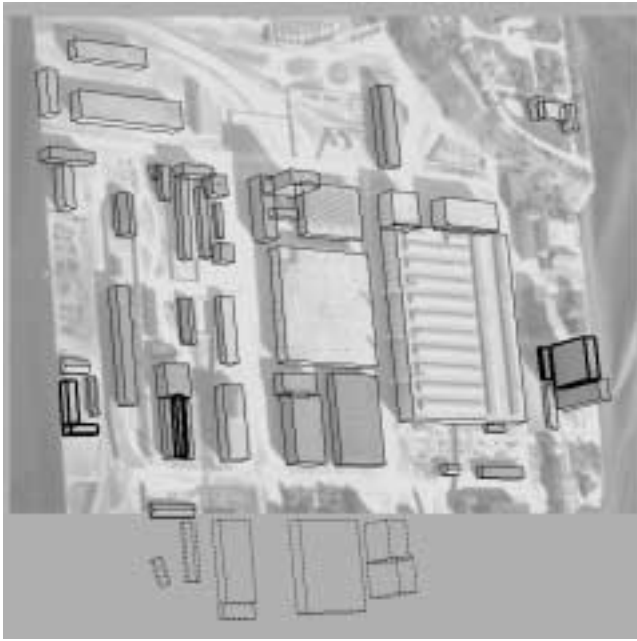


information extracted from the image



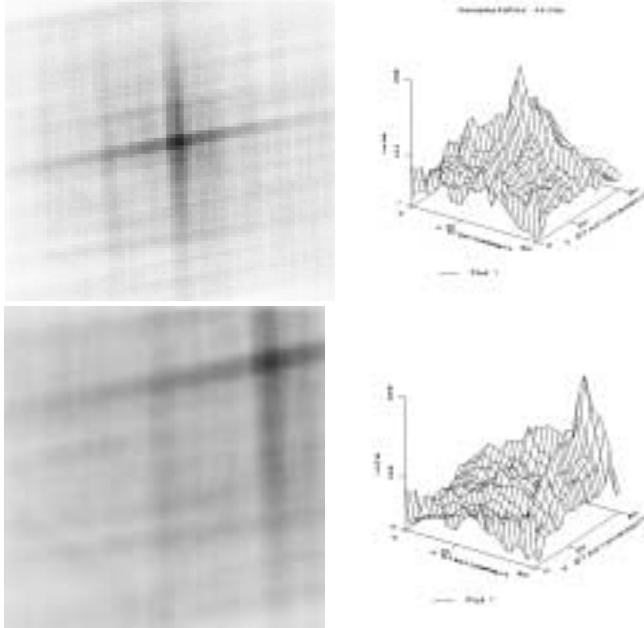
**Figure 10** Validation result for image J2. Objects outside the field of view are shown with dashed lines. Non-validated objects are shown in bold lines.

Although our results are very encouraging, there is certainly room for improvement, both at the registration level and at the verification level. This is part of our future plans. Inaccuracies in the model should be tolerated to a good extent at this stage, for example. These improvements will help processing images that lack shadow definition or shadows altogether.

## References

- [1] O. D. Faugeras, Q.-T. Luong, S.J. Maybank. *Camera self-calibration: theory and experiments*, in Proceedings of the 2nd European Conference on Computer Vision, pages 321-334, Santa Margherita, Ligure, Italy, June 1992.
- [2] R. I. Hartley, R. Gupta, T. Chang. *Stereo from uncalibrated cameras*, in Proceedings of the IEEE conf. on Computer Vision and Pattern Recognition, pages 761-764, Champaign, Illinois, June 1992.
- [3] B. K. P. Horn. *Robot Vision*, Mc Graw Hill, 1986.
- [4] B. K. P. Horn. *Relative orientation revisited*, in Journal of the Optical Society of America A., Vol. 8, Number 10, pages 1630-1638, October 1991.
- [5] A. Huertas, R. Nevatia. *Detecting buildings in aerial images*, Computer Vision, Graphics and Image Processing 41, 131-152 (1988).
- [6] R. Lillestrand, *Techniques for Change Detection*, IEEE Trans. on Computers, Vol. C-21, Number 7, 654-659 (July 1972).
- [7] C. Lin, A. Huertas, R. Nevatia. *Detection of Buildings using Perceptual Grouping and Shadows*, accepted for publication in the Proceedings of the IEEE conf. on Computer Vision and Pattern Recognition, Seattle, June 1994.
- [8] H.C. Longuet Higgins. *A computer algorithm for reconstructing a scene from two projections*, in Nature, Vol. 293, pages 133-135, September 1981.
- [9] G. Médioni, A. Huertas, M. Wilson. *Automatic Registration of Color Separation Films*, Machine Vision and Applications (1991) 4:33-51.
- [10] R. Nevatia, R. Babu. *Linear feature extraction and description*, Computer Vision, Graphics and Image Processing 13, 257-269 (1980).
- [11] M.S. Ulstad. *An Algorithm for Estimating Small Scale Differences Between two Digital Images*, Pattern Recognition, Vol. 5, 323-333 (1973).

