows that this goal was achieved.

### **17. Domain Level Reasoning**

The bulk of work in IU research has been on developing algorithms that operate at the pixel level and are able to recognize geometric objects. Common examples are edge detectors, building detectors and vehicle detectors. In our work, we have been investigating the next stage of image understanding. In other words, we are concerned with the question of what sort of processing would we like to have happen once the low-level detectors have finished their work.

We feel that the next step involves reasoning with the aid of domain models—models of the world. This raises the level of abstraction of the interface between

the image analyst and the computer system. Instead of operating at the level of pixels or geometric shapes, one would like to have the interface operate at a level that has the appropriate semantic content for the task at hand. This level would allow interaction in terms of headquarters rather than buildings, convoys rather than isolated vehicles. By raising the level of interaction, better use of an image analyst's time can be made.

By increasing the level of abstraction and allowing queries at that level, it becomes easier to select appropriate images for viewing out of a large library. By raising the level of abstraction, we are also able to describe events that cover multiple images naturally and locate them efficiently.

## **APPENDIX A**

### **Image Understanding Environment Support**

The Image Understanding Environment (IUE) is an object oriented software development system for supporting research in image understanding and for facilitating the exchange of results and programs among different research groups. The IUE provides a conceptual framework for image understanding describing algorithms and data and the implementation of standard techniques within the IUE will allow for performance evaluation and comparisons between different techniques.

The process of designing the IUE began in 1989 with a number of meetings where existing systems and approaches were evaluated and the goals of the program were specified. In 1991, the IUE Technical Committee (IUETC) was established to generate the final specifications. This design process resulted in a design with over 500 classes and over 800 pages of documentation. The implementation effort is led by AAI, but the initial class definitions and empty method definitions are generated directly from the documentation that was developed by the IUETC in its several years of effort. We participated in the initial evaluation effort and in the later specification and evaluation effort while serving on the IUETC.

In the initial specification phase of the project our primary role was in the description of image features, the spatial index and data exchange. While the implementation of the basic IUE is being done by AAI, a number of programs (called the library) are not included in their requirements.

Our role in this design process included work on the general specifications, through the many meetings of the committee. We provided detailed specifications in the area of image features, the spatial index, and data exchange. Image features are the primary building blocks of image understanding programs. These include point (0-D) features such as simple edge elements, line (1-D) features such as extracted edge or line features, and area (2-D) features such as image regions, and groupings of these basic features such as line junctions and ribbon features.

In the development of the IUE, the initial specification of image features was treated separately from other objects (indeed the initial specification of many parts was separate from other related objects). Through the committee discussions it became apparent that the general group of spatial objects and image features were similar, but different. This led to the development of the current image feature description in terms of the more general spatial object hierarchy. In many cases the only difference between an image feature and the corresponding spatial object is the relationship of the feature to an image.

We developed programs to extract edge elements from the image, group the edges into sequences, extract image-feature line segments from the sequences, and finally to group the line segments into parallel pairs (for simple ribbon features). These programs were based on the earlier work of Nevatia and Babu [Nevatia and Babu 1980].

For many operations it is necessary to answer a variety of spatial questions about an object. For example “what line segments are near some point?” or “what line segments are within a given distance from a line?” The typical storage of the line segment feature in a list works when there are few features, but results in large computation times for images typically used in many applications (which can easily have 10,000 or 50,000 line segments). To efficiently answer these questions, we use a spatial index. In the simplest form, this is just an array that maps directly to image locations (usually the array is smaller than the image so that blocks of the image map to one array location). This means that spatial questions (e.g. near a point, near a line, etc.) can be approximately answered by reference to the array, with exact answers still requiring testing of the limited number of objects that are returned from the array reference. The basic spatial index design was completed with the implementation being done by AAI under their IUE implementation contract.

One of the early goals of the IUE was predictable data exchange of structures normally used by image understanding programs. For this purpose, the IUE committee defined an IUE data exchange standard. The standard must be functional (able to describe all objects), compatible, portable and extensible. We adopted a character-based, human readable format that uses a Lisp-like syntax (primarily the use of parentheses as delimiters) with generic representations. The data exchange standard thus provides a means for transfer between users of the IUE, but also provides a means to transfer data, with consistent interpretation of the values, between the IUE and other systems or between different image analysis systems.

While AAI implemented the IUE DEX system, we implemented a Lisp-Based DEX reader/writer so that data could be exchanged between the IUE and the RCDE used in the program.

To support the goal of providing working programs in the IUE and to test the completeness of image feature descriptions we developed programs to extract edge elements from the image, group the edges into sequences, extract image-feature line segments from the sequences, and finally to group the line segments into parallel pairs (for simple ribbon features). These programs were based on the earlier work of Nevatia and Babu [Nevatia and Babu 1980]. The initial implementation of these programs came before many of the image feature classes were implemented, but the transition to the later versions of the IUE was simplified by using the same basic descriptions of the objects (since the enhanced capabilities that come with the IUE objects were not needed).

