Rule-Based Interpretation of Aerial Imagery

DAVID M. McKEOWN, JR., MEMBER, IEEE, WILSON A. HARVEY, JR., AND JOHN McDERMOTT

***Abstract*—In this paper, we describe the organization of a rule-based system, SPAM, that uses map and domain-specific knowledge to interpret airport scenes. This research investigates the use of a rule-based system for the control of image processing and interpretation of results with respect to a world model, as well as the representation of the world model within an image/map database. We present results on the interpretation of a high-resolution airport scene where the image segmentation has been performed by a human, and by a region-based image segmentation program. The results of the system's analysis is characterized by the labeling of individual regions in the image and the collection of these regions into consistent interpretations of the major components of an airport model. These interpretations are ranked on the basis of their overall spatial and structural consistency. Some evaluations based on the results from three evolutionary versions of SPAM are presented.**

***Index Terms*—Artificial intelligence, computer vision systems, knowledge utilization, production systems, rule-based aerial photo interpretation.**

I. Introduction

SPAM, System for Photo interpretation of Airports using MAPS, is an image-interpretation system. It coordinates and controls image segmentation, segmentation analysis, and the construction of a scene model. It provides several unique capabilities to bring map knowledge and collateral information to bear during all phases of the interpretation. These capabilities include the following.

- The use of domain-dependent spatial constraints to restrict and refine hypothesis formation during analysis.
- The use of explicit camera models that allow for the projection of map information onto the image.
- The use of image-independent metric models for shape, size, distance, absolute and relative position computation.
- The use of multiple image cues to verify ambiguous segmentations. Stereo pairs or overlapping image sequences can be used to extract information or to detect missing components of the model.

A. *The Nature of the Task*

The task of airport image analysis has many interesting properties. First, airports are a complex organization of man-made structures placed over a large ground area. While the actual spatial arrangement of typical structures such as runways, terminal buildings, parking lots, etc., varies greatly between airports, the types of structures normally found in an airport scene are well understood. The airport task provides a "knowledge rich" environment, where functional relationships between structures preclude arbitrary spatial arrangements and provide spatial constraints.

Second, a body of literature [1], [2] on airport planning is readily accessible and provides general design constraints. Knowledge acquisition for spatial constraints, therefore, does not involve examination of large numbers of sample airports. It can be observed that there are two major classes of airports, commercial and military, whose organization varies widely from large-scale international airports to small county and private airstrips. Both general knowledge (class-specific) and site-specific knowledge can be expected to help in the interpretation process.

Image-processing techniques are inherently errorful. For complex, uncontrived natural scenes, image segmentation results are highly ambiguous. The correspondence between regions in the segmented image and physical objects in the scene is generally many to one. Boundaries between objects may not be distinguishable due to occlusions, objects with similar spectral properties, and the intrinsic resolution of the image. Thus, the assumption that regions in the segmented image *directly* correspond to objects in the scene is not useful unless we are able to reason about the segmentation process and reconstruct a meaningful portion of the original object.

Finally, for aerial photo-interpretation tasks, it is crucial that the metrics used by the analysis system be defined in cartographic coordinates, such as ⟨*latitude/longitude/elevation*⟩, rather than in an image-based coordinate system. Systems that rely on descriptions such as "the runway has area 12 000 pixels" or "hangars are between 212 and 345 pixels" are useless except for (perhaps) the analysis of one image. Further, spatial analysis based on the semantics of *above*, *below*, *left-of*, *right-of*, etc., are also inappropriate for general interpretation systems. To operationalize metric knowledge one must relate the world model to the image under analysis. This should be done through image-to-map correspondence using camera models which is the method used in SPAM. We can directly measure ground distances, areas, absolute compass direction, and recover crude estimates of height using a camera model computed for each image under analysis. Direct projection of known map information is also possible in SPAM. For example, the position and orientations

Manuscript received June 12, 1985. This work was supported in part by the Defense Advanced Research Projects Agency under ARPA Order 3597, and monitored by the Air Force Avionics Laboratory under Contract F33615-78-C-1551. The views and conclusions in this paper are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Projects Agency or the U.S. Government.

The authors are with the Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA 15213.

of known airport features such as runways and terminals could be directly factored into the scene interpretation as *a priori* knowledge. We are working on a system that can bring both class-specific and site-specific knowledge to bear; however, here we report on the use of class-specific knowledge.

B. Related Work

There is a long history of research in model-based vision. Binford [3] provides a good summary of a variety of work including Brooks [4], the VISIONS project [5], SRI [6], [7], and work by Matsuyama [8] and Ohta [9].

Work by Bullock [10], based on the ACRONYM system developed by Brooks and Binford [11], uses image registration to a geographic model, and identifies preselected regions of interest and attempts to locate and identify predefined object instances within these areas. ACRONYM is an example of a model-based system which incorporates viewpoint-insensitive mechanisms in terms of its model description. Its recognition process is to map edge-based image properties to instances of object models. So far, results have been reported for the recognition of a small number of models (3) for wide-bodied jets in aerial photographs. It is not clear how spatial knowledge would be directly integrated into the ACRONYM framework.

Matsuyama [12], [13] has demonstrated a system for segmentation and interpretation of color-infrared aerial photographs containing roads, rivers, forests, and residential and agricultural areas. It uses rules to make assignments based on region adjacency and multispectral properties. It uses two-dimensional (2-D) shape descriptions and performs region merging to generate object descriptions. It generates good descriptions of a variety of fairly complex aerial scenes getting a great deal of constraint from directly mapping the multispectral properties of regions into scene hypotheses.

In his dissertation, Selfridge [14] proposes using adaptive threshold selection for region extraction by histogramming and region growing using an image-based "appearance model." Recently, Hwang [15] has explored this method, coupled with the use of domain knowledge to guide interpretation of suburban house scenes in monochromatic aerial imagery. Hwang uses a test-hypothesize-act cycle to generate large numbers of potential hypotheses which are then grouped into consistent interpretations.

II. SPAM Components

SPAM is composed of three major system components: a Washington, DC, image/map database (MAPS), image-processing tools, and the rule-based system. In this paper, we describe the rule-based component, but it is important to emphasize the necessity of a complete system design in order to realistically explore aerial photo-interpretation tasks. Fig. 1 is a block diagram of the current system. SPAM is organized to view information extracted from image(s) uniformly, that is, without knowledge of what method was used to extract the image features. Further, all *image-based* descriptions are converted into *map-based* descriptions using an image-to-map transformation based on the camera model stored for the image(s) in the MAPS database. Thus, it is possible to integrate image feature information from different segmentation or analysis methods, as well as from multiple scenes of the same airport. This architecture allows us to add and delete various sources of knowledge and to measure their effect on the quality of the resulting interpretation.

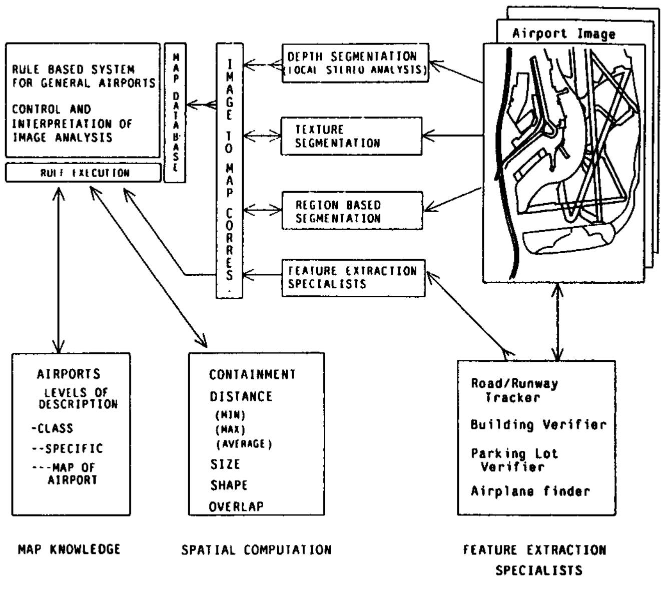

Fig. 1. SPAM system organization.

A. Image/Map Database

The MAPS [16], [17] database stores facts about man-made or natural feature existence and location; this allows SPAM to perform geometric computation in *map space* rather than *image space*. Differences in scale, orientation, and viewpoint can be handled in a consistent manner using a simple camera model. The function of the image/map database is to tie database feature descriptions to a geodetic coordinate system ⟨*latitude*, *longitude*, *elevation*⟩, and to use camera models (image-to-map correspondence) to predict their location and appearance in the aerial photography. MAPS also provides facilities to compute geometric properties and relationships of map database features such as *containment*, *adjacency*, *subsumed by*, *intersection*, and *closest point*.

B. Image-Processing Tools

There are currently four image-processing tools that can be invoked with actions specified by the rule-based system. They are a region-growing segmentation program [18], a road/road-like feature follower [19], a stereo analysis program [20], and an interactive human segmentation system. These tools perform low-level and intermediate-level feature extraction. Processing primitives are based on linear feature extraction and region extraction using edge-based and region-growing techniques. The goal of the image processing component is to identify feature-based islands of interest and extend those islands constrained by the geometric model provided by MAPS and model-based goals established by the rule-based component.

C. Rule-Based System

The rule-based component[1] provides the image-processing system with the best next task based on the strength/promise of expectations and with constraints from the image/map database system. It also guides the scene interpretation by generating successively more specific expectations based on image-processing results.

Airport scene interpretation is a classic case of the knowledge-based system approach to signal (image) analysis. This class of problems is characterized by uncertainty in the underlying data, and multiple, ambiguous interpretations for any particular hypothesis. It is our contention that hypotheses generated from raw region data should be able to be reliably verified by looking at other interpretations in its vicinity. Those interpretations emerging with the highest confidence should be those regions that can be collected into consistent cliques that will model the majority of the airport. These cliques can then be manipulated as individual pieces of a puzzle, putting appropriate ones together to form a complete interpretation (or model) of the airport in question. Multiple consistent models may be present. In SPAM, hypotheses are grouped into cliques according to their function and spatial proximity.

