10-1-1987

PSEIKI: A Production System Environment for Integrating Knowledge with Images

K. M. Andress
Purdue University

A. C. Kak
Purdue University

PSEIKI: A Production System Environment for Integrating Knowledge with Images

K. M. Andress
A.C. Kak

TR-EE 87-36
October 1987

School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

PSEIKI: A Production System Environment for Integrating Knowledge with Images[*]

K. M. Andress and A. C. Kak

Robot Vision Lab
School of Electrical Engineering
Purdue University
W. Lafayette, IN 47907

Technical Report TR-EE 87-36
October 1987

[*]This work was supported partly by the Army Night Vision and Electro-Optics Lab and partly by the National Science Foundation.

TABLE OF CONTENTS

ABSTRACT

A description of work-in-progress on PSEIKI is presented. PSEIKI is a computer vision system designed to use multiple sources of knowledge to aid in the image understanding task. In this report we describe the concepts used in PSEIKI and how the incorporation of world knowledge is used to make PSEIKI expectation driven. The world knowledge in the system is represented as a line drawing of the expected scene. The system is implemented as a 2 panel / 6 level blackboard and uses the Dempster-Shafer formalism to accomplish inexact reasoning in a hierarchical space.

1. INTRODUCTION

A fundamental goal of computer vision is the development of systems that would be capable of carrying out scene interpretations with the aid of all available knowledge. As an example, if a helicopter-based computer vision system is looking at a snow covered terrain, then that knowledge must be taken into account explicitly in a target recognition procedure -- clearly the processing required for a snow-covered background is different from that for, say, wooded areas in spring.

As a simpler example of knowledge-based processing, consider the problem of self-location for a vehicle-mounted vision system [1]. Let's say the vehicle's whereabouts are known approximately from the position encoders mounted on the wheels -- the precision of this information being limited by the extent of slippage in the wheels, etc. Given this approximate information, is it possible to make a more precise fix on the location of the vehicle by integrating the vision data with the map knowledge, while the two are out of registration? This problem of robot self-location was the original goal of PSEIKI. We felt that this simple exercise in knowledge-based processing would give us the expertise to set up more complex reasoning structures for incorporating other kinds of knowledge sources in an image interpretation task. The acronym PSEIKI stands for a Production System Environment for Integrating Knowledge with Images.

As it stands now, PSEIKI, a production system in OPS83, is designed to use multiple sources of knowledge to aid in an image understanding task. The world knowledge is presented to the system as a line drawing of the expected scene.* The system is implemented as a 2 panel/6 level blackboard and uses the Dempster-Shafer formalism to accomplish inexact reasoning in a hierarchical space.

Although PSEIKI was originally was developed for carrying out knowledge-based experiments in robot self-location, the current implementation is general enough to be used in any application where a good estimate of the expected scene is available to the vision system. The system can be used as a general vision verification module either in a robotics context or for automatic target recognition.

PSEIKI contains two features that keep it domain independent. First, the knowledge used by PSEIKI consists of a line drawing of the expected scene (which in most applications would not be in registration with the observed image). For example, for robot navigation applications, line drawings can be easily be generated from road maps. For verification vision, a line drawing of the object whose identity, location and orientation need to be verified, can be generated from a 3D model of the object. In more industrial 2D vision applications, computer

*Note that for applications such as automatic target recognition the line drawing representation of the expected scene can include environmental effects.

graphics or CAD systems can be used directly to generate the line drawings of the expected scenes. The other feature that provides the system domain independence concerns how the system presents its results. The output of PSEIKI consists of a mapping from elements detected in the input image to elements in the expect scene.*

The mapping generated by PSEIKI is expressed by labeling the detected edges with the names of the corresponding lines in the expected scene; a belief value is also attached to each label indicating the confidence of the mapping found. Furthermore, a belief value is estimated for the entire mapping process. If this overall belief value does not exceed a threshold, the entire mapping is rejected. As a simple illustration of what PSEIKI does, if Fig. 1a is a line-drawing rendition of an expected scene and Fig. 1b a depiction of the edges that might be found in the vision data collected for the scene, then PSEIKI would produce an output similar to the one in Fig. 1c.

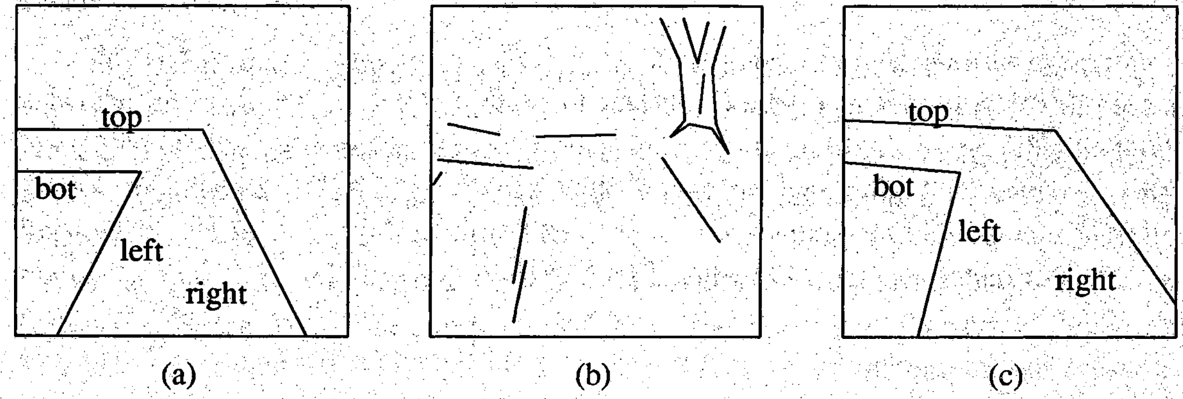


FIGURE 1. Typical images used by PSEIKI

Since PSEIKI only generates a mapping from the edges in the input image to the expected scene, it is left to a higher level system to make global interpretations based on the mapping found.

PSEIKI is also a testbed for carrying out experiments in how inexact reasoning can be achieved on hierarchical representations of scenes. Gordon and Shortliffe [2] discuss a technique that allows the Dempster-Shafer formalism [3] to be used in a system that hierarchically groups hypotheses. That article deals with diagnostic reasoning (in medicine) where the hypotheses can be grouped into strict hierarchies. The methods of Gordon and Shortliffe can not be used by us directly because PSEIKI does not employ strict hierarchies (an edge can be a member of two faces if it is a part of the border between them). In the current implementation of PSEIKI, the blackboard architecture is exploited to permit exact and inexact reasoning in a tangled hierarchy. The Dempster-Shafer formalism is used for pooling uncertain evidence in the hierarchy.

*We will be referring to this mapping throughout the report.

PSEIKI is able to handle significant perspective effects. Many previous systems, again most notably aerial interpretation systems, were able to assume that the images were obtained by an orthographic imaging system. Although perspective distortions make image interpretation difficult because metric properties, such as length and orientation, depend on the object's position in the image, they also provide clues to the structure of objects in the image.

PSEIKI was also developed to investigate how different sources of knowledge can be integrated into the image interpretation task. Although knowledge of the expected scene is the main source of information being investigated, the system is general enough to affect integration over multiple images in structural stereo or optic flow. The investigation of how system control flow affects the image understanding task is also underway.

1.1. RELATED IMAGE UNDERSTANDING SYSTEMS

- ACRONYM (Brooks, et al. [4]) is a model-based image understanding system. The system's task consists of finding instances of known objects in the image. To perform object identification, the system first builds a *Picture Graph* of the image and an *Observability Graph* that specifies information about objects that could be in the image. The system identifies instances of objects in the image by matching nodes of the Observability Graph with sets of nodes in the Picture Graph. The objects in the Observability Graph are represented in *slot - filler* structures where any slot that can accept numeric values can also accept algebraic constraints expresses as inequalities. The system can then manipulate these constraints and determine if properties of objects detected in the image meet these constraints. The objects used to generate the Observability Graph are represented as generalized cones. Inexact reasoning is not used and the system uses only backward chaining to arrive at an interpretation.
- Davis and Hwang describe the SIGMA image understanding system [5] for aerial image interpretation. The system uses both forward and backward chaining to arrive at an interpretation, and it represents its object classes hierarchically using frames. Furthermore, the system is able to integrate hypotheses about specific objects in the scene. The system does not use uncertain reasoning but instead is able to control its focus of attention based on the strength of a situation.
- SPAM [6], a system designed by McKeown, Harvey and McDermott is also an aerial image interpretation system. The system originally was constructed to interpret airport scenes but has been expanded with a rule generator so it can now interpret scenes from other domains. SPAM uses confidence values to aid labeling and can manipulate these values based on the consistency of the various labelings.
- VISIONS (Hanson and Riseman [7]) is a blackboard expert system designed to analyze color images. The system uses a flexible control scheme, hierarchical scene

representation, and a number of knowledge sources to accomplish the scene interpretation task. VISIONS is domain independent but uses schemas to tune the system for a particular application.

- The image segmentation expert system developed by Nazif and Levine [8] contains two global memories. The global long term memory contains rules that are applied to the data stored in its short term memory. The system is rule based and uses modules to update lines, regions and areas in the image. The expert system also contains a set of metarules and can control its focus of attention.
- Barnard describes a system that deals with perspective images [9]. The system is able to use the Gaussian sphere to determine the vanishing points of the scene being analyzed. The backprojection of angles and curvatures also is used to aid the interpretation task.
- Barrow and Tenenbaum discuss the problem of interpreting line drawings in [10]. They are able to use junction libraries and knowledge of differential geometry to discriminate between extremal and discontinuity boundaries. This knowledge then is used to determine how the surfaces should be constrained.