x 50 allowing a 25-pixel misregistration offset in any direction. Initial coarse registration using the USGS resection technique was in the order of a few pixels. An example of accumulator array is shown in figure 9. The peak is very sharp and allowing unambiguous determination of the translation. The two ridges in the horizontal and vertical dimension correspond to the dominant orientations of the segments in the model (the buildings are nearly all constructed parallel to each other).



**Figure 9** The accumulator array. On top, the case where the initial coarse registration brings the model to fit the image within a few pixels. The translation found in this case was (2,1) pixels. On the bottom, the result obtained after artificially translating the model segments by (15,15) pixels. The correct translation was found.

## 4.2 Summary of Verification Results

We summarize our validation results in the table below. The percentages are the averages after processing 18 images and show the percentage of validated buildings after the two stages of the algorithm: after the matching process (strong matches) and after the shadow verification process. Objects not visible in the image are easily determined and thus are not included in the figures. The images did not all have the same resolution, but we have grouped them into two groups of “approximate” resolution for convenience in the presentation of the results: 45 cm/pixel (full resolution) and, 90 cm/pixel (one-half resolution). The size of the images at full resolution is typically 1300x1000 pixels.

Note that at high resolution improved matching accuracy yields more strong matches as can be expected. Shadow verification however suffers as inaccuracies in the model are magnified. In some instances, the modeled height of an object is incorrect and thus, the projected shadows we derive fall long or short their actual

locations in the image. This and other factors (the accuracy in registration, the quality and visibility of shadows, and the model itself) should be considered when evaluating the performance of the system..

**Table 1: Percentage of verified buildings for images j1-j8 and k1-k10**

Stage → ↓ Resolution	Strong Matches (%) time <sup>a</sup>	After Shadow verification (%) time
1/2 (~0.9 m/pixel)	47% (t1=7 min.)	80.1% (t2 = t1+5.5s)
1 (~0.45 m/pixel)	58.8% (t1=17.5 min.)	75.9% (t2=t1+6s)

a. t1 is the time taken for edge detection and matching. t2 is t1 plus the time taken for the shadow verification process. This shows that t2-t1 is negligible compared to t1+t2.

Although our results are very encouraging, there is certainly room for improvement, both at the registration level and at the verification level. Inaccuracies in the model should be tolerated to a good extent at this stage, for example. These improvements will also help processing images that lack shadow definition or shadows altogether.

As far as processing time is concerned, we observe from the table that the time taken for shadow verification is negligible compared to t1, which divides equally between the line extraction and the segment matching. We have to note that the times indicated here are given for comparison purposes only, no serious effort was made to optimize our implementation.

A typical result of our validation system is shown in figure 10. The model objects are shown superimposed on the scene (J2 in this example). The objects shown in dashed lines are not visible in the image. The objects shown in bold lines were not validated. All others were validated. Note that non-validated objects correspond to dark buildings with poorly visible shadows.

## 5 Conclusion

We have presented a model-based system capable of verifying the presence in an image of objects described in a 3-D site model. This task, called model validation, is the first step towards the construction of a full change detection system.