Region data can be derived from a number of sources. Typical sources are image-based segmentation, texture-based segmentation, multispectral analysis, and segmentation of depth map images compiled from stereo image pairs. SPAM has been tested with both hand-generated and machine-generated image-based segmentations. Obviously, perfect data are not generated by image analysis methods, so SPAM must have shape constraints flexible enough to compensate for this. The system, then, must perform the following two basic tasks:

1) generate a complete and consistent model of the airport given "perfect" hand-generated data; and

2) extract reasonable hypotheses for actual features from nonideal data.

In Sections III, V, and VI, we describe the organization of the rule-based system. In Section IV, we present several examples of SPAM processing both hand- and machine-generated data. Some performance statistics are presented in Section VIII.

III. Interpretation Primitives

The task of the rule-based system is to generate plausible, complete analyses of the airport image by building successively more specific interpretations based on the initial image segmentation. Rules to generate and evaluate such interpretations manipulate four simple working memory elements: *regions*, *fragments*, *functional areas*, and *models*.

[1] We use the OPS5 production system language [21] as the implementation language for our rule-based system.

A. Regions

The region working memory element contains low-level image properties of each segmentation region, such as shape, texture, spectral properties, etc. Regions represent the low-level data extracted from the image or specified by a human interpreter. The region properties are used to determine which classes of airport features a particular region best represents.

Since the criteria for initial class determination are weak discriminators, multiple hypotheses may be generated for each region as a different plausible interpretation. Current classes and subclass specializations are given in Fig. 2. These interpretations are represented as fragments, which are described in Section III-B. Fig. 3 illustrates a region working memory element and gives a general description of each of the attribute names. The *symbolic-name* is a unique identifier that allows functions external to the interpreter access to the segmentation region, the image, and the map database.

B. Fragments

A fragment is an interpretation for an image region. Fig. 4 shows a sample fragment working memory element. Each fragment contains class and subclass interpretation information, as well as interpretation status information used to control rule execution. Status information includes whether the fragment has been extended, aligned with other fragments, and whether consistency has been performed. Fragments which represent alternative interpretations of regions can be identified by their region-token attribute name.

C. Functional Areas

A functional area (FA) is a distinct spatial subdivision of the airport scene. It comprises a collection of man-made and natural features, normally found in close physical proximity and often related in function. For example, runways, taxiways, tarmac, and grassy areas form one functional area. Some of the characteristics of this FA are: the landing and takeoff of planes; routes of planes from terminal to takeoff or landing points; a buffer area with minimal 3-D structures (obstructions); and is centrally located within the airport area. SPAM currently supports four functional area types:

- *FA1*: *terminal, parking apron, parking lots, roads
- *FA2*: *roads and grassy areas
- *FA3*: *hangars, roads, tarmac, and parking apron
- *FA4*: *runways, taxiways, grassy areas, tarmac.

Interpretations marked with * are seed fragments for the creation of each functional area. For example, if there are no hangar fragment interpretations then no *FA3* functional areas will be generated. Fig. 5 shows a sample functional area working memory element. Included in the functional area is its confidence, its coverage in the image, and a list of fragments which support the interpretation.

D. Models

The airport-model working memory element represents a mutually consistent collection of functional areas. At

Class:	Sub-class:
linear:	runway, taxiway, access road
compact:	terminal building, hangar
small-blob:	parking lots, parking aprons
large-blob:	tarmac, grassy areas

Fig. 2. Class and subclass interpretations.

```
(region
 ^token g00005                         ;Machine generated unique id.
 ^seg-number 6471                      ;Machineseg region number.
 ^statefile ALL                        ;Machineseg state file.
 ^symbolic-name ALL-L.6471_0           ;Database access id.
 ^spectral-characteristics nil         ;Currently unknown.
 ^texture nil                          ;Currently unknown.
 ^shadow-type nil                      ;Currently unknown.
 ^shadow-class nil                     ;Currently unknown.
 ^location-lat 139844.728507           ;latitude of feature center.
 ^location-lon 277380.289332           ;longitude of feature center.
 ^orientation 1.132534                 ;Orientation of fourier ellipse.
 ^ellipse-width 10.046023              ;Width, fourier ellipse (meters).
 ^ellipse-length 344.298377            ;length, fourier ellipse (meters).
 ^mbr-length 370.092492                ;Length, MBR (meters).
 ^height nil                           ;Height, feature (meters).
 ^eclipsed t                           ;Partially covered by another region.
 ^curvature curved                     ;Curvature wrt curvature criterion.
 ^ellipse-linearity 34.272108          ;Linearity using fourier ellipse
 ^mbr-linearity 44.59714               ;Linearity using MBR
 ^compactness 0.00413                  ;Compactness (ratio).
 ^fractional-fill 0.047454             ;Fractional-fill (ratio).
 ^area 3441.397917                     ;Area (square meters).
 ^perimeter 912.836763                 ;Perimeter (meters).
 ^linear t                             ;fits criterion for linear regions?
 ^compact nil                          ;     compact regions?
 ^large-blob nil                       ;     large-blob regions?
 ^small-blob nil                       ;     small-blob regions?
 ^road t                               ;     roads?
 ^runway nil                           ;     runways?
 ^taxiway nil                          ;     taxiways?
 ^terminal nil                         ;     terminal-buildings?
 ^hangar nil                           ;     hangars?
 ^parking-lot nil                      ;     parking-lots?
 ^parking-apron nil                    ;     parking-aprons?
 ^grassy-area nil                      ;     grassy-areas?
 ^tarmac nil                           ;     tarmac?
 ^special-region nil                   ;Special region (shadow or given)
 ^region-origin bottom-up              ;Generation: bottom-up, top-down.
 ^region-status interpreted)           ;One of: active, to-unknown,
                                       ;uninterpretable, to-be-interpreted,
 )                                     ;deleted, or interpreted.
```

Fig. 3. Region working memory element.

least one member of each functional area type must be present in order to produce a complete model. Additional functional areas that are consistent with the four primary functional areas are also represented. A confidence is assigned to each model based on the confidence of each functional area, and how well the collection of functional areas "explain" or cover the airport scene.

The conflict working memory element is used to keep track of conflict recognized in the model generation process. This structure represents conflicts of interpretation between two functional areas that may share common fragment interpretations. It is generated as SPAM attempts to group functional areas together as constituents of a model. The presence of conflict working memory elements invoke rules to resolve conflicts and are specific to particular types of conflicts (see Fig. 6). This is discussed in more detail in Section VI-G.

IV. Some Examples

Fig. 7 shows one of the high resolution (1:12000) images of the National Airport in Washington, DC, stored in the MAPS database. This is one of several test images that we are currently using to develop the SPAM rule-based component. The image is 2280 rows by 2280 columns digitized to 8 bits of intensity per pixel. Approximately 12 views of the airport at image scales from 1:12000 to 1:60000 taken over a six-year period are available in the database.

```
(vector-attribute frag-list)
(literalize fragment
 fragment-token    ;Unique ID for this fragment.
 name              ;A human readable name for the fragment.
 object-type       ;LINEAR, SMALL BLOB, LARGE-BLOB, or COMPACT.
 hypothesis        ;Sub-class specialization for object-type.
 confidence        ;goodness of this interpretation 0.0< x <1.0
 region-token      ;ID of the region this fragment represents.
 symbolic-name     ;The region symbolic name.
 extension-flag    ;Current status during extension trials.
 frag-status       ;The 'age' of the fragment.
 evaluation-flag   ;The current state of evaluation.
 eval-history      ;Number of times fragment has been evaluated.
 frag-origin       ;Origin, machineseg, handseg.
 consistency-flag  ;Consistency strategy: checked or unchecked.
 aligned           ;Has fragment been used in an alignment.
 assoc-shadow      ;sym-name of associate shadow region for frag
 fa-seed           ;Is this a functional area seed.
 in-fa             ;Whether this fragment is in a function area.
 eclipse           ;Is this fragment eclipsed by another region.
 generic-flag      ;A general-purpose flag.
 frag-cnt          ;A count of the # of consistent fragments.
 frag-list         ;Fragment ids consistent with this fragment.
 )
```

Fig. 4. Fragment working memory element.

```
(vector-attribute elements)
(literalize functional-area
 fa-id          ;unique id for this element.
 fa-type        ;functional area type.
 fa-gen         ;generation number.
 confidence     ;the goodness of this area.
 coverage       ;the percentage that this FA is explained.
 count          ;the number of fragments represented.
 elements       ;the list of fragments ids contained in FA.
 )
```

Fig. 5. Functional area working memory element.

```
(vector-attribute other-fas)
(literalize airport-model
    mo-id        ;the model id.
    confidence   ;the goodness of this model.
    count        ;the number of functional areas in model.
    terminal-fa  ;terminal functional area.
    runway-fa    ;runway functional area.
    hangar-fa    ;hangar functional area.
    road-fa      ;road functional area.
    other-fas    ;other functional areas included.
)

(literalize conflict
    cid            ;so we can identify this conflict.
    status         ;whether or not this conflict has been resolved.
    in-favor-of    ;which interpretation was the decision in favor of.
    symbolic-name  ;the region id.
    fid1           ;the id of the 1st interpretation.
    type1          ;and its interpretation type.
    fid2           ;the id of the 2nd interpretation.
    type2          ;and its interpretation type.
    confidence-difference ;the difference in confidence between the
                          ;alternative interpretations.
    fa1            ;the id of the FA where the 1st fragment occurs.
    fa2            ;the id of the FA where the 2nd fragment occurs.
    model          ;the id of the model in which this conflict occurs.
)
```

Fig. 6. Model and conflict working memory elements.

Fig. 8 shows the results of applying our region-growing segmentation system to the image in Fig. 7. The segmentation is run over multiple windows of the image, each window is approximately 512 × 512 pixels. Artifacts of these windows can be seen as the straight boundaries in Fig. 8. The segmenter is supplied with image scale-dependent criteria from the MAPS database and performs region growing while searching for linear, compact, and blob regions [18].

It is evident that the image segmentation is quite errorful. Few features are completely segmented, many are merged together or broken into arbitrary fragments, and some are missing entirely. For example, in the runway beginning in the lower left and running to the upper right,

Fig. 7. SPAM: National Airport, Washington, DC.