PSEIKI differs from the above system in the following three main areas:

Firstly, PSEIKI's task differs from those of previous systems. Most of the other systems were designed to find object instances in the image and, through such discoveries, to arrive at a global interpretation of the image. PSEIKI's task is limited to integrating expected scene information with the observed image -- the result is a set of consistent labels, with associated belief values, for the edge elements in the image.

PSEIKI differs from SPAM and SIGMA, and to a certain extent VISIONS, in not relying on domain-dependent information. For example, SPAM uses airport design knowledge when interpreting airport scenes. Context-cues have also been used extensively in past computer vision systems. For example, if SIGMA has detected a driveway in an image, it would then search for a house and for roads connected to the driveway. Because PSEIKI is provided with a good estimate of the expected scene, it does not have to perform inferences of this type. Although it might be said that context-cues are indispensable for scene interpretation because they make deductions more powerful, their use necessarily introduces some domain dependence. Therefore, it is our philosophy to separate the generation of the mapping from the formation of an overall interpretation of the scene. If the use of context-cues is desired by a system using PSEIKI, then it is up to the higher level system to provide PSEIKI with a line drawing incorporating the information contained in the cues.

PSEIKI also differs from previous systems in its method of performing inexact reasoning. Many systems, including ACRONYM, SIGMA and the system by Nazif and Levine use no uncertain reasoning in the image interpretation process. Because of the overwhelming amount of data in an image, most of the inexact reasoning schemes used in the past have been fairly simple to avoid becoming bogged down in certainty value computations. On the other hand, inexact reasoning in PSEIKI is based on the Dempster-Shafer formalism in a tangled hierarchical space. The use of a hierarchy curtails the number of uncertainty calculations and is made possible by the use of the blackboard architecture.

1.2. OVERVIEW OF PSEIKI

There are two main sections to PSEIKI, a low-level preprocessor which performs pixel-to-symbol conversion and a rule-based edge labeler. PSEIKI's architecture is shown in Fig. 2. Because PSEIKI tries to establish a mapping from the input image to a line drawing of the expected scene, the preprocessor produces an edge-based segmentation. The final output of the preprocessor is a collection of edges detected in the image which are represented as piecewise-linear segments.

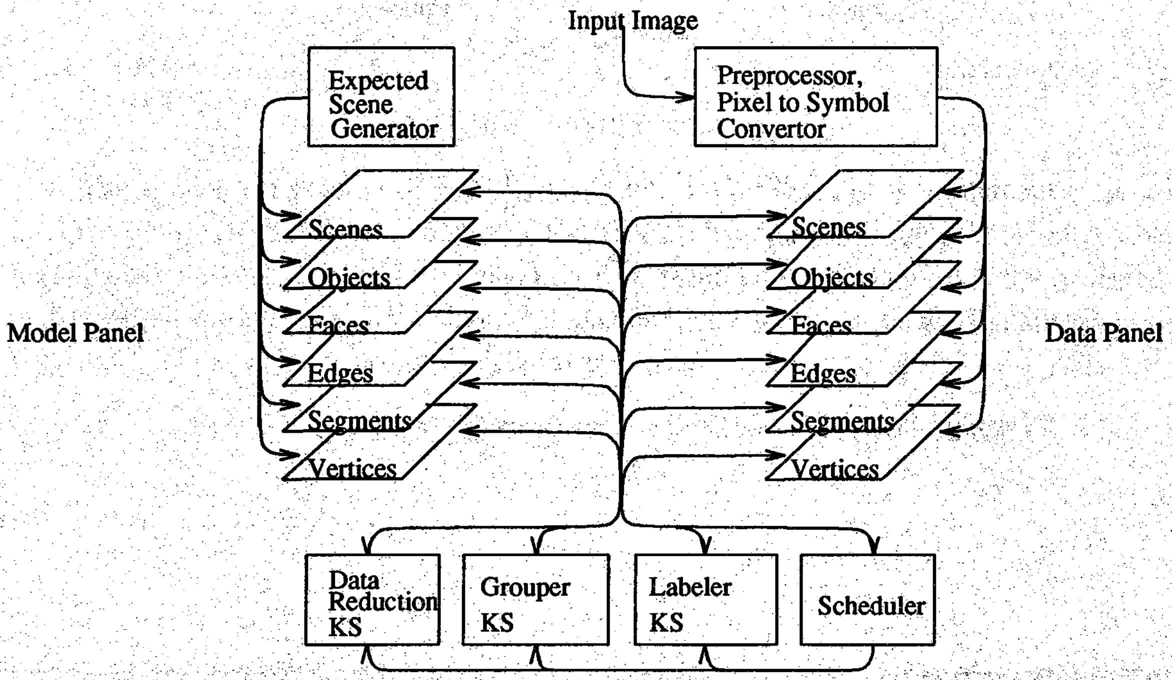


FIGURE 2: PSEIKI's Architecture

The rule based portion of PSEIKI is written in OPS83 and is implemented as a blackboard system. The expected and detected scenes are hierarchically represented on a 2 panel / 6 level blackboard. One panel of the blackboard is reserved for data defining the expected scene and is called the *model* panel. The other panel, called the *data* panel, contains data derived from the input image.* Currently, data on the model panel is static once the expected scene

*Note that these two panels correspond with the Observability and Picture graphs in ACRONYM.

data is deposited on it. However, it is thought that the data in this panel will be dynamic in future versions of the system. The ability to change the model panel could be exploited in future versions of PSEIKI.

For example, if the expected and observed scenes are misregistered by a large amount, then PSEIKI will not be able to establish a complete mapping. However, if it is able to provide the higher level system with a partial mapping, then the system may be able to generate an improved estimate of the expected scene. The improved estimate could then be deposited onto PSEIKI's model panel producing a greater correspondence between the expected and observed scenes. This new information would hopefully aid in the generation of the mapping. Currently, it is not necessary to perform this change because a high degree of correspondence is required between the observed and the expected scenes.

The ability to change the data in the model panel also could be used if multiple images were being fused to provide stereo vision capabilities. In this case, data from a second camera could replace the model information; PSEIKI should be able to use many of the same techniques to perform structural stereo fusion on the data in the two panels.

Each blackboard panel contains the following levels to represent the images: scenes, objects, faces, edges, segments and vertices. Each element in a level is defined by a finite collection of elements on lower levels. For example, a scene is made of a union of objects and a face is defined by the group of edges which form its borders. Figure 3 shows the data on the model panel for a trivial expected scene, a simple block. It shows each element's label and the sub-elements from which it is composed (note that the segment level is omitted).

The following is a short description of the data stored at each level.

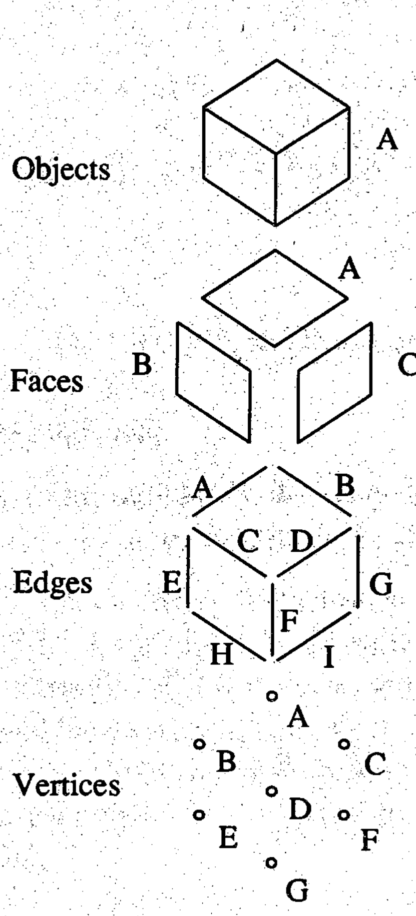
6) **Scenes** -- The entire scene (expected or observed) is represented on this level. The scene is defined as the union of all objects in level 5 of the hierarchy. It provides a way of labeling multiple objects that otherwise would not be possible.

5) **Objects** -- Each element on this level corresponds to a distinct physical object. The objects are defined as the union of all boundary faces from level 4.

4) **Faces** -- The elements on this level represent the polygonal faces that form a boundary representation of the observable portion of the objects. A face is defined by the edges from level 3 which form its border.

3) **Edges** -- These elements form the boundaries of the faces in level 4 of the hierarchy. This level is included to provide a way to compensate for segmentation deficiences. Highly collinear segments from level 2 are grouped to produce an edge in this level.

2) **Segments** -- The piecewise linear segments produced by the low level vision system are represented on this level. It should be noted that this level and the Edge level are identical in the model panel because the line drawing depicting the expected scene should not



$O_A = \{F_A, F_B, F_C\}$

$F_A = \{E_A, E_B, E_C, E_D\}$

$F_B = \{E_C, E_E, E_F, E_H\}$

$F_C = \{E_D, E_F, E_G, E_I\}$

$E_A = \{V_A, V_B\}$

$E_B = \{V_A, V_C\}$

$E_C = \{V_B, V_D\}$

$E_D = \{V_C, V_D\}$

$E_E = \{V_B, V_E\}$

$E_F = \{V_D, V_G\}$

$E_G = \{V_C, V_F\}$

$E_H = \{V_E, V_G\}$

$E_I = \{V_F, V_G\}$

FIGURE 3: Example of Data on Levels of the Model Panel

need to be improved.

1) **Vertices** -- The vertices are the endpoints of the segments and edges from the next two higher levels. Most of the vertices are also provided by the low level vision system.

Levels 1 - 4 are currently implemented; the rest will be implemented in the near future. We have found these four levels to be sufficient for the integration of map knowledge with vision data for a mobile robot with downward slanted cameras, the robot roving over a network of sidewalks and cameras being able to see only the sidewalk and the ground points in its immediate vicinity. However, we do believe that the levels 5 and 6 (and, perhaps, even some further intermediate levels) are important for complex scenes containing 3-D objects in arbitrary orientations.