We perform the model validation in three steps: first we register the image on the model, then we perform segment matching and build hypothesis to represent the possible presence of each model object in the image, and finally we verify those hypothesis using the shadow

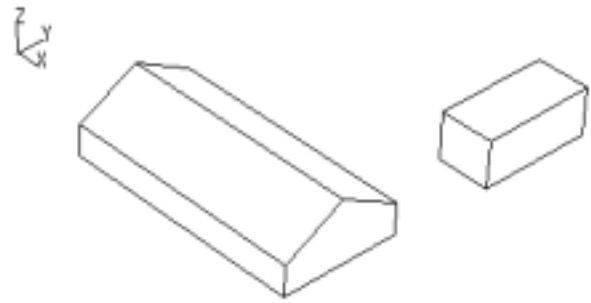
tems. The site model describes geometrically a scene containing several buildings and other objects, constructed at the approximate scale of 1:500. The images are 1320x1035 in size with a ground sampling distance between 18.5 and 32.5 inches. Despite the ‘artificiality’ of this model (due to the indoor setting, no-clouds lighting, the use of small models for the buildings), there is sufficient added noise to consider the data set fairly realistic and adequate. A typical image is shown in figure 6. The features in these images consist of buildings (of



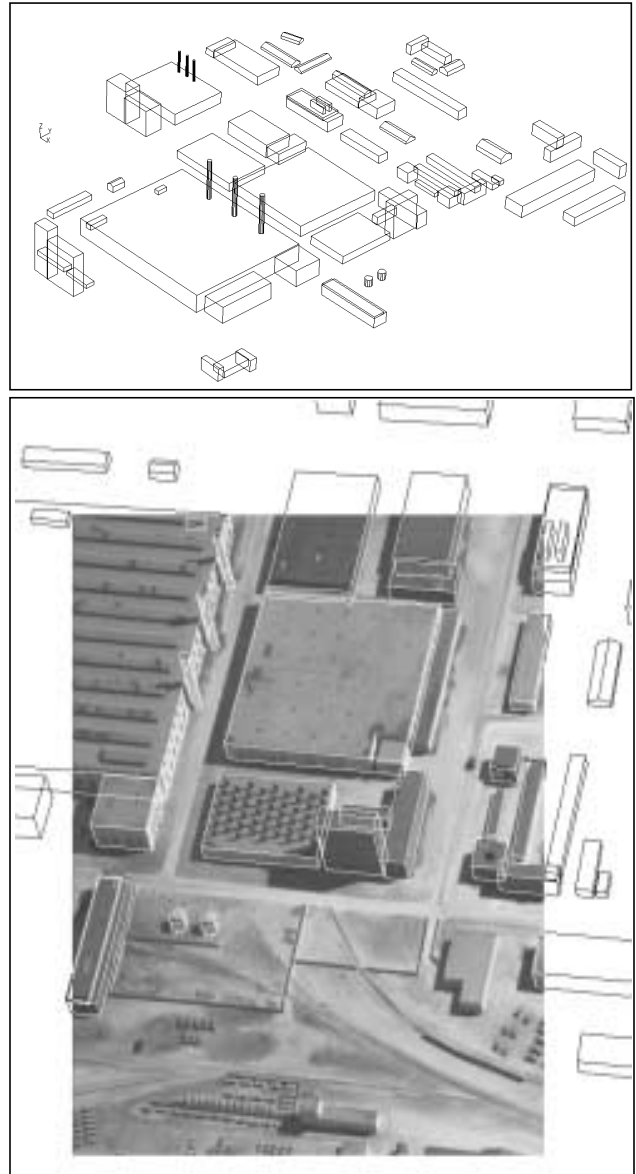
**Figure 6 Model Board image k10.**

‘cubic’ shape), houses (which are simple buildings with a gable) and storage tanks and chimneys (cylindrical shapes). The model contains 60 cubes and houses. We removed six cubes from the model as they were considered too small or inaccurate. The basic shapes are shown in figure 7, a view of the complete site model is shown in figure 8. We have used in our experiments the house and cube shapes, but extension to other similar shapes is straightforward.

We have tested our model validation system using 18 images (RADIUS data set j1-j8 and RADIUS data set k1-k10), at full and at one-half resolution. As a global result, *all* the objects described in the model were present in the set of 18 images. The system software is written in Common Lisp and runs on a Sun Sparcstation 10.



