AD-P000 129

SCENE ANALYSIS USING REGION-BASED CONSTRAINT FILTERING

Les Kitchen

Computer Vision Laboratory, Computer Science Center
University of Maryland, College Park, MD 20742

ABSTRACT

A general-purpose scene-analysis system is described which uses constraint-filtering techniques to apply domain knowledge in the interpretation of the regions extracted from a segmented image. An example is given of the configuration of the system for a particular domain, FLIR (Forward Looking InfraRed) images, as well as results of the system's performance on some typical images from this domain.

1. Introduction

An image (whether on the human retina, on photographic film, or in some electronic device) is formed by a complicated interaction of light with objects in three-dimensional space (a scene). Scene analysis is the process of unravelling this interaction: inferring from an image the arrangement of lighting and objects that produced it. In theory, this problem is indeterminate: A given image may result from many different scenes, all of which happen to appear identical from the observer's viewpoint. But in practice there are usually sufficient restrictions on allowable scenes to permit essentially only one interpretation of the image. The problem is to find this interpretation efficiently. Humans are clearly able to do this. Can computers achieve similar performance?

In this paper we present a method for scene analysis based on the application of constraint-filtering techniques to a network of regions extracted from an image. Such an approach has two chief advantages. First, its conceptual simplicity: It provides a clean separation between the general processing algorithm and the knowledge about a particular domain, which is expressed declaratively as constraints. Second, its computational speed: Constraint-filtering can be decomposed into many almost independent processes which can all be run in parallel on a suitable multiprocessor computer.

To try this approach, we have implemented a prototype system that does scene analysis by constraint filtering. A diagram giving an informal overview of the system is shown in Figure 1. In the interest of expediency we have made many simplifications. For example, only a few crude measurements are made on the extracted regions. In the following sections, we will describe the prototype system, taking note of these simplifications. We will also show how the system is used in a particular domain -- forward-looking infrared (FLIR) images of battlefield scenes. This domain was chosen partly because of its military interest, but primarily because its moderate complexity is just about right for fully exercising the prototype system. Then we will discuss the system's performance, taking care to distinguish those failures that are inherent in the method from those that are merely the result of simplifications made in this implementation, and finally, we will suggest directions for further progress.

2. Segmentation

A digital image is merely an array of light intensity (or color) values. There seems to be no way of going directly from these values to a description of a scene in terms of the objects in it. As argued by Barrow and Tenenbaum [],2], Marr [3], and numerous others, several stages of processing are needed, each with its own intermediate representations of the information contained in the image. A first step is to organize the pixels into groupings that correspond more closely to the objects in the scene.

Typically, this is done by segmenting the image into regions of fairly homogeneous brightness. For many scenes, this is a reasonable thing to do. In most cases, the regions will correspond to the objects themselves, or else to significant pieces of them. By this means the myriads of pixels in an image can be reduced to a few score, or a few hundred regions, considerably decreasing the amount of data that must be processed, but with little loss of information. Furthermore, since regions more closely correspond to objects, expectations about the appearance of objects can be more readily applied to the regions than to unorganized pixels.

The support of the Defense Advanced Research Projects Agency and the U.S. Army Night Vision Laboratory under Contract DAAG-53-76C-0138 (DARPA Order 3206) is gratefully acknowledged, as is the help of Clara Robertson in preparing this paper. The author would like to express his debt to Azriel Rosenfeld, Teresa Silberberg, Larry Davis, and Michael Shneier for helpful and stimulating discussions during the development of this system and to Robert Kirby for his quite excellent implementation of LISP.

However, segmentation into regions is not without problems. An initial process of segmentation normally applies a single criterion of homogeneity over the entire image. Unfortunately, a difference in brightness that is insignificant in some contexts (such as a fluctuation in a textured background) may well be very significant in other contexts (such as part of the border of an object with its surroundings). Most segmentation processes take little account of this sort of contextual information, and so make errors of two sorts: oversegmentation and undersegmentation. Oversegmentation breaks into pieces what should ideally be a single region. Properties of the region as a whole (such as shape and area) and relations with other regions (such as adjacency and surroundedness) cannot be computed directly, but can only be recovered by attempting to merge pieces that are likely to belong to the same region. More serious is undersegmentation, by which several regions that should be distinct are fused together. Again, region properties and relations are lost, but recovering them is a more difficult business of attempting to split the fused region into parts.

Several attempts have been made to overcome this problem. Tenenbaum and Barrow in IGS (Interpretation Guided Segmentation) [4] used domain knowledge to guide the low level segmentation. Constraints about the relationships between objects were used to guide the merging of pixels into regions. Feldman and Yakimovsky [5] also used semantic constraints to guide segmentation. Another approach, used by Nagao and Matsuyama [6], first performs as unguided segmentation and later corrects the errors in this segmentation by a semantically controlled process of merging and splitting regions. We assume that undersegmentation never occurs, and that oversegmentation is not serious: that an object is at worst broken into two or three pieces. We augment our domain model to cover fragments of objects, but without making any attempt to integrate them into wholes. For the simple domain used as an example, the initial segmentation can usually be fine-tuned by hand to fit our assumptions above. Even so, failures are not uncommon, indicating that a more subtle treatment of segmentation errors is needed.

First we smooth the image using an edge-preserving smoothing technique in order to reduce noise. The particular technique used does not matter greatly, but usually we have used Narayanan and Rosenfeld's histogram-guided smoothing technique [7], which has proved quite effective. Next, we requantize the image into a small number of gray levels (typically five), following the peak structure of the histogram of the smoothed image.

After this, the regions themselves can be extracted by a connected components analysis. At the same time, we make a few measurements on each region; these measurements serve as a description of the region for all subsequent processing. We construct the bounding upright rectangle around each region (see Figure 2) and measure the image location of its lower left corner, its width and height, as well as the area and average brightness of the region itself.

As mentioned above, these measurements provide only a crude description of each region, but sufficient for this prototype system. A full implementation would need a more complete description of shape, perhaps the chain code of the boundary of each region. Since any region description is necessarily incomplete, it may ultimately be necessary to refer back to the original image to check for properties that cannot easily or efficiently be extracted by preprocessing operations.

3. Constraint filtering

After segmentation, scene analysis becomes mostly a matter of labelling the regions with their identifications as objects or object parts. (For now we ignore the problem of organizing the parts of objects into wholes.) Clearly, only those labellings are valid that can be derived from an arrangement of real objects in space. Properties of objects, and relationships between them, imply corresponding properties and relationships of the image regions that result from these objects. These projected properties and relationships constrain the possible labelling of regions with object identifications. Thus scene analysis can be reduced to a constraint satisfaction problem. (The early work of Barrow *et al.* [8,9], used this approach, with the constraints derived from a relational structure which provided a single but inflexible scene model.)

