Application of a User-Programmable Vision System to Inspection of Complex Assemblies

Peter W Woods[†] **and Christopher J Taylor**
Wolfson Image Analysis Unit
Department of Medical Biophysics
University of Manchester
Oxford Road
MANCHESTER M13 9PT

This paper describes how a framework for user-programmable image interpretation has been implemented as a computer vision system capable of executing a declarative task description. The system has been applied to an industrial inspection task, and its performance has been compared with that of an existing procedural program written specifically for the same task. These show that the system is capable of achieving a similar level of performance but can adapt its analysis strategy more flexibly.

Ia a previous paper [1], we described a framework for representing and executing visual tasks. The system has now been implemented more fully and applied to a real industrial inspection task. The most significant development reported here involves an application independent control strategy which attempts to satisfy explicitly stated, task-specific goals by reasoning about the collection and interpretation of evidence.

The overall objective of the project is to develop an application generator for complex but specific visual tasks, based on definitions of the task goals and expected image appearance rather than prescriptions of how the tasks are to be performed. The potential advantages of this approach are that: the end-user can develop and modify applications directly because expert knowledge of machine vision is not required; the applications are robust because the system can make systematic use of task knowledge and reason about the extent to which goals have been satisfied; the effort involved in developing new applications is minimised by separating out the task-specific aspects of the problem from the generic.

The remainder of this paper describes, briefly, the process of defining a task, gives an overview of the execution control strategy and describes its implementation, demonstrates the application of the system to the problem of inspecting vehicle brake assemblies and compares the results with those obtained by a less systematic but exhaustively tested solution to the same problem [2]. The principle issue addressed is the practicality of generating robust applications for complex visual tasks without specialist knowledge of machine vision. The nature of the information which the user must provide is critical. The ergonomics of the user interface will also be important in a final system, but this is not considered here.

DEFINING VISUAL TASKS

The new work described here builds on an existing framework for representing and executing visual tasks which is described elsewhere [1]. What follows is a brief overview.

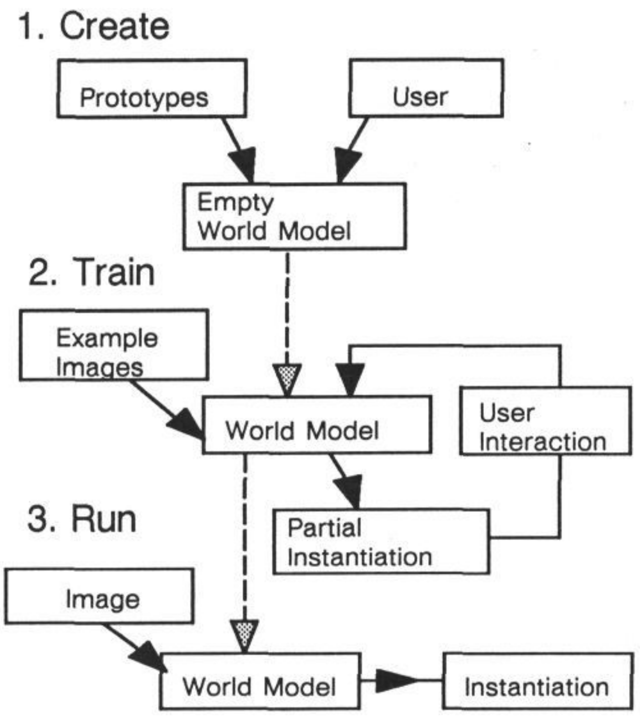

Figure 1. Stages in the development of a world model

The stages in the creation and execution of a task description are shown schematically in figure 1. The user is assumed able to specify the task in an appropriate way but not to know how it is to be executed. The specification takes the form of an inter-related set of sub-models (model elements), arranged in a structure (the world model) which both defines the task and models image appearance. Task goals are embodied as explicitly labelled goal elements. Each model element is defined by the user to be a specialisation of some system prototype, chosen

[†]Current address: The Technology Partnership, Melbourn Science Park, Cambridge Road, Melbourn, Royston, Herts. SG8 6EE.