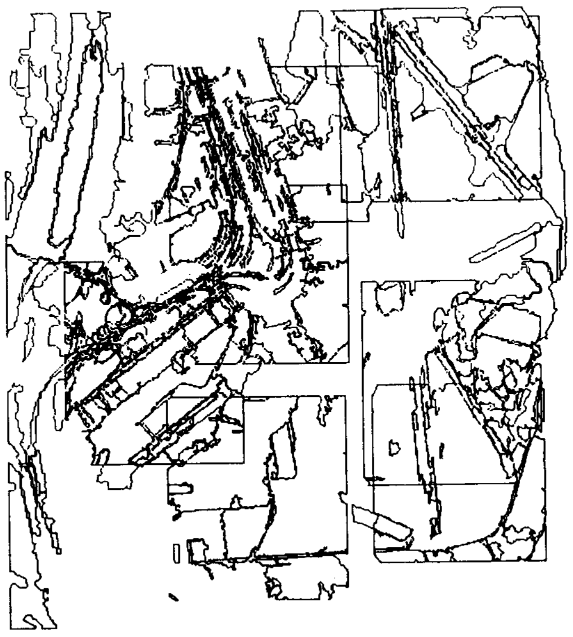
Fig. 8. SPAM: Region-based segmentation for Fig. 7.

many of the small taxiways are merged with the adjacent tarmac. In spite of this, the results are not atypical of the current state of the art in region-based segmentations using monocular views and monochrome photography with complex scenes.

The segmentation in Fig. 9 illustrates the problem of broken and missing regions for the access roads directly behind the hangars in the center-right of Fig. 7. Fig. 10 shows the area of the image that corresponds to Fig. 9. Note that even the nearly homogeneous bright parking apron is broken due to aircraft and shadows cast by aircraft on the apron. Interpretation systems simply must be able to accommodate errorful segmentations in order to perform scene analysis tasks.

Fig. 11 shows a set of 280 machine segmentation regions selected from the original 477 regions shown in Fig. 8. The 280 regions were selected to cover a representative number of airport features in the scene.

Fig. 12 is a human segmentation of the airport scene containing 95 regions. All of the road features have been broken into several segments to crudely simulate the problems noted in Fig. 9. SPAM has been tested using machine-generated and human segmentations using the identical rule base in both cases. We have also run SPAM on collections of machine-generated segmentations ranging from 20 to 280 regions per collection for the purpose of timings and rule base validation. The segmentations in Figs. 11 and 12 are the most complex tested to date.

Results are presented for two versions of the SPAM system, version 2 (V2), and version 3 (V3). The versions differ in that V3 has texture and height information supplied by associating a vector of probabilities with each of the 95 hand-segmented regions. These probabilities give estimates for ***highly-textured***, ***moderately-textured***, and ***lightly-textured*** as well as ***height*** $>$ 15 ***meters***, ***height*** $>$ 5 ***meters***, and ***height*** $<$ 5 ***meters***. Machine-segmented regions acquire their texture and height probabilities through calculation of their proportional overlap with the hand-

Fig. 9. SPAM: Detail of segmentation near hangar area.

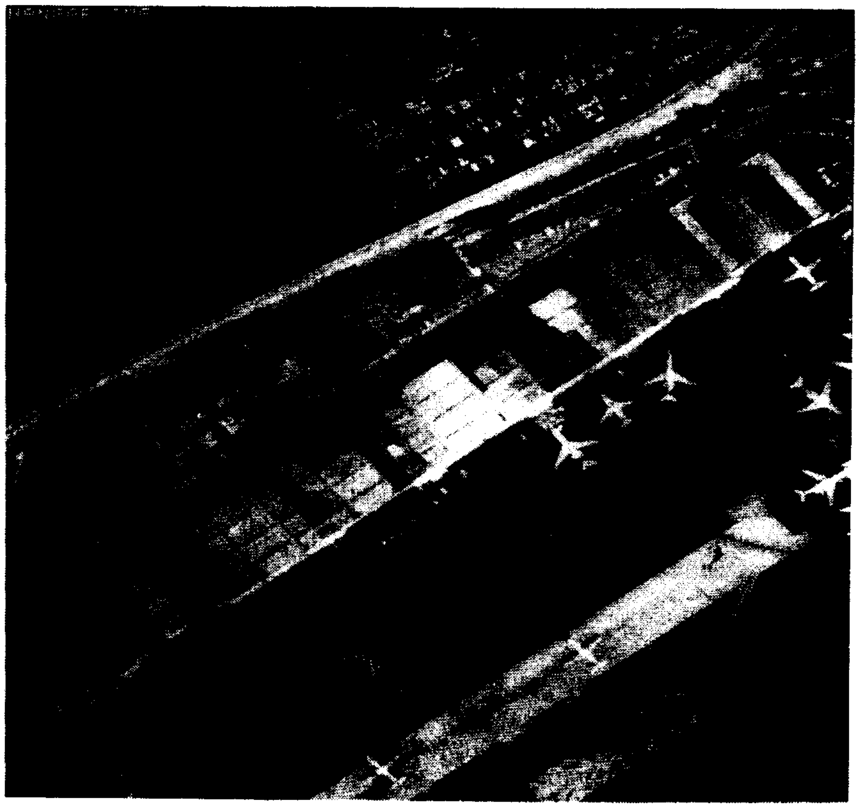

Fig. 10. SPAM: Full resolution image for segmentation in Fig. 9.

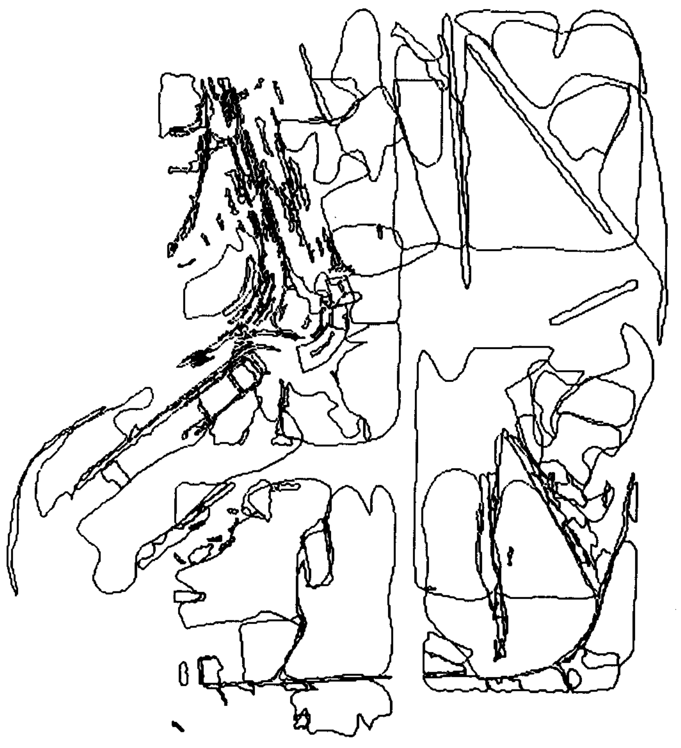
Fig. 11. SPAM: 280 region test machine segmentation.

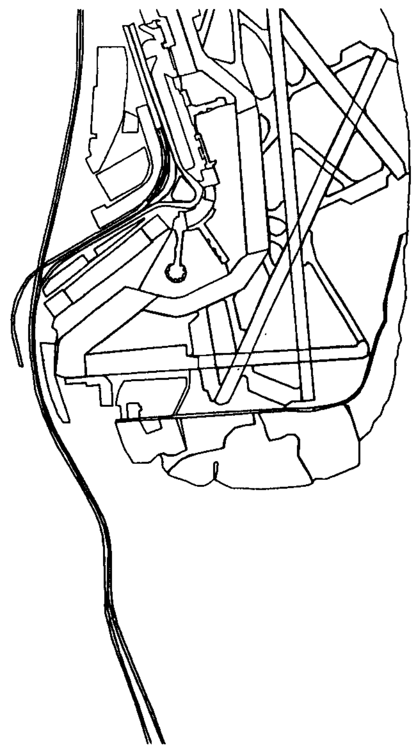
Fig. 12. SPAM: 95 region human segmentation.

segmented regions. V3 contains no additional rules to utilize the simulated texture and height information, the existing region-to-interpretation rules simply make use of the information when it is available.

Figs. 13–15 show the results of SPAM V2 on the machine segmentation data in Fig. 11. Fig. 13 shows the five highest rated fragments for each of the nine subclass interpretations. Fig. 14 shows the fragments that comprise the model in Fig. 16. Fig. 15 shows the fragments that were compatible with the model in Fig. 16, but were not included as part of the model. Fig. 14 is the collection of functional areas that comprise the "best" model generated by SPAM in terms of model confidence.

Our method for generating functional areas relies on the identification of seed fragments. These seed fragments are ordered by relative confidence and evaluated in turn. Therefore, it is important that 1) interpretations of terminal, access roads, hangars, and runway be generated and that 2) the "correct" regions be highly rated. The emergence of correctly identified runways and hangars in each of Figs. 13, 17, 21, and 25 is an important factor in the generation of runway and hangar FA's for the models in Figs. 16, 20, 24, and 28.

Figs. 17–20 show the same results for SPAM V2 run on the hand-segmentation data in Fig. 12. Figs. 21–24 show the results for SPAM V3 run on the machine-segmentation data. Figs. 25–28 show the same results for SPAM V3 run on the hand-segmentation data.

V. Interpretation Phases

SPAM is loosely organized into five processing phases. These phases are *build*, *local evaluation*, *consistency*, *functional area*, and *model evaluation*. The processing strategy is to move sequentially through the five phases. There is no prescribed relationship with the rule classes described in Section VI and the phases described in this section. For example, rules to determine the consistency of a runway with a taxiway might be activated during the functional area phase as well as during the consistency phase. This is achieved by generating the appropriate evaluation context for the fragment hypotheses.

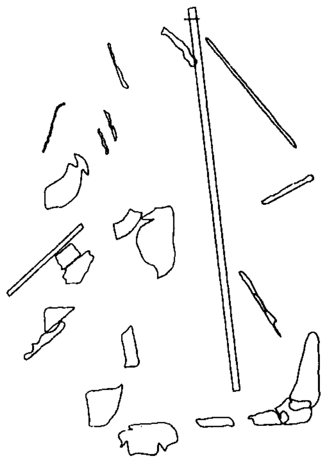
Fig. 13. V2MACH: Best 5 fragment interpretations in each subclass.