1.2.1. Reasoning Scheme

PSEIKI has two main sub-processes that it uses to accomplish the goal of establishing a mapping from the observed to the expected scene; these processes are performed by the *labeler* and *grouper* knowledge sources (KS's). These processes can be understood by considering the data on the different levels of the hierarchy. The grouper KS determines which data elements in the lower levels of the hierarchy should be grouped to form a data element on a higher level. Since this process is primarily data driven, it is accomplished via a forward chaining scheme. The second activity that must be accomplished is the labeling of data elements; the labeler KS is used to perform this function. Because this process can be expressed most readily as a goal to be achieved, backward chaining is used.

These two subprocesses are heavily interdependent. For example, when some data elements are grouped into a higher level construct, a goal requesting that this new element be labeled is generated. Conversely, the labeler can also request that the data elements from lower levels be regrouped if it hypothesizes that the regrouping will aid in labeling. Control flow is opportunistic and is currently under investigation.

2. PREPROCESSING AND CONVERSION TO SYMBOLIC FORM.

The preprocessor accepts digitized images from the robot and outputs binary edges represented as piecewise linear segments. A block diagram representation of the preprocessor is shown in Fig. 4.

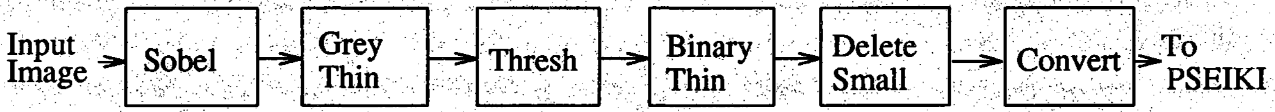


FIGURE 4. Block Diagram of PSEIKI's Preprocessor

To convert the image to a form usable by PSEIKI, the edges in the image are detected by applying a Sobel operator to the digitized gray scale image. These edges are then thinned via Eberlein's algorithm [11] and thresholded. The resulting binary images are thinned again to produce edges that are at most one pixel wide. Small edges are also deleted by the preprocessor. At this point, the image is ready to be converted into symbolic form.

The conversion to symbolic form is accomplished via an algorithm based on the Duda-Hart iterative end-point fit algorithm [12]. In this process, the following steps are performed. First, some pixels are labeled as vertices. The pixels so labeled are edge endpoints and the points at which two or more edges intersect. The edges in the segmented image are then traced from the starting to ending vertices and are represented as broken line segments. The symbolic form of each edge contains the following information: edge number, start vertex, end vertex, length and strength (average gradient magnitude). Likewise, each vertex contains the following information: row coordinate, column coordinate, vertex number and degree. Currently, low level processing is one of the most time consuming activities performed by PSEIKI; it is hoped that the speed of low level system will be increased by replacing much of it with a ridge-following algorithm.

3. EXPECTED SCENE GENERATION

As mentioned before, one way that PSEIKI achieves domain independence is by accepting expected scene information as a line drawing. An obvious method of generating line drawings of an expected scene is via a CAD or computer graphics system; because any type of graphics system could be used to provide this information, it is considered to be separate from PSEIKI proper. However, PSEIKI does require that the graphics system be capable of performing hidden line removal. The fact that PSEIKI can be interfaced with a CAD system can be useful for 2D vision sensing in an integrated manufacturing environment where the same information used to specify a part could also be used by the vision system during the manufacturing process.

When the graphics program presents its data to PSEIKI, PSEIKI requires that the locations of the model vertices be defined first. Higher level constructs (segments, edges, etc.) are then defined by listing the subelements of which they are composed. The vertices of the edges in the expected scene can be specified in two ways: First of all, the locations of the model vertices can be specified in terms of their *world* coordinates. PSEIKI also can accept input in which the vertices of the model are specified by their *pixel* locations. If the input is of this form, PSEIKI uses its camera calibration information and immediately projects the vertices onto a plane (usually the ground plane for mobile robot applications). This projection is done so that PSEIKI can work in the world coordinate frame.

For example, for sidewalk-navigation applications a simple 2D graphics program is used to generate the line drawings of expected scenes from the stored sidewalk maps. To generate the expected scene, the graphics system first accounts for the robot's hypothesized current position and orientation by performing a coordinate transform on the global sidewalk map. Next, a clipping algorithm is applied to determine which edges of the sidewalk should be visible to the robot. Each section of sidewalk is then considered to be a face of a single-object scene. When the graphics system presents the line-drawings to PSEIKI, it passes the vertex coordinates in the world coordinate frame.

4. HIGH LEVEL VISION SYSTEM

Currently, there are three knowledge sources in the PSEIKI system. Besides the grouper and labeler which have been discussed briefly, there is also a data reduction knowledge source. Each of these knowledge sources will be discussed in more detail in this section of the report.

4.1. Data Reduction Knowledge Source

The goal of this knowledge source is to overcome segmentation deficiencies and reduce the amount of data seen by the sections that use inexact reasoning. This KS by itself does not employ an uncertain reasoning scheme. While the process performed by this KS is fairly conservative, it does reduce the amount of data by a significant amount. The need for this KS is dictated by the following two deficiencies in the image segmentation process.

First the segmentation procedure produces artifacts that break lines into smaller line segments. The data-reduction KS tries to compensate for this fact by rejoining these broken line segments. Also, if possible, this KS combines segments that are joined at a degree-two vertex into a single segment. The segmentation process also generates small edges that are caused by noise. Although many of these edges are eliminated during the preprocessing phase, others remain because they are connected to longer segments. These "dangling" edges (all segments which are shorter than a specified length and have a degree one vertex) are eliminated by the data-reduction KS. The actions performed by this KS are shown in Fig. 5.

The overall result of these subprocesses is a cleaner image containing a substantially reduced number of line segments. Experimental results demonstrate that the amount of pruning is greater than 50%; this is obviously a large reduction in the amount of data.

4.2. Labeler Knowledge Source

The second subsystem performs element labeling and confidence estimation. This KS uses the Dempster-Shafer formalism to combine the certainty values used for uncertain reasoning. The combinatorial explosion of uncertainty calculations usually associated with the Dempster-Shafer scheme is avoided by the use of a hierarchical reasoning space.

The hierarchical structure of the blackboard data provides a natural basis for a hierarchical reasoning space. The levels of this space correspond naturally with the levels of the data elements on the blackboard. This can be demonstrated with the help of Fig. 3. For example, assume that F_A on the model panel is composed of edges $\{E_A, E_B, E_C, E_D\}$[*]. Also assume that

[*] Note, in the following discussion the elements on the model panel have capital letters as subscripts while elements on the data panel have numeric subscripts.

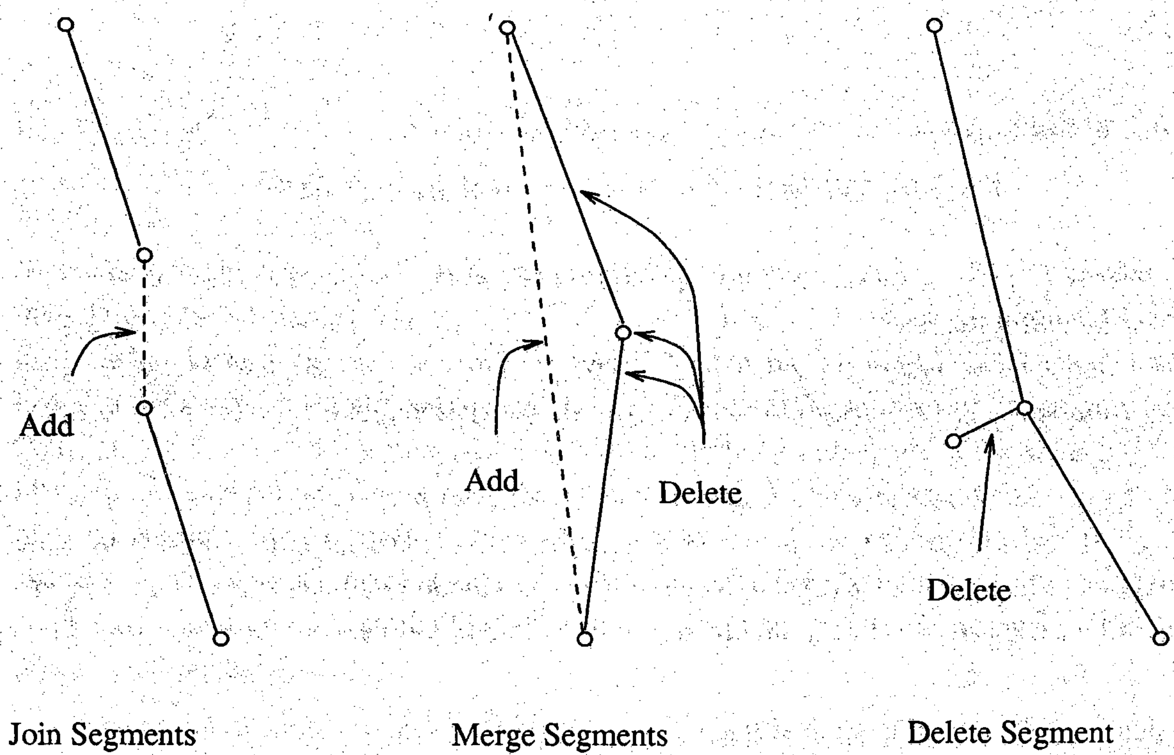


FIGURE 5. Actions performed by the Data Reduction KS

on the data panel edge E_1 is part of the group that composes face F_1. If F_1 is labeled as F_A, then E_1 can be labeled as one of $\{E_A, ..., E_D\}$ only. If the data were not arranged hierarchically, it would be necessary to consider every element on the model panel when assigning labels and when performing consistency checks. To curtail the number of uncertainty calculations, consistency checks are not made directly between two elements at the same level of the hierarchy if their parents do not have the same label. In the previous example, edge E_1 would only be checked for consistency with edges whose parents were also labeled as F_A. Consistency checks between two non-siblings can be made indirectly by propagating an element's confidence value up through the hierarchy until a common ancestor is reached and then back down to the second element.