**Figure 7 The two shapes used in our experiments with the Model Board 1 site 3D model: ‘house’ and ‘cube’.**



**Figure 8 View of the model used in our tests (top), the model superimposed on image k4 after coarse registration (bottom)**

#### 4.1 Matching Results

The matching process gives excellent estimates of misregistration. We use an accumulator array of size 50

the registration has been completed. We also assume that the image contains reasonable shadow information.

To verify model objects we use the sun angles to generate the shadows they are supposed to cast. We then match these shadows to the shadow evidence found in the image. More details are given below. For this we make one assumption concerning the site itself. In the absence of a terrain elevation model (DTM) we consider the ground plane to be locally planar, so that the projections of shadows to the ground are simple to compute (see figure 4). In the future we plan to use available DTMs for improved accuracy.

Next we describe the verification process using shadows. It involves labeling image boundaries as potential shadows, and to compare these against shadows generated from model information.

### 3.3.1 Shadow Detection

We label image edges or segments as potential shadow segments by noting the consistency of the “dark” side of the segment with respect to the direction of illumination. Segments oriented parallel to the direction of illumination also correspond to possible shadow lines cast by vertical object edges. Similarly, we detect shadow junctions. The L-junctions formed (allowing for gaps) by potential shadow lines are labeled potential shadow junctions. For more details on the shadow labeling of segments and junctions see [5] and [7].

### 3.3.2 Validation of hypotheses

The 3-D position of the objects having weak or no matches, is known from the model. This information is used to predict their position in the image. Thus, the process consists of calculating the shadows cast in 3-D space and predict their location in the image. Then we search around the predicted locations for evidence of shadows among the potential shadows extracted from the image. If we find a sufficient evidence of shadows, then the presence of the building is confirmed.

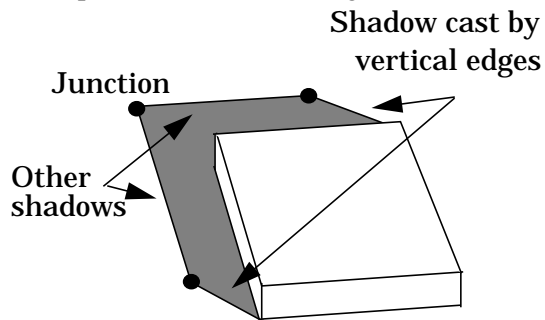


Figure 4 Typical shadows cast by an isolated “cubic” building. There are three junctions, two shadows cast by vertical segments and two other

Some objects may not be sufficiently isolated and its shadows may not be cast on the ground. Also, shadows may be cast on other objects and sometimes will

not even be visible from the viewpoint. In figure 5, for example, the shadow 1-3 cast by building C is not visible (occluded by building B) and the shadow 1-2 is projected on building A, not on the ground. We cannot take this into account until we have verified building B or C.

In our system we compare the evidence of shadows found with the visible shadow evidence. Visible shadows are determined by the knowledge of which objects have already been verified. In the “counting” (or accumulation) of shadow evidence, we also give greater importance to shadows cast by vertical edges and to shadow junctions that appear to correspond geometrically to shadow casting object structures. We consider this very strong and reliable evidence.

Not all weak matches are verified in a single pass due to poor contrast, occlusion, self-occlusion, lack of a DTM, and errors in the model itself. We however use the knowledge of previously verified buildings. This knowledge also allows us to predict more accurately the position of shadows on a subsequent pass. We repeat the process until no further verification of weakly matched objects is possible.

Note that for model buildings that have *no* matched segments, we are still able to predict their position in the image, by relation to the position of already confirmed buildings. We proceed in the same manner to verify these. In this case however the absence of shadows confirms the absence of the building. On the other hand, the presence of shadows does not guarantee the presence of a building. In this case attention is focused at this location for further study, including the application of object detectors, such as the one described in [7], to validate the presence of an object. This is a topic of current study in our group, as part of our work on change detection.