This work is supported by an SERC grant and has been carried out as part of Alvey project MMI-093: 'Techniques for User-Programmable Image Processing'.

from a documented library. A repertoire of representational forms and image analysis methods are associated with prototypes as submodels, and are inherited by model elements. The world model specification only becomes useful after an interactive training session which uses example images to define free parameters in the model. Training also allows a utility value to be assigned to model elements and submodels to indicate their usefulness in the particular task.

A task is executed by attempting to instantiate the goal elements, which generally leads to instantiation of the world model as a whole. Control of instantiation is achieved through a production rule system with an application-independent rule-base. Most rules are predicated on the state of model elements and their inter-relationships. Rule firings can alter the state of the world model, invoke procedural activity such as image processing, or both.

The system currently comprises a prototype library catering for scenes containing rigid objects of axially symmetric, circular, or polyline shape forms. The control system deals with the application of geometric constraints, cue generation, and the verification of hypothetical instantiations of the model.

CONTROL STRATEGY

The instantiation of an object from an image requires three steps: the application of constraints, cue generation, and verification. In this paper only geometric constraints and constraints on the expected number of instances of an object, or cardinality, are considered. The latter are further assumed only to take on the values 0 or 1.

The strategy for the exploitation of geometric relationships in object location is first to use relations which apply directly to the image itself or some object of known position, then to consider relations which involve one or more levels of indirection. Relations labelled as fixed are applied first in each case. If the object position becomes established to better than a certain precision this process will terminate, otherwise all possible relations with a greater than a minimum utility will be exploited. Whenever an object becomes located, the constraints on related unlocated objects are updated.

If geometric constraints are sufficiently precise, the prediction of the object position is itself an object cue. If they are less constraining, it is sometimes possible (for classes of object for which orientation is not important) to generate a line segment which can be expected to cross the object, and to obtain a positional cue by searching along this line and matching to a grey-level profile model. If neither of these cases applies, cues are generated by some operator applied over the region of interest defined by the geometric constraints.

A cue generator can yield a number of cues which may not be in one-to-one correspondence with actual objects, due to confusing features and other causes. The way in which the system responds to a cue depends on the confidences associated with all the cues, the expected number of instances, and the task goals. If the favoured cue generation method yields no cues, the system will tentatively assume the object is not present. However, any evidence to the contrary may cause the system to try to resolve this by applying an alternative method. At present, the system can only operate properly in cases where there is only ever expected to be at most one instance of each object in each image. In this case the cue with highest confidence is assumed to arise from the object and the others are initially ignored. Failure to verify, or external evidence that the interpretation is inconsistent, might cause verification to be repeated with an alternative interpretation of the cues.

Verification is the process of determining if a cue actually corresponds to an occurrence of the sought object, and if so instantiating the relevant model element. Model elements, and more particularly their submodels, are each attributed with a confidence value when they become instantiated. This is a probability estimate derived from the match between measured parameters of the instance and the corresponding model. The nature of these values is independent of both the task and the methods which generate them, and as such they are the primary parameters used to determine control.

It would be optimistic to expect that in every case a cue will provide adequate information to allow full instantiation of the object to follow directly. In some cases cues must be refined in one or more stages to provide a suitable input for an instantiation method. Similarly, verification might involve several steps if more than one submodel needs to be instantiated.

IMPLEMENTATION

The Rule Base

The strategy outlined above is implemented as a production rule system. The control rules have been designed to be independent of specific model types, allowing the system to be extended by the addition of new prototypes. Some information about specific methods, prototypes, or user-defined model elements is also best represented in the form of rules. This is only useful if such rules can complement the existing rule base without the control strategy needing to become either ad hoc, or unstructured and therefore potentially inefficient. Such rules cannot therefore directly cause actions but instead assert priorities for possible actions. Several such assertions, or none, might apply in a particular circumstance such as in attempting to choose the best cue generation method. If no special conditions apply, a low priority rule in the main rule-base will fire to select the method of greatest utility. If other rules apply, these will fire first to cause a particular method to be assigned higher utility and so take precedence.

In the current implementation all parameters of all submodels are modelled as independent variables by first order statistics of the values encountered in training. Confidence values are calculated using Bayes' rule to combine matches from the different parameters. This assumes, purely for simplicity, that parameter distributions can be adequately represented by normal distributions although this is not always true. In order to generate more reliable confidence factors it will be necessary to use a more general model of parameter variability.