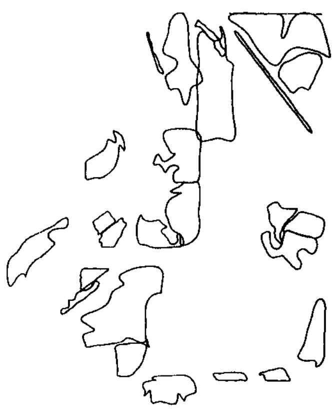
Fig. 14. V2MACH: Fragments comprising the model in Fig. 16.

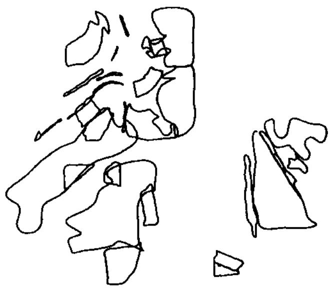
Fig. 15. V2MACH: Fragments compatible with the model in Fig. 16.

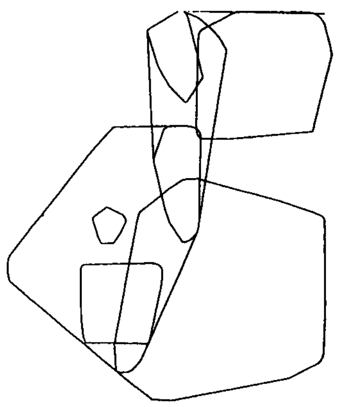
Fig. 16. V2MACH: Functional areas comprising the best model.

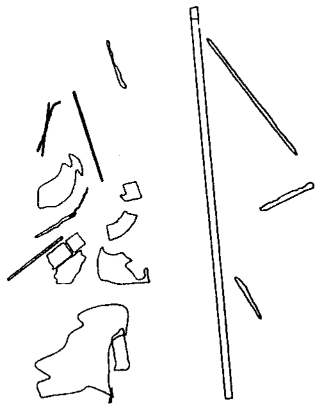
Fig. 21. V3MACH: Best 5 fragment interpretations in each subclass.

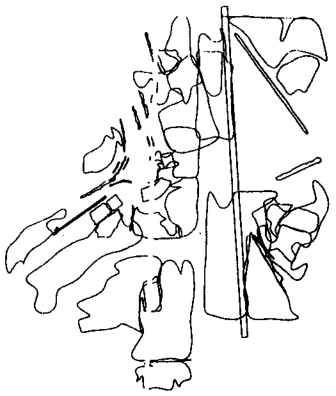
Fig. 22. V3MACH: Fragments comprising the model in Fig. 24.

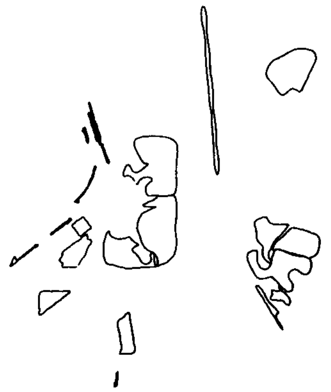
Fig. 23. V3MACH: Fragments compatible with the model in Fig. 24.

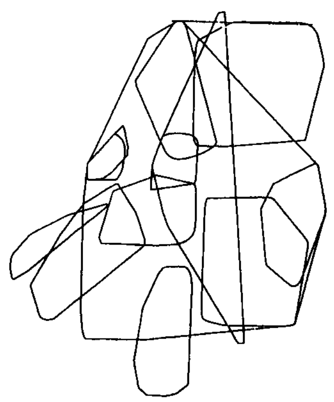
Fig. 24. V3MACH: Functional areas comprising the best model.

ible type. If satisfied, a region is created whose boundary is the convex hull of all the regions represented by the fragment interpretations. An initial confidence value is assigned to the FA based on the average of the confidences of the member fragments. The computation of the FA boundary allows us to compute two factors. The first is the area "explained" by the fragments that compose the FA; the second is the area which is "uninterpreted," or not covered by fragments in the functional area. Further, a list of all fragments lying on or within the boundary of the FA that have a *compatible* interpretation with the FA type is calculated. For example, for a runway functional area, tarmac and hangars lying inside the boundary would be in the compatible list. These fragments will not have been found to be consistent with the functional area, but this does not preclude a weaker compatibility. A list of *incompatibles* consisting of those fragments lying on or within the boundary of the FA that have an incompatible interpretation with the FA type is also calculated. A final confidence value for the functional area is based on the relative strength of these compatible and incompatible fragments. Fig. 31 is an example of the best runway functional area calculated for the hand segmentation data in version V3.

E. Model Evaluation

The model evaluation phase begins when there are sufficient functional areas generated by the previous phase. At a minimum, one of each of the four functional area types must be present in order to begin model generation. Much of model evaluation is organized to resolve inconsistencies between overlapping functional areas. In general, when an overlap is recognized, one of three following situations may occur:

- *case 1:* the area of overlap does not contain any fragment interpretations;
- *case 2:* the fragment interpretations within the area of overlap are the same, or compatible; and
- *case 3:* the fragment interpretations within the area of overlap are incompatible.

In the third case, reasoning about which fragment interpretations are in error is possible by recognizing chains of inconsistency or preferred interpretations. For example, if a hangar interpretation falls inside of a highly rated runway functional area, and the region to which the hangar is attached has an alternative taxiway interpretation as well, it may be determined that the hangar interpretation should be removed. One side effect of the deletion is to improve the confidence of the runway functional area. Another method for resolving the conflict is to simply select the fragment with higher confidence.

If a functional area is lacking support in a certain subarea, SPAM could use this information to verify the functional area by specifically looking for missing fragment interpretations. For example, if a terminal functional

Fig. 29 outlines the paradigm used in SPAM at each interpretation phase. At each phase, knowledge is used to check for consistency among hypotheses, to predict missing components using context, and to create contexts based on collections of consistent hypotheses. Prediction is restrained in SPAM in that a hypotheses cannot predict missing components at their own representation level. A collection of hypotheses must combine to create a context from which a prediction can be made. These contexts are refinements or spatial aggregations in the scene. For example, a collection of mutually consistent runways and taxiways might combine to generate a runway functional area. Rules that encode knowledge that runway functional areas often contain grassy areas or tarmac may predict that certain subareas within that functional area are good candidates for finding such regions. However, an isolated runway or taxiway hypothesis cannot directly make these predictions. In SPAM the context determines the prediction. This serves to decrease the combinatorics of hypothesis generation and to allow the system to focus on those areas with strong support at each level of the interpretation. In the following sections, we describe each of the interpretation phases.

A. Build

During the build phase, region data, instantiated by an initialization rule, invoke the region-to-interpretation rules. A region is "chosen" by the interpreter and tested against each class (one of small blob, large blob, linear, or compact). If that region fits a particular class, a generic fragment interpretation for that class is created.

For each of the generic class interpretations, the region is tested against more specific criteria for each subclass. For example, a region may fit the criterion for the linear class, so it is tested against the criteria for access road, taxiway, and runway. Some of the criteria are listed in Fig. 33. A region may fit the criterion for a class, but not fit the criteria for any of the subclasses. Similarly, a region may also fit several classes (and therefore several subclasses), so several class (and subclass) fragment interpretations are created. The existence of generic class interpretations, in addition to more specific subclass interpretations for the same region, allows SPAM to reevaluate regions by changing the criteria for subclass due to changed expectations.

An *initial confidence value* is calculated based on how well a region fits the criterion for a class or subclass. A confidence is assigned to each interpretation. For example, if a region is very long, wide, and straight, then it is a good candidate for a runway and, because of its length, not such a good candidate for a taxiway. These facts are reflected in the confidence values for each of the two fragment interpretations for that region. This phase terminates when all instantiated regions have been tested for interpretation.

B. Local Evaluation

The local evaluation phase allows some processing to take place on the fragment interpretations generated by the build phase. For example, straight and curved alignment of linear segments is performed and the resulting regions are added to working memory. They are recognized as new regions and are evaluated by region-to-interpretation rules. This process continues until no new alignments occur.

Currently, regions are assumed to be geometrically distinct; that is, no two regions occupy the same two-dimensional space. This assumption may be violated when alignments or further image processing is performed. Regions resulting from these operations can occupy nearly the same portion of the image. SPAM determines possible candidates for overlap using knowledge stored in the region descriptors. Rules test those candidates to determine whether an overlap is present, and what percentage of each feature has been "eclipsed." For simple cases, where one region subsumes another and both have the same class interpretation, we inactivate the subsumed fragment's subclass interpretation. Inactivation (instead of deletion) allows the system to go back to the original hypotheses in the event that an aligned region is not justifiable in the image. Thus, if two runway hypotheses are generated through multiple pairwise alignments, the "smaller" hypothesis will be inactivated. Its linear class interpretation will remain, but is already marked to preclude further alignments. Other strategies that prune interpretations under these conditions need to be explored, especially within the context of choosing the "best" representative for a spatial area using more sophisticated analysis.

C. Consistency

The consistency phase is invoked once the build/evaluation phase terminates. Only fragments with subclass interpretations participate in this phase. A fragment interpretation is chosen, and a consistency strategy is created. Consistency strategies comprise of a list of applicable tests specific to the subclass. As each test is applied, an incremental confidence score is computed based on the consistency test.

Each test is a collection of rules that execute the test, as well as routines outside of the rule-based system that actually perform spatial evaluation. At the end of the consistency phase, each fragment contains a list of the fragments with which it is consistent. Its confidence reflects how well it satisfied the tests that were applied to it. Fig. 30 shows a road fragment interpretation and the interpretations with which it was found consistent. The subclass name, region identification, fragment identification, and fragment confidence are displayed.

D. Functional Area

The functional area (FA) phase evaluates fragments that have specific subclass interpretations. These *seed fragments* are terminals, access roads, runways, and hangars. For each seed fragment, the list of fragments with which it is consistent is traced to determine whether there is sufficient evidence for a functional area to be formed.

The current criterion for sufficient evidence is if the consistency trace finds one or more fragments of compat-

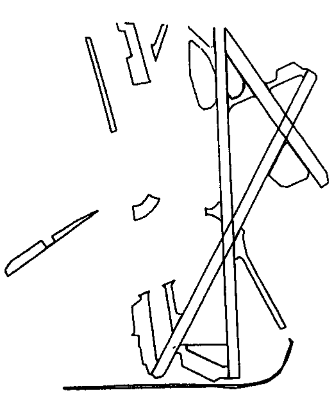
Fig. 17. V2HAND: Best 5 fragment interpretations in each subclass.