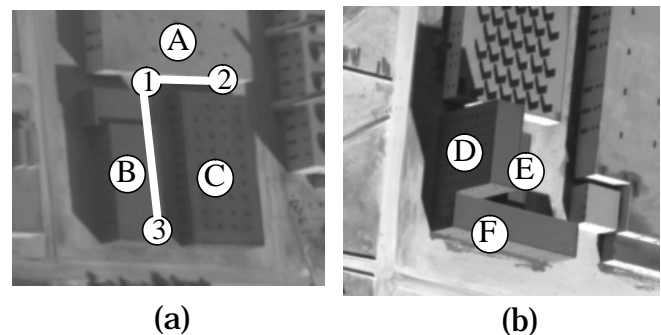


Figure 5 Example of a situation where building verification is made difficult, for buildings C and E, because of the presence of surrounding buildings.

## 4 Experimental Results

The validation system has been tested with a 3-D site model and images denoted as the Model Board 1 data set, provided to us by the RADIUS program. RADIUS is a U.S. government sponsored program for research and development of image understanding sys-

ing pair (model feature, image feature) denotes a hypothesis that the object the model feature is part of, has a corresponding object in the image. For this purpose we use the algorithm described in [9]. The algorithm assumes that there is only a translation between the two sets of features to match. We now briefly describe the matching process.

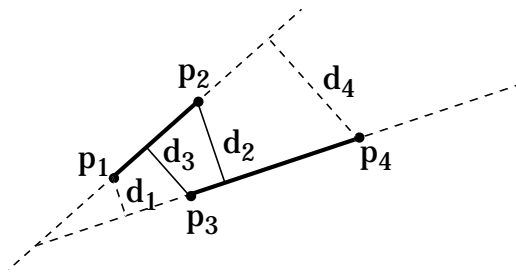
### 3.2.1 Input

We have chosen to match two sets of line segments. The first set consists of line segments extracted from the image using our LINEAR feature extraction system [10]. The second set is constructed from the 3-D wire-frame model (edges of buildings and roofs) projected on the image, and consists of the visible segments only. Here, “visible” means segments of a model building that are not occluded by that same building (i.e. only self-occlusion is considered). Also, we consider segments longer than a certain length (10 pixels in our current system).

### 3.2.2 The Matching Algorithm

We have a *candidate* pair of matching segments (one from the image set and one from the model set) when the two segments overlap, that is, a segment ends project inside the other segment (see figure 2). The segments also must lie within a certain “distance” of each other. The calculation of “distance” is as follows: If the two segments intersect, then the distance is zero. Otherwise the distance is the smallest projected distance of each segment end on the other segment, if it falls inside that segment.

Each pair of candidate segments produces a *vote* as a function of the distance between the two segments and the differences of segment orientation and length.



**Figure 2** Distance between two segments  $p_1p_2$  and  $p_3p_4$ . Each segment end is projected onto the other segment. The distance is the minimum of the projection distances, if the projected point is inside the segment. In this case  $p_1$  and  $p_4$  project outside the other segment, therefore  $d(p_1p_2, p_3p_4) = \min(d_2, d_3)$ .

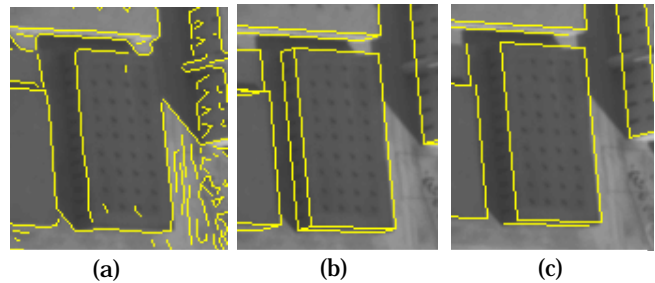
We compute votes for candidate pairs and add their contribution into an accumulator array. The array axes denote translation. For noise effect reduction, the vote is cast not only at the point corresponding to the translation between the two segments (the translation between

the two segment centers), but in a rectangular region around these points. For details see [9].

At the end of the voting process, a peak detector in the accumulator array gives the position of the best translation between the two sets of segments. Knowing the position of the peak, a second pass of the algorithm is used to collect the matching pairs that contribute to the peak in the accumulator array. As a result of applying this two-pass process, we know for each segment of the model whether a corresponding segment was found in the image.