Applying Geometric Constraints

Geometrical relationships can be defined between any two objects and are limited to the image plane. Each object position is represented by a system-defined vector which can be considered as the axis of an object-centred local coordinate system. In order that relationships can be established with respect to the image (or between images) each image also has an associated vector. All distances are in user-defined world units which can be related to pixel units by a calibration process. Each shape representation for an object has a well defined relationship with the object vector. The length of each object vector is related to the size of the object, to optionally allow geometric relationships to be scaled so that distances are relative to the object size. The problem of defining a unique vector for an object with several symmetry axes is not addressed here. However, where the vector orientation is ambiguous or undefined this is taken into account when defining associated relationships. In particular this means that a relationship between two objects might not be as constraining as its inverse.

Geometric relationships are defined by the relative position of two vectors in terms of the distance between their origins, the angle between the first vector and the line joining the origins, and the relative orientation between the two vectors. The user only specifies the special cases of the relationship being nominally fixed, or scaled. The three parameters are modelled independently from statistics on their values acquired during training. Initially, the positions of objects are indicated by the user, by reference to an iconic display overlayed on each training image. Subsequently, wherever possible, the model is updated using the positions of objects located by the system itself. Consistent feedback between the user and the system regarding the estimated and actual location of an object is achieved through an iconic display. This can be arbitrarily defined by the user, but defaults to the object outline as defined by one of the shape models for the object.

The chief use of geometric relationships is to define constraints on the position of an object vector, in terms of an area containing the possible locations of the vector origin together with a range of allowed orientations. This is based on the extremes and standard deviation of values observed during training. Constraints can also be propagated through more than one relationship and their effects can be combined. In order to transform the range of possible vector positions into the minimum area containing the entire object (which is usually what is required for the application of a cue generator), it is necessary to convolve the former with the shape of the object. This is approximated by a simpler calculation in which the minimum enclosing rectangle for the object is combined with the vector constraints to produce a region of interest bounded by an image-aligned rectangle. Figure 2 shows how the area of interest for object B is derived from its vector constraint which has been calculated from a range of possible positions for the related object A which is in turn related to the image.

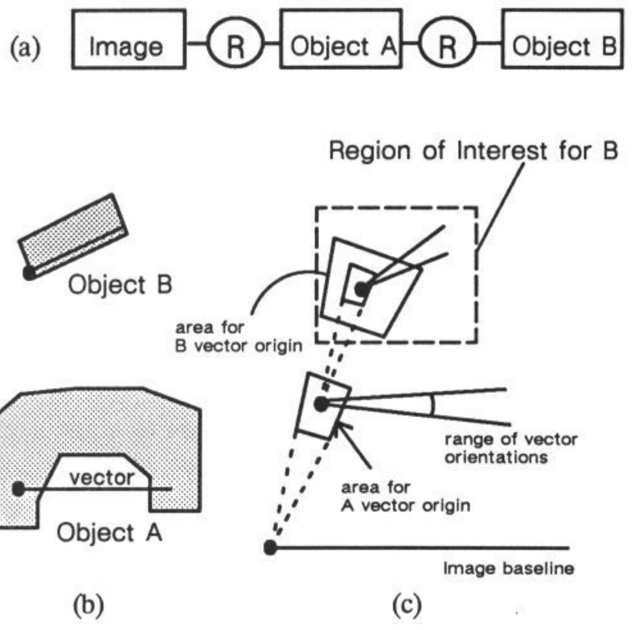

Figure 2. (a) shows a fragment of world model indicating relationships between objects A and B which appear as in (b). The propagation of constraints is shown in (c).

Cue Generation

At present the system contains centre of gravity [3], threshold, and edge detection cue generators. The appropriateness of these operators are learned during training as follows.

Cue generation methods are ranked in a prior order of utility based on their suitability for the object class and their computational expense. This means for example that the system never normally attempts to use axial cues for circular objects, and that simple thresholding is usually applied first. During training, the user is able to indicate that a displayed result is unsuccessful and consequently the utility of the method for that object is reduced. Conversely, a consistently successful method will attain high utility. When attempting to locate and identify a particular object, the system might try several methods in succession, until the user confirms that one has given a correct or at least an acceptable result.