# APPENDIX B

## Model Board 2 Images

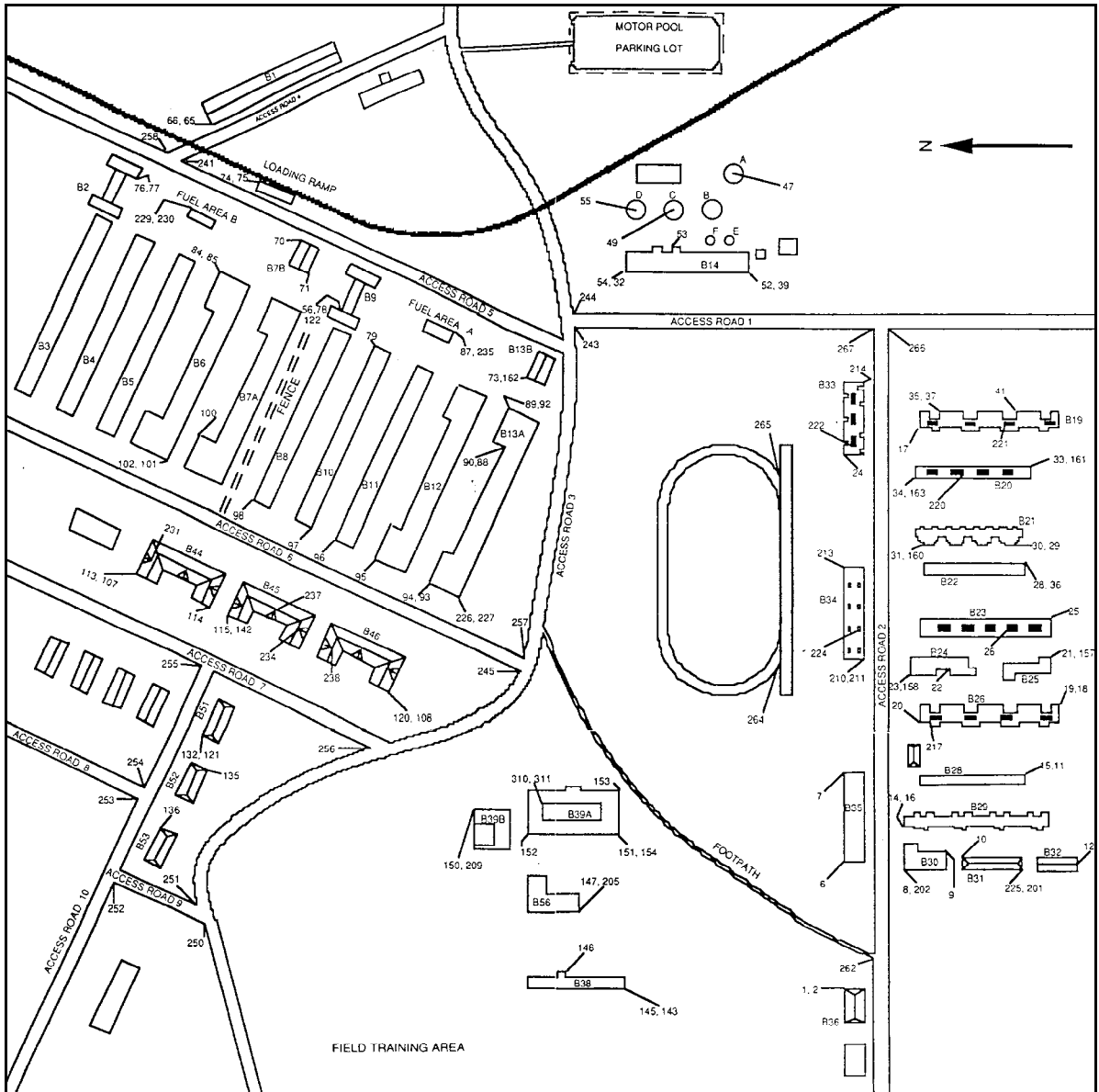


Figure 19. Map of Model Board 2



**Figure 20. Image M 4**



**Figure 21. Image M 6**



**Figure 22. Image M 8**



**Figure 23. Image M12**



**Figure 24. Image M16**



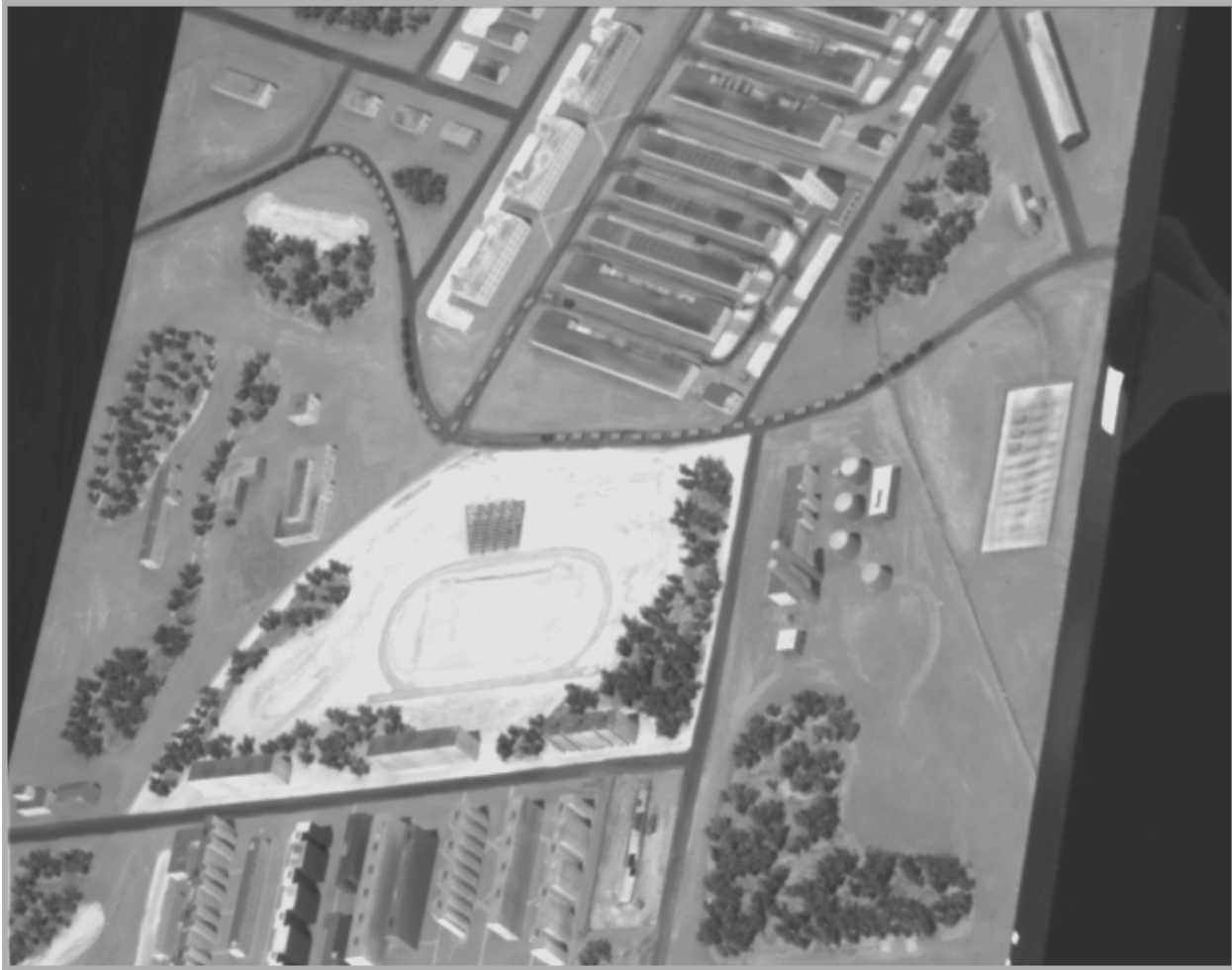
**Figure 25. Image M19**



**Figure 26. Image M24**



**Figure 27. Image M27**



**Figure 28. Image M32**



**Figure 29. Image M33**



# APPENDIX C

## Vehicle Location Data

These tables contain the information that was used to augment the site models provided with the RADIUS distribution.

Since the project did not have a reliable vehicle detector available, the vehicle locations, vehicle types and vehicle group assignments were done by humans.

