VERIFICATION VISION FOR PROGRAMMABLE ASSEMBLY

Robert C. Bolles*
Stanford University
Stanford, Ca. 94305

Abstract

This paper is a condensed version of the author's thesis [Bolles 1976], which investigates a subclass of visual information processing referred to as *verification vision* (abbreviated VV). VV uses a model of a scene to locate objects of interest in a picture of the scene. The characteristics that distinguish VV from the other types of visual information processing are: (1) the system has a great deal of prior knowledge about the type, placement, and appearance of the objects that form the scene and (2) the goal is to verify and refine the location of one or more objects in the scene. VV includes a significant portion of the visual feedback tasks required within programmable assembly. For example, locating a screw hole and determining the relative displacement between a screw and the screw hole are both VV tasks. Two types of VV tasks are discussed in the thesis: *inspection* and *location*. This paper only discusses location tasks, but essentially the same capabilities are required for both types of tasks.

This paper describes (1) a structure for a VV system that makes it easier for programmers who are not experts in computer vision to program VV feedback, and (2) a set of combination rules that are capable of using the results of several different types of operators to estimate the confidences and precisions that are necessary within VV.

An interactive VV system based upon these ideas has been implemented. It helps the programmer select potentially useful operator/feature pairs, provides a training session to gather statistics on the behavior of the operators, automatically ranks the operator/feature pairs according to their expected contributions, and performs the desired task. The VV system has also been interfaced to the AL control system for the mechanical arms and has been tested on tasks that involve a combination of touch, force, and visual feedback.

Introduction

Verification vision is a type of visual information processing that uses a model of a scene to locate objects of interest in a picture of the scene. The characteristics that distinguish verification vision (abbreviated VV) from the other types of visual information processing are: (1) the system has a great deal of prior knowledge about the type, placement, and appearance of the objects that form the scene and (2) the goal is to verify and refine the location of one or more objects in the scene. The following situation illustrates a typical use of VV:

> During the assembly of a pump, a mechanical arm places the pump base in a vise. The next step is to insert an aligning pin into one of the screw holes in the base. But the location of the screw hole is not known precisely enough to insert the pin directly. The programmable assembly program needs to improve its estimate for the location of the hole. VV is one way to accomplish this subtask.

In this task the pump base may be mispositioned to the extent of perhaps plus or minus half an inch and rotated plus or minus fifteen degrees, but there will not be any big surprises: the base will not be upside down or at the other end of the workstation. VV is designed to use this predictability to minimize the cost of reducing the uncertainties associated with the location of an object. It bridges the gap between the known tolerances on an object and the desired tolerances *when the initial tolerances greatly restrict the possible appearance of the object.*

If the locations and appearances of the objects are greatly restricted, why not use any of several correlation operators to match corresponding features and use the positions of these matches to deduce the location of the screw hole? If the constraints on the objects guarantee unique matches for all of the features, this procedure may be acceptable. However, there are two main reasons why it is difficult to guarantee unique matches:

(1) Constraints, even quite tight constraints, often do not guarantee a single match. For example, if a visual operator is designed to locate a certain corner, often there are other corners that are near the desired corner and look almost identical to it. Sometimes the operator may locate one of these decoys instead of the desired corner. The desired feature and the decoy features will be jointly referred to as *known alternatives* for the operator.

(2) Visual operators, such as correlation operators, are not completely reliable; they do not always locate one of the known alternatives. Low level operators are notorious for occasionally locating something completely unexpected. This type of match will be referred to as a *surprise*.

Thus, almost any interesting class of VV tasks includes situations in which the visual operators do not locate unique features. The VV system has to decide which known alternative or surprise has been matched by each operator.

In VV there are several sources of information that can be used to help make these decisions. For example, the system may have previous pictures of the scene and constraints on the locations of the objects. The previous pictures can be used to predict the range of values produced by an operator and the constraints can be used to determine the portion of the picture in which a feature might appear.

There are also sources of information that can be used to simplify the programming of VV tasks. For example, previous pictures of the scene or models of the objects in the scene can be used to suggest potentially useful features to be located.

This paper describes a VV system that has been designed to take advantage of these diverse sources of knowledge. The second and fourth sections outline a structure for a VV system that makes it easier for programmers who are not experts in computer vision to program VV feedback. The third section presents a set of combination rules that are capable of combining the results of several different types of operators into the confidences and precisions that are required within VV.