Each cue method has a model whose form is independent of the associated object. The purpose of these models is to define any parameters required by the cue operator, and to model typical properties of the cues to allow a confidence value to be given to each cue. This value is important in aiding control but can allow grossly abnormal cues to be rejected immediately. The model is a set of parameters likely to aid discrimination of true cues from false ones. For example, the centre-of-gravity operator gives false cues corresponding to external object symmetries. These can usually be easily separated due to the different grey-level ranges in the vicinity of an external cue and a true object cue. This grey-level value is therefore one of the properties of the associated model. It cannot be guaranteed that in all cases these parameters will be useful, but the intention is to provide a small fixed number which will be useful in the majority of cases.

All cues are of the same form: a line segment together with an arc or region description and a confidence value. For example, a threshold cue consists of the thresholded region and the associated line is the maximum chord of this region. Methods exist to transform this standard form into whatever is required as the input to each shape instantiation method, but the transformation is usually trivial, involving a relationship between the line segment and the object axis.

Verification

Verification usually involves an attempt to instantiate a shape model for the object given a cue, and using both shape and grey-level information [4]. Verification can, however, in some cases be achieved by instantiating other properties of the object. The system contains circular shape models (for circular objects), axially symmetric shape and axis profile models (for axially symmetric objects), and a polyline shape model (potentially for all rigid objects). Many of the associated image analysis methods are based on the systematic use of grey-level profile models associated with line segments. These are used principally to model object boundaries in order to facilitate boundary detection leading to shape instantiation, and in other ways, such as to represent axial profiles. The choice of model to be used for verification is determined by utility as for cue models and also by an importance value. Importance effectively takes on one of two values, and is set up as a predetermined property of each submodel. Cue methods all have low importance which means that the existence of a specific type of cue is neither a sufficient nor necessary condition for object instantiation. Conversely, most shape models have high importance which means their instantiation is sufficient to instantiate the object. Model elements which are essential to task goals are automatically labelled with high importance prior to training.

A further way of inferring the existence of an object is through instantiation of its sub-parts. For example, an unimportant might be related geometrically to other components which are important, and so its position could be determined as a means to locating these objects. The existence and position of the component can be inferred from those of any modelled sub-parts it may possess, without its boundary needing to be verified.

Confidence values represent the intrinsic probability that detected image features really correspond to an instance of the model element. Typically, a definite decision has to be made as to whether an object is present or not. This decision does not necessarily correspond to the most probable interpretation based on the confidence factor, but depends also on the nature of the task. For example the system should treat the case of an object being instantiated with confidence 0.5 differently if the aim is to detect all occurrences of the object with as low a false negative rate as possible (for instance if the object corresponds to a rare but important defect), than if the task is to ensure that all cases of the object being missing are detected. The user must therefore supply this information, by specifying target error rates for the relevant goal elements. For this reason the model element for the object is not itself the goal frame, but is related to an 'is-present' goal frame which is dependent on the confidence factor of the object, its expected cardinality, and the task specific conditions.

THE BRAKE ASSEMBLY APPLICATION

The system has been applied to the problem of detecting specific faults in a car rear drum-brake assembly. A typical view of an assembly is shown in figure 3. Each of two brake shoes is imaged separately but we consider only one of these views. The goals of the inspection task are : to determine whether the assembly is right-handed or left-handed (distinguished principally by the presence of springs but no lever or *vice versa*); if left-handed, to verify that two springs are present; and in either case to verify that a retaining cap is present and that the brake lining thickness is within a specified tolerance limit.

Figure 4 is a schematic of the relevant components as they appear in the camera views. Figure 5 shows the resulting world model.