The traditional technique for solving such problems is backtracking. However, backtracking is inherently a sequential technique, which does not lend itself well to parallel processing. Even if we restrict our attention to sequential processing, simple backtracking has a serious defect, especially on the problems arising in scene analysis: It suffers greatly from "thrashing" behavior [10, 11,12]. When a failure is discovered, only the most recent labelling is reconsidered. If the true cause of the failure lies in an earlier labelling, it will take the program many steps of blind backtracking before it can undo the incorrect labelling. To overcome these problems, a number of authors [10,11,14,15,16,17,18,21] have proposed "constraint filtering", "relational consistency", or "discrete relazation" techniques for constraint satisfaction problems. Some have emphasized the suitability of these methods for parallel processing, while others have stressed the avoidance of thrashing. We feel that the chief advantage of these methods lies in their potential parallelism, especially since Gaschnig [13] has shown that more sophisticated backtracking methods can outperform sequential implementations of constraint filtering.

In order to perform constraint filtering, it is necessary that those nodes (regions) that constrain each other be connected in a network. It is at least theoretically possible for the labelling of a region to be influenced by any other region in the image, so ideally the constraint network should be a complete graph, connecting each region to every

other region. But in practice this is not feasible, since the number of interconnections grows as the square of the number of regions -- far too fast. Having too many interconnections is undesirable for two reasons: First, it increases the cost of building a hardware network for constraint filtering (though some architectures, such as that of ZMOB [19], permit arbitrary interconnection at no extra hardware cost). Second, and more important, it increases processing time, since the amount of computation done for each region is roughly proportional to the number of regions it is connected to.

Therefore, it is desirable to limit the number of interconnections. This must be done carefully, since the correctness and effectiveness of the constraint filtering depend on the completeness of the interconnections, and the lack of a necessary connection might prevent or mislead the application of an important constraint. We would like to build a network of interconnections that is as sparse as possible, but still produces the same results as the complete graph. Obviously, this cannot be known ahead of time--the best we can do is to connect those regions that have a good chance of being relevant to each other. This is a matter that requires much further investigation, but for now we have implemented a simple notion of relevance: A region is connected to all regions that are very close to it (because these make up its immediate context), and to all very large regions in the image (because these give a good basis for judging it in its global context). Thus the number of interconnections is roughly constant for each node and overall is proportional to the number of regions in the image. This interconnection scheme is imperfect, but appears to work with few errors, at least for the domain of FLIR images used in this report.

Once the configuration of the network is complete, the constraint filtering proper can begin: We first of all attach to each region a list containing all the labellings that it might possibly bear. (Currently, these will be all labellings possible in the domain, although it should be possible to use context and the taxonomy of labels to reduce this initial list considerably.) Each label has associated with it a special "when-proposed" procedure, which is executed for each region whenever that label is first proposed for the region. This permits the calculation of certain parameters that make sense only if the region is interpreted as a particular sort of object. For example, if a region is hypothesized to correspond to an object of a certain intrinsic size, then it may be useful to use the region's apparent size, in conjunction with the camera geometry, to compute the object's range and location in space. Notice, however, that this computation makes sense only under this hypothesis.

Next, the label lists are filtered using what are called here "unary constraints". That is, knowledge about the intrinsic properties of objects is used to eliminate incorrect labellings from each region's label list. Regarded as propositions, the unary constraints have the form: "If a region is to bear this label, then the region must have these properties". The constraint is actually used in the contrapositive form: "If a region does not have these properties, then it cannot bear this label." These properties may be immediate properties of the region, or they may be those computed indirectly by the appropriate when-proposed procedure.

Clearly, these three steps (hypothesizing labels, computing parameters, and filtering labels) could be done at one swoop, with some improvement in computational efficiency. Here they are done separately, for clarity of presentation and ease of programming.

After the region labels have been filtered, we can attach to each interconnection (or arc) a list of label pairs that is the cross-product of the sets of labels on the two regions at either end of the arc. This list represents the joint labellings that are simultaneously possible for the two regions considered. Then all these label-pair lists can be filtered by binary constraints, that is, those joint labellings can be eliminated that violate a constraint on the labelling of pairs of regions. These constraints have the propositional form: "If two regions (say r_1 and r_2) are to simultaneously bear the labels ℓ_1 and ℓ_2 respectively, then r_1 and r_2 must stand in certain relations to each other." Again the constraint is used in its contrapositive form: If the two regions fail to stand in the required relations to each other, the appropriate pair of labels can be deleted from the arc joining them.

Following all this, three more filtering processes can be applied. One of them, filtering by existential constraints, enforces constraints of the following form: "If a region is to bear a certain label then there must exist other regions that have certain properties and stand in certain relationships with the given region." This is very much like a unary constraint, except that the properties of the other regions include the requirement that they bear certain labels, and that those labels are permitted simultaneously with the labelling to which the existential constraint is being applied. Thus existential constraints must examine the arc labellings. Unary constraints need be applied just once, but since the allowable labellings of arcs change during the constraint processing, an existential constraint that is satisfied early may later be violated because a labelling that it depended on has been rejected. Hence the filtering by existential constraints should be redone every time the arc labellings change.

The other two filtering processes, arc-upon-node interaction and node-upon-arc interaction, attempt to enforce consistency between the node labellings and arc labellings. The first process ensures that every node labelling has support from every arc that impinges on it. By "support" we mean that there exists on each arc at least one label pair that has the same label as the region for its first or second component, as appropriate, depending on which end of the arc the node lies at.

If a node labelling lacks support it is deleted. The second process ensures that every arc labelling has support from the nodes at either end of it. Here, by "support" we mean that every label pair on an arc should have as its first element a label that is represented in the labelling of the node at the appropriate end of the arc, and have as its second element a label that is represented in the labelling of the node at the other end of the arc. Any arc label that lacks such support is deleted.

These three interdependent filtering processes provide a simple but effective way of propagating inferences about the identification of regions through the constraint network. They provide the system with a rudimentary form of reasoning about scenes in the sense that its conclusions, if justified logically, would take several proof steps from the given axioms (these are the region properties and relations, and the domain constraints). Of course, all this reasoning is done by a mechanical process of propagating the effects of deleting node and arc labellings, but it can be regarded as a limited form of logical deduction.

Notice that all the filtering processes work strictly by refuting and eliminating labellings. This means that, after the initial labelling generation processing, all the filtering processes could be run independently and asynchronously on the regions in any order, without the fear of race conditions occurring. That is, the results of the constraint filtering will be the same, no matter in what order the individual filtering processes are applied to each node, provided all processes are applied until the network stabilizes--when no further deletion of labellings can be made. However, in the interests of efficiency and simplicity and in order to simulate an actual parallel implementation, we apply the various processes synchronously in parallel over the entire network. As described above, we first perform all the initial node labellings, next all the node filtering by unary constraints, then all the generation of joint labellings on arcs, followed by arc filtering by binary constraints. Now, the arc-upon-node interaction and the existential constraint filtering use the arc labellings to update the node labellings; and the node-upon-arc interaction updates the arc labellings using the node labellings. Therefore it is appropriate to apply these three propagation processes in a cycle of three (in the order given) repeatedly until the network stabilizes (when no further deletions of labellings can be made).