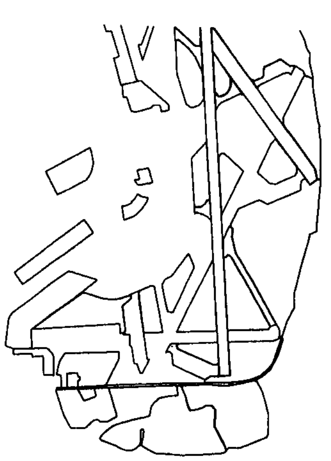
Fig. 18. V2HAND: Fragments comprising the model in Fig. 20.

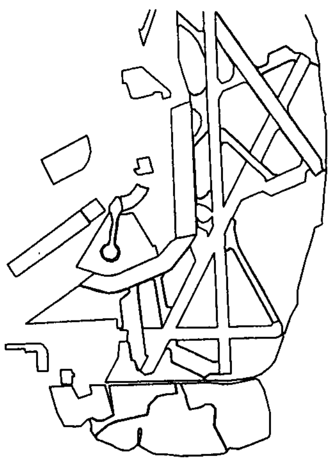
Fig. 19. V2HAND: Fragments compatible with the model in Fig. 20.

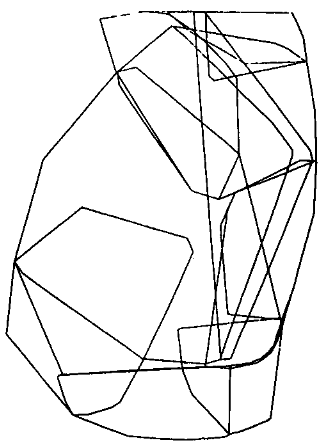
Fig. 20. V2HAND: Functional areas comprising the best model.

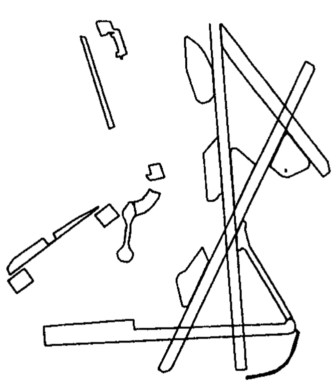
Fig. 25. V3HAND: Best 5 fragment interpretations in each subclass.

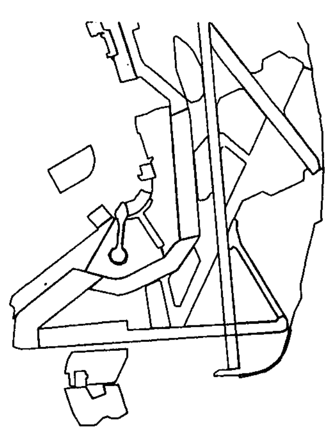
Fig. 26. V3HAND: Fragments comprising the model in Fig. 28.

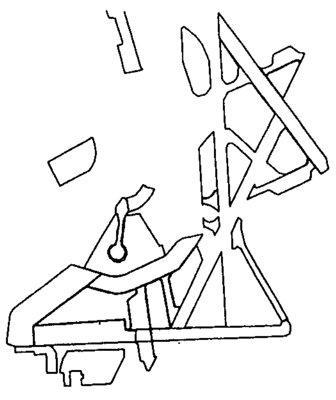
Fig. 27. V3HAND: Fragments compatible with the model in Fig. 28.

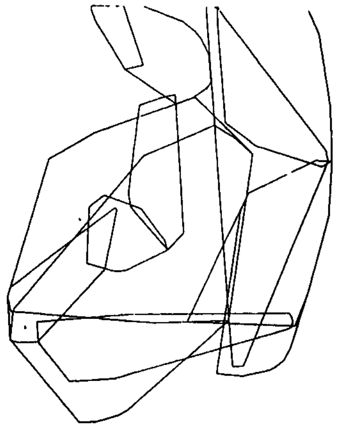
Fig. 28. V3HAND: Functional areas comprising the best model.

area contains roads and parking lots, but no parking apron, rules invoke image analysis tools to go look for regions whose shape and texture properties match with SPAM's *build* and *local evaluation* model for parking aprons. This form of prediction is consistent with our strategy outlined in Fig. 29.

When a large number of functional areas is generated, due to multiple consistent interpretations of fragments that have not been rejected by consistency tests, SPAM will generate multiple alternative models. We can make two observations about this. First, the difference between models appears to be small, the inclusion or exclusion of one or two functional areas can generate two or four different models. Second, the models all share a core set of functional areas indicating that the general organization of the airport appears to be relatively stable. In the following section, we will describe classes of rules that are used by SPAM during the interpretation phases discussed in this section.

VI. Rule Classes

Rules, as currently implemented in SPAM, are loosely organized into six classes. Within each class, sets of rules operate on various specific airport hypotheses. The classes generally correspond to the interpretation phases discussed in Section V, although the effects of interpretation in one phase can cause rules in another phase to become active. There are currently about 450 OPS5 rules in the system. Fig. 32 is a schematic that contrasts the interpretation phases in SPAM with the approximate number of rules in each class. An estimate of the potential for growth in the number of rules is given and discussed in Section IX.

A. Initialization Rules

The initialization rules (2 rules) initialize the interpretation goal states, *a priori* knowledge from map database, airport class expectations, and load working memory with the low-level image region segmentation. *A priori* knowledge such as the number of runways or the size or type of airport can be instantiated at the beginning of a run. They supply SPAM with a method to handle expectations input by the user. Alternatively, this allows one to experiment with SPAM by setting external goals and allowing the system to attempt to satisfy them. Fig. 33 shows a subset of the interpretation constants and gives a sample specification that can be used to bias interpretations for a large urban area around the airport.

B. Region-to-Interpretation Rules

The region-to-interpretation rules (35 rules) create initial fragment hypotheses for each region based on its intrinsic properties of size, shape, texture, and elevation

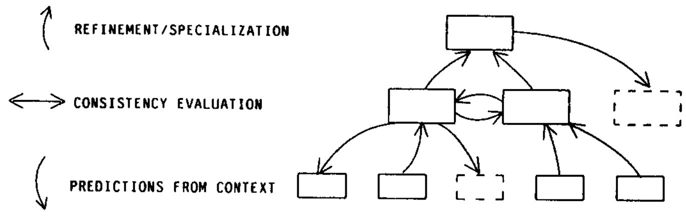

Fig. 29. Refinement, consistency, and prediction in SPAM.

```
ALL-L.6471_14
  road  g00354  0.87101
        perimeter   GIVEN-N.129_0     N129    1.0
        terminal    Hand36809-N.75_0  g00134  0.92802
        terminal    Hand36809-N.73_0  g00136  0.89112
        hangar      TERM2-C.7205_0    g00245  0.91008
        parking-lot TERM2-B.1134_0    g00262  0.79519
        hangar      HANGAR-C.3832_0   g00295  0.71099
        hangar      HANGAR-C.2983_0   g00297  0.84008
```

Fig. 30. Consistency relationships.

```
FUNCAREA-N.7_0
  runway 1 0.7395 0.64524 4
        runway      Hand36809-N.3_0  g00260 0.82994
        grassy-area Hand36809-N.26_0 g00158 0.63281
        grassy-area Hand36809-N.5_0  g00162 0.65813
        taxiway     Hand36809-N.16_0 g00241 0.83732
```

Fig. 31. Functional area description.

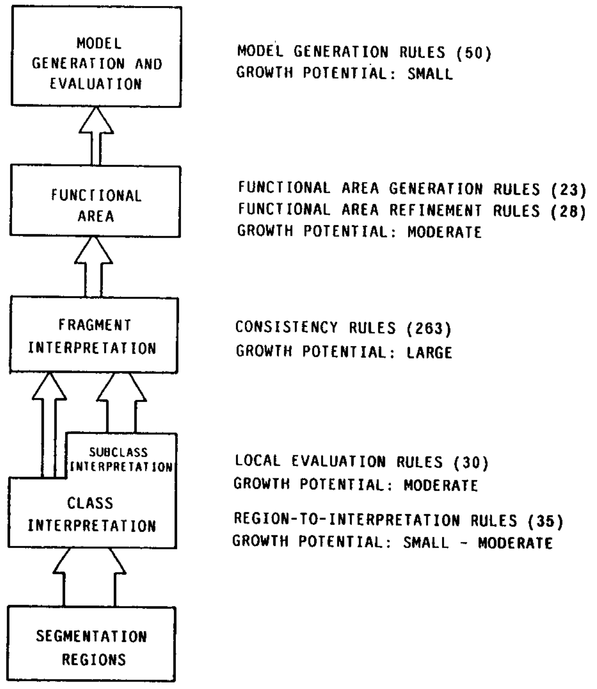
MODEL GENERATION AND EVALUATION
MODEL GENERATION RULES (50)
GROWTH POTENTIAL: SMALL
FUNCTIONAL AREA
FUNCTIONAL AREA GENERATION RULES (23)
FUNCTIONAL AREA REFINEMENT RULES (28)
GROWTH POTENTIAL: MODERATE
FRAGMENT INTERPRETATION
CONSISTENCY RULES (263)
GROWTH POTENTIAL: LARGE
SUBCLASS INTERPRETATION
LOCAL EVALUATION RULES (30)
GROWTH POTENTIAL: MODERATE
CLASS INTERPRETATION
REGION-TO-INTERPRETATION RULES (35)
GROWTH POTENTIAL: SMALL - MODERATE
SEGMENTATION REGIONS