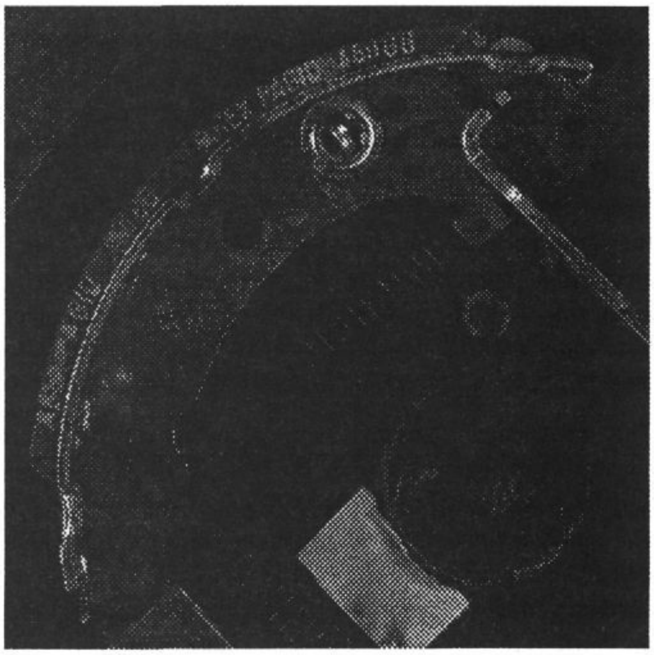

Figure 3. Typical image used for analysis.

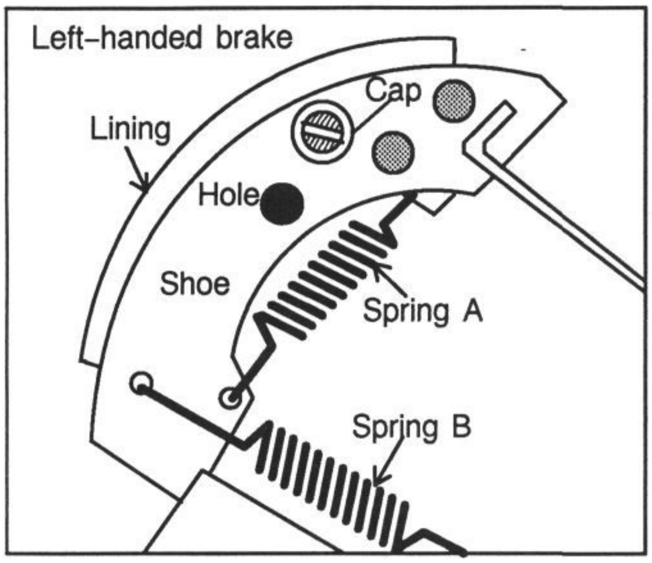

Figure 4. Schematic sketch of important components of a brake image.

RESULTS

The principal result is that the system is able to successfully use the world model description to produce an execution strategy which is efficient and which compares well with an existing hand-crafted solution to the same problem.

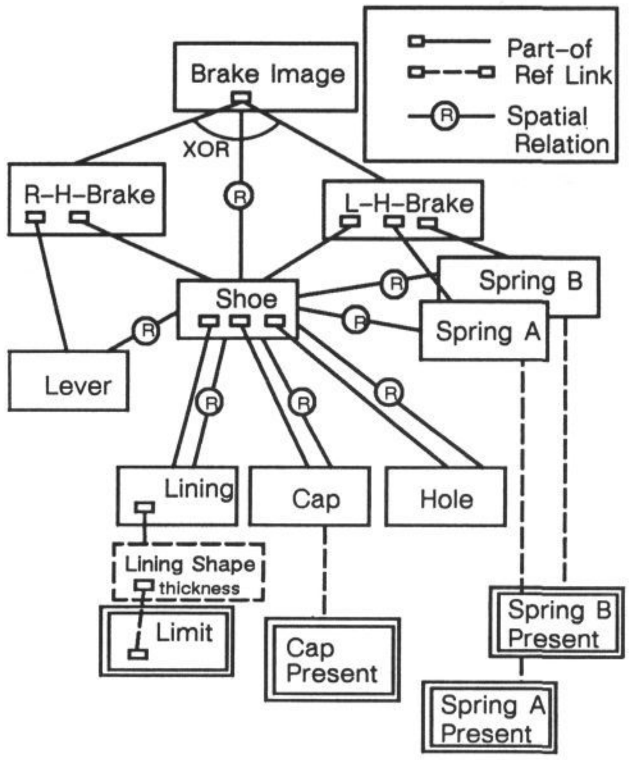

Figure 5. The World Model for the Brake Inspection Task. Large boxes are model elements. Goal elements have double borders; system defined sub-models are not shown except for the lining shape model (dashed).