After the constraint filtering has stabilized and terminated, we can turn our attention to the interpretation of its results. Unfortunately, these results will not necessarily be correct in the sense of being a valid solution to the given constraint satisfaction problem. Ideally, we would like to see every region correctly and uniquely labelled with its identification as an object or object part. However, given the way we have decomposed the problem so as to make it amenable to parallel processing, such an outcome cannot be guaranteed. Before discussing these erroneous results in detail, we should stress that both in the example domain used here, and in other domains [17,20], we have not found these errors to be a serious problem in practice. Other authors [21], report similar findings.

One sort of error is that after the filtering a region may retain several labels, not just one. This situation can arise from two causes. First, it may well be that there is more than one valid solution to the constraint satisfaction problem, that is, the image admits of several distinct interpretations. So a given region may have more than one correct identification ascribed to it, either of itself, or in conjunction with multiple identifications of neighboring regions (neighboring in the sense of being directly connected in the network). From one point of view, this can hardly be considered an error: a region can have several different interpretations, and all of these are retained. But if a number of regions all have multiple labels, it may be of interest to discover which unique labellings of all of them are simultaneously possible, and this sort of unravelling cannot be done by mere constraint filtering. Related to this is the second cause of multiple labelling: There may exist in the network an ambiguity that can be resolved in principle, but cannot be resolved by pairwise constraint filtering--its resolution requires the simultaneous examination of the labellings of three or more nodes. Errors such as these are not serious, since they tend to occur infrequently--for most domains it seems that pairwise interaction is sufficient for essentially unambiguous interpretation. Even when they occur they can easily be resolved, by some sort of backtracking technique alone, or in combination with further constraint filtering, as used by Barrow and Tenenbaum in MSYS [21], and by Haralick and Shapiro [15,16]. In most cases, the bulk of the disambiguation will have been done by the constraint filtering, leaving very little work to be done by the final backtracking. However, we have not implemented such a post-processing phase for the current system because our main interest is in the filtering itself.

It is worth remarking that if only unary and binary constraint filtering are used, or existential constraints are used but the network is sufficiently complete, extra labelling (as discussed above) is the only sort of error that can occur. Under these circumstances constraint filtering will be safe in that it will never reject a correct labelling, even though it may retain some incorrect labellings. This means that if every region bears a single label, then we can be sure that all these labellings comprise the unique, correct solution to the constraint satisfaction problem. If any regions bear multiple labels, then we know that unique interpretations could be found, if necessary, by a later backtracking process. Unfortunately, if existential constraints are used, then the required completeness of the network cannot be guaranteed unless it is a complete graph, which is seldom feasible in practice.

The other sort of error that can occur is that a correct labelling is mistakenly rejected. As implied above, this can only happen when existential

constraints are applied to a network that lacks some necessary interconnections. Since the completeness of the interconnections cannot always be determined ahead of time, it cannot be foreseen whether such errors will occur. If they do occur, they they are irreparable, since a label once lost cannot conveniently be reinstated. But once again, these errors, while theoretically possible, have not occurred in our examples because our simple configuration rule mentioned earlier ensures sufficient interconnection in the network--at least for the existential constraints used in the example domain.

Related to this is a problem that occurs if a region loses all of its labellings, that is, all possible identifications of it can be refuted. This means that the region is unrecognizable as anything from the assumed domain. But once any node in the network is unrecognizable, its effect will be propagated until all nodes lose all their labels. Strictly speaking, this is perfectly correct: If a scene contains objects that cannot be recognized, then the scene could not possibly be from our chosen domain, and therefore the whole scene is essentially unrecognizable. The problem is that the constraint filtering implicitly assumes that the set of labels and constraints correctly account for everything that might possibly appear. If this assumption is violated, then the entire image must be rejected, even though the image could be successfully interpreted if the alien object were not there. While theoretically justifiable, this behavior is undesirable in practice. If such a vision system were turned loose on the world, we would not want it to effectively go blind every time an unexpected object chanced into its field of view. One solution to this problem is to postulate a catch-all label for which there are no constraints whatsoever. Any region in the image can therefore bear this label, even those that are otherwise unrecognizable. Of course, all recognizable regions will also bear this label in addition, and there will be numerous additional label pairs attached to the arcs. This will cause no problem with the interpretation, but it does introduce a certain computational overhead which may not be negligible. Another solution, which does not suffer from these problems, is this: When a node loses all its labels, it should be marked as unrecognizable, and then removed from the network, with its connecting arcs as well, so that the undesirable effects cannot spread further. Both of these solutions have ramifications that we shall not go into here. Because of this, and because the problem only arises when the model embodied in the constraints is inadequate, we have not made any special provision for handling it in the prototype system. If any region is found to be unrecognizable, we go back and revise the model to account for the misrecognition.

This brings us to one final matter: How are the constraint models for a particular domain constructed in the first place? Winston [22] has proposed an automatic system for building scene models, that uses inductive inference over a set of training examples. Such an approach is certainly possible here, but the problem of separating relevant from irrelevant features can be expected to be very difficult for all but the simplest scenes. For now, we expect that models will be built by hand. A user of the system relies on his own introspection and knowledge of the domain to construct an initial model, applies it to some well-chosen examples, diagnoses any errors, and then corrects the model accordingly. For many applications, this is a quite acceptable way of building models.

In order to illustrate the matters presented above, we now give examples of the operation of our prototype system in a particular domain.

4. An example domain -- TANKSWORLD

In order to test out the ideas described in the previous sections, we have implemented a prototype system for scene analysis using constraint filtering, and applied it to a domain of forward-looking infra-red (FLIR) images of tanks and other military vehicles on fairly open ground. The image segmentation and region extraction programs were written in the programming language C. The constraint filtering system was written in LISP. This includes the constraint filtering procedures themselves, and also a number of auxiliary procedures, including those that provide the relational primitives out of which the constraints were constructed. The constraints were written as logical expressions in these primitives, using special conventions to mark the variables for the regions.

Example images from this chosen domain (dubbed "TANKSWORLD") can be seen in Figure 3. We admit five principal region labels in TANKSWORLD:

GROUND (corresponding to the ground or any patch of ground)

SKY (the sky or any patch of sky)

SMOKE (a puff of smoke or similar bright compact object)

TANK (a tank, or any vehicle)

TREE (a tree or shrub)

Only TREE and TANK have any real size restriction, so only for these is oversegmentation a problem. Therefore we provide two additional labels, TANK-FRAGMENT and TREE-FRAGMENT, to cover pieces of these objects.