Fig. 32. Interpretation phases in SPAM.

```
(literalize constants   ;Stores constants used in productions.
 GENERAL-area-filter    ;Used to filter out small regions.
 COMPACT-min-compact-filter  ;Compactness filter.
 COMPACT-max-area-filter
 COMPACT-min-area-filter
 COMPACT-perimeter-filter  ;Perimeter filter, compact regions.
 LINEAR-ell-lin-filter     ;Ellipse-linearity filter.
 LINEAR-mbr-lin-filter     ;MBR-linearity filter, linear regions.
 confidence-filter         ;min confidence, eliminate all sub-class
                           ;interpretations lower than this.
 runway-minlength-filter   ;min length, linear region as runway.
 runway-maxlength-filter   ;max length, linear region as runway.
 runway-minwidth-filter    ;min width, linear region as runway.
 runway-maxwidth-filter    ;max width, linear region as runway.
 adjacency-criterion       ;max distance, two features are adjacent.
 curvature-criterion       ;threshold, linear regions are curved.
 curvature-minimum         ;length threshold, to compute curvature.
 texture-criterion         ;threshold, region considered untextured.
 orientation-criterion     ;distance threshold, extension along
 )                         ;orientation direction.

(AP-specs
 ^spec-type                ;Type of specification.
   urban-area-around-airport
 ^operation nil            ;Hypothesis only, no oper performed.
 ^prediction-value nil     ;Expected value for specification.
 ^probability 90           ;Probability expectation is correct.
 )

(AP-specs
 ^spec-type number-of-runways
 ^operation equal          ;Exact number of runways to be found.
 ^prediction-value 5       ;Expected value for runways.
 ^probability 100          ;Probability expectation is correct.
 )
```

Fig. 33. Evaluation criteria and expectations.

```
region-to-fragment::initialize-region
  Interpretation of each region begins by creating a context where matching
  against specific shape attributes for each class and subclass occurs.
region-to-fragment::generate-linear-class-match
  Test the shape attributes of this region against linear class prototype.
region-to-fragment::make-linear-fragments
  If region matched the linear characteristics successfully make an instance
specific::get-region-curvature::uncalculated-1
  If a linear fragment, we need the curvature to do any further matching.
region-to-fragment::generate-compact-class-match
  Match this region against attributes of the "ideal" compact fragment.
region-to-fragment::failing-compact-match
  The match against the compact attributes failed. No further interpretation
  in the compact class is performed.
region-to-fragment::generate-small-blob-class-match
  Match this region against attributes of a small-blob region.
region-to-fragment::failing-small-blob-match
  The small-blob match has failed.
region-to-fragment::generate-large-blob-class-match
  Match this region against attributes of a large-blob region.
region-to-fragment::failing-large-blob-match
  The large-blob match has also failed.
region-to-fragment::generate-road-subclass-match
  We match this region against all the subclasses of those classes that
  matched successfully.
region-to-fragment::failing-road-match
  This region should not be interpreted as a road.
region-to-fragment::generate-taxiway-subclass-match
  Generate the match against the second linear subclass, taxiway.
region-to-fragment::interpret-as-taxiway
  The match against the "ideal" taxiway succeeds. Create a fragment
  hypothesis with this region as a taxiway.
region-to-fragment::generate-runway-subclass-match
  Generate the match against the last linear subclass, runway.
region-to-fragment::interpret-as-runway
  The match against the "ideal" runway succeeds. Create a separate fragment
  hypothesis with this region as a runway.
region-to-fragment::exit-context
  No more matching to be done on this region. The system chooses another
  region to interpret.
```

Fig. 34. Region-to-interpretation rules.

(depth). Linear regions are evaluated with respect to curvature. These rules hypothesize a class interpretation for the image region as described in Section III-A. Multiple fragment interpretations can be created for each image region. Once a class interpretation has been created, a specialization of the class interpretation is attempted. New fragment interpretations are created for those fragments that satisfy the subclass specialization. Fig. 34 gives a sequence of region to interpretation rules operating on a linear region.

C. Local Evaluation Rules

Local fragment evaluation rules (30 rules) are used to invoke image processing tools for region enlargement, extension, and join/merge. These rules are specialized within feature classes. For example, given several linear image regions, hypothesize a new linear region that encompasses each original region. These rules attempt to verify this hypothesis by invoking a linear feature extraction module using the new (hypothesized) linear region as a guide. In lieu of image analysis, other linear alignment rules are used to search for possible linear fragment interpretations that have not been specialized into runway or road interpretations, and to hypothesize (hallucinate) a connected runway fragment. Thus, fragment interpretations can be generated in cases where complete low-level support is lacking. Such fragment interpretations are marked as to whether they have support in the image, as well as maintaining a list of supporting fragments. Alignments are specialized for nearly straight and curved regions. The actual evaluations are performed outside of the rule interpreter using a procedure that attempts to align linear fragments and bridge gaps in the segmentation, using splines for local interpolation [22].

From a practical standpoint, external calls by the production system need only determine the operation to be performed and the region elements participating in that operation. A process outside of the interpreter can determine the actual image data using the symbolic name and extract any geometric information it might need. Since all of the spatial computation and image analysis is organized outside of the OPS5 rule interpreter, it is important to keep this interface simple.

D. Consistency Rules

Consistency rules (263 rules) recognize groups of potentially mutually consistent fragment interpretations. For each subclass specialization, a consistency strategy is defined. This strategy contains a list of rules that apply spatial and contextual constraints to modify the confidence of the fragment hypothesis. These rules are relatively simple tests which look for supporting evidence for the fragment hypothesis. Each strategy is implemented as a rule set that may invoke geometric or image-processing routines. The following is a portion of the consistency strategy for fragments having the runway interpretation:

- runways-are-oriented-parallel-to-terminal-building
- check-runway-distance-from-terminal
- taxiways-are-oriented-orthogonal-to-runways
- grassy-areas-border-runways-or-taxiways
- taxiways-are-close-to-runways
- parking-lots-are-far-from-runways
- runways-do-not-have-curved-segments
- taxiways-intersect-runways.

Consistency rules are the major source of knowledge in SPAM. They encode simple observations that are generally true about airport scenes. For example, hangars are usually adjacent to other hangars, access roads lead to parking lots and the terminal building, runways are (usually) not oriented into the terminal building, and are further away than some tarmac and most parking aprons. In SPAM, the effect of a large number of relatively simple rules is that correct fragment interpretations are supported much more frequently by correctly interpreted partners than the random support acquired by incorrect interpretations due to fortuitous spatial alignments. However, since these rules only test local consistency, it is necessary to select the "best" fragment interpretations and aggregate them together to form more global pieces of the airport model. This is the role of the functional area rules discussed in the following section. Other consistency rules include the following sample:

- access-roads-generally-have-curved-segments
- access-roads-are-close-to-perimeter
- access-roads-lead-to-terminal-building
- taxiways-orient-toward-terminal
- terminal-building-is-centrally-located
- terminal-building-is-near-parking-lots
- hangars-adjacent-to-tarmac
- grassy-areas-separate-runways-and-taxiways
- tarmac-between-taxiways-terminal.

E. Functional Area Rules

Functional area rules (23 rules) recognize situations where fragment interpretations can be grouped together based on propagation of compatibility into more global collections. These collections of fragment interpretations are called functional areas. Rules identify those fragments that are consistent with at least one other fragment and are potentially members of the same functional area. The convex hull of these fragments forms the initial functional area. Fragments that fall within the functional area and are compatible with the functional area type, but not necessarily consistent with the initial set of fragments, can be added to the functional area. These fragments are marked as compatible. Similarly, fragments that fall within the convex hull of the functional area that are inconsistent, i.e., hangar within runway functional area, are noted. Rules evaluate the confidence of the functional area in light of the compatibles and incompatibles. If there is not sufficient support for the new functional area, or if an existing functional area of the same type covers nearly the same spatial area, it is not activated. Otherwise, fragments are marked as being a member of a functional area, or having failed to generate one. A typical sequence of functional area rules is given in Fig. 35.

F. Goal-Generation Rules

We have incorporated goal-generation rules (28 rules) which recognize small numbers of situations that are inconsistent with general knowledge of airport layout. These rules recognize situations that may cause large numbers of weakly consistent fragment interpretations to build functional area interpretations. The combinatorics dictate that pruning these weakly consistent fragments can greatly reduce the number of resulting functional area interpretations. These rules monitor the global state of working memory and use goal specification as the focus of attention. For example, since few airports have more than 10 runways, if there are more than 10 valid runway fragments in working memory, we invoke routines to coalesce or possibly eliminate weak fragment interpretations. Methods rely on the invocation of existing rules from the local evaluation and consistency classes with specific fragments as contexts, and with more stringent spatial constraints. For example,

- search for and attempt to join aligned fragments with compatible subclass interpretations;
- remove weak runway fragments that have significantly stronger support as taxiways or roads; and
- reevaluate runway fragments with respect to their position in the scene using more stringent proximity constraints.

Similar situations arise with large numbers of competing terminal and hangar hypotheses. Another type of goal-directed pruning occurs upon recognition that an abnormal percentage of the overall scene is explained by fragment interpretations of a particular subclass. For example, we do not expect that the entire airport is covered with grassy areas, tarmac, or parking lots. Rules to recognize

```
FA::fragment-initialization   Choosing fragment hypothesis
FA::trace-consistents         All the hypotheses that are consistent.
FA::spawn-FA-creation         If at least one valid start creation.
FA::create-functional-area    Instantiate into working-memory
FA::compute-convex-hull       Physical representation of FA
FA::reduce-fa-size            Spatial outlying hypotheses are eliminated.
FA::compute-convex-hull       With the outlyers removed.
FA::find-compatibles          Inside and compatible with FA definition.
FA::find-incompatibles        Inside and incompatible with FA definition
FA::re-evaluate-confidence    Re-evaluate the confidence of the FA
FA::get-fa-superset           Another FA of the same type?
FA::insufficient-overlap      Yes, but not enough overlay to deactivate.
FA::activate-functional-area  Add to working memory as active.
```

Fig. 35. Functional area rule execution.

these situations can further prune weak hypotheses making use of spatial constraints, and looking for overlaps with incompatible fragment interpretations having highly rated hypotheses.

G. Model-Generation Rules

Model-generation rules (50 rules) gather mutually consistent functional areas and build an *airport model.* Initially, the functional area with the greatest confidence is selected. These rules then recognize situations where they can enumerate conflicts between the current model and a newly chosen functional area. The methods for resolution of those conflicts, the decision to augment a model or spawn a new model, are also encoded in this rule set. Model generation rules continue to be active until all the active functional areas have been included in some model. Fig. 36 gives an example of a sequence of model-generation rule execution.

H. Rule Distribution By Fragment Class

In the previous sections, we have given the number of rules in each interpretation class. It is interesting to note the actual distribution of these rules in terms of the types of fragments that they recognize. Fig. 37 gives a breakdown of the frequency of rules by class and subclass.