The frame of discernment** (FOD) for any element is defined by the labels which could be given to the element; this is determined by the label of the element's parent. In the example above, since F_1 is labeled as F_A, the frame of discerment for edge E_1 would be

** Some terminology used in the Dempster-Shafer formalism will be defined here. The *frame of discernment*, Θ, for an object is the set of all possible values that the object may assume. A *basic probability number* for a subset of Θ is the amount of belief assigned that subset and which cannot be further subdivided. The *basic probability assignment* (bpa) for that object over Θ is the collection of the basic probability numbers for all possible subsets of Θ. See Shafer [3] for further explanations.

$$\Theta = \{E_A, E_B, E_C, E_D\} \tag{1}$$

An element's label is defined to be the label of the element from its frame of discernment which has the greatest belief value attached to it. If the belief value of an element on an upper level of the hierarchy is changed, then all of its descendents must change their FOD. Thus, it is advantageous to perform compatibility checks between elements on upper levels of the hierarchy first to avoid performing unnecessary calculations on lower levels when frames of discernment are changed. This necessity for checking global consistency before checking local consistency seems reasonable.

Two metrics are required when updating the label belief functions for elements on any level of the hierarchy. The two metrics must provide measures of the *compatibility* and the *incompatibility* between two elements. To facilitate the correspondence between them and certainty values, both metrics should range between 0.0 and 1.0. Obviously, the metrics need not be the same for all levels of the hierarchy.

The two metrics need only provide a measure of the (in)compatibility of two elements if they are believed to correspond to the same model element. For example, if E_1 and E_2 are edges in the data panel and are thought to correspond to the same model edge, then **collinearity**(E_1, E_2) is the measure of compatibility between them. We have defined collinearity as:

$$\text{collinearity}(E_1, E_2) = \frac{D_{max} - D_{seg}}{D_{max}} \times \cos(\Theta) \tag{2}$$

Where Θ is the angle between the two segments; D_{seg} is the distance from the middle of E_2 to E_1 and D_{max} is the maximum possible distance between the two segments (see Fig. 6).

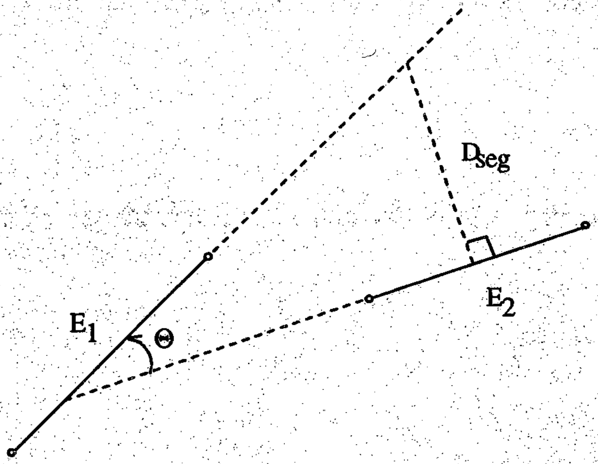


FIGURE 6. Geometry used in definition of collinearity

Likewise the measure of incompatibility for the segment and edge levels,

noncollinearity(E_1, E_2), can be defined as:

$$\text{noncollinearity}(E_1, E_2) = \frac{D_{seg}}{D_{max}} \times \text{scale}(E_1) \times \sin(\Theta) \tag{3}$$

where scale(E_1) depends on the length of E_1[*]. When establishing the initial belief function for a data element, it is desirable to be able to find the compatibility between it and elements in the model panel; we shall discuss how this is done in a later section of the report.

If two elements correspond to different model elements, a rigid motion transform is applied to one of them before the metric is applied. This has the effect of enforcing relational constraints between the two data elements. For example, if edges E_1 and E_3 are thought to correspond to model edges E_A and E_B respectively, then the measure of compatibility between E_1 and E_3 would be defined as

$$\text{compatibility}(E_1, E_3) = \text{collinearity}(E_1, T_{E_B \to E_A}(E_3)) \tag{4}$$

where $T_{E_B \to E_A}$ is the rigid motion transformation that makes model edge E_B collinear with model edge E_A. In other words,

$$\text{collinearity}(E_A, T_{E_B \to E_A}(E_B)) = 1.0 \tag{5}$$

For example, if we assume that E_1 is labeled as E_A, then when using data element E_2 to update the belief function for data element E_1 the new evidence is defined to be

$$m_{update}(\{E_A\}) = SF \times \text{compatibility}(E_2, E_1) \tag{6}$$

$$m_{update}(\{\neg E_A\}) = SF \times \text{incompatibility}(E_2, E_1) \tag{7}$$

Where SF is a scale factor ($0.0 \leq SF \leq 1.0$) that determines the maximum amount of evidence that can be obtained by checking data element compatibility. The symbol "$\neg$" as used in this context denotes set negation (i.e. $\{\neg E_A\} = \Theta - \{E_A\}$). Once m_{update} is computed, Dempster's rule is used to combine it and the original bpa to yield the new bpa for the element.

4.2.1. Computing Initial Belief Values for Elements in a Group

To more fully describe how belief values are computed for, say, the edge elements at level 3 for the data panel in Fig. 2, consider the following example. Let's say that at this level, the information in the model panel and the data panel is as shown in Fig. 7.

The grouper KS (to be described later) will hypothesize that the edges $\{E_1, ..., E_9\}$ should be grouped together; let this grouping be designated by the face F_1. In order to label the edges in the data, the labeler KS will now construct the following frame of discernment for each edge

[*] The scale factor is provided to limit the amount disconfirmatory evidence provided by small edges which may be due to noise.

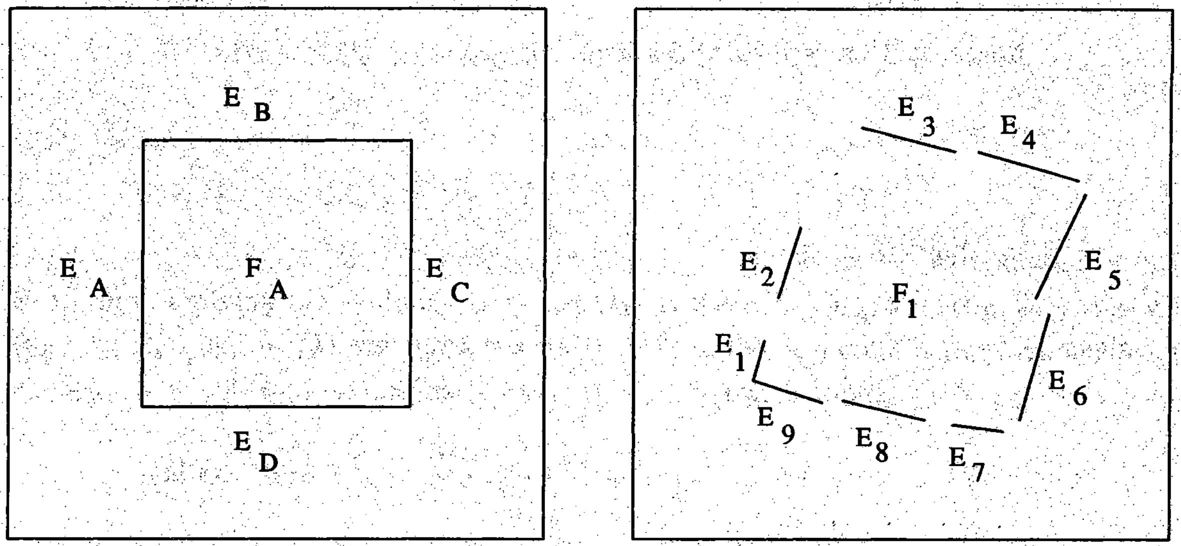


FIGURE 7. Example data on the edge level of the blackboard

in the group:

$$\Theta = \{E_A, E_B, E_C, E_D\} \tag{8}$$

Let's now focus on the labeling process for edge E_1. The labeler KS will compute a bpa over Θ by first applying the collinearity procedure to the pairs (E_A, E_1), (E_B, E_1), (E_C, E_1) and (E_D, E_1). In other words, we check the collinearity of the data edge in question against all the elements in Θ. To find the collinearity of, say, (E_A, E_1) we compute

$$\text{collinearity}(E_A, E_1) = \frac{D_{max} - D_{seg}}{D_{seg}} \times \cos(\theta) \tag{9}$$

where, as shown in Fig. 8, D_{max} is the maximum allowable distance between the two segments, D_{seg} is the distance from the middle of E_A to E_1, and θ is the angle between the segments. Note that although E_A and E_1 exist in different frames, it is possible to speak of distances and angles between them because, for the purposes of blackboard processing, they are both projected into the same world coordinate system. For a mobile robot navigating on a flat ground plane, all the edges are transformed onto the ground plane.