In general, the spatial position of an object cannot be determined from a single image. All that can be said is that the object must lie along a certain line of sight. However, we know that TANKS and TREES (we use the labels here informally to stand for the classes of objects they represent) must stand on the ground, and if we assume that the ground is an approximately level plane, we can use projective geometry and a simple camera model in order to fix their actual spatial locations, and from this determine their ranges, and their actual sizes from their apparent sizes in the image. The simple camera model used for these computations is shown in Figure 4. It assumes that the image is formed by a simple pin-hole camera, with known parameters. For objects in space we use Cartesian

coordinates x,y,z, with the origin on the ground vertically below the camera's pinhole. The x axis runs along the ground to the right from this origin, the y axis directly forward, and the z axis vertically. In the image we use coordinates ξ (horizontal) and η (vertical) relative to an origin at the center of the field of view. These coordinates are related by

$$\frac{\xi}{f} = \frac{x}{y\cos\varphi - (z-h)\sin\varphi}$$

$$\frac{\eta}{f} = \frac{y\sin\varphi + (z-h)\cos\varphi}{y\cos\varphi - (z-h)\sin\varphi}$$

where h is the height of the pinhole above the ground, f is the distance from the pinhole to the film plane, and φ is the dip angle below the horizontal of the optical axis of the camera. These equations give an adequate approximation for any camera, provided the field of view is not too wide. In practice, these parameters should be known. For the images used here they were not known, but were estimated by taking measurements on the images of objects whose size was approximately known.

So for the labels TANK and TREE, we have a when-proposed function that computes spatial location on the ground and approximate vertical and horizontal extent. For the corresponding fragments the when-proposed function can compute only bounds on these values, but these bounds are nontheless useful.

There are 62 constraints used in the current model. The unary constraints are used to enforce the size restrictions on TANKS, TREES and their fragments, limits on the height to width ratio for TANKS and TREES, and restrictions on the position of SKY and GROUND relative to the horizon. The binary constraints are used in two ways: First, to express that the region for a compact object such as TANK, TREE, SMOKE cannot surround the region for any other sort of object (except that TANKS and TREES can surround their respective fragments). Second, to enforce restrictions on the relative brightness of objects, that SMOKE is brighter than anything else, TANK is not brighter than anything else, and that objects of the same class have roughly the same brightness, except for GROUND which has considerable variation. (Notice that the system is given no knowledge of the absolute brightnesses of objects--only relative brightness is used. This was done deliberately in order to demonstrate the ability of the constraint filtering.) Finally, the existential constraints capture the requirement that TREES and TANKS must rest upon a piece of GROUND, and that a fragment of an object must have next to it another fragment of the same sort such that the two taken together do not exceed the size restrictions for the corresponding whole object.

In Figure 5, we show some typical subimages from this domain. Figure 6 shows these images after segmentation with boxes drawn around the regions. Because of memory limitations of the present implementation, regions below a certain size were ignored, and interconnections were made only between regions whose boxes were immediately adjacent or overlapping, and to the one or two largest regions in each image. The constraint-filtering system was run on these examples, and the results are presented in Figures 7 and 8. (For each label, unambiguously labelled regions are shown in white; ambiguously labelled regions, which bear other labels as well, are shown in gray.) In all cases the constraint filtering stabilized after only a few iterations of the propagation processes.

As can be seen, the results are quite good, especially considering the noisiness of the original image, and the blind simplification done by the segmentation and the region extraction. A number of problems with the results are worth discussing, since they illustrate limitations of this approach.

Since there are so few constraints on GROUND, many other sorts of objects will retain this label. In a sense, this is perfectly unobjectionable. In these images there is no way of distinguishing a tank from a patch of ground with the same shape and coloration as a tank. In this domain the TANK interpretation is more likely, but the constraint filtering has no mechanism for expressing preference between two logically irrefutable labellings.

Another problem is that because there is often little contrast between sky and ground, quite a number of regions straddle the horizon, and thus admit both the labels SKY and GROUND. It is clear that the segmentation is wrong, but the current system can only accept uncritically the regions it receives from the segmentation. A more sophisticated system could attempt to modify the segmentation when such a contradiction was detected. In a few cases, an object clear to the eye is merged with another object because of a short segment of low contrast boundary between them and is thus lost altogether. Detecting and repairing such a mistake in the segmentation is really quite difficult.

The limited context provided by the limited interconnection of regions causes some difficulties. There are a few regions that retain the label SMOKE, not because they are the brightest regions in the image, but merely because they are brighter than anything they are connected to. In some other images it happens that a cluster of TREE-FRAGMENTS support each other, even though altogether they are too large or too small to comprise an entire TREE. The system takes into account only the pairwise interactions of the fragments, without trying to organize them into a coherent whole.

There are some other misidentifications that can be blamed on the simplified shape description used here. A number of odd-shaped regions are labelled as TREES just because the regions happen to fix boxes of about the right size and shape, even though it is apparent that they look nothing like TREES in their actual shape.

Despite these problems, it is clear that the constraint filtering can accomplish almost all the task of analyzing these scenes. In the next section, we will discuss some of the issues raised by

these problems, and consider ways of extending the constraint filtering process to overcome them.

5. Discussion

We have seen in the previous section that constraint filtering is a feasible technique for scene analysis, even if implemented in a very simple way. We will discuss here some extensions to this technique that would overcome most of the shortcomings of the current approach, and lead to a more powerful and flexible scene analysis system.

One straightforward improvement would be to provide a more accurate shape description for regions, which would permit more realistic computation of region properties and relations. The representation of regions by boxes is convenient, but hardly satisfactory. In a few cases, a spurious adjacent or surround relation will hold between the boxes of two regions, when in fact it is not true of the regions themselves. This can lead to errors in interpretation.

It would be desirable to provide some facility for indicating preference between several labels for a region, all logically equally possible, but one far more likely. The unavoidable labelling of TANKS also as GROUND, mentioned in the previous section, illustrates this problem. More generally, as suggested by numerous authors [14,21,23], it would be useful to attach probabilities or confidence measures to all the hypotheses, properties and relations in the system, and provide a calculus for combining these confidence measures. As an example, the relation same-brightness just checks that the difference in brightness between two regions is below some given threshold. In most domains this is unsatisfactory. There is no sharp cut-off between "same" and "not the same". All we can say is that the greater the difference in brightnesses between two regions, the less reasonable it is to regard them as having the same brightness.

Related to this is the need for a more subtle combination of evidence. A certain label may have a number of constraints applicable to it. If a certain region passes all but one of these constraints it would lose that label. But in some circumstances it may be more reasonable to suspect that the labelling is correct but that some error has been made in the evaluation of the failed constraint. Perhaps an important piece of evidence was obliterated by noise, occlusion, or poor segmentation. Ideally a scene analysis system should be able to tolerate such lost evidence, and even attempt to recover it by a closer re-examination of the original image.