The VEIL program used the group and location information to infer the status of groups as being in convoys, in garrison areas or in the field. As can be seen, the only semantic content added by hand are vehicle type and group assignments. VEIL automatically infers all additional semantic information.

Coordinates are given in terms of the underlying RADIUS Model Board 2 site model's geographic coordinate system. They can be used by Other researchers using this site model.

Image M 4			
Group	Type	X	Y
None	Crane	1407.0	-352.3
	Crane	1412.7	-235.8
	Vehicle	1413.7	-468.3
	Vehicle	1448.1	-350.0
	Vehicle	1448.4	-373.7
	Vehicle	1457.2	-439.5
	Vehicle	1470.3	-112.5
	Vehicle	1481.9	76.7
Group 1	Vehicle	705.8	714.1
	Vehicle	740.8	716.6
	Tank	775.6	738.5
	Tank	786.9	774.9
	Tank	809.7	828.2
	Tank	832.5	871.6
	Tank	846.7	902.9
	Tank	863.6	946.3
	Tank	892.1	979.7
	Tank	943.7	972.0
	Tank	986.7	950.0
	Tank	1032.3	923.3
	Vehicle	1249.3	815.0
Group 3	Vehicle	1437.2	718.3
	Tank	1540.7	933.4
	Vehicle	1553.8	966.0