Let's assume, for the sake of discussion, the collinearity calculation produces the following results for E_1.

$$\begin{aligned} \text{collinearity}(E_A, E_1) &= 0.7 \\ \text{collinearity}(E_B, E_1) &= 0.1 \\ \text{collinearity}(E_C, E_1) &= 0.4 \\ \text{collinearity}(E_D, E_1) &= 0.05 \end{aligned} \tag{10}$$

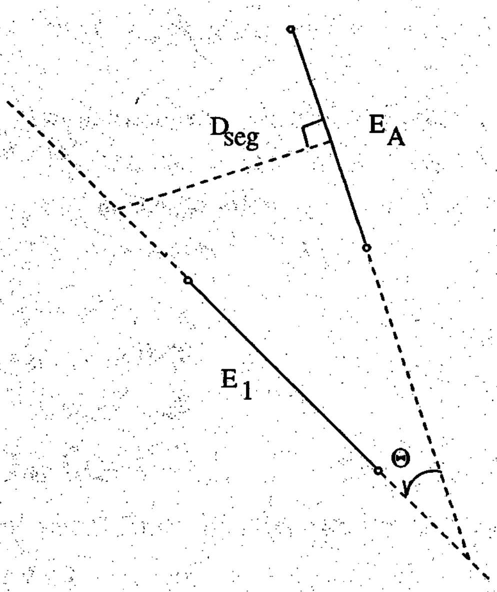


FIGURE 8. Collinearity geometry for the computations of initial bpa's

Note that although E_1 is approximately parallel to both E_A and E_C, the collinearity calculation yields a larger result for E_A because of the distance dependence of the calculation. If the sum of the collinearity measures exceeds unity, as is the case in our example, they are normalized by the summed value. Therefore, in our example

$$\begin{aligned}&\text{collinearity}(E_A, E_1) = 0.48\\&\text{collinearity}(E_B, E_1) = 0.08\\&\text{collinearity}(E_C, E_1) = 0.28\\&\text{collinearity}(E_D, E_1) = 0.04\end{aligned} \tag{11}$$

These collinearity measures define a bpa over Θ, for our example as given by

$$\begin{aligned}&m_{E_1}(\{E_A\}) = 0.48\\&m_{E_1}(\{E_B\}) = 0.08\\&m_{E_1}(\{E_C\}) = 0.28\\&m_{E_1}(\{E_D\}) = 0.04\\&m_{E_1}(\cdot) = 0.0 \text{ for all other subsets of } \Theta\end{aligned} \tag{12}$$

A slightly different approach is taken when the sum of collinearity measures is less then 1. Assume for a moment that due to the distance cutoff, D_{max}, the collinearity measure of (E_C, E_1), is zero. Now the collinearity measures are

$$\begin{aligned}&\text{collinearity}(E_A, E_1) = 0.7\\&\text{collinearity}(E_B, E_1) = 0.1\\&\text{collinearity}(E_C, E_1) = 0.0\\&\text{collinearity}(E_D, E_1) = 0.05\end{aligned} \tag{13}$$

Since the measures now sum to less then unity, there is no reason to normalize. Instead, we now convert them directly into bpa's in the following manner

$$\begin{aligned} &m_{E_1}(\{E_A\}) = 0.7 \\ &m_{E_1}(\{E_B\}) = 0.1 \\ &m_{E_1}(\{E_C\}) = 0.0 \\ &m_{E_1}(\{E_D\}) = 0.05 \\ &m_{E_1}(\Theta) = 0.15 \\ &m_{E_1}(\cdot) = 0.0 \text{ for all other subsets of } \Theta \end{aligned} \tag{14}$$

Note that we have now assigned 0.15 belief to Θ, 0.15 being what's left after we subtract from 1 the sum of collinearity measures. This seems intuitively plausible for the simple reason that collinearity is a good measure of "E_1 is a part of E_A." Clearly, if E_1 does not match with any of E_A, E_B, E_C or E_D to a sufficiently high degree then we may leave some belief uncommitted. In the above assignment, $m_{E_1}(\Theta) = 0.15$ represents the uncommitted portion of our belief.

Let's say that the procedure just described yields the following bpa's for all the data edges:

$$\begin{aligned} &m_{E_1}(\{E_A\}) = 0.7 \;\; m_{E_1}(\{E_B\}) = 0.1 \;\; m_{E_1}(\{E_C\}) = 0.0 \;\; m_{E_1}(\{E_D\}) = 0.05 \;\; m_{E_1}(\Theta) = 0.15 \\ &m_{E_2}(\{E_A\}) = 0.8 \;\; m_{E_2}(\{E_B\}) = 0.1 \;\; m_{E_2}(\{E_C\}) = 0.0 \;\; m_{E_2}(\{E_D\}) = 0.00 \;\; m_{E_2}(\Theta) = 0.1 \\ &\cdots \\ &\cdots \\ &\cdots \\ &m_{E_9}(\{E_A\}) = 0.2 \;\; m_{E_9}(\{E_B\}) = 0.1 \;\; m_{E_9}(\{E_C\}) = 0.1 \;\; m_{E_9}(\{E_D\}) = 0.60 \;\; m_{E_9}(\Theta) = 0.0 \end{aligned} \tag{15}$$

These then constitute the initial bpa's for the data edges.

4.2.2. Revising Belief by Enforcing Mutual Collinearity Constraints

From the set of initial bpa's, the labeler KS then seeks those observed edges that have maximum bpa for the same model edge. For illustration, in the example here, E_1 and E_2 exhibit maximal beliefs for the same model edge, E_A.

The edge-labeler will now apply a local consistency enforcer to the belief values. Let's say that since E_1 and E_2 both have maximal similarities with E_A, we wish to use E_2 to revise our beliefs regarding E_1. To do so, the labeler will measure the collinearity of E_1 and E_2, and then the noncollinearity of the same two edges. Let's say we get

$$\text{collinearity}(E_2,E_1) = 0.8 \tag{16}$$

$$\text{noncollinearity}(E_2,E_1) = 0.1 \tag{17}$$

We now construct an "updating" bpa for E_1 as follows. By multiplying $m_{E_2}(\{E_A\})$ with 0.8, we obtain

$$m_{\underset{2\to1}{\text{update}}}(\{E_A\}) = m_2(\{E_A\})\text{collinearity}(E_2, E_1) \times SF \tag{18}$$

$$= 0.64 \quad \text{if } SF = 1.0$$

$$m_{\underset{2\to1}{\text{update}}}(\{\neg E_A\}) = m_{E_2}(\{E_A\})\text{noncollinearity}(E_2, E_1) \times SF \tag{19}$$

$$= 0.08 \quad \text{if } SF = 1.0$$

$$m_{\underset{2\to1}{\text{update}}}(\Theta) = 0.28 \tag{20}$$

Where once again, the bpa for the frame of discernment was set to the uncommitted portion of belief. To find the new bpa over Θ for E_1, we now use Dempster's rule to combine m_{E_1} with $m_{\underset{2\to1}{\text{update}}}$ to obtain

$$\begin{aligned} m_{E_1}(\{E_A\}) &= 0.87 \\ m_{E_1}(\{E_B\}) &= 0.04 \\ m_{E_1}(\{E_C\}) &= 0.0 \\ m_{E_1}(\{E_D\}) &= 0.02 \\ m_{E_1}(\{\neg E_A\}) &= 0.02 \\ m_{E_1}(\Theta) &= 0.05 \end{aligned} \tag{21}$$

4.2.3. Revising Belief Values by Enforcing Relational Constraints

The labeler KS also contains procedures for enforcing relational constraints within each grouping; this is done in the following manner. Consider the bpa's over Θ for the edges E_1 and E_9 as illustrated in Fig. 7. While the bpa for E_1 takes its maximum value for the label E_A, the bpa for E_9 is a maximum at E_D. Let's say we wish to compute, using relational constraints, the bpa $m_{\underset{9\to1}{\text{update}}}$, which is the additional belief generated by E_9 for E_1's label. To find $m_{\underset{9\to1}{\text{update}}}$, we note that the geometrical relationship between E_A and E_D is already known, since their positions in the world coordinate system are known. Let $T_{E_A \to E_D}$ be the transform that makes the model edges E_A and E_D coincident. To get a measure of the extent to which the geometrical relationship between E_1 and E_9 is the same one as the one that exists between E_A and E_D, the labeler carries out the following (in)compatibility computations:

$$\text{compatibility}(E_9, E_1) = \text{collinearity}(E_9, T_{E_A \rightarrow E_D}(E_1)) \tag{22}$$

$$\text{incompatibility}(E_9, E_1) = \text{noncollinearity}(E_9, T_{E_A \rightarrow E_D}(E_1)) \tag{23}$$

Clearly, if compatibility$(E_9, E_1) = 1.0$ (when that happens, we will also have incompatibility$(E_9, E_1) = 0.0$), that would imply that the geometrical relationship between E_1 and E_9 in the data is exactly the same as between E_A and E_D in the model. To explain how the relational information is used to update the bpa distribution for, say, E_1, we will go back to our example that started with Fig. 7. Let's say that the compatibility calculations yielded the following results for computing the contribution of E_9 to the belief values for E_1:

$$\begin{aligned} \text{collinearity}(E_9, E_1) &= 0.7 \\ \text{noncollinearity}(E_9, E_1) &= 0.4 \end{aligned} \tag{24}$$

Since these measures sum to more then 1, they are normalized by their sum to yield the final (in)compatibility measures

$$\begin{aligned} \text{collinearity}(E_9, E_1) &= 0.64 \\ \text{noncollinearity}(E_9, E_1) &= 0.36 \end{aligned} \tag{25}$$

We now construct another bpa for E_1 as follows. The quantity $m_{E_9}(\{E_B\})$ is multiplied by 0.64 to yield

$$\begin{aligned} m_{\underset{9 \rightarrow 1}{\text{update}}}(\{E_A\}) &= m_{E_9}(\{E_A\})\text{compatibility}(E_9, E_1) \times SF \\ &= 0.13 \quad \text{if } SF = 1.0 \end{aligned} \tag{26}$$

$$\begin{aligned} m_{\underset{9 \rightarrow 1}{\text{update}}}(\{\neg E_A\}) &= m_{E_9}(\{E_A\})\text{incompatibility}(E_9, E_1) \times SF \\ &= 0.07 \quad \text{if } SF = 1.0 \end{aligned} \tag{27}$$

$$m_{\underset{9 \rightarrow 1}{\text{update}}}(\Theta) = 0.2 \tag{28}$$

By using Dempster's rule to combine this update bpa with the one in (21), we obtain a further revision of the belief values over the frame of discernment for E_1.