This brings us to the matter of the interaction between the scene analysis system and the image data. In the current system there is a strictly one-way flow: segmentation, then analysis of the segmentation. It would be preferable to have a mechanism whereby the higher-level analysis could, under certain conditions, call for a re-examination of parts of the original image in order to search for features that may have been lost in the initial processing. The scene analysis system should also be provided with an arsenal of assorted image processing routines, in addition to segmentation, in order to capture lines, spots, and other features that are likely to be lost during segmentation.

The current scheme for interconnecting regions into a network, while effective, is fairly ad hoc. It should be possible, by an analysis of the constraints, to make a more rational decision about the connection of regions. A region may need only to be connected to regions that stand in certain relations to it and that retain certain labels after the unary constraint filtering, for these are the only regions that could possibly falsify the applicable constraints. More generally, it may be advantageous to permit the reconfiguration of the network during processing, although this would have to be done with great care in order to retain the desirable properties of constraint filtering.

One deficiency of the current system is its clumsy notation for expressing constraints that apply to a number of labels. As mentioned earlier, to express the notion that SMOKE is the brightest object in the domain we must provide a separate brighter-than constraint for every other label in the domain. This could be overcome, at some cost in efficiency, by permitting some sort of quantification over labels, allowing constraints like "For all labels ℓ, not equal to SMOKE, SMOKE is brighter than ℓ." But this is only a cosmetic change, which does not address the underlying problem. The current system regards all the labels as being quite independent of each other. This becomes quite a serious computational inefficiency as the number of labels becomes large (as it will for any realistic domain), especially as the amount of calculation on each arc is roughly proportional to the square of the number of labels in the domain. But in reality, the labels in a particular domain will usually show certain similarities among themselves and share many constraints. It is wasteful to independently re-evaluate for each label these shared constraints. This inefficiency can be naturally and effectively overcome by organizing the labels of a domain into a taxonomy based on similarity and shared constraints. The system could initially propose generic labels, which stand for whole classes of objects, and test these labels by applying only those constraints common to whole classes of objects. Later, when no further progress could be made by such general reasoning, the generic labels could be replaced by more specialized labels, and more specialized constraints could be applied. For example in TANKS-WORLD, we could group all objects that must lie below the horizon into a single class, and eliminate this class label from all regions that lie above the horizon. Once this had been done we could specialize objects below the horizon into classes of compact and extended objects. Later, the compact objects could be subdivided into TANKS and TREES. If necessary, TANKS and TREES could be further classified into their different models and varieties. While this scheme is intuitively clear, some work still remains to be done in order to properly formalize it, especially in regard to the

interactions between nodes at different levels of specialization.

The current system also suffers from a one-level treatment of network nodes. It is possible for a cluster of regions to retain the label TREE-FRAGMENT, even though the cluster, considered as a unit, looks nothing like a TREE. What is needed is a mechanism for creating new nodes having existing nodes as parts. This becomes more acutely necessary in more complex domains whose objects may be built up from distinct parts. Even in TANKSWORLD, at slightly better resolution, a TREE would be seen to consist of a trunk, branches and foliage; and a TANK would show wheels, turret, gun-barrel and other details. Such techniques for hierarchical constraint filtering have been studied in simpler domains [24,25], but require further development for more complex domains, especially if they are to be applied in an efficient manner.

Recently, Davis [26] has shown that constraint filtering, expressed formally in logic, can be regarded as a limited form of inferencing. This raises the possibility that more powerful forms of constraint filtering could be devised. Currently, these techniques work by falsifying simple hypotheses about the individual and joint identifications of nodes in network. More powerful techniques could conceivably reason about other properties and relations between nodes, for example, occlusion relations between objects. Formal logic and theorem-proving would also provide a convenient means of treating some of the other extensions of constraint filtering described above.

In conclusion, we have shown that constraint filtering is an effective means of scene analysis in a domain more complex than has previously been used with such techniques. The deficiencies of the approach, as revealed by the results we have obtained, have suggested a number of improvements and extensions to constraint filtering. The development of these extensions is the object of current research.

REFERENCES

1. H. G. Barrow and J. M. Tenenbaum, "Recovering intrinsic scene characteristics from images," pp. 3-36 in Computer Vision Systems, A. R. Hanson and E. M. Riseman, eds., Academic Press, New York, 1978.

2. H. G. Barrow and J. M. Tenenbaum, "Representation and use of knowledge in vision," SIGART, no. 52, June 1975; also SRI AI Center Technical Note 108, July 1975.

3. D. Marr, "Early processing of visual information," Phil. Trans. Roy. Soc., vol. B275, 1976, pp. 483-524; also MIT AI Lab Memo 340, Dec. 1975.

4. J. M. Tenenbaum and H. G. Barrow, "Experiments in interpretation-guided segmentation," Artificial Intelligence, vol. 8, 1977, pp. 241-274.

5. J. A. Feldman and Y. Yakimovsky, "Decision theory and artificial intelligence: A semantics-based region analyzer," Artificial Intelligence, vol. 5, 1974, pp. 349-371.

6. M. Nagao and T. Matsuyama, A Structural Analysis of Complex Aerial Photographs, Plenum Press, New York, 1980.

7. K. A. Narayanan and A. Rosenfeld, "Image smoothing by local use of global information," IEEE Trans. Syst. Man, Cybern., vol. SMC-11, 1981, pp. 826-831.

8. H. G. Barrow and R. J. Popplestone, "Relational descriptions in picture processing," pp. 377-396 in Machine Intelligence 6, B. Meltzer and D. Michie, eds., University of Edinburgh Press, 1971.

9. H. G. Barrow, A. P. Ambler and A. M. Burstall, "Some techniques for recognizing structures in pictures," pp. 1-29 in Frontiers of Pattern Recognition, S. Watanabe, ed., Academic Press, New York, 1972.

10. D. Waltz, "Understanding line drawings of scenes with shadows," pp. 19-92 in The Psychology of Computer Vision, P. H. Winston, ed., McGraw-Hill, New York, 1975.

11. A. K. Mackworth, "Consistency in networks of relations," Artificial Intelligence, vol. 8, 1977, pp. 99-118.

12. J. Gaschnig, "A general backtrack algorithm that eliminates most redundant tests," 5th IJCAI, Cambridge, MA, Aug. 1977, p. 457.

13. J. Gaschnig, "Experimental case studies of backtrack vs. Waltz-type vs. new algorithms for satisficing(sic) assignment problems", Proc. Canadian Soc. for Comp. Studies of Intelligence, Toronto, July 1978, pp. 268-277.

14. A. Rosenfeld, R. Hummel and S. Zucker,"Scene labeling by relaxation operations," IEEE Trans. Syst. Man Cybern., vol. SMC-6, 1976, pp. 420-433.