Image M 6			
Group	Type	X	Y
Group 1	Tank	-142.1	352.5
	Tank	-138.2	313.1
	Tank	-135.5	400.8
	Tank	-133.0	274.9
	Tank	-129.4	242.9
	Tank	-128.5	436.7
	Tank	-109.5	472.7
	Tank	-93.1	508.7
	Tank	-67.3	539.8
	Tank	-43.7	556.0
	Tank	-24.2	573.4
	Tank	-18.7	615.3
	SP-Artillery	34.4	349.0
	SP-Artillery	48.9	395.9
	SP-Artillery	59.9	429.2
	Vehicle	75.6	1029.3
	Vehicle	118.6	95.9
	Vehicle	121.7	125.3
	Vehicle	125.0	1061.0
	Vehicle	137.7	119.5
	Vehicle	155.5	1106.2
	Vehicle	171.0	381.7
	Vehicle	254.3	1012.3
	Vehicle	293.9	490.7
	Vehicle	303.5	521.2
	Vehicle	313.0	321.7
	Vehicle	333.1	351.2
	Vehicle	358.5	374.7
	Vehicle	381.2	398.1
	Vehicle	404.1	416.6
Vehicle	426.7	1008.9	
Vehicle	442.1	1014.9	
Vehicle	459.9	1025.7	
Vehicle	474.6	961.2	

Group	Image Type	M 8 X	Y
None	Vehicle	532.4	1658.5
	Vehicle	539.2	1153.2
	Vehicle	752.8	1150.3
	Vehicle	784.0	761.6
	Tank	839.9	864.6
	Tank	904.2	998.6
	Vehicle	1008.8	647.8
	Vehicle	1018.1	-99.8
	Vehicle	1489.0	601.1
	Group 1	Tank	1066.6
Tank		1086.2	865.1
Tank		1102.7	858.3
Tank		1109.5	999.3
Tank		1118.5	848.6
Tank		1130.4	989.9
Tank		1137.3	840.4
Tank		1148.2	979.0
Tank		1153.2	830.6
Tank		1166.2	968.0
Vehicle		1184.4	1199.9
Tank		1185.7	962.9
Tank		1198.6	812.9
Tank		1201.4	953.3
Vehicle		1204.3	1187.9
Tank		1216.7	801.7
Tank		1222.4	943.9
Tank		1234.6	1059.1
Tank		1237.3	799.5
Tank		1239.8	940.1
Tank		1260.1	927.7
Tank		1355.6	754.4
Vehicle		1488.5	1160.8
Vehicle		1509.5	1151.6
Group 3		Vehicle	1469.7
	Vehicle	1519.0	776.3
	Vehicle	1535.3	904.3
	Tank	1556.6	949.3

Note: there is no group 2 in this image.

Group	Image Type	M12 X	Y	
None	Vehicle	60.1	-71.1	
	Vehicle	212.5	575.2	
	Vehicle	231.3	1452.3	
	Vehicle	768.6	-97.5	
	Vehicle	1193.4	1602.9	
	Vehicle	1470.5	-329.8	
	Vehicle	1517.0	298.3	
	Tank	1754.1	1417.5	
	Group 1	Vehicle	1023.7	1080.6
		Tank	1064.5	877.2
Tank		1092.6	878.2	
Tank		1106.0	853.8	
Tank		1109.7	1011.1	
Tank		1124.8	993.1	
Tank		1124.9	846.0	
Tank		1143.9	834.9	
Tank		1147.4	982.7	
Tank		1164.5	972.4	
Tank		1181.9	817.6	
Tank		1183.4	963.1	
Tank		1197.2	810.9	
Tank		1206.1	954.3	
Tank		1212.5	802.6	
Tank		1231.7	793.0	
Tank		1250.0	933.9	
Tank		1254.8	783.9	
Tank		1269.3	921.2	
Tank		1275.2	814.7	
Tank		1290.3	914.5	
Vehicle		1299.1	1037.2	
Tank		1311.6	902.8	
Vehicle		1329.4	1125.8	
Vehicle		1335.9	1007.6	
Vehicle	1366.8	994.8		
Vehicle	1373.6	1104.3		
Vehicle	1505.8	1046.1		
Group 2	Tank	1313.9	1396.1	
	Tank	1336.5	1388.0	
	Tank	1363.1	1375.7	
	Tank	1372.9	1507.8	
	Tank	1395.6	1356.7	
	Tank	1397.4	1501.1	
	Tank	1416.6	1349.1	
	Tank	1416.6	1487.8	
	Tank	1430.0	1342.1	
	Tank	1435.7	1476.1	
	Tank	1456.6	1470.3	
	Tank	1470.5	1322.4	
	Tank	1475.7	1465.0	
	Tank	1489.9	1310.3	
	Tank	1506.7	1445.2	
	Tank	1523.9	1440.4	
	Tank	1529.9	1315.2	
	Tank	1658.9	1420.9	
Group 3	Tank	1421.9	869.8	
	Tank	1457.5	706.3	
	Tank	1485.7	878.7	