* The author is now at Stanford Research Institute, 333 Ravenswood, Menlo Park, Ca. 94025.
This research was supported by the Hertz Foundation and the Advanced Research Projects Agency of the Department of Defense under Contract DAHC 15-73-C-0435. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing the official policies, either expressed or implied, of Stanford University, Hertz Foundation, ARPA, or the U.S. Government.

A VV system based upon these ideas has been implemented and interfaced to the AL control system for mechanical arms [Finkel 1976]. The VV system helps the user select potentially useful operator/feature pairs, provides a training session to gather statistics on the behavior of the operators, automatically ranks the operator/feature pairs, and performs the task. The fifth section briefly describes the results of using this system.

There are a few excellent, special-purpose systems that perform VV tasks (e.g., see [Baird 1976] and [Kashioka 1976]). There are also a few systems that handle a subclass of VV tasks (e.g., see [Fischler 1971], [Agin 1973], [Chien 1975] and [Holland 1975]), but none of these programs deal with the wide variety of available information or concentrate on the combination of precision and confidence to the extent required by the general class of VV tasks.

The motivation for this research came mainly from the domain of programmable assembly. Some of the techniques have been optimized to take advantage of specific properties of this environment, but the basic methods are more widely applicable. Other possible applications include aerial photograph interpretation and medical image processing.

A Structure for VV

This section briefly describes the basic philosophy of a VV system and introduces some of the terminology. It characterizes the subtasks that are involved in setting up and performing VV tasks. Later sections will describe specific techniques that can be used to perform some of these subtasks.

A VV system can be (somewhat arbitrarily) partitioned into four stages:

(1) PROGRAMMING TIME: the user states the goal of the task, calibrates the camera, and chooses potential operator/feature pairs.

(2) TRAINING TIME: if previous pictures of the scene are available, the system applies the operators to several sample pictures and gathers statistical information about their effectiveness.

(3) PLANNING TIME: the system ranks the operators according to their expected contribution, determines the expected number of operators to be needed, and predicts the cost of accomplishing the task.

(4) EXECUTION TIME: the system applies operators one at a time, combines the results into confidences and precision, and stops when the desired levels have been reached or a cost limit has been exceeded.

These stages represent different conceptual steps in the development of a VV program. Different types of knowledge and techniques are applicable at different stages.

To accomplish a VV task requires progress from one stage to the next in the order shown above. However, for clarity, the execution time will be discussed first.

If one ignores the intermediate *details*, the VV process can be roughly characterized as follows: the programmer states the goal of the task in terms of the following three quantities:

(a) the *confidence* that the system has found the correct object(s),

(b) the *precision* within which the system has located the object(s),

and (c) the *cost* involved in determining this information;

and the system, at execution time, tries to extract useful information from a picture and combine the results of these extractions into estimates of the quantities of interest.

The implementation of the VV system discussed in this paper gathers information by applying *operators*, such as edge operators, correlation operators, and region growers, that are designed to locate and describe *features*, such as line segments, correlation points, and regions. The information produced by such operators can be roughly classified into two types: *value* information and *position* information. Value information includes the value of a correlation coefficient, the contrast across an edge, and the intensity of a region. Position information, in addition to (x,y) or (x,y,z) information, may include orientation information. For example, an edge operator might return (1) the (x,y) position of a point on an edge, (2) an estimate of the orientation of the edge, (3) the confidence that there is an edge at that position, and (4) the contrast across the edge. The first two quantities are types of position information; the second two are examples of value information.

Given the position and value information from several operators, what is the best estimate of the location of an object? What is the precision associated with that estimate? What should the combination rules be? The next section will describe a set of mathematical tools that form a set of combination rules for the class of VV tasks. The basic approach is to use Bayesian probability to estimate the confidences in the assignment of known alternatives to the matches and a least-squares technique to combine the available position information to form a current, best estimate of the location of the object (plus a tolerance about that estimate). These techniques are well-known, but they combine particularly nicely to answer the various needs of a VV system.

Combination Rules for Location

Within VV the purpose of applying operators to a picture of a scene is to establish a correspondence between features on an object in the scene and their two-dimensional positions in the picture. Given this correspondence, it is possible to use the camera calibration and a fitting scheme to determine the current location of the object in the scene. If the correspondence is correct, the deduced location will be correct. If the correspondence is not correct, the fitting scheme may be able to detect the inconsistency and possibly even identify the incorrect matches. However this checking process is not reliable enough to be depended upon as the main filtering mechanism for incorrect matches. Thus, it is important to produce as good a correspondence as possible before it is given to the fitting routine. This section reviews some of the attributes of one of the most common fitting procedures, least-squares fitting, and then develops a set of combination rules to express the confidence that an assignment of features to a set of matches is correct.