15. R. M. Haralick and L. G. Shapiro, "The consistent labeling problem: Part I," IEEE Trans. Patt. Anal. Mach. Intel., vol. PAMI-1, 1979, pp. 173-184.

16. R. M. Haralick and L. G. Shapiro, "The consistent labeling problem: Part II," IEEE Trans. Patt. Anal. Mach. Intel., vol. PAMI-1, 1979, pp. 193-203.

17. L. Kitchen and A. Rosenfeld, "Discrete relaxation for matching relational structures," IEEE Trans. Syst. Man Cybern. vol. SMC-9, 1979, pp. 869-874.

18. J. J. McGregor, "Relational consistency algorithms and their application in finding subgraph and graph isomorphisms," Information Sciences, vol. 19, 1979, pp. 229-250.

19. C. Rieger, "ZMOB: Doing it in parallel," Proc. IEEE Workshop on Computer Architecture for Pattern Analysis and Image Database Management, Hot Springs, VA, November 1981, pp. 119-124.

20. L. Kitchen and E. V. Krishnamurthy, "Fast, parallel, relaxation screening for chemical patent data-base search," to appear in Journal of Chemical Information and Computer Sciences.

21. H. G. Barrow and J. M. Tenenbaum, "MSYS: A system for reasoning about scenes," SRI AI Center Tech. Note 121, April 1976.

22. P. H. Winston, "Learning structural descriptions from examples," pp. 157-210 in The Psychology of Computer Vision, P. H. Winston, ed., McGraw-Hill, New York, 1975.

23. L. Kitchen, "Relaxation applied to matching quantitative relational structures," IEEE Trans. Syst. Man Cybern., vol. SMC-10, 1980, pp. 96-101.

24. L. S. Davis and A. Rosenfeld, "Hierarchical relaxation for waveform parsing," pp. 101-109 in Computer Vision Systems, A. R. Hanson and E. M. Riseman, eds., Academic Press, New York, 1978.

25. L. S. Davis and T. C. Henderson, "Hierarchical constraint processes for shape analysis," IEEE Trans. Patt. Anal. Mach. Intel., vol. PAMI-3, 1981, pp. 265-277.

26. L. S. Davis, "A logic model for constraint propagation," Tech. Report 137, Computer Sciences Dept., University of Texas, Austin, TX, Feb. 1980.

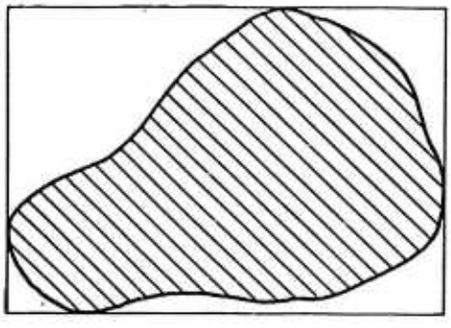


Figure 2. Circumscribing upright rectangle ("box") used to describe a region

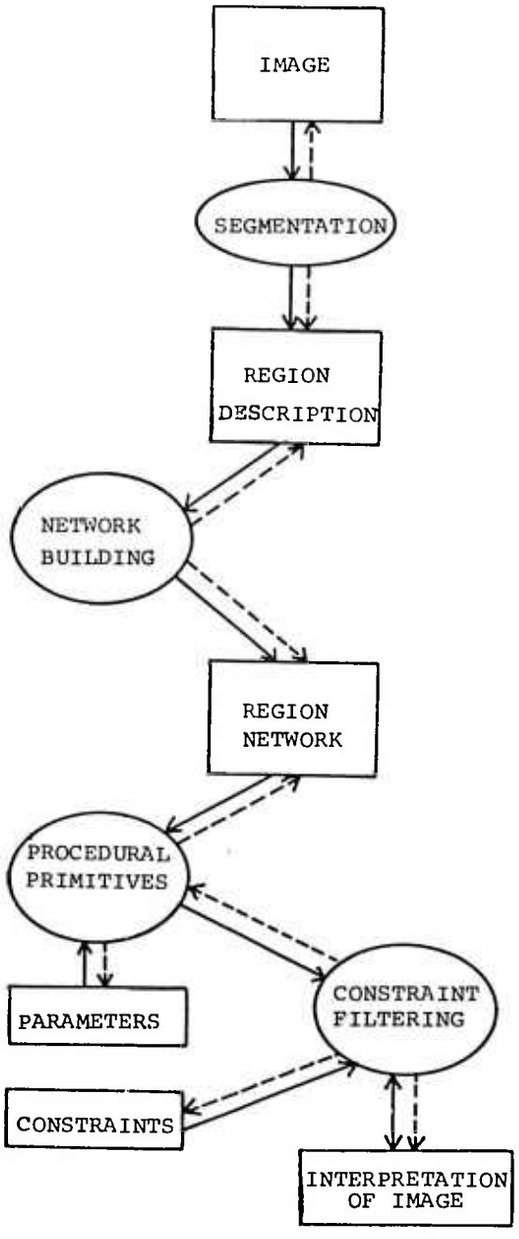
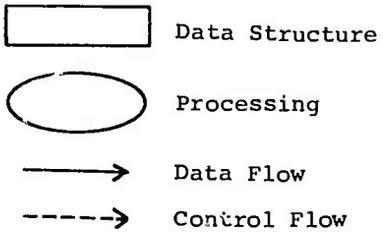