4.2.4. Combining Belief Revisions from Relational Constraints and Mutual Collinearity Constraints

Actually, a single procedure is used for enforcing both the local collinearity constraints and the relational constraints within a group. Note that the (in)compatibility calculations from relational constraints reduce to the computation of local collinearity measures, as described in the beginning of Section 4.2, if we use an identity transformation for $T_{E_X \rightarrow E_Y}$.

Therefore, we are able to employ the following integrated algorithm for enforcing both the relational constraints and the local collinearity constraints. Suppose n edges, $E_1, E_2, ..., E_n$, have been grouped together in the data panel and are hypothesized to belong to a single face. Also, suppose that the edges in one of the possible corresponding faces in the model panel are $E_A, E_B, ..., E_Z$. Clearly, the frame of discernment, Θ, for each edge is

$$\Theta = \{E_A, ..., E_Z\} \tag{29}$$

Let's further say that, as described in Section 4.2.1, measuring local collinearities of the data edges against those of the model edges yields the following initial bpa distribution for each data edge:

$$m_{E_i}(\{E_\alpha\}) \quad \text{for} \quad \begin{matrix} i = 1, ..., n \\ \alpha = A, ..., Z \end{matrix} \tag{30}$$

with the assumption that initially

$$m_{E_i}(\Theta) = 1 - \sum_{\alpha=A}^{Z} m_{E_i}(\{E_\alpha\}) \tag{31}$$

$$\text{and} \quad m_{E_i}(\cdot) = 0.0 \quad \text{for all other non-singleton subsets of } \Theta$$

For each data edge E_i, let $E_{\alpha_{max}}$ be the model edge for which the bpa takes a maximum value. That is,

$$m_{E_i}(\{E_{\alpha_{max}}\}) \geq m_{E_i}(\{E_\alpha\}) \quad \text{for } \alpha = A, ..., Z \tag{32}$$

Since for each edge E_i there exists a single model edge, $E_{\alpha_{max}}$, with maximum bpa, for every data edge E_i, we can denote the maximum model edge as $E_{\alpha_{max}(i)}$. (in case there is a tie due to two or more model edges yielding the same value for bpa, we arbitrarily select one of them).

When incorporating new evidence for an element's label, we can ease the computational load by making use of a property of Dempster's rule of combination. If we update an element's bpa incrementally with every piece of new evidence from element's siblings, then the new bpa will be computed as

$$m_{new} = (((m_{old} \oplus m_{\underset{1\rightarrow i}{update}}) \oplus m_{\underset{2\rightarrow i}{update}}) \oplus \cdots \oplus m_{\underset{n\rightarrow i}{update}}) \tag{33}$$

Where $\oplus$ is denoting Dempster's rule of combination and $m_{\underset{j\rightarrow i}{update}}$ is the updating bpa for the new evidence that element j is providing for element i. Since Dempster's rule of combination is invariant with respect to the order of combination, the new bpa can be expressed as

$$m_{new} = m_{old} \oplus m_{update} \tag{34}$$

where

$$m_{update} = ((m_{\underset{1\to i}{update}} \oplus m_{\underset{2\to i}{update}}) \oplus \cdots \oplus m_{\underset{n\to i}{update}}) \tag{35}$$

Now we can take advantage of the fact that the updating bpa's are binary frames of discernment (if element E_i is labeled as E_A then the only nonzero elements correspond to the subsets $\{E_A\}$, $\{\neg E_A\}$ and Θ itself). Barnett [13] describes how binary frames of discernment can be used to allow the combination of bpa's with linear time complexity (with respect to the size of the FOD). Currently, we do not use Barnett's formulas, but we do take advantage of the fact that there are now 4 elements in the power set of the updating frame of discernment instead of $2^{|\Theta|}$ elements.

A function, **update-belief**, used to provide an updating bpa for an arbitrary element, E_i, using compatibility and incompatibility measures is shown in Fig. 9. It takes the element being updated as a parameter and it returns the updating bpa for that element. Thus the bpa for E_i could be updated with the following function call

$$m_{E_i} = m_{E_i} \oplus \textbf{update-belief}(E_i) \tag{36}$$

4.2.5. Evidence Propagation Between Levels in the Hierarchy

Mutual consistency constraints and geometric constraint relations are not the only sources of knowledge used to update an element's belief function. A mechanism is also provided for passing belief values between different levels of the hierarchy. This is done to satisfy the intuitive argument that says any evidence confirming an element's label also should provide evidence that its parent's label is correct (for example, it is very unlikely that an edge be labeled correctly, yet be grouped into a face that is incorrectly labeled). Conversely, it is also intuitively pleasing that disconfirming evidence be passed to lower levels of the hierarchy (e.g. if we think that a face is mislabeled, then all of its constituent edges are also most likely mislabeled).

We use the updating bpa, m_{update}, as computed by the **update-belief** function when passing evidence up the hierarchy. To do this, we combine m_{update} not only with the bpa for the element in question, but also with that element's parent. Combining the updating bpa with an element's parent makes intuitive sense because all new evidence generated on a level comes from the the (in)compatibility between elements on that level. If the children of an element have consistent (compatible) labels, then these child elements should provide evidence that the label given to that element is correct. Likewise children with inconsistent labels provide

```
function update-belief ($E_i$)
    ;; initialize updating bpa as a vacuous belief function
    $m_{update}(\Theta) = 1.0$
    $m_{update}(\cdot) = 0.0$        ; all other subsets
    ;; use all n elements in the data panel with the same parent group
    ;; when updating $E_i$
    For each $j = 1, 2, ..., n$
        begin
            $c_j = compatibility(E_i, E_j)$
            $d_j = incompatibility(E_i, E_j)$
            ;; normalize (in)compatibility measures if needed
            if $(c_j + d_j > 1.0)$
                begin
                    $c_j = \frac{c_j}{c_j + d_j}$
                    $d_j = \frac{d_j}{c_j + d_j}$
                end
            ;; create updating bpa created by element $E_j$
            $m_{\underset{j \to i}{update}}(\{E_{\alpha_{max}(i)}\}) = c_j \times m_{E_i}(\{E_{\alpha_{max}(i)}\})$
            $m_{\underset{j \to i}{update}}(\{\neg E_{\alpha_{max}(i)}\}) = d_j \times m_{E_i}(\{\neg E_{\alpha_{max}(i)}\})$
            $m_{\underset{j \to i}{update}}(\Theta) = 1.0 - m_{\underset{j \to i}{update}}(\{E_{\alpha_{max}(i)}\}) - m_{\underset{j \to i}{update}}(\{\neg E_{\alpha_{max}(i)}\})$
            ;; accumulate into updating bpa
            $m_{update} = m_{update} \oplus m_{\underset{j \to i}{update}}$
        end        ; (for loop)
    ;; return the accumulated sum of new evidence
    return ( $m_{update}$ )
End.    ; update-belief
```

FIGURE 9: algorithm used to generate updating bpa

evidence that an element's label is incorrect. Thus, by passing the updating bpa's to each parent element on a higher level of the hierarchy, we are providing new evidence for those elements based on the consistency or the inconsistency of its descendents.

Evidence from an element cannot be applied directly to its parent because the FOD's of an element and its parent are composed of different types of objects. However, we will show with the help of an example that it is possible to build a FOD that can be used to update the belief functions of elements on a higher level of the hierarchy. Assume for our example that the model panel is shown in Fig. 3 and that edge E_1 on the data panel is a child of face F_1. Furthermore, assume that E_1 is labeled as E_A and F_1 has label F_A. Because the basic probability number for edge E_1 given by $m_{\underset{E_1}{update}}(\{E_A\})$ arises from the consistency of E_1's label with its sibling's labels, it may be considered as a weighted vote of confidence that face F_1's label is correct. Likewise, because $m_{\underset{E_1}{update}}(\{\neg E_A\})$ arises from the inconsistency of E_1's label with its sibling's labels, it may be considered as a (weighted) vote of no confidence in face F_1's label. Thus, $m_{\underset{E_1}{update}}(\Theta)$ can be considered to be the amount of ignorance about F_1's label. Using this rational, we may define an updating bpa for face F_1 with the following non-zero basic probability numbers

$$\begin{aligned}
&m_{\underset{E_1 \to F_1}{update}}(\{F_A\}) = m_{\underset{E_1}{update}}(\{E_A\}) \\
&m_{\underset{E_1 \to F_1}{update}}(\{\neg F_A\}) = m_{\underset{E_1}{update}}(\{\neg E_A\}) \qquad (37) \\
&m_{\underset{E_1 \to F_1}{update}}(\Theta) = m_{\underset{E_1}{update}}(\Theta)
\end{aligned}$$

Now we will show that $m_{\underset{E_1 \to F_1}{update}}$ is a bpa for F_1. As described in the last section, after initial belief values are assigned to the FOD for an element's updating bpa is really binary in nature. Thus the only non-zero elements of the updating bpa for E_1 are $m_{\underset{E_1}{update}}(\{E_A\})$, $m_{\underset{E_1}{update}}(\{\neg E_A\})$, and $m_{\underset{E_1}{update}}(\Theta)$. Because these are the only non-zero basic probability numbers from E_1's bpa, they sum to 1. Given this fact, it is trivial to show that $m_{\underset{E_1 \to F_1}{update}}$ is also a bpa. We can now express the total accumulated new belief for face F_1 from its children $E_1, E_2, \ldots, E_n$ as

$$m_{\underset{F_1}{update}} = ((m_{\underset{E_1 \to F_1}{update}} \oplus m_{\underset{E_2 \to F_1}{update}}) \oplus \cdots \oplus m_{\underset{E_n \to F_1}{update}}) \qquad (38)$$

Information is passed down the hierarchy only if it is disconfirmatory. Currently, this downward propagation of information takes the form of the reassignment of frames of discernment caused by the ancestor of an element having its label changed. In the previous example, this could happen if the label for face F_1 is changed to F_B. Using information from the model panel (as shown in Fig. 3) we note that the FOD for E_1 would be reassigned to

$$\Theta = \{E_C, E_E, E_F, E_H\} \qquad (39)$$

4.3. Grouper Knowledge Source

The grouper knowledge source builds the data elements on the upper levels of the hierarchy from the segments and vertices deposited by the low level vision system. It does this in a data-driven manner by first grouping segments into edges and then grouping the edges into faces, etc. The KS is triggered by a request to find the parent of a *seed* element. After the KS is triggered, it must find elements that could be members of the same group and then form the parent element on the next higher level of the hierarchy. To create the parent element, the grouper must first find all of its children. After the children are identified, they are grouped into the parent element.