Least-squares Fitting

There are two important quantities in a location task: (1) an estimate for the location of the object and (2) the precision associated with that estimate. In the context of VV the *location of an object* refers to the position and orientation of the object's coordinate system in terms of some other coordinate system (e.g., the workstation coordinate system). In the most general case the location of an object may involve three rotations and three displacements. Often, however, there are constraints on the location of the object that reduce the number of unknown parameters. For example, if an object is known to be sitting up-right on a table, there are only three unknown parameters: two displacements and one rotation. The purpose of the fitting scheme is to use the correspondence between object features and picture positions to estimate the unknown parameters and produce precisions about these estimates.

A least-squares fitting routine was chosen for the first implementation of a VV system because it provides the necessary location and precision information and a generalized, non-linear, least-squares fitting routine was available (see [Gennery 1975]). Given estimates of the location parameters, one can predict the location of any point on the object. If the routine is given the uncertainties associated with each match, it can produce an estimate for the precision about each of the location parameters. Given the precisions about the location parameters, one can estimate the precision about points on the object. This ability to propagate the precisions is of major importance. It makes it

possible to estimate the precision about the features of interest, such as the screw hole. It also makes it possible to use the location of some initial features to predict the location and uncertainty about new features to be found. For example, if five features on the pump base have been located and the least-squares fitting routine has been called, the precision of the matches can be translated into a precision about the screw hole. If the precision is not sufficiently tight, the system needs to locate more features. The parameters and uncertainties produced by the first five matches can be used to predict the location and uncertainty region for a sixth feature.

The ability to estimate locations and precisions depends upon a correct correspondence between the object features and their picture positions. If there are several incorrect associations between operator results and object features, a fitting technique is probably useless because it will produce incorrect parameter values and will not be able to distinguish between correct and incorrect associations. If there are only a few incorrect assignments, however, the correct assignments may be able to override the incorrect assignments sufficiently to produce reasonable parameter values. If that is the case, the residuals associated with each assignment can be used to cull bad assignments. (The *residual* associated with an assignment is the difference between the position at which the operator located the image of the feature and the position of the feature predicted by the parameter values.)

The ability of a least-squares fitting routine to do this culling, however, is limited. For example, consider figure 1.

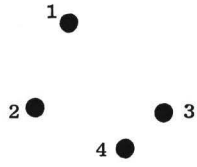

Figure 1.a

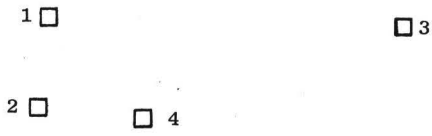

Figure 1.b

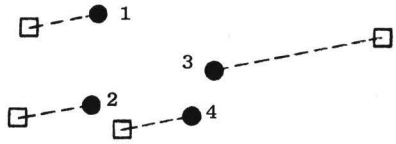

Figure 1.c

Figure 1.a shows the actual locations of the four features. Figure 1.b shows the positions returned by the operators. If the object is known to be sitting up-right on a table so that there are only three unknown parameters, the best fit to this data will probably be at the position shown in figure 1.c. In that case the incorrect assignment has a very large residual and the culling procedure would work. However, if all six parameters are unknown, the least-squares routine will take full advantage of the available flexibility in order to try to reduce the residuals, even it means rotating the object one-hundred twenty degrees and placing it at the other end of the table. The size of the resulting residuals would indicate that something is wrong, but the residual associated with the incorrect assignment would probably not be the largest. In fact, it may very well be the smallest. Its distance from the other group of matches acts as a lever arm that forces the fitting routine to minimize its error at the expense of the others. Thus, one has to be careful about using the fitting routine to cull incorrect matches.

The conclusion is to insure that the assignments are as correct as possible before they are handed to the fitting routine. The information available to make these assurances consists of the value and position information returned by the operators. The remainder of this section develops a sequential decision procedure to estimate the confidence that the assignment of a feature to the results of an operator is correct. The idea is to use only as much information as necessary. For example, if the value information returned by an operator clearly indicates the known alternative being matched, the system immediately makes the indicated assignment and adds it to the correspondence. However, if the confidence produced by the value information is not high enough, the system uses progressively more information from other operators to raise (or lower) the confidence associated with the assignment.