The next step in the validation process involves complete model objects. For each building object in the model, we have a record of the matching of its component edges. We denote a *strong* match when at least four of the possible nine visible segments in its model have a corresponding match among the segments from the image. Otherwise the match is denoted a *weak* match.

Objects having a strong match are considered validated (see figure 3 for an example.) Model buildings having weak matches, or no matches require verification before they are considered validated. Our verification technique uses shadows, detected and processed using the techniques described in [7]. The next section gives details of our method.



**Figure 3** Example of segment matching on building 36. (a) Image segments, (b) segments from the projected model, and (c) segments of the projected model that

### 3.3 Verification of Objects

Objects having weak matches or no matches require additional verification in an attempt to confirm their presence in the image. Weak matches can be the result of several conditions, such as inaccurate registration, poor contrast or occlusion by other objects. Verification can be achieved by means of 3-D clues from stereo or from evidence of the shadows cast. Our current work processes monocular images and thus, uses shadow clues for verification.

We assume that the sun angles (direction of illumination and incidence angle) are known *a-priori*. Otherwise they can be computed from the time of the day at which the image was taken, and the latitude and longitude coordinates of the site. If the time is unknown, the direction of illumination could easily be at least approximated by a trained operator from the image itself, after

for any of the objects. *Model validation* refers to the task of confirming the presence of model objects in the image. Specifically, we apply our technique to validating the *buildings* in the model. Extension to most objects of similar structure and geometry is straightforward.

Model validation requires processing three steps:

- *Coarse registration* of the model and the image. This is equivalent to finding the correct position and orientation of the camera at the time the image was taken.
- *Matching* model to image. Once the viewpoint is known, we project model features onto image coordinates, and use them for matching to equivalent features extracted from the image. We use the edges, or segments, of the wire-frame model representation of the model to match against line segments extracted from the image. Matching features allow us to form hypotheses that represent the presence –or the absence– of a model feature in the image.
- *Validation* of hypotheses. We attempt to verify the hypotheses made with the help of the shadow information extracted from the image.

In the following sections we describe in more detail each of the three steps of our model validation algorithm.

### 3.1 Registration

The orientation and position (the external parameters) of the camera are usually known approximately for a given image. Otherwise, if the intrinsic parameters of the camera –focal length and position of the principal point (the perpendicular projection of the focal point onto the image plane)– are known, then it is possible to derive this information. This problem is known as *relative* orientation for the case of image-to-image registration, and *exterior* orientation for the image-to-model registration case.

#### 3.1.1 Relative Orientation between two images

Most algorithms proposed for this problem in the literature, are based on the use of a set of conjugate points (the matched control points) given by the user. In theory, given three parameters for rotation, three for translation, and allowing for an overall scale factor (which cannot be determined from two projected images), there are five constraints. The knowledge of a pair of conjugate points gives three equations (denoting the transformation of the three coordinates) but brings up two additional unknowns (the depth, in both coordinate systems). Therefore, only five points are needed to compute the transformation (see [3]). In this case however, the equations to be solved are non-linear.

These methods are usually computationally expen-

sive and in general, a unique solution is not possible when the minimal number of points are given. An example of a powerful non-linear solution can be found in [4].

It is possible, however, to solve the problem using only linear equations, if more information is given. The linear algorithms (Longuet-Higgins [8], reexamined later by Hartley [2] and Faugeras [1]), need at least 8 points correspondences. The main advantage of linear methods is that they are very fast and guarantee the uniqueness of the solution, except for degenerate cases. They however exhibit sensitive behavior in the presence of real, noisy data.

#### 3.1.2 Exterior Orientation between an Image and a 3-D Model

The exterior orientation problem, which is the one we need to solve here, is just a special case of the relative orientation problem, thus the algorithms mentioned above are applicable with only slight modifications. Instead of giving two sets of 2-D coordinates to describe the conjugate points, we will give for a ‘conjugate point’ its 3-D coordinates in the site model and its 2-D coordinates in the image.