Overall, linear fragments provide more constraints in the airport scene and can be more reliably determined than the other classes of interpretations. For example, runways can be distinguished fairly reliably based on shape. Roads are harder to distinguish based solely on shape since they are combinations of straight and curving linear regions, which must often be aligned before they can be hypothesized. Roads give many key constraints to the overall scene, such as determining plausible locations of the terminal and hangar areas. We believe that the relative importance of each class is reflected in the number of rules in that class as is our ability to recognize and employ appropriate tests.

Within some classes, it is more difficult to distinguish members of the subclass, therefore, more knowledge is needed to eliminate faulty hypotheses. This fact is clearly represented in the cases of roads and hangars. Since roads can be curved or straight, some small runways and most taxiways have alternative road interpretations that are faulty. Similarly, many small compact regions which are found toward the center of the runway/taxiway area are mistaken for hangars. However, as we will see in the following sections, these alternative interpretations rarely have global support. Consequently, their confidence is significantly weaker than the correct interpretations.

```
MG::initialize
  Create context where only model-generation rules are active.
MG::choose-best-FA
  Choose highest-rated active functional-area.
MG::end-conflict-test
  There were no other functional-areas being considered.
MG::choose-best-FA
  Choose the next highest-rated active functional-area.
MG::test-best-for-conflicts
  Check if there are conflicts between hypotheses.
MG::resolve-conflicts::resolve-by-confidence-1
  Choose the interpretation with the highest confidence.
MG::resolve-conflicts::remove-duplicates-1
  This conflict has been resolved before, therefore remove it.
MG::resolve-conflicts::parking-apron/parking-lot-delay
  For this conflict, there is specific knowledge for evaluating the two
  hypotheses in light of the current model.  Therefore, because we know
  that we need a complete model to resolve this conflict, delay the
  resolution of the conflict until that model is created.
MG::unset-delayed-conflicts
  Activate those conflicts that were delayed.
MG::resolve-conflicts::parking-aprons-and-parking-lots
MG::resolve-conflicts::choose-parking-lot-over-parking-apron
  Case-specific knowledge for this conflict is applied and, for this
  instance, the parking-lot hypothesis was determined to be more likely
  in the context of this model.  For this conflict, the knowledge applied
  was that a parking-apron will occur between the terminal functional-area
  and the runway functional-area, whereas a parking-lot will occur as
  far as possible from the runway functional-area.
MG::enumerate-conflicts
  A new, active functional-area of any type is chosen and is evaluated with
  respect to the functional-areas in the current model(s).
MG::resolve-conflicts::unknown-conflict-type-2
  Unknown conflicts are those interpretations that cannot safely be
  discriminated by confidence alone because the difference in the
  confidence values is very small (less than 0.1).  These conflicts are
  marked as irreconcilable.
MG::fork-new-model2
  Because we have found at least one irreconcilable conflict, the current
  model is split into two models: one containing the first conflicting
  interpretation and one containing second.  The models are tested for
  completeness with respect to the definition of the airport-model.
MG::fork-new-model:failed-to-find-replacement-terminal
  The current model could not be split because the conflict occurred
  with the model's terminal functional-area, and there was no alternate
  terminal functional-area in the model to replace the first one.
MG::resolve-conflicts::grassy-areas-and-tarmac
MG::resolve-conflicts::choose-grassy-area-over-tarmac
MG::augment-airport-model
  This functional-area can be added to this particular model because all
  conflicts were reconcilable.
```

Fig. 36. Model rule execution.

Class / Subclass	Number of Rules	
Linear	80	
Runway		24
Taxiway		29
Road		38
Compact	64	
Hangar		58
Terminal		22
Small-blob	40	
Parking-apron		16
Parking-lot		32
Large-blob	32	
Grassy-area		24
Tarmac		16

Fig. 37. Rule distribution by interpretation of class.

VII. Evaluation Functions for Hypotheses

As briefly discussed in Sections V-A and C, SPAM assigns confidence values to fragment interpretations. This evaluation occurs in two stages. During the initialization and local evaluation phases, the fragment is assigned a confidence value based on how well the region it represents fits the criteria for class and subclass specialization. During the consistency phase, confidence values are assigned based on the results of compatibility with other fragment interpretations.

The initial confidence function is as follows:

$$\text{confidence} = (io) * (t/v) \hat{}\, 2 + mcv$$

where:

io initial offset:

$io + mcv =$ minimum confidence value (-0.3)

t	the threshold (criterion) for this class
v	the actual region value being tested ($v \,! = 0$)
mcv	maximum allowable confidence value (currently 0.5).

This produces a confidence value between 0.2 and mcv, with the function asymptotically approaching mcv.[2] There are some problems with this, however. Currently, this function allows us to rate an interpretation based only on one criterion, although some class tests require the regions to fit several criteria at once. A better confidence function would accommodate from 1 to $\boldsymbol{n}$ different threshold tests where $\boldsymbol{n}$ is not known *a priori*.

The incremental confidence function used during consistency checking is as follows:

$$\text{increment} = Cr * Cs * [Fr *(1.0 - Co)]$$

where

Cr	is the confidence of the test itself $0 < Cr < 1$
Cs	is the normalized score received from the test $0 < Cs < 1$
Fr	is the fraction of the maximum allowable increment $0 < Fr < 1$; that is, given a perfect score ($Cs = 1$) and a perfect test ($Cr = 1$), this determines the fraction of the maximum increment
Co	is the old confidence value ($0 < Co < 1$).

The increment function factors a relative goodness or selectivity of the test, the current goodness of the fragment being evaluated, and how well the fragment scored on the test. The use of a relative goodness factor allows us to weight tests according to our subjective belief as to how well it disambiguates competing hypotheses. Under this model, a test has weight 1.0 if it uniquely distinguishes one subclass from all of the other members of the class. In practice, test weights range from 0.5 to 0.8. This reflects a lack of particularly strong compatibility tests in a domain where the underlying certainty of the signal data is weak and the belief that real action is a result of the application of many simple tests. Another property of this evaluation function, which asymptotically approaches 1, is that as the score of the fragment increases, it becomes increasingly difficult to increase fragment goodness.

VIII. Performance Analysis

In this section, we give some performance statistics for the examples described in Section IV. Statistics for fragment, functional area, and model generation compare the relative effects of the texture/depth experiments, the effect of hand versus machine segmentations. An additional set of statistics for MACH200 using both the V2 and V3 system is given. MACH200 is a subset of MACH280 reported in Section IV in that it has 80 fewer regions. Most of these regions were small linear features, comprising roads around the terminal building in the left-central area

Fragments Generated

Version	Linear	Compact	S-Blob	L-Blob	Total	# regions
MACH200v2	30	27	26	30	113	197
MACH280v2	40	27	26	30	123	277
HANDv2	15	30	50	44	139	95
MACH200v3	37	9	25	32	103	197
MACH280v3	59	9	25	32	125	277
HANDv3	15	7	48	43	113	95

Linear Class Interpretations

Version	Road	Runway	Taxiway	Uninterpreted
MACH200v2	6	14	19	11
MACH280v2	6	21	24	16
HANDv2	7	4	1	8
MACH200v3	10	10	29	8
MACH280v3	10	16	45	14
HANDv3	7	5	1	8

Compact Class Interpretations

Version	Hangar	Term	Uninterpreted
MACH200v2	20	25	2
MACH280v2	20	25	2
HANDv2	25	30	0
MACH200v3	2	5	4
MACH280v3	2	5	4
HANDv3	4	4	3

Small-Blob Class Interpretations

Version	PApron	Plot	Uninterpreted
MACH200v2	24	26	0
MACH280v2	24	26	0
HANDv2	40	50	0
MACH200v3	25	21	0
MACH280v3	25	21	0
HANDv3	47	33	1

Large-Blob Class Interpretations

Version	GrassA	Tarmac	Uninterpreted
MACH200v2	29	28	1
MACH280v2	29	28	1
HANDv2	35	41	3
MACH200v3	30	20	2
MACH280v3	30	20	2
HANDv3	39	19	4

Fig. 38. Fragment generation statistics.

Functional Areas And Models Generated

Version	Road	Runway	Hangar	Term	Total	Models
MACH200v2	4	5	24	19	52	9
MACH280v2	6	3	24	19	52	12
HANDv2	1	4	4	11	20	5
MACH200v3	7	8	5	2	22	1
MACH280v3	12	8	5	2	27	1
HANDv3	1	5	4	4	14	3

Fig. 39. Total functional area interpretations.

Rule Executions by Phase

Version	Build+LE	Consist	Functional	Model
MACH200v2	1816	36528	3941	1174
MACH280v2	2355	39637	3981	1619
HANDv2	1469	8079	1354	357
MACH200v3	1876	11606	1140	507
MACH280v3	2515	14879	1476	969
HANDv3	1405	4318	590	684

Fig. 40. Rule execution statistics by interpretation phase.

of Fig. 7. These statistics show the effect on the execution time and number of productions fired as the number of interpretation regions is varied.

A. Fragment/Functional Area/Model Distributions

Fig. 38 gives a breakdown of the total number of fragments generated by class and subclass. Fig. 39 gives a breakdown of the types of functional areas generated and the total number of models that were generated from these functional areas. Fig. 40 gives rule execution frequencies by interpretation phase.

As described in Section IV, SPAM versions V2 and V3 differ only in that V3 has elevation and texture knowledge supplied. This information is applied by rules which recognize, for example, man-made objects with elevation above the ground plane, as well as those that distinguish textured grassy areas from nearly smooth tarmac and

[2] It is assumed in this equation that $t > = v$, therefore, if a test requires that a value be above a threshold ($t < = v$), then one must replace the quotient (t/v) with (v/t).

parking aprons. The effects of such knowledge is more evident in the reduction of subclass interpretations than in the reduction of generic class interpretations.

Version V3 generated a much reduced set of functional areas. This is encouraging since a proliferation of functional areas can pose a major problem for global model recognition. As seen in the earlier examples, both versions generated good sets of functional areas, so that using our relative "best" model we may also be able to accommodate large numbers of errorful functional areas.