Confidences from Value Information

In order to use the value information produced by an operator one needs either an analytic method or an experimental method to estimate the expected ranges of values associated with the different possible matches. Since analytic methods are still quite limited, the current VV system uses experimental data. A supervised training session is used to apply each operator to several example pictures and gather three types of information:

(1) an estimate of the *a priori* probability that the operator will locate a certain known alternative,

(2) an estimate of the distribution of values produced by the operator when it locates a certain known alternative (actually an estimate of the density function),

and (3) an estimate of the density of values produced by the operator when it locates a surprise.

Figure 2 displays this information for an example operator. This simple model of the operator only distinguishes between two possibilities: (a) the operator locates the correct feature and (b) the operator locates a surprise. At execution time if the operator is applied to a picture and it returns a value of 1.67 (see figure 2), what is the probability that the operator has located the correct feature? Bayes' theorem (e.g., see [Hoel 1971]) is a standard way of combining the *a priori* probabilities with the density functions to answer this question. Bayes' theorem expresses the estimate of the desired *a posteriori* probability that the operator has located

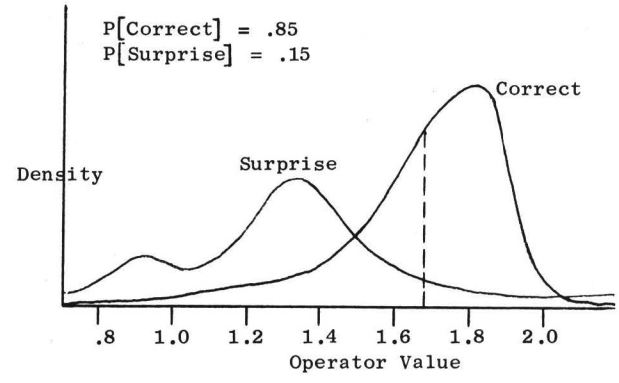

Figure 2

the correct feature in terms of the *a priori* probabilities and conditional probabilities as follows:

$$\{1\}\quad P[\text{Correct} \mid V] = \frac{1}{1 + \dfrac{P[V|\text{Surprise}]}{P[V|\text{Correct}]} * \dfrac{P[\text{Surprise}]}{P[\text{Correct}]}}.$$

where V is the value of the operator.

Given a particular value of V, the ratio of conditional probabilities (i.e., the *likelihood ratio*) is simply the ratio of the values of the density functions at that value. This observation makes it particularly easy to evaluate formula 1. It is also important to point out that formula 1 can be evaluated for any density functions. No particular form, such as a normal distribution, is required.

Formula 1 can be applied to the VV problem in a straightforward way: apply an operator, estimate the probability that it has located the correct feature, if that probability is above a certain threshold, add the feature and its match position to the correspondence to be used by the fitting routine, otherwise discard the results. This approach guarantees a certain probability of correctness for each operator used by the fitting routine.

If there are several known alternatives for an operator, this simple, two-possibility model is not sufficient. For example, consider the operator whose density functions are shown in figure 2. If several of the *surprise* matches are actually the result of the operator locating another feature that looks similar to the original one, the density function and the *a priori* probability associated with the *surprise* matches can be divided into two parts. See figure 3. In this case, when the operator returns a certain value at execution, the system has to decide which of the following situations is the most likely:

kal ≡ <the op. has located known altern. 1>
ka2 ≡ <the op. has located known altern. 2>
or surprise ≡ <the op. has located a surprise>,

where kal is the same as the *correct* match in the two-possibility model.

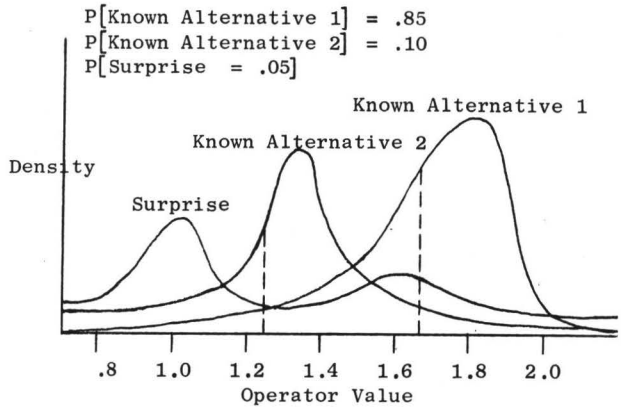

Figure 3

Bayes' theorem can be extended to produce the probabilities associated with the three possible explanations for the results of the operator. For example,

$$\{2\}\quad P[\text{kal} \mid V] = \frac{1}{1 + \dfrac{P[V|\neg(\text{kal})]}{P[V|\text{kal}]} * \dfrac{P[\neg(\text{kal})]}{P[\text{kal}]}}.$$