Figure 1. Informal overview of system



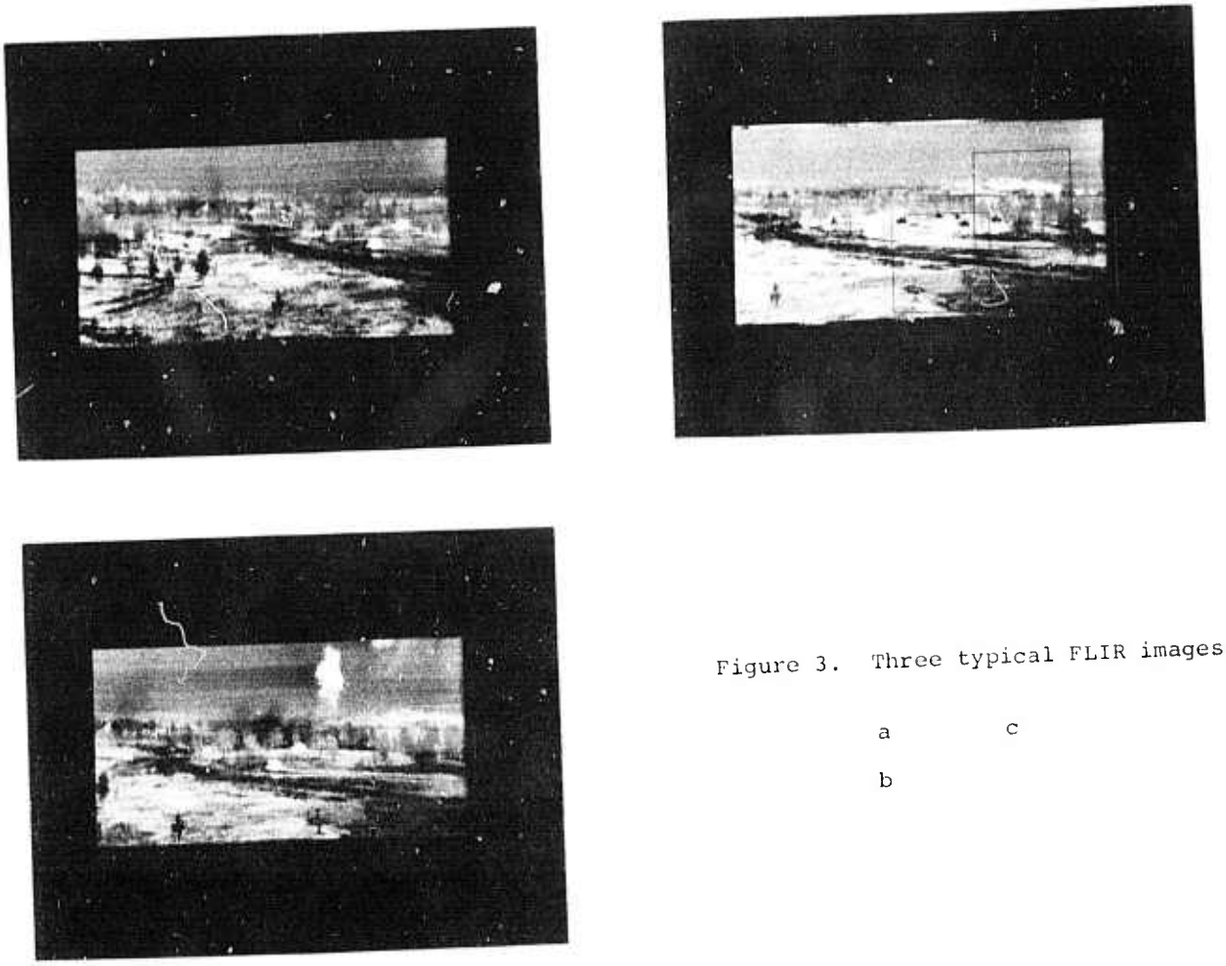


Figure 3. Three typical FLIR images.

a c

b

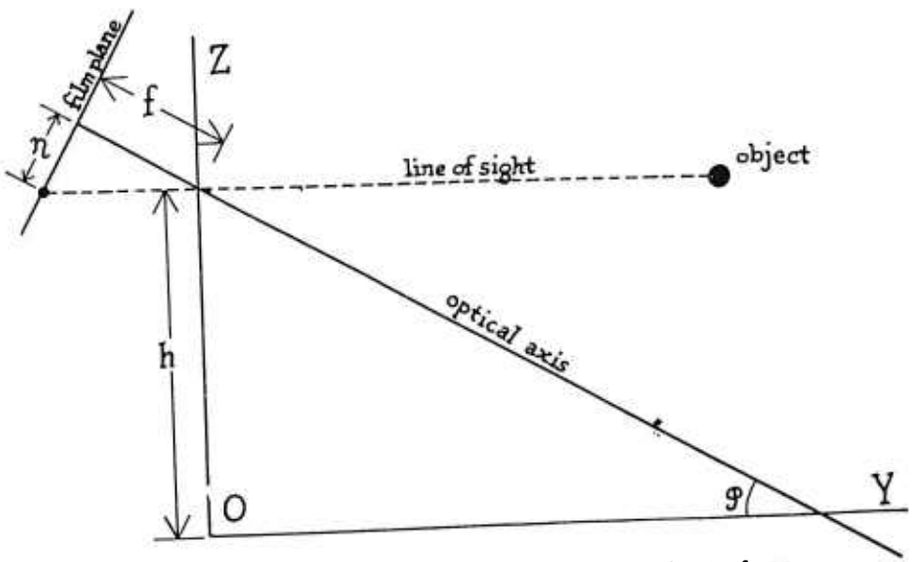
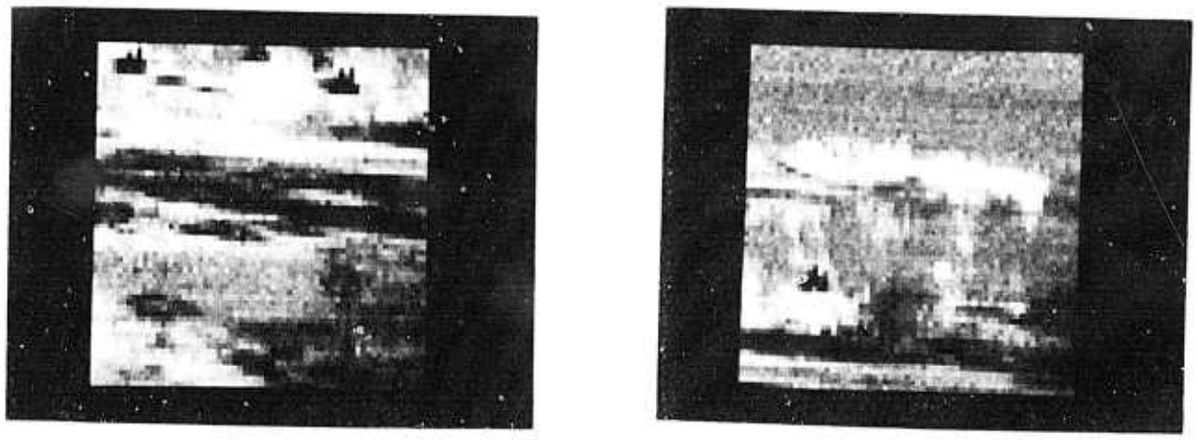
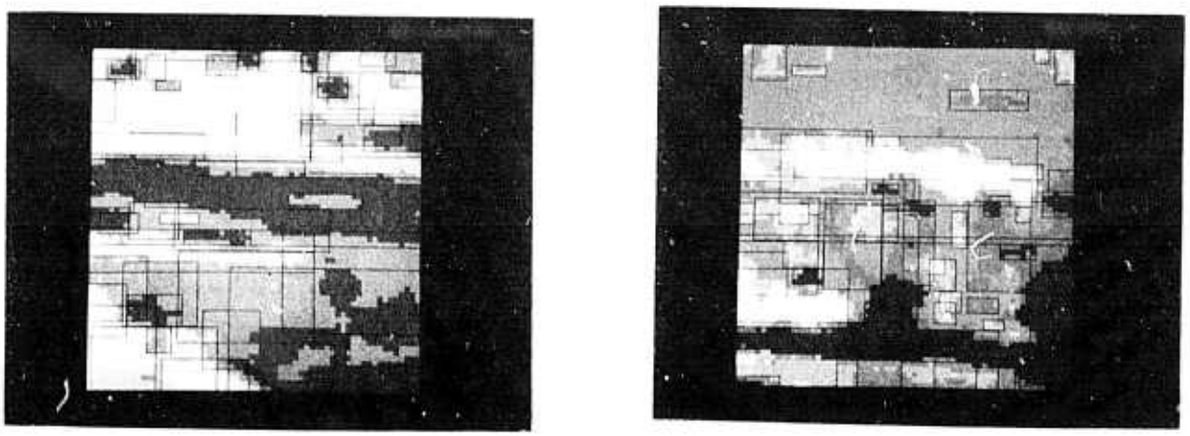