Here we have more information than in the relative orientation problem. There are three rotation parameters and three translation parameters (in this case the overall scale factor can be determined). Each point brings up three equations as before, but only one additional unknown emerges. Therefore it is sufficient to have, in theory, only 3 points to solve the registration problem. However, the equations in this case being non-linear, will result in more than one solution (up to eight); a fourth point is needed to completely solve the problem (see Horn, [3]). Using a larger number of points is desirable to improve accuracy of the solution with a least-squares method. We have found in our experiments that 20 reference points is adequate.

We have used two algorithms, both giving good results. First, we used the linear method of Hartley (see [2]) that we adapted to the exterior orientation problem. The non-linear algorithm we have used is the USGS resection algorithm provided as part of the RADIUS Common Development Environment (RCDE), the environment used for DARPA sponsored RADIUS research work, including our own Aerial Image Analysis research at USC. Both methods give a good coarse approximation of the orientation, with a preference for the non-linear method which gave results within a few pixels (see figure 8 on page 15 for an example of registration on image k4, using the USGS resection algorithm).

#### 3.2 Matching

In order to obtain an accurate registration between the model and the image, we first match features computed from the information in the site model with features extracted from the image. Secondly, each match-

Another approach is to use general classification algorithms after spatial registration to assign each pixel to a limited number of classes, thus transforming the intensity range of a pixel into a reduced number of levels. This assumes that preferably more than one band of image data is available, as in Landsat's multiple band images, for example. Depending on a pixel's position in the  $n$ -dimensional data space, and relying on the statistical description of each class (which is user-defined), a pixel is assigned a class. The classes could represent water, forest, roads, urban areas, and so forth. Once the classification is done for each image, the detection of changes becomes straightforward.

Next, we outline our approach to change detection, and give details of our model validation technique. We also discuss experimental results, applied to building structures, using a site model and associated imagery supplied to us by RADIUS, a government sponsored research program on image understanding.

## 2 Overview of the Change Detection Task

The task of change detection can be broken into two sub-tasks: *Detection* of changes and *Description* of changes. In the detection phase we determine whether a *significant* change has taken place since the last look at the site. In the description phase, we obtain a description of the change. The description may consist of the size and shape of the new (or altered) structure and surface properties. This step is similar to the process of constructing a site model, and, in fact, one result of this step may be an updated site model.

In figure 1 we show a flowchart of the complete change detection system. The process clearly requires a comparison of new imagery with the old imagery (and the models constructed from the old imagery). However, we need to make a distinction between changes in the images and changes in the site. We are only interested in those changes in the image that come from some changes in the site rather than from changes in imaging conditions. The techniques of simple image differencing are inadequate for this task. We follow a four step approach:

- *Registering Site Model to Image:* The first step in change detection is to register the new image(s) to the model(s) contained in the site folder.
- *Model Validation:* After a coarse registration between the image and model has been made, we verify the presence in the image of the model objects. We use feature matching techniques for this step. Model features that are not present in the image represent likely changes. Some missing features are due simply to viewing conditions. Most of these, however, can be predicted and explained from the site model itself. The task of finding objects in the image that

are not in the model is more difficult since the model is no longer as useful in directing the processing. We plan to do this in the future by the next two steps.

- *Focus of Attention:* This will be a collection of techniques that will draw attention to significant structures in the image that are not explained by the existing model. To separate the significant changes from the insignificant ones, a *perceptual grouping* operation that organizes lower level features into higher level structures will be necessary (see [7]). Matching features between multiple images (if available) will help distinguish between structures on the ground and above the ground. In monocular images, an important cue to significant changes will be presence of shadows. Collateral information may also be useful in determining the focus of attention.
- *Detailed Analysis:* In this step, we analyze in detail the structures indicated by the focus of attention processes. This step requires the development of automated or semi-automated site modeling techniques.