In general, two elements on the same level must satisfy two requirements if they are to be grouped together. First, the elements must satisfy a level-specific adjacency constraint. These constraints usually force the KS to consider only elements that are physically close when it forms groups. The other requirement is obtained from the compatibility metric used by the labeler KS; i.e., the elements must be highly consistent in order to be grouped. For example, two segments must be highly collinear and two edges must lie on a common plane to be grouped.

Although the KS should be able to group elements based solely on their geometry, it must be able to use any label information that the labeler KS has provided. PSEIKI's grouper KS currently does this by refusing to group data elements with an incompatible labels.

Although eventually the grouper and labeler KS's work in concert on an opportunistic basis, some initial groupings must be formed for the labeler KS to act at all. In other words, the grouper KS must be able to generate some initial groupings. To illustrate how this is done, consider the example in which the low-level preprocessor has deposited the data shown on the right of Fig. 10 on the segment level of the data panel. Information about the expected scene on the model panel is shown on the left.

Again, remember that the information in both the model and the data panels is in the same world coordinate frame. For a mobile robot with cameras slanted downwards, this corresponds to the flat ground plane. For generating initial groupings, every segment in the data panel is compared with all the model edges that are in the vicinity of the segment; the basis of comparison is collinearity. For each data segment, we retain that model edge label that yields the highest value for collinearity. In Fig. 11, the left hand side shows the data segments and the right hand side a possible label for each segment. All the adjoining segments that have the same initial labels are now joined together (in a manner similar to what is accomplished by the data-reduction KS). Joining by replacing segments by edges, the result for our example is shown in Fig. 12. Now it is a simple matter to generate initial groupings for the data. Using the labeling shown in the right hand side of Fig. 11, we deduce that the data edges as shown in Fig. 12. must be grouped in the following manner.

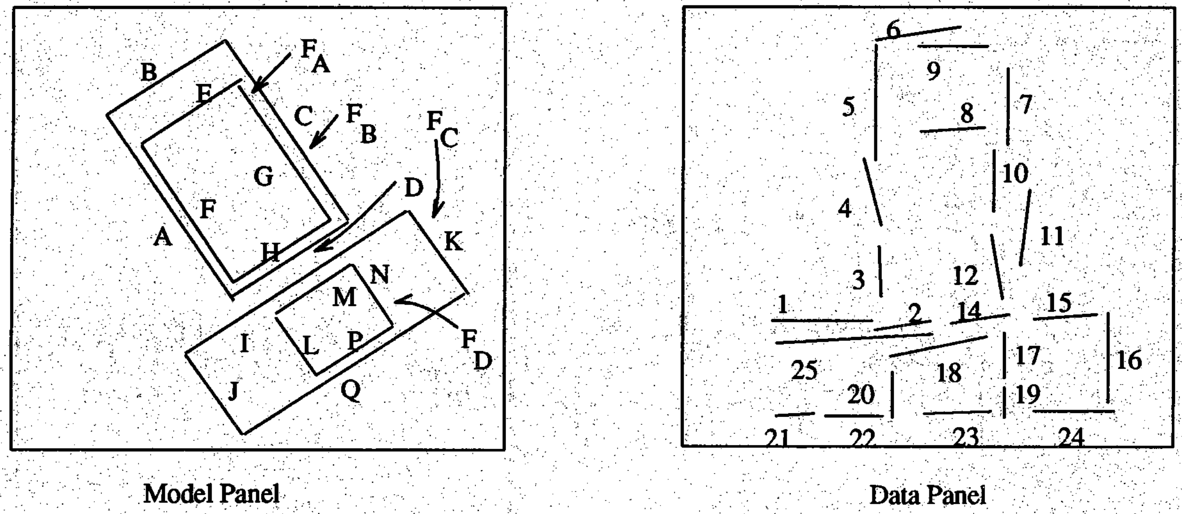


FIGURE 10. Example data presented to the grouper KS on the segment level

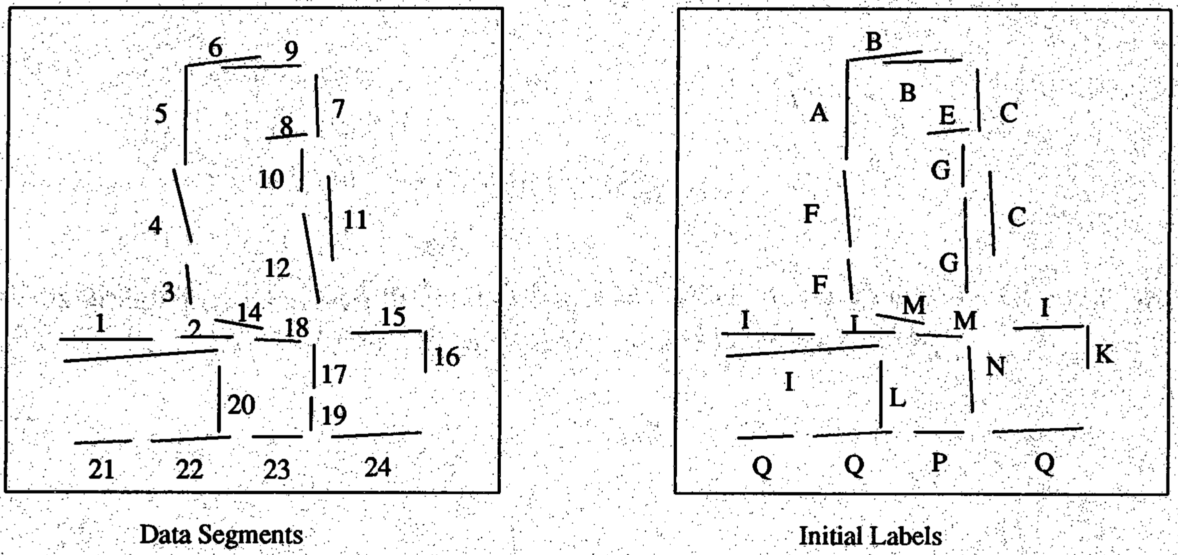
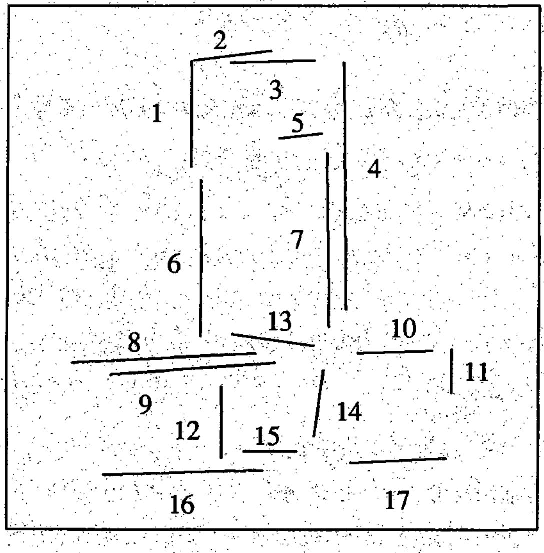


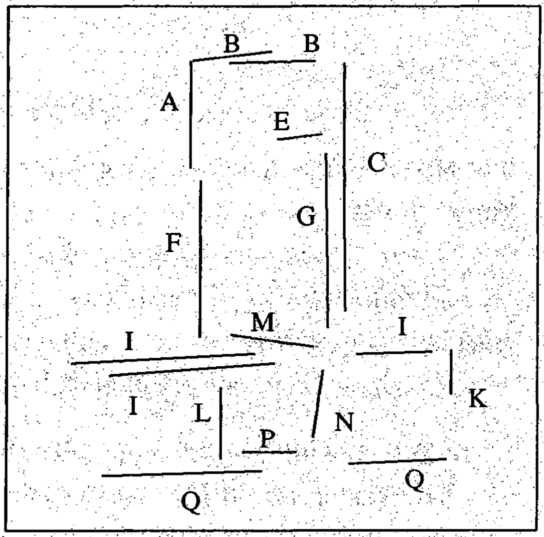
FIGURE 11. Initial labels for data segments

$$
\begin{aligned}
F_1 &= \{E_5, E_6, E_7\} \\
F_2 &= \{E_1, E_2, E_4, E_3\} \\
F_3 &= \{E_8, E_{11}, E_{16}, E_{17}, E_9, E_{10}\} \\
F_4 &= \{E_{12}, E_{13}, E_{14}, E_{15}\}
\end{aligned}
\tag{40}
$$

This grouping is then used by the labeler. The labeler, by enforcing mutual collinearity and relational constraints will drop E_2 or E_3 from F_2; and E_8 or E_{10} from F_3.



Data Edges



Labels

FIGURE 12. Edges with initial labels derived from data segments

4.3.1. Mechanisms That Initiate Relabeling and Regrouping

The labeler KS used in PSEIKI uses backward chaining to establish initial labels and belief values of data elements. Although this procedure works satisfactorily during initial labeling, the processing needs to become more opportunistic when labels must be reassigned or when elements must be regrouped. Two of the mechanisms that trigger relabeling and regrouping will be discussed here. Theses triggers are generated by the enforcement of relational consistencies at the level of edges and above in the data panel of the blackboard.