Group	Image Type	M16 X	Y
None	Vehicle	1117.8	1491.6
	Group 1	Tank	41.5
Group 1	Tank	74.7	1401.8
	Tank	117.5	1414.5
	Tank	158.7	1429.9
	Tank	195.0	1439.8
	Tank	243.5	1449.3
	Tank	288.9	1445.0
	Tank	328.5	1432.5
	Tank	364.6	1398.0
	Tank	399.2	1366.3
	Tank	429.5	1329.1
	Tank	467.8	1260.8
	Tank	492.3	1220.7
	Tank	508.1	1183.6
	Tank	524.4	1140.7
	Tank	538.3	1097.7
	Tank	550.0	1055.0
	Vehicle	553.6	1051.6
	Vehicle	563.4	1017.0
	Vehicle	582.9	947.4
	Vehicle	588.8	915.5
	Tank	595.2	877.6
	Tank	607.9	836.5
	Tank	625.2	801.2
	Tank	646.0	768.6
	Tank	674.2	736.0
Tank	711.9	709.1	
Tank	756.5	702.9	
Tank	778.2	738.1	
Tank	792.5	773.2	
Tank	812.5	825.5	
Tank	835.8	865.8	
Tank	852.2	900.2	
Tank	872.3	945.8	
Group 2	Tank	1281.2	1400.4
	Tank	1300.7	1397.0
	Tank	1322.3	1385.6
	Tank	1351.3	1376.6
	Tank	1366.1	1365.5
	Tank	1384.5	1351.5
	Tank	1403.1	1345.6
	Tank	1426.8	1334.1
	Tank	1444.5	1325.5
	Vehicle	1448.1	1715.4
	Tank	1473.3	1308.6
	Tank	1476.9	1446.7
	Tank	1494.1	1302.6
	Vehicle	1495.1	1579.4
	Tank	1504.0	1432.5
	Vehicle	1517.6	1565.5
	Tank	1530.5	1423.7
	Vehicle	1539.4	1556.8
Vehicle	1557.5	1543.1	
Group 3	Vehicle	1124.2	1104.7
	Vehicle	1143.0	1206.6
	Vehicle	1152.4	1090.2
	Vehicle	1234.4	1157.5
	Tank	1468.1	869.0
Tank	1489.4	902.5	

Group	Image Type	M19 X	Y
Group 1	Tank	589.1	928.4
	Vehicle	606.3	870.4
	Vehicle	647.8	772.8
	Vehicle	674.1	742.9
	Vehicle	701.6	716.1
	Vehicle	730.8	687.5
	Vehicle	771.6	724.1
	Tank	799.4	793.1
	Tank	821.9	835.7
	Tank	843.5	880.7
	Tank	869.8	941.9
	Tank	940.6	1077.6
	Group 2	Tank	1462.1
Tank		1481.9	646.1
Tank		1500.8	682.7
Tank		1508.4	594.3
Tank		1513.5	720.9
Tank		1536.4	762.4
Tank		1552.8	595.0
Tank		1558.4	804.5
Vehicle		1582.9	851.4
Tank		1600.5	600.0
Tank		1643.7	609.0
Tank		1686.7	619.7
Tank		1730.3	633.4
Tank		1770.3	652.0
Tank		1809.4	663.9
Tank		1843.8	687.0
Tank		1881.4	712.5
Vehicle		1911.5	730.2
Vehicle		1938.2	743.6
Vehicle		1965.5	761.7
Vehicle	2002.8	781.4	
Group 3	Tank	959.0	964.3
	Tank	1040.5	1053.8
	Tank	1106.4	886.9
	Tank	1205.4	840.2
	Tank	1249.9	779.5
	Vehicle	1374.2	735.9
	Tank	1413.8	871.1
	Tank	1450.2	706.2
	Tank	1473.3	881.1
Group 4	Tank	1286.0	1404.5
	Tank	1303.9	1395.5
	Tank	1327.7	1388.1
	Tank	1350.6	1376.3
	Tank	1478.0	1311.1
	Tank	1493.9	1439.1
	Tank	1507.4	1433.9
	Tank	1516.8	1316.5

Group	Image Type	M24 X	Y
Group 1	Vehicle	573.6	-108.0
	Vehicle	617.1	-111.0
	Vehicle	628.4	809.3
	Tank	1036.0	887.5
	Tank	1055.0	874.4
	Vehicle	1065.0	-137.0
	Tank	1075.0	866.1
	Tank	1090.0	857.8
	Tank	1091.0	1005.0
	Tank	1110.0	849.6
	Tank	1113.0	998.7
	Tank	1124.0	838.9
	Tank	1130.0	990.3
	Tank	1143.0	829.5
	Tank	1146.0	984.4
	Tank	1163.0	974.8
	Tank	1164.0	820.1
	Tank	1181.0	811.9
	Tank	1183.0	963.0
	Tank	1199.0	802.6
	Tank	1200.0	951.1
	Tank	1219.0	944.0
	Tank	1219.0	792.0
	Tank	1233.0	936.9
	Tank	1242.0	781.5
	Tank	1253.0	926.3
	Tank	1269.0	915.6
	Tank	1291.0	908.6
	Tank	1306.0	899.2
	Vehicle	1498.0	568.8
	Vehicle	1504.0	281.4
Vehicle	1505.0	194.6	
Tank	1537.0	946.9	
Vehicle	1539.0	596.4	