The current system has been tested using a small sample of assemblies viewed using an identical imaging geometry to the existing system. The performance of the existing program is well characterized from several years of routine use and each test has an error rate in the range 0.1 – 0.3 %. No equivalent large-sample figures are available for the current system. Since the methods for object verification are similar in many cases in the two versions it can be assumed that the performance of the current system will be at least as good in these cases and results on small samples confirm this.

The current system is able to reproduce non-obvious behaviour of the existing system. For example, the springs (axially symmetric objects) are visible not because of a clear boundary but from the pattern of highlights from the coils along their length. Verification is in this case carried out with respect to an axis profile model. The system learns this by discovering that this is the only important submodel which can be instantiated with consistent success, ie which also has high utility.

As an example of the difference in behaviour of the two systems, consider the location of the brake shoe. The programmed system locates the shoe by finding three points on the circumference of a circular hole within it, and one special point on the shoe boundary. Measurements on other objects are made with direct reference to the located shoe position. These are subject to failure if for example one circumference point is incorrect, leading to a large error in position which is not immediately detected. If location of the shoe fails the entire analysis has to be aborted. These shortcomings are acceptable in this application because the error rate in finding the points is very low (less than 0.1 %), however in general the inflexibility of this approach may lead to unacceptably poor reliability. The current system is capable of learning that the shoe itself cannot be directly located and so can mimic the programmed method. It usually uses the cap and the hole to infer the presence and location of the shoe. Figure 6 shows the shoe icon and the initial position range. Figure 7 shows how this is updated after the cap has been instantiated. High precision in location is unnecessary since the shoe position is only used as a guide to locating other objects by enabling cues from geometry to be generated. Both versions use similar methods to locate the hole but the current system should be even more robust since it detects more boundary points and a confidence factor is used to check that the location has been successful. If this is not the case, alternative strategies will come into force which can enable the interpretation to succeed by for instance locating the shoe using other features, or by instantiating related objects by other means.

The system currently uses 150 rules in total, comprising about 110 for the main rule-base and the rest for handling geometric relations and representing generic knowledge. The world model compiles into about 1250 application specific facts in addition to 1000 facts representing the prototype hierarchy. A typical analysis involves about 200 rule firings.

The speed of the existing system is just under 2 seconds per image of which control is a negligible proportion, compared to a total of about three minutes for the current implementation. The bulk of the extra overhead in the

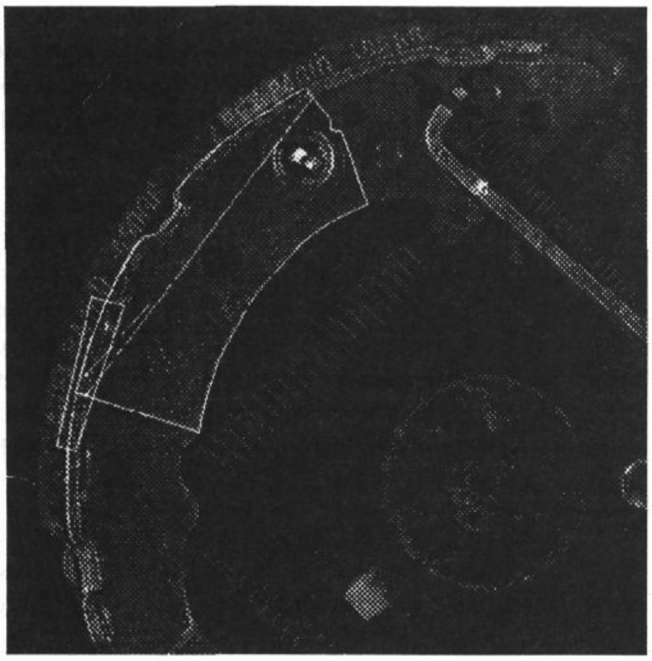

Figure 6. Shoe icon and uncertainty in shoe position with respect to the image.

current implementation arises from the production rule system which is implemented in Inference ART. There is a small additional overhead due to an inter-process communication mechanism which links the production rule system and the image processing subsystem.