Since

$$\{3\}\quad P[\neg(\text{kal})] = P[\text{ka2}] + P[\text{surprise}],$$

formula 2 can be expanded into

$$\{4\}\quad P[\text{kal}|V] = \frac{1}{1 + \dfrac{P[V|\text{ka2}]}{P[V|\text{kal}]} * \dfrac{P[\text{ka2}]}{P[\text{kal}]} + \dfrac{P[V|\text{surp}]}{P[V|\text{kal}]} * \dfrac{P[\text{surp}]}{P[\text{kal}]}}.$$

Formula 4 can be generalized to handle operators with N known alternatives and a surprise. Let Aj, for j equal 1 to N, represent the N known alternatives and let A0 represent the surprise. Then

$$\{5\}\quad P[Aj|v] = \frac{1}{1 + \sum_{i \neq j} \dfrac{P[v|Ai]*P[Ai]}{P[v|Aj]*P[Aj]}} \qquad (\text{for } 0 \le j \le N).$$

This formula is convenient because it states the desired probability in terms of the *a priori* probabilities and the likelihood ratios. Given the value of an operator, the probability of each alternative can be computed, and the alternative with the highest probability is declared the best match. If the example operator returns a value of 1.67, as it did earlier, the probability that it has matched the first known alternative is still the same. If the operator happens to return a value of 1.22, the old formulation would have estimated that the operator had located a surprise. In this new formulation, the system would predict that the operator has located the second known alternative with a probability of .94. Thus, knowing about ka2 and incorporating it into the formula has increased the percentage of times the system can associate a known alternative with the results of the operator. It has also increased the percentage of times the system can make the assignment with sufficient confidence to add the assignment to the correspondence.

Confidences from Position Information

In the scheme discussed so far, if the probability associated with the best alternative is less than the acceptance threshold, the results of the operator are simply discarded. If the probability is less than the threshold, but close, is there any way to incorporate additional information in order to raise or lower the probability? One possibility is to extend Bayes' theorem to incorporate the position information returned by the operator. In fact, it is possible to estimate P[kal | v,p], but the position information of one operator by itself usually does not add any significant constraints because the operator is just as likely to find the feature at one location as another (within a certain tolerance region).

The position information produced by each individual operator may not be helpful, but the relative positions of two or more features can add important, discriminating information. For example, consider figure 4.a, which is a picture of a screw with two features marked. Assume that the location of the screw is known within plus or minus one quarter inch along each axis and that the orientation of the screw is known to be within plus or minus fifteen degrees of vertical. If the operators trying to locate these two features happen to find their best matches at the positions shown in figure 4.b, what is the probability that they have located the correct features? The probability is small because it would require the screw to be rotated 140 degrees from vertical, which is highly unlikely, given the initial uncertainties. If the two matches are at the positions shown in figure 4.c, the probability that the matches are correct is significantly higher. Thus, the relative structure of two or more matches can be an important source of information. It can be used to measure the consistency of a set of matches.

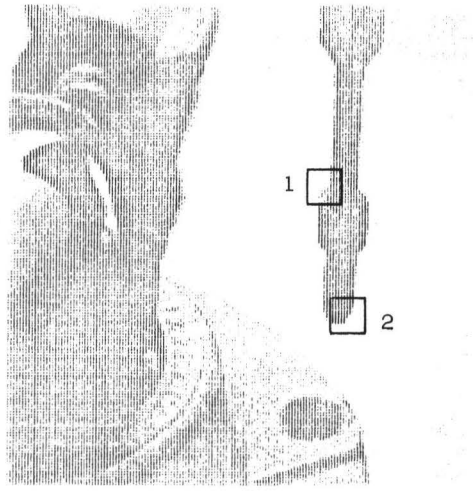

Figure 4.a

Figure 4.b

Figure 4.c

This intuitive idea can be captured in a form of Bayes' theorem. For example, assume that two operators have been applied and their value information did not clearly indicate which known alternatives they matched. The relative position of their matches may be able to clarify the situation. The probabilities of interest are the probabilities associated with the different possible assignments of known alternatives to the results of the two operators. That is, given the fact that the operators have located their matches at positions p1 and p2, what is the probability that the first operator has located its jth known alternative and the second operator has located its kth known alternative? Let fj represent the proposition that the jth known alternative is the best assignment to the results of the first operator and let gk represent the proposition that the kth known alternative is the best assignment for the second operator. Then the probability of interest is:

$$\{6\} \qquad P[fj,gk \mid p1,p2],$$

Since

$$\{7\} \qquad P[fj,gk \mid p1,p2] = \frac{P[fj,gk,p1,p2]}{P[p1,p2]}$$

and the denominator in formula 7 is the same for all j's and k's, the program can simply choose the assignment that has the highest value of P[fj,gk,p1,p2].

The probability P[fj,gk,p1,p2] represents the probability that the first operator locates its match at p1, the second operator locates its match at p2, the best assignment for operator one is known alternative fj, and the best assignment for operator two is known alternative gk. In effect this probability is the probability that the object would be at a location such that the features would appear at the indicated positions in the picture. One way to express this probability is in terms of the parameters that describe the location of the object. For example, consider a task in which there are three unknown parameters: two displacements, dx and dy, and a rotation, dt. Statistics can be gathered at training time that describe the expected density functions for these parameters. Then if the results of two operators imply that dx=.1, dy=-.2, and dt=-2.3, the probability P[fj,gk,p1,p2] can be expressed as

$$\{8\} \qquad P[fj,gk,p1,p2] = P[dx=.1;dy=-.2;dt=-2.3].$$

To a first approximation the fuzzy logic convention can be used to approximate this probability of a conjunction by the minimum of the probabilities of the conjuncts.

Once the best assignment has been found, the probability that it is the correct assignment can be estimated by the following formula:

$$\{9\} \qquad P[fj,gk \mid p1,p2] = \frac{1}{1 + \sum_{\neg(x=j \,\&\, y=k)} \frac{P[p1,p2,fx,gy]}{P[p1,p2,fj,gk]}}.$$

Formula 9 can be extended to evaluate the assignments for more than two operators, but unfortunately the number of terms in the denominator quickly becomes prohibitively large. For example, ten operators, each with two alternatives, require 1023 probabilities to be computed in order to determine the best assignment. This fact implies that formula 9 should only be used to assign alternatives to small subsets of the operators. The overall probability that the complete assignment is correct can be estimated by a function of the probabilities associated with the subsets. For example, it is possible to modify a maximal clique algorithm so that it determines the most consistent structure of pairwise evaluated matches (i.e., subsets of order two) [Barrow 1976].

If subsets of the operators are used to develop confidences, since it may be computationally expensive to evaluate all possible subsets, it is important to choose subsets that are expected to produce distinct patterns. For example, consider figure 5, which shows the known alternatives for four operators. If the program is trying to assign one of the two alternatives to the results of operator 1, and the relative position information is needed to distinguish between the two alternatives, which of the other operators should be used? Operator 2 may not be helpful because it is difficult to distinguish between assignments [1a,2b] and [1b,2a]. If the initial constraints allow for a small angular uncertainty and a small scale uncertainty, it may be difficult to distinguish between assignments [1a,4] and [1b,4]. However, even if the scale is allowed to change fifty percent and the orientation in the plane is completely unconstrained, it is still possible to distinguish between assignments [1a,3a] and [1b,3a], or between [1a,3b] and [1b,3b]. Therefore, a good subset to be evaluated is the subset consisting of operator 1 and operator 3.

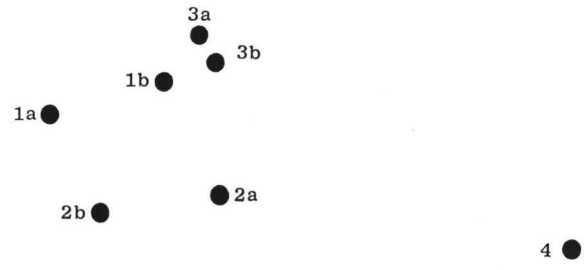

Figure 5

Since the known alternatives for the operators are known in advance and the task constraints are known in advance, it is easy to estimate the distinctness of patterns and choose the best subsets at planning time.