Figure 4. Camera geometry, projected on plane YOZ. x and ξ axes not shown



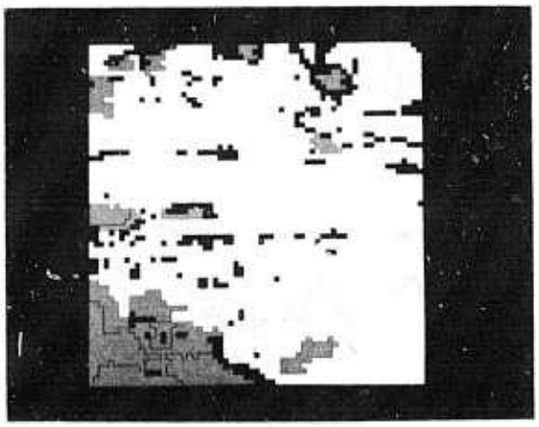
(a) (b)

Figure 5. Two FLIR subimages (boxed in Fig. 3c)

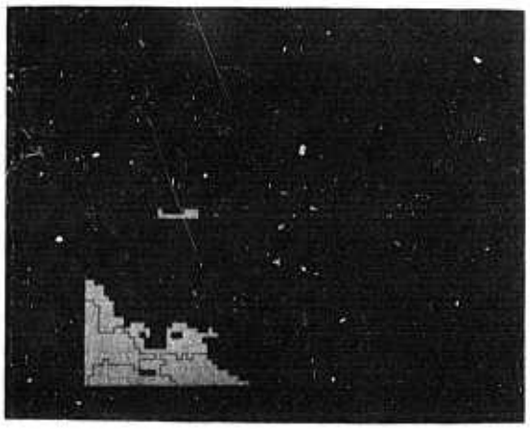


(a) (b)

Figure 6. The subimages in Figure 5 after segmentation, with boxes around regions.



Ground

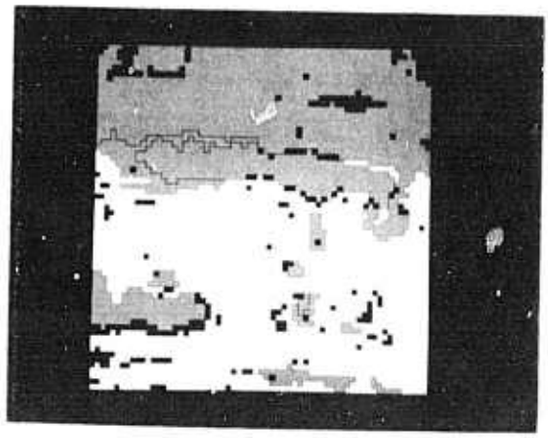


Smoke

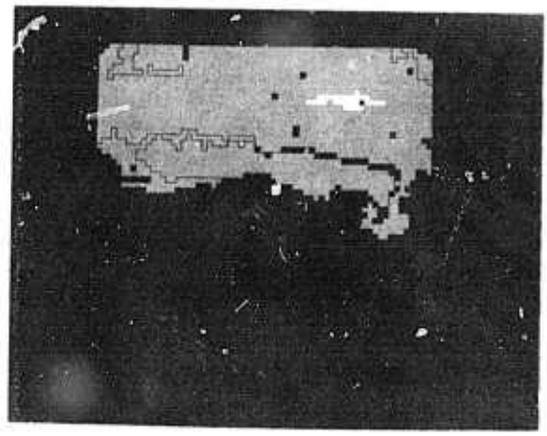
Tree

Tank

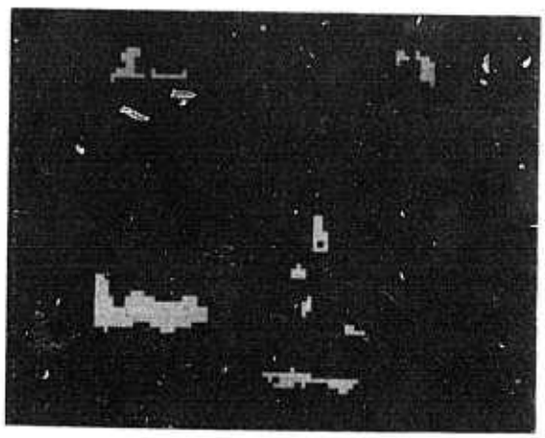
Figure 7. Results of constraint filtering for Fig. 5a.



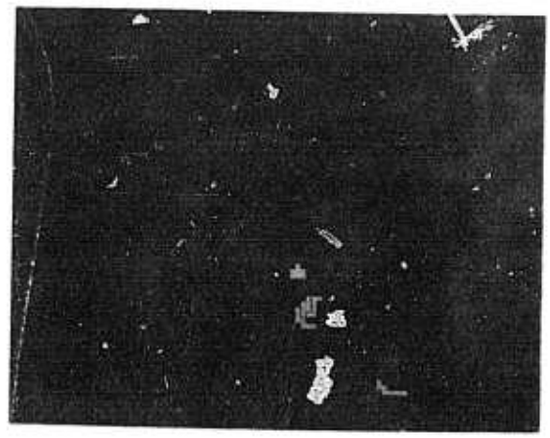
Ground



Sky

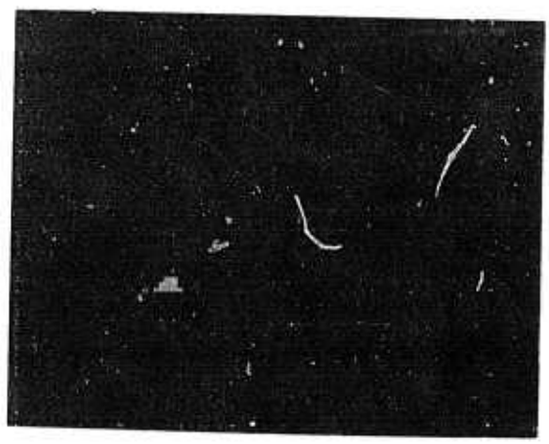


Smoke



Tree

Tree fragment



Tank

Figure 8. Results of constraint filtering for Fig. 5b.