Group	Image Type	M27 X	Y	
None	Vehicle	155.4	1441	
	Vehicle	169.8	-80.1	
	Vehicle	525.7	1164.2	
	Vehicle	755.0	1407.6	
	Vehicle	904.9	1029.9	
	Vehicle	1017.7	-94.5	
	Vehicle	1468.1	640.4	
	Group 1	Tank	1053.7	874.3
		Tank	1079.1	866.6
		Tank	1093.4	860.0
Tank		1101.0	1000.9	
Tank		1110.5	850.9	
Tank		1121.8	993.7	
Tank		1128.9	845.1	
Tank		1138.7	984.5	
Tank		1146.5	833.1	
Tank		1156.0	973.9	
Vehicle		1171.1	1198.3	
Tank		1176.6	966.8	
Tank		1191.5	812.5	
Tank		1191.8	955.9	
Vehicle		1195.7	1191.5	
Tank		1208.7	802.1	
Tank		1211.7	951.8	
Tank		1223.0	1065.1	
Tank		1228.9	796.7	
Tank		1231.5	938.7	
Tank		1253.9	932.0	
Tank		1346.6	756.8	
Vehicle		1476.9	1163.7	
Vehicle		1501.9	1151.5	
Group 2		Tank	1293.8	1401.9
		Tank	1314.0	1394.4
		Tank	1339.4	1381.8
	Tank	1356.6	1536.1	
	Tank	1364.1	1372.0	
	Tank	1375.6	1524.0	
	Tank	1376.5	1362.1	
	Tank	1397.5	1350.4	
	Tank	1402.1	1514.3	
	Tank	1449.9	1350.5	
	Vehicle	1465.5	1483.4	
	Vehicle	1487.8	1473.3	
	Tank	1507.5	1335.5	
	Vehicle	1513.2	1457.9	
	Vehicle	1542.4	1443.1	
	Tank	1576.5	1561.4	
	Tank	1617.2	1529.3	
Group 3	Vehicle	1457.9	851.8	
	Tank	1550.4	949.7	
	Tank	1570.9	991.0	
Group 4	Vehicle	1720.2	1250.0	
	Tank	1827.9	1552.6	
	Tank	1846.4	1538.0	
Group 5	Tank	2087.8	529.3	
	Vehicle	2088.7	285.2	
	Vehicle	2090.0	321.0	
	Tank	2090.0	502.8	
	Vehicle	2090.1	304.0	
	Tank	2150.6	549.3	
	Tank	2168.1	382.2	
	Tank	2169.4	409.3	
	Tank	2169.6	433.4	
	Tank	2169.8	309.0	
	Tank	2170.0	333.1	
	Tank	2170.1	357.2	
	Tank	2170.9	452.0	

Group	Image Type	M32 X	M32 Y
None	Vehicle	207.8	578.7
Group 1	Vehicle	-93.1	1379.5
	Vehicle	-59.0	1386.7
	Vehicle	-21.6	1395.5
	Vehicle	9.6	1403.2
	Vehicle	46.4	1408.6
	Vehicle	87.2	1417.8
	Vehicle	121.3	1428.3
	Vehicle	166.4	1440.0
	Vehicle	207.3	1452.4
	Vehicle	243.2	1464.7
	Tank	283.7	1461.0
	Tank	323.5	1442.5
	Tank	358.4	1417.1
	Tank	387.7	1388.3
	Tank	415.4	1355.5
	Tank	443.6	1319.3
	Tank	462.9	1285.0
	Tank	485.0	1248.0
	Tank	502.6	1209.9
	Vehicle	525.6	1156.4
	Vehicle	571.2	986.7
	Vehicle	577.8	954.9
	Vehicle	589.8	913.5
	Vehicle	602.8	873.2
	Vehicle	620.2	834.4
	Tank	643.9	787.2
	Tank	669.1	749.7
	Tank	705.6	719.0
	Tank	753.4	712.0
	Tank	779.2	768.4
	Tank	802.6	819.9
	Tank	828.9	868.9
	Tank	853.2	916.1
Tank	886.9	974.7	
Tank	934.2	980.0	
Vehicle	990.5	953.5	
Group 2	Vehicle	990.0	646.1
	Tank	1025.8	641.2
	Tank	1067.9	637.3
	Tank	1110.3	631.3
	Tank	1160.9	621.8
	Tank	1212.3	612.2
	Tank	1260.7	605.2
	Tank	1310.8	599.9
	Tank	1348.9	599.7
	Tank	1403.5	596.7
	Tank	1456.4	596.0
	Tank	1502.2	597.5
	Tank	1503.3	597.3
	Tank	1551.2	597.4
	Tank	1596.8	603.9
	Tank	1640.1	610.7
	Tank	1682.4	622.7
	Tank	1724.7	633.7
	Vehicle	1760.3	646.7
	Vehicle	1801.4	663.8
Vehicle	1829.5	677.1	
Vehicle	1862.5	699.5	

Image M32 (continued)			
Group	Image Type	M32 X	M32 Y
Group 3	Vehicle	1293.5	1043.1
	Tank	1413.0	872.6
	Tank	1449.8	708.0
	Tank	1471.8	884.7
	Vehicle	1498.4	1050.2
Group 4	Tank	1282.0	1417.3
	Tank	1300.6	1406.8
	Tank	1319.9	1398.3
	Tank	1347.6	1385.8
	Tank	1474.5	1318.5
	Tank	1490.0	1450.3
	Tank	1506.0	1446.8
	Tank	1515.8	1327.1
Group 5	Tank	2094.8	473.0
	Tank	2095.7	443.5
	Tank	2096.8	503.2