Summary

This section described a fitting scheme and an ordered set of rules to estimate confidences. The least-squares fitting scheme is well-known and produces the desired location and precision information, *if the assignment of known alternatives is correct.* The combination rules provide ways to estimate the confidences associated with the assignments. Some of the rules use the value information produced by the operators, some of them use the position information.

Techniques to Simplify VV Programming

The purpose of this section is to present a sample of the possible techniques that can simplify the programming of VV tasks. These techniques are designed to automate the programming process to such an extent that a programmer who is not an expert in computer vision can set up programs to perform VV tasks. A version of each of the techniques mentioned in this section has been implemented in the current VV system. Out of necessity the descriptions are brief. More techniques and more complete descriptions can be found in the author's thesis [Bolles 1976].

Estimate the Desired Resolution of a Picture

Given an initial set of constraints and a desired precision about some point of interest, the VV program can use the expected precisions of the operators to help determine a good resolution for the picture. The resolution is important because (a) it may not be possible to achieve the desired precision with the available operators if the resolution is too low, and (b) the amount of searching may be excessive if the resolution is too high.

Given a calibrated camera it is possible to convert the two-dimensional uncertainty about the position of an operator's match in the picture coordinate system into a three-dimensional uncertainty about the feature in the workstation coordinate system. Given a set of features and their associated uncertainties, the least-squares routine can predict the precision about the point of interest.

Suggest Operator/Feature Pairs

In addition to providing a convenient environment within which a programmer can experiment with different operators, there are two methods that a VV system can use to suggest potentially useful operator/feature pairs:

(1) The VV program can analyze a typical picture of the scene and produce a list of visually distinct features. For example, an edge operator can be applied to a picture in order to pick out all pairs of line segments that form a corner of a certain minimum size. Hannah and Moravec have each implemented *interest operators* of this type that search for good correlation features (see [Hannah 1974] and [Moravec 1976]).

(2) The VV program can analyze a model of the scene and produce a list of features on the objects that can be seen and are expected to be distinctive. For example, a three-dimensional model and a hidden-line elimination scheme may be used to predict visible corners. Bolles has partially implemented a suggestion system of this type [Bolles 1976]. It is based upon an object modelling system designed by Miyamoto and Binford [Miyamoto 1975] and it suggests potentially useful curve segments.

Both of these automatic suggestion methods reduce the amount of detailed work required of the programmer. The programmer only has to filter out suggestions that may be difficult to locate reliably or that produce unreliable position information. In theory, the second method should produce better suggestions because it can have access to all of the knowledge associated with the real objects: their appearance, their structure, and their function. For example, the first method may suggest a corner feature, one side of which is formed by a shadow. Such a feature may not produce reliable position information if the location of the shadow changes as the object moves about within its range of uncertainty. The second method, on the other hand, could directly determine that one side of the corner is formed by a shadow and not make such a suggestion.

Gather Statistics

Given a set of operator/feature pairs, the programmer has to supervise the training session. Each operator is applied to each training picture and the superviser has to specify which known alternative or surprise was matched. This process could be tedious unless it is carefully human-engineered.

The current system displays two pictures side-by-side, a reference picture with the known alternatives marked and the training picture with the matches of the operators marked. The system uses the value and position information to assign known alternatives to matches and asks the user for confirmation. In this way the interaction between the programmer and the training system is minimized.

Determine the Order of the Operators

There are several factors that can enter into the ordering of the operators: for example, the expected cost, the expected precision of the match, and the expected contribution to the confidence associated with the best known alternative. There are relatively straightforward methods to estimate the first and second quantities, but the third is more complicated. Consider the following definition of the contribution of an operator that has N known alternatives and has returned a value of V:

$$\{10\}\ \langle\text{contrib. at } V\rangle = \begin{cases} 0 & \text{(if surprise)} \\ \text{MAX}(P[A1|V]\ldots P[An|V]) & \text{(if KA).} \end{cases}$$

The expected value of this function can be computed as follows:

$$\{11\}\ \langle\text{exp. contrib.}\rangle = \int_{-\infty}^{+\infty} \langle\text{contrib. at } V\rangle * \text{density}(V)\,dV.$$

Since it is difficult to expand this integral symbolically, a numerical integration technique can be used instead.