CONCLUSIONS

It has been demonstrated that the system can successfully employ a declarative task description in order to execute significantly complex visual tasks with performance comparable to that of a conventionally programmed approach. The control mechanism allows the analysis to be more robust, although it imposes an overhead on the execution efficiency. In the task involved, interpretations of objects are independent and so the control strategy does not need to support multiple interpretations.

The principal disadvantage of this system is the speed of operation. However it is important to distinguish between inefficiency due to implementation and that arising from an inherently inefficient control strategy. The ART implementation is ideal for experimental development but is not suitable for for a real inspection system. For tasks like the example considered here, it would not be difficult to implement a simple but specialized production rule system which could reproduce the behaviour of the current system, but be far more efficient. This would take advantage of the small number of types of relationship used in the representation, and use fixed forms of representation where possible. This might allow a rule-based system to achieve a practical level of execution efficiency.

Assuming the world model structure is sound, and that appropriate prototype methods are available, the performance of any application using this system will be determined by the validity of the statistical models created during training. One major shortcoming of the current implementation is the simplicity of the model of parameter variability, compounded by the fact that both model parameters and estimates of matching performance are set up simultaneously during training. This means that an inadequately trained system will operate with unknown unreliability and may only give poor performance. More complex statistical modelling may well require a larger set of training examples to become adequately defined. Any calculation of confidences should take into account the uncertainty in the model data.

It is intended to continue this work to apply the system to the task of chromosome analysis, where the scenes and the model are less well constrained [5]. This will involve extending the control strategy to deal with multiple interpretations, and expanding the range of prototypes to include non-rigid object representations.

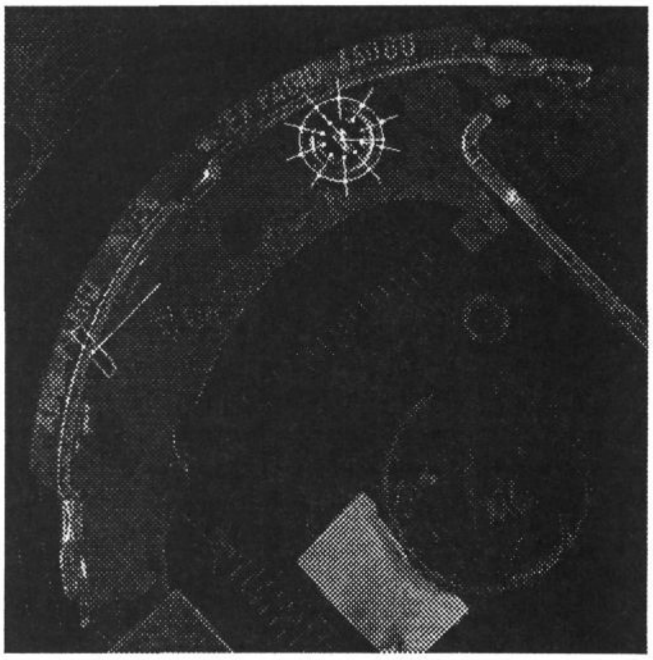

Figure 7. Instantiated cap and uncertainty in shoe origin.

REFERENCES

1. **Woods P W, Pycock D, and Taylor C J** 'A framework for modelling and executing visual tasks' *Image and Vision Comput.* Vol 7 No. 2 (1989) pp 102.
2. **Woods P W, Taylor C J, Cooper D H and Dixon R N** 'The use of geometric and grey-level models for industrial inspection'. *Patt. Rec. Letters* Vol. 5 (1987) pp 11.
3. **Thornham A, Cooper D H and Taylor C J** 'Object cues for model-based image interpretation' *Proceedings 4th Alvey vision conference, Manchester, September 1988.*
4. **Cooper D C, Bryson N and Taylor C J** 'An object location strategy using shape and grey-level models' *Image and Vision Comput.* Vol 7 No. 1 (1989) pp 50 - 55.
5. **Pycock D** 'Chromosome classification in a general purpose frame-based interpretation system' *Proceedings 5th Alvey vision conference, Reading, September 1989.*