Image M33			
Group	Image Type	M33 X	M33 Y
None	Vehicle	1125.1	625.8
Group 1	Vehicle	244.5	1453.6
	Vehicle	298.1	1451.2
	Vehicle	343.4	1429.8
	Vehicle	380.8	1391.8
	Vehicle	419.7	1348.9
	Vehicle	437.5	1317.9
	Tank	462.9	1277.2
	Tank	486.4	1238.9
	Tank	505.3	1200.5
	Tank	526.9	1164.5
	Tank	535.7	1121
	Tank	544.8	1079.8
	Tank	555.9	1036
	Tank	572.4	982.4
	Tank	583.6	938.4
	Vehicle	603.3	874.7
	Vehicle	654.1	771.5
	Vehicle	675.9	747.1
	Vehicle	704.4	720.3
	Vehicle	732.5	691.0
	Vehicle	777.2	734.4
	Tank	799.0	796.8
	Tank	827.2	841.7
	Tank	843.2	883.7
	Tank	873.5	942.9
	Tank	939.2	1081.7
	Tank	960.1	966.1
Tank	1250.4	781.4	
Tank	1280.6	915.4	
Tank	1301.5	903.7	
Tank	1409.3	869.4	
Tank	1448.2	705.1	
Group 2	Tank	1284.0	1411.7
	Tank	1308.9	1400.1
	Tank	1327.5	1390.9
	Tank	1349.8	1517.5
	Tank	1354.9	1381.7
	Tank	1372.7	1508.3
	Tank	1373.5	1372.5
	Tank	1388.9	1356.3
	Tank	1393.0	1496.8
	Tank	1410.6	1354.0
	Tank	1413.6	1487.6
	Tank	1429.9	1478.4
	Tank	1430.4	1337.9
	Tank	1449.0	1328.7
	Tank	1452.7	1469.3
	Tank	1473.0	1457.8
	Tank	1477.9	1314.9
	Tank	1493.6	1448.6
	Tank	1501.4	1310.3
	Tank	1507.4	1437.2
Tank	1528.3	1430.3	
Vehicle	1569.0	1557.8	
Vehicle	1593.4	1546.4	
Vehicle	1621.6	1530.5	

## Bibliography

[Abella 1995] Alicia Abella., *From Imagery to Salience: Locative Expressions in Context*, Ph.D. thesis, Columbia University, 1995.

[Bonissone *et al* 1987]. Piero Bonissone, Stephen Gans, and Keith Decker, "RUM: A Layered Architecture for Reasoning with Uncertainty," in *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, pp. 891-898, AAAI, August 1987.

[Brachman 1979] Ronald J. Brachman, *On the Epistemological Status of Semantic Networks*, pages 3-50, 1979. (Reprinted in Ronald J. Brachman and Hector J. Levesque, editors. *Readings in Knowledge Representation*, Morgan Kaufmann Publishers, Inc., Los Altos, CA, 1985. pp. 192-215)

[Brachman *et al.* 1983] Ronald J. Brachman, Richard E. Fikes, and Hector J. Levesque, KRYPTON: A functional approach to knowledge representation, *IEEE Computer*, **16**(10):67-73, 1983. (Revised version reprinted in Ronald J. Brachman and Hector J. Levesque, editors. *Readings in Knowledge Representation*, Morgan Kaufmann Publishers, Inc., Los Altos, CA, 1985., pp. 412-429.)

[Brill 1993] David Brill, *Loom Reference Manual*, Version 2.0, USC/ISI, 4676 Admiralty Way, Marina del Rey, CA 90292, December 1993.

[Burhans *et al.* 1994] Debra Burhans, Rajiv Chopra, Rohini K. Srihari, Venugopal Govindaraju, and Mahesh Venkataraman, Use of Collateral Text in Image Interpretation, *Proceedings of the ARPA Image Understanding Workshop*, Morgan Kaufmann Publishers, Inc. pp. 897-907, 1994.

[Canny 1986] J.F. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, Vol. 8, No. 6, pp. 679-698, November 1986.

[Collins *et al.* 1989] R. Collins, J. Brolio, A. Hanson, and E. Riseman, "The Schema System," *International Journal of Computer Vision*, Vol. 2, No. 3, pp. 209-250, 1989.

[Draper *et al.* 1989] B. A. Draper, R. T. Collins, J. Brolio, A. R. Hanson and E. M. Riseman, "The Schema System," in *International Journal of Computer Vision*, Vol. 2, 1989, pp. 209-250.

[Faugeras *et al.* 1981] O. Faugeras and K. Price, "Symbolic Description of Aerial Images using Stochastic Labeling," in IEEE Transactions on PAMI, Vol.3, pp 633-642, November 1981.

[Haarslev *et al.* 1994] Volker Haarslev, Ralf Möller and Carsten Schröder, Combining Spatial and Terminological Reasoning, *KI-94: Advances in Artificial Intelligence – Proceedings of the 18th German Annual Conference in Artificial Intelligence*, published as *Lecture Notes in Artificial Intelligence*, vol. 891 (Springer-Verlag: Berlin), pp. 142-153, 1994.