The first mechanism enforces the constraint that elements that are non-intersecting in the model should also be non-intersecting after they have been processed by the labeler KS. To illustrate, suppose the labeler has produced the following two groupings of edges into faces.

$$F_3 = \{E_8, E_{11}, E_{16}, E_{17}, E_9, E_{10}\} \tag{41}$$

$$F_4 = \{E_{12}, E_{13}, E_{14}, E_{15}\} \tag{42}$$

Since theses two groupings are intersecting, while their corresponding model faces are not, the labeling KS is set to work again on the initial set

$$\{E_{10}, E_9, E_8, E_{11}, E_{17}, E_{16}\} \tag{43}$$

to generate an alternative grouping. For generating alternative groupings, we take a set difference of groupings in (42) and (43); we thus find the competing edge elements that did not get included in (43). In our example, the edge element E_8 competes with E_{10} since the former did not get included in (42) and is parallel and proximal to the latter. We therefore now delete E_{10} from (43) and send the result back to the labeler.

Another mechanism for initiating relabeling and regrouping uses the notion of minimum bounding rectangles (MBRs). After a set of edges is grouped together, the area of the minimum bounding rectangle of the group is compared with the area of the MBR of the face in the model panel. If the area of the minimum bounding rectangle in the data is the larger of the two, one of the edges is *released* from the grouping with other faces, especially those that are adjoining and whose data MBR area is less then what it should be.

As can be seen from the discussion here, a part of the grouper KS's responsibility is to exchange edges between neighboring faces in the data panel. If face attributes, as exemplified by MBR, and face relational constraints cannot be satisfied by regrouping and relabeling, then the offending edges are discarded and new edges are formed from the segment data.

5. EXPERIMENTAL RESULTS

PSEIKI was run on data from a number of digitized images typical of those gathered by a mobile robot; the images in Fig. 13 show partial results for a typical scene. The example depicts a scene typical of what the robot would see when approaching an intersection. Figure 13a shows the edges representing the expected scene; each edge's label is indicated also. The observed scene is shown in Fig. 13b; note that it differs from the expected by a significant amount. Because the robot was slightly to the right of its expected position and its orientation also was slightly off, two edges that are in the expected scene are missing entirely. Shadows on the sidewalk are also a problem that the robot encounters frequently; the example also contains this type of degradation.

The other images in Fig. 13 show partial results produced by PSEIKI. The output of the preprocessor is shown in Fig. 13c. Figure 13d shows the output of the data reduction module; it was able to reduce the number of segments to 59 from 183 in the original image. The final result, Fig. 13e, shows the edges and labels provided by PSEIKI. Even though the expected and observed scenes differed by a significant amount, PSEIKI was able to label all but one edge. As Fig. 13 demonstrates, the results that PSEIKI produces are sufficient to provide a higher level navigation system the feedback need to successfully navigate a known sidewalk map.

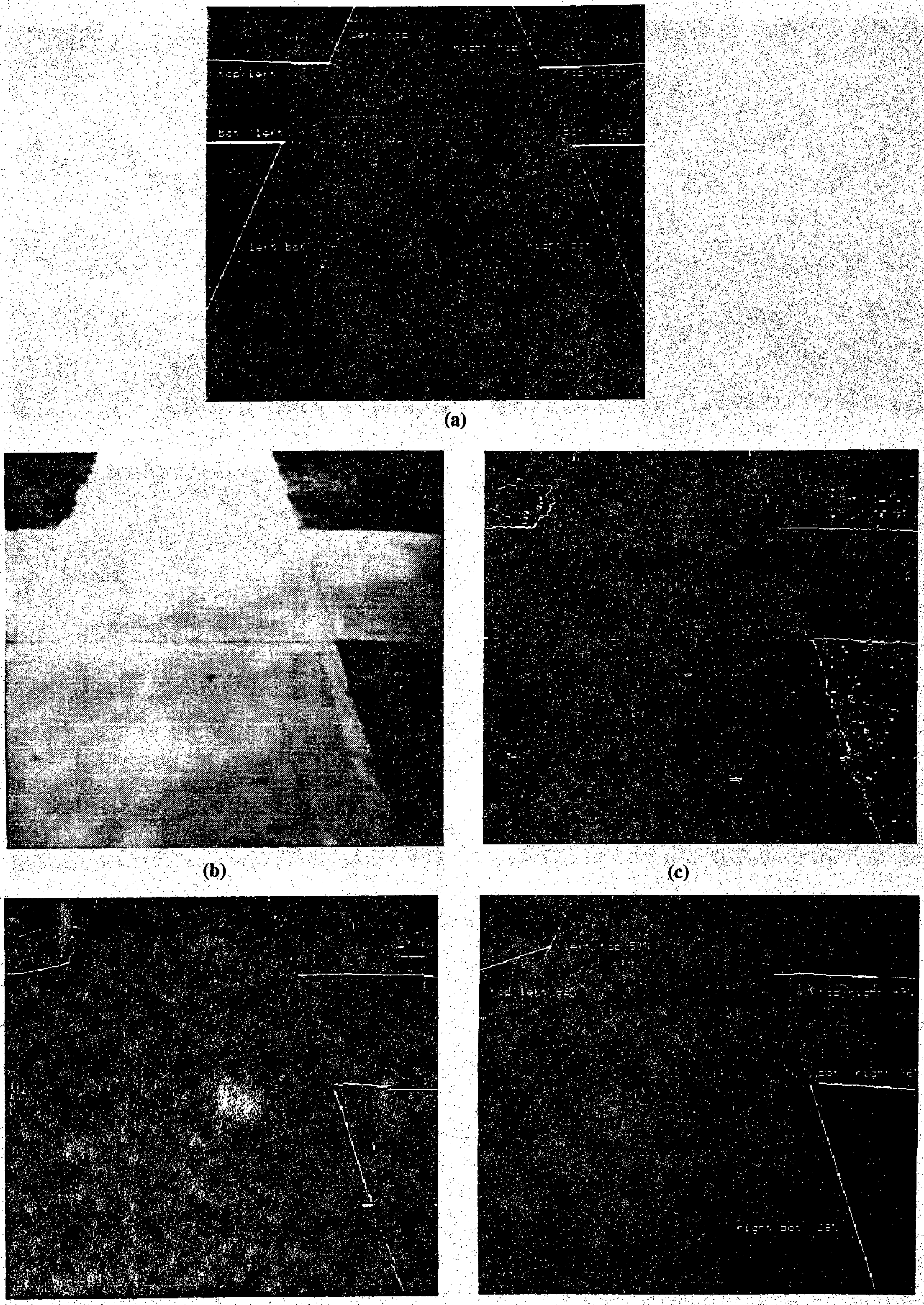


FIGURE 13: Example images from PSEIKI

6. CONCLUSIONS

This report describes work in progress on PSEIKI, a domain independent vision system. It demonstrates how a line drawing of the expected scene can be used to aid in the image understanding task. It describes the architecture and reasoning schemes used by PSEIKI to accomplish its task. It details how the Dempster-Shafer theory of evidence can be applied to labeling and grouping processes in a hierarchical scene representation.

7. ACKNOWLEDGEMENT

We would like to thank Rich Coe for his work on the design and development of PSEIKI's debugging software.

REFERENCES

[1] A. C. Kak, B. A. Roberts, K. M. Andress and R. L. Cromwell, ''Experiments in the Integration of World Knowledge with Sensory Information for Mobile Robots,'' *Proc. IEEE Int. Conf. Robotics Automat.*, Vol 2., 1987, pp. 734-741.

[2] J. Gordon and E. H. Shortliffe, ''A Method for Managing Evidential Reasoning in a Hierarchical Hypothesis Space,'' *Artificial Intelligence,* Vol. 26, 1985, pp. 323-357.

[3] G. Shafer, *A Mathematical Theory of Evidence,* Princeton University Press, 1976.

[4] R. A. Brooks, ''Symbolic Reasoning Among 3-D Models and 2-D Images,'' *Artificial Intelligence,* Vol. 17, 1981, pp. 285-348.

[5] L. S. Davis and S. S. V. Hwang, ''The SIGMA Image Understanding System,'' *IEEE Proc. Comp. Vision: Rep. and Cont.,* 1985, pp. 19 - 26.

[6] D. M. McKeown, Jr., W. A. Harvey, Jr. and J. McDermott, ''Rule-Based Interpretation of Aerial Imagery,'' *IEEE trans. on Pat. Anal. and Mach. Intel.,* Vol. PAMI-7, No. 5, 1985, pp. 570-585.

[7] A. R. Hanson and E. M. Riseman ''VISIONS: A Computer System for Interpreting Scenes,'' *Computer Vision Systems* Academic Press, 1978, pp. 303-333.

[8] A. M. Nazif and M. D. Levine ''Low Level Segmentation: An Expert System,'' *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. PAMI-6, No. 5, 1984 pp. 555-577

[9] S. T. Barnard, ''Interpreting Perspective Images,'' *Artificial Intelligence,* Vol. 21, 1983, pp. 435-462.

[10] H. G. Barrow and J. M. Tenenbaum, ''Interpreting Line Drawings as Three-Dimensional Surfaces,'' *Artificial Intelligence,* Vol. 17, 1981, pp. 75-116.

[11] R. B. Eberlein, ''An iterative gradient edge detection Algorithm,'' *Computer Graphics and Image Processing,* Vol. 5, 1976, pp. 245-253.

[12] R. O. Duda and P. E. Hart *Pattern Classification and Scene Analysis,* Wiley, New York, 1973.

[13] J. A. Barnett, ''Computational Methods for a Mathematical Theory of Evidence,'' *Proc. IJCAI,* pp. 868-875, 1981.