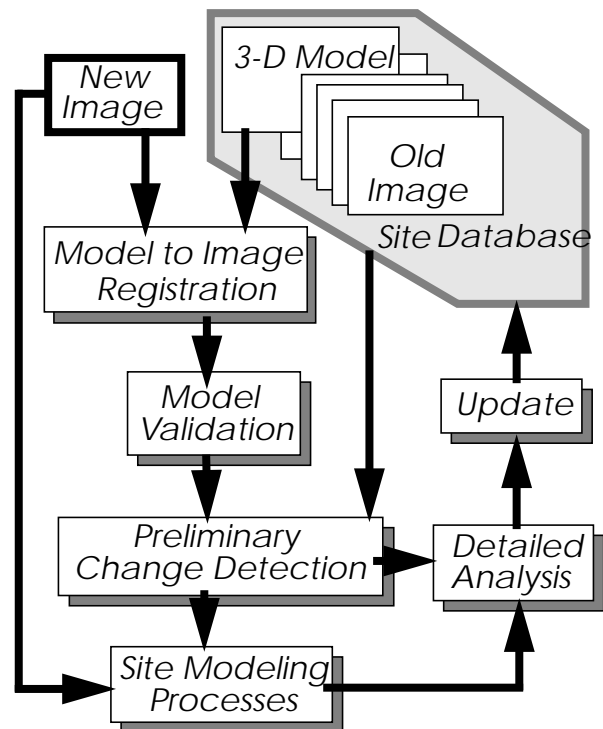


Figure 1 Flowchart of Change Detection System

## 3 Validation of Buildings

We are given a 3-D site model (consisting of a set of objects such as buildings, houses, and storage tanks, describing a site) and an image of this site, taken at a later time. The site model consists of geometric information only: we do not have any data about albedo or color

# Model Validation for Change Detection\*

M. Bejanin, A. Huertas, G. Medioni and R. Nevatia

Institute for Robotics and Intelligent Systems

Powell Hall, Room 204

University of Southern California

Los Angeles, CA 90089-0273

## Abstract

An important application of machine vision is to provide a means to monitor a scene over a period of time and report changes in the content of the scene. We have developed a validation mechanism that implements the first step towards a system for detecting changes in images of aerial scenes. By validation we mean the confirmation of the presence of model objects in the image. Our system uses a 3-D site model of the scene as a basis for model validation, and eventually for detecting changes and to update the site model. The scenario for our present validation system consists of adding a new image to a database associated with the site. The validation process is implemented in three steps: registration of the image to the model, or equivalently, determination of the position and orientation of the camera; matching of model features to image features; and validation of the objects in the model. Our system processes the new image monocularly and uses shadows as 3-D clues to help validate the model. The system has been tested using a hand-generated site model and several images of a 500:1 scale model of the site, acquired from several viewpoints.

## 1 Introduction

Change detection is an important task in the process of photo-interpretation. The task of change detection consists of finding significant differences between the new data and the model derived from the older data. The significance of the differences may be task specific though in most cases man-made changes are more important than changes caused by natural factors such as seasonal changes. We are only interested in those changes in the image that come from some changes in the site rather than from changes in imaging conditions.

Change detection is a tedious task as it requires careful comparison of images (and their models) taken at different times under possibly vastly varying conditions. We believe that even partial automation of this task will greatly increase human productivity and possibly also enhance the reliability of the results.

The task of model validation in the context of change detection, involves comparing a *new* image of a site (or a collection of images) to the information associated with that site. These information may consist of a site model and one or more previous images and results of previous analyses on these images. We assume that in all cases, a site model of suitable resolution and complexity is available.

In this paper we address the problem of *model validation*, the process of confirming the presence of model objects in images. This task requires that we first register the new image to the site model and second that we validate the model objects. In our work we have chosen to work with features extracted from model information and from the new image, rather than developing pixel based techniques.

Previous work on change detection relies on image differencing, which is unable to separate the effects of changes resulting from different viewing conditions (such as different viewing positions, different illumination and seasonal changes) from important structural changes.

One example of previous work is by Lillestrand and Ulstad [6] [11]. Given a reference image and a new image, the reference image is first corrected for any spatial distortion so that it is registered with the reference. This is done at each point by finding the best conjugate point on a simple correlation-based criterion. Then global intensity corrections are made so that the first two moments of the image match, allowing for any differences in luminosity or film sensitivity. At this point a simple subtraction of the two images is made to reveal small scale changes between the two views. This algorithm works very well for this particular problem of detecting changes between two images.

---

\* This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by the U.S. Army Topographic Engineering Center under Contract No. DACA76-93-C-0014. Mathias Bejanin is a Visiting Researcher at the University of Southern California, supported in part by Matra Cap Systèmes, Velizy-Villacoublay, France.