[Huertas and Nevatia 1988] A. Huertas and R. Nevatia, "Detecting Buildings in Aerial Images," in Computer Vision, Graphics and Image Processing Journal, No.41, pp 131-152, 1988.

[Huertas *et al.* 1989] A. Huertas, W. Cole, and R. Nevatia, "Using Generic Knowledge in Analysis of Aerial Scenes: A Case Study," in Proceedings 10th International Joint Conference on Artificial Intelligence, Detroit, Michigan, August 1989.

[Huertas *et al.* 1990] A. Huertas, W. Cole, and R. Nevatia, "Detection of Runways in Complex Airport Scenes," in Computer Vision, Graphics and Image Processing Journal, No.51, pp 107-145, 1990.

[MacGregor 1988] MacGregor, Robert, "A Deductive Pattern Matcher," in Proceedings of AAAI-88, The National Conference on Artificial Intelligence, 1988a.

[MacGregor 1990] Robert MacGregor "LOOM Users Manual," in USC/ISI Technical Report, ISI/WP-22, 1990a.

[MacGregor 1990] Robert MacGregor "The Evolving Technology of Classification-based Knowledge Representation Systems," to appear in Principles of Semantic Networks: Explorations in the Representation of Knowledge, John Sowa, Ed., Morgan-Kaufman, 1990b.

[MacGregor and Bates 1987], Robert MacGregor and Raymond Bates, The Loom knowledge representation language, Technical Report ISI/RS-87-188, USC Information Sciences Institute, 1987.

[MacGregor and Burstein 1991] R. MacGregor and M. Burstein, "Using a Description Classifier to Enhance Knowledge Representation," IEEE Expert, Vol 6, No. 3, pages 41-46, June 1991.

[MacGregor and Yen 1988] Robert MacGregor and John Yen, "The Knowledge Representation Project at ISI," in USC/ISI Technical Report, RR-87-199, 1988b.

[McKeown *et al.* 1985] D. M. McKeown, Jr., W. A. Harvey, and J. McDermott, "Rule Based Interpretation of Aerial Imagery," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol 7, No. 5, September. 1985, pp. 570-585.

[McKeown 1989] D. M. McKeown., W. A. Harvey and L. E. Wikson, "Automating Knowledge Acquisition for Aerial Image Interpretation," in *Computer Vision, Graphics and Image Processing*, Vol 46, 1989, pp. 37-81.

[Medioni *et al.* 1984] G. Medioni, and R. Nevatia, "Matching Images Using Linear Features," in *IEEE Transactions on PAMI*, Vol.6, No.6, November 1984.

[Mohan 1989] R. Mohan and R. Nevatia, "Using Perceptual Organization to Extract 3-D Structures," in *IEEE Transactions on PAMI*, Vol 11, No. 11, November 1989.

[Mundy *et al.* 1993] J. Mundy and The Gang of Ten, "The Image Understanding Environment: Overview," in *Proc. ARPA Image Understanding Workshop*, Washington, DC, April 1993, pp. 283-288.

[Nevatia and Babu 1980] R. Nevatia and K. R. Babu, "Linear Feature Extraction and Description," *Comp. Graphics and Image Processing*, Vol. 13, 1980, pp. 257-269.

[Nevatia and Price 1982] R. Nevatia, and K. Price. "Locating Structures in Aerial Images," in *IEEE Transactions on PAMI*, Vol.4, No.5, Sepember 1982.

[Payton 1984] D. W. Payton, "A symbolic pixel array representation of spatial knowledge," in *Proceedings of the IEEE Conference of Computers and Communications*, Phoenix, 1984, pp. 11-16]

[Price *et al.* 1994] Keith Price, Thomas Russ, and Robert MacGregor, Knowledge representation for computer vision: The VEIL project, *Proceedings of the ARPA Image Understanding Workshop*, Morgan Kaufmann, pp. 919-927, 1994.

[RADIUS Manual 1993], SRI International and Martin Marietta, *RADIUS Common Development Environment: Programmer's Reference Manual*, version 1.0 edition, July 1993.

[Silberberg 1987] T. Silberberg, "Context Dependent Target Recognition", in *Proceedings of the Image Understanding Workshop*, Los Angeles, 1987, pp. 313-320.

[Srihari *et al.* 1996] R. K. Srihari, Z. Zhang, M. Venkatraman, R. Chopra, ,Using speech input for image interpretation and annotation, *Proceedings of the ARPA Image Understanding Workshop*, Morgan Kaufmann, pp. 501-510, 1996.

[Winston 1975] P. H. Winston, "Learning Structural Descriptions from Examples," in *The Psychology of Computer Vision*, P. H. Winston, Ed., New York:McGraw Hill, 1975, Chapter 5.

[Yen *et al.* 1989] John Yen, Robert Neches, and Robert MacGregor, "Using Terminological Models to Enhance the Rule-based Paradigm," in *Proceedings of the Second International Symposium on Artificial Intelligence*, Monterrey, Mexico, October 1989.