B. Timings

The timings in Figs. 41 and 42 illustrate the effects of increasing the number of initial regions and, more interestingly, the effect of adding elevation (depth) and texture knowledge to the system. Timings are averaged over several runs of the system for each machine segmentation data set on a VAX 11/780, with 8 Mbytes of memory running the UNIX operating system.

```
# regions          Avg User time   Avg System time
Build and Local Evaluation Phases:
    95             27:09.2         2:48.7
    200            27:21.0         4:24.5
    280            47:54.2         5:41.2
Local Consistency Phase:
    95             2:23:41.8       50:47.6
    200            14:18:31.6      4:03:59.7
    280            17:32:58.3      5:19:18.3
Functional Area Phase:
    95             2:02:56.1       16:31.3
    200            4:58:52.6       37:55.3
    280            5:10:51.3       42:25.2
Model Generation Phase:
    95             10:09.6         1:21.7
    200            18:03.4         1:00.5
    280            59:23.0         5:14.7
```

Fig. 41. V2 timings (CPU h:min:s.tenths).

```
# regions          Avg User time   Avg System time
Build and Local Evaluation Phases:
    95             20:24.2         2:31.9
    200            32:48.7         3:38.9
    280            1:02:22.9       5:47.3
Consistency Phase:
    95             1:04:16.6       21:13.4
    200            3:56:10.9       1:10:09.6
    280            5:32:58.3       1:26:36.9
Functional Area Phase:
    95             37:05.2         5;31.2
    200            1:01:44.8       10:31.4
    280            1:19:18.3       14:06.3
Model Generation Phase:
    95             8:16.5          35.4
    200            14:25.2         43.9
    280            32:23.0         1:19.3
```

Fig. 42. V3 timings (CPU h:min:s.tenths).

A prototype version, V1, spent much of its time (70 CPU hours) in the OPS5 match and conflict resolution phases. This very inefficient implementation structured rules such that the matcher matched *all* fragments and *all* regions against each production rule. Examples of pragmatic issues in building OPS5-based systems can be found in Brownston *et al.* [23]. We performed some optimizations on rule evaluation order and restructured all rules to make them context dependent, matching only when an appropriate context was generated. We also allowed the region and fragment working memory elements to overlap by putting redundant interpretation information in the region structure and low-level database information in the fragment structure. With these changes, the OPS5 matcher had significantly fewer match candidates since it is matching (effectively) on one working memory element.

The significant decrease in evaluation times in V3 over V2 is primarily due to the decrease in the total number of subclass fragment interpretations that participate in consistency and functional area phases. Further, certain particular interpretations such as terminal, hangar, and taxiway, that have large number of consistency rules, were greatly reduced. This is easily noted in Fig. 40.

IX. Conclusions

SPAM represents research in progress. Currently, the system can extract and identify some runways, taxiways, grassy areas, and buildings from region-based segmentations for several images of National Airport in Washington, DC. It uses these primitives to build multiple plausible functional areas descriptions which support the runway and terminal building, hangar, and access road components of the scene model. These components generate multiple models that represent the spatial layout of the airport. The models generated by SPAM using both machine and hand segmentations compare favorably and appear to give a good description of the airport scene. Many problems remain, some in reliable low-level feature extraction from the imagery, others in the design and implementation of effective recognition strategies using the rule-based approach.

As mentioned in Section VI, Fig. 32 gives an indication as to the potential for growth within each rule class as SPAM is engineered to accommodate a variety of airport organizations. Based on preliminary experimentation with hand segmentation data for the interpretation of Los Angeles and Dulles airports, we believe that the largest growth of rules will be due to the expansion of subclass interpretations to include features such as maintenance, air cargo, and service buildings, and a greater variety of runway/taxiway spatial organizations. The implication is that spatial relationships between all existing and newly created subclass interpretations must be expressed in terms of consistency rules, hence the potential for large growth. Rules that aggregate fragment interpretations into functional areas will also be effected by the increase of new subclasses or more general spatial organizations. However, we do not expect a large increase in the number of model generation and region-to-fragment rules.

A near-term agenda for continuing our work in rule-based aerial image interpretation is as follows.

- Continue to measure the effect of the number of machine-generated regions and various image domain cues such as texture and depth with respect to the quality and completeness of the models generated.
- SPAM needs to be able to refine a more complete and consistent airport interpretation through analysis of the initial models that it generates. In some sense, the model generation phase could be viewed as the beginning of a cycle of analysis phases that would focus image analysis tools with very specific goals in promising portions of the scene. For example, within a hangar functional area, look for planes on the parking apron.

• Explore techniques to determine what information is currently lacking in the airport model(s) and methods for acquiring that knowledge from the image or by further evaluation of the fragment interpretations. Develop more sophisticated goal generation rules to detect and explain missing components of the scene model.

• Apply the SPAM system to other airport scenes.

Given the results to date, we believe that the integration of map knowledge, image processing tools, and rule-based control and recognition strategies will be shown to be a powerful computational organization for automated scene interpretation in aerial imagery.

REFERENCES

[1] C. Froesch and W. Prokosch, *Airport Planning*. New York: Wiley, 1946.

[2] R. Horonjeff and F. X. McKelvey, *Planning and Design of Airports*, 3rd ed. New York: McGraw-Hill, 1983.

[3] T. O. Binford, "Survey of model-based image analysis systems," *Int. J. Robot. Res.*, vol. 1, no. 1, pp. 18-64, Spring 1982.

[4] R. A. Brooks, "Symbolic reasoning among 3-D models and 2-D images," *Artif. Intell.*, vol. 17, pp. 285-348, 1981.

[5] A. R. Hanson and E. M. Riseman, *VISIONS: A Computer System for Interpreting Scenes*. New York: Academic, 1978, pp. 303-333.

[6] J. M. Tennenbaum and H. R. Barrow, "Experiments in interpretation-guided segmentation," *Artif. Intell.*, vol. 8, pp. 241-274, 1977.

[7] M. A. Fischler, J. M. Tenenbaum, and H. C. Wolf, "Detection of roads and linear structures in low resolution aerial imagery using a multisource knowledge integration techniques," *Comp. Graph. Image Processing*, vol. 14, pp. 201-223, Mar. 1981.

[8] M. Nagao and T. Matsuyama, *A Structural Analysis of Complex Aerial Photographs*. New York: Plenum, 1980.

[9] Y. Ohta, "A region-oriented image-analysis system by computer," Ph.D. dissertation, Kyoto Univ., Kyoto, Japan, Mar. 1980.

[10] B. L. Bullock *et al.*, "Image understanding application project: Status report," in *Proc. DARPA Image Understanding Workshop*, Sept. 1982, pp. 29-41.

[11] R. A. Brooks, "Symbolic reasoning among 3-D models and 2-D images," *Artif. Intell.*, vol. 17, pp. 285-349, 1981.

[12] T. Matsuyama, "A structural analysis of complex aerial photographs," Ph.D. dissertation, Dep. Elec. Eng., Tech. Rep., Apr. 1980.

[13] M. Nagao, T. Matsuyama, and H. Mori, "Structural analysis of complex aerial photographs," in *Proc. 6th Int. Joint Conf. Artif. Intell., IJCAI*, Tokyo, Japan, Aug. 1979, pp. 610-616.

[14] P. G. Selfridge, "Reasoning about success and failure in aerial image understanding," Ph.D. dissertation, Univ. Rochester, Rochester, NY, Tech. Rep. 103, May 1982.

[15] S.-S. V. Hwang, "Evidence accumulation for spatial reasoning in aerial image understanding," Ph.D. dissertation, Univ. Maryland, College Park, MD, 1984.

[16] D. M. McKeown, "MAPS: The organization of a spatial database system using imagery, terrain, and map data," in *Proc. DARPA Image Understanding Workshop*, June 1983, pp. 105-127; Dep. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-CS-83-136.

[17] —, "Digital cartography and photo interpretation from a database viewpoint," in *New Applications of Databases*, G. Gargarin and E. Golembe, Eds. New York: Academic, 1984, pp. 19-42.

[18] D. M. McKeown and J. L. Denlinger, "Map-guided feature extraction from aerial imagery," in Proc. Second IEEE Comput. Soc. Workshop Comput. Vis. Represent. Contr., May 1984; Dep. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-CS-84-117.

[19] —, "Image analysis using cooperating knowledge sources: Road tracking," Dep. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA, Tech. Rep. in preparation, 1985.

[20] C. A. McVay, B. D. Lucas, and D. M. McKeown, "Stereo verification in aerial image analysis," Dep. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA, Tech. Rep. in preparation, 1985.

[21] C. L. Forgy, "The OPS5 user's manual," Dep. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA Tech. Rep., 1981.

[22] D. M. McKeown and J. F. Pane, "Alignment and connection of fragmented linear features in aerial imagery," in *Proc. IEEE Comp. Vis. Pattern Recogn. Conf.*, June 1985; Dep. Comput. Sci., Carnegie-Mellon Univ., Tech. Rep. CMU-CS-85-122.

[23] L. Brownston, R. Farrell, E. Kant, and N. Martin, *Programming Expert Systems in OPS5*. Reading, MA: Addison-Wesley, 1985.

David M. McKeown, Jr. (S'71-M'72) received the B.S. degree in physics and the M.S. degree in computer science from Union College, Schenectady, NY.

From 1972 to 1974 he was an Instructor in Computer Science and electrical engineering at Union College. During 1974-1975 he was a Research Associate at George Washington University, Washington, DC, and a member of the Technical Staff at Goddard Space Flight Center, Greenbelt, MD. He is currently a Senior Project Scientist with the Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA. His research interests include image understanding for aerial photointerpretation, digital mapping and image/map database systems, computer graphics, and artificial intelligence. He is the author of over 25 papers and technical reports, and is an active consultant for government and industry in these areas.

Mr. McKeown is a member of the Association for Computing Machinery, American Association of Artificial Intelligence, American Society for Photogrammetry and Remote Sensing, and Sigma Xi.

Wilson A. Harvey, Jr., was born in Baltimore, MD, in 1962. He is currently completing the B.S. degree in both physics and mathematics at Carnegie-Mellon University, Pittsburgh, PA.

Since 1983 he has been a Research Programmer for the Department of Computer Science, Carnegie-Mellon University. His interests include artificial intelligence, especially knowledge representation and organization, distributed processing, and quantum mechanics.

John McDermott, for photograph and biography see this issue, p. 522.