Given values for the expected cost, the expected precision, and the expected contribution for all of the operator/feature pairs, one simple ordering scheme is to rank them according to:

$$\{12\}\quad \frac{\langle\text{expected contribution}\rangle}{\langle\text{expected cost}\rangle * \langle\text{expected precision}\rangle},$$

largest first.

The exact ordering is not critical because several operators are applied for each task. As long as the better operators are applied first, the overall cost will be close to the minimum.

The linear ordering of operator/feature pairs is a simple example of a strategy. Other researchers have investigated more general strategies and more general strategy optimization techniques within similar domains (e.g., see [Taylor 1976] and [Sproull 1977]).

Results

The VV system has been used to locate fifteen to twenty different objects, such as a vise, a screw dispenser, a pencil sharpener base, and an engine casing. The programming time for a new task (using 4 to 5 training pictures) was about an hour. Programming consisted of (1) stating the task; (2) calibrating the camera; (3) choosing potential operator/feature pairs; (4) setting up, taking, and analyzing training pictures; and (5) letting the system rank the operator/feature pairs. The execution time depended upon several factors, the most important of which was the ratio of known precision to desired precision. For a typical task in which this ratio was 1:10, four or five operators were required and the execution time was a few seconds. The time to teach a person to use the system was about an hour or two, most of which was spent watching the system being used.

Cross-correlation was the main operator used for these tasks, however, an edge operator and a blob characterizer were used in a few tasks. Approximately thirty percent of the operators had more than one known alternative. If an operator had more than two known alternatives, it was generally not cost effective to use it because of the expense involved in determining which known alternative had been matched.

The training pictures were used for two slightly different purposes in different tasks: (1) to set a threshold that distinguishes *reasonable* operator values from values that indicate that the operator did not find a known alternative, and (2) to determine a set of density functions that could be used to estimate the confidence that the operator matched a particular known alternative. Four or five training pictures were sufficient for the first purpose; forty or fifty were needed for the second. The structural information was more reliable because it was difficult to control the lighting, background, and camera sensitivity from one day to the next. Therefore, the value information was mainly used to eliminate obvious mistakes and the structural information was used to make the final assignments of known alternatives and to determine the location of the object.

Conclusion

There are two main conclusions of this paper:

(1) A large class of visual feedback tasks can be formulated and accomplished within one framework.

(2) The framework can be implemented in such a way that it is relatively easy for a programmer who is not an expert in vision research to construct programs that perform visual feedback tasks.

The justification for each of these statements was essentially a proof by construction. A class of visual feedback tasks, referred to as verification vision tasks, was characterized and an interactive system that greatly simplifies the programming of such tasks was implemented.

A VV task is a task in which the scene is highly predictable; there are no big surprises. The problem is to coordinate all of the available knowledge in order to accomplish the task in as efficient way as possible. This paper outlined the different subtasks within VV and the different types of information that can be used to perform these subtasks. It presented an ordered set of rules that evaluate the results of operators and a scheme to produce the desired location and precision information. Finally, the paper gave a sample of the techniques that can be used to simplify the programming of VV tasks.

References

Agin, Gerald J. and Binford, Thomas O. [1973], "Computer Description of Curved Objects," Proc. of 3IJCAI, Stanford, August 1973, pp. 629-640.

Baird, Michael L. [1976], "Sequential Image Enhancement Technique for Automotive Parts on Conveyor Belts," Submitted to Proc. of 5IJCAI, November 1976.

Barrow, H.G., Burstall, R.M. [1976], "Subgraph Isomorphism, Matching Relational Structures, and Maximal Cliques," Information Processing Letters, Vol. 4, No. 4, January 1976, pp. 83-84.

Bolles, Robert C. [1976], "Verification Vision within a Programmable Assembly System," Stanford A. I. Memo No. 295, December 1976.

Chien,R.T. and Jones,V.C. [1975], "Acquisition of Moving Objects and Hand-Eye Coordination," Proc. of 4IJCAI, Tbilisi, Georgia, USSR, September 1975, pp. 737-741.

Finkel, Raphael A. [1976], "Construction and Debugging of Manipulator Programs," Stanford A. I. Memo No. 284, August 1976.

Fischler, Martin A., and Elschlager, Robert A. [1971], "The Representation and Matching of Pictorial Structures," Lockheed Missiles and Space Company, LMSC-D243781, September 1971.

Gennery, Donald B. [1975], "Least-Squares Stereo-Camera Calibration," Stanford A. I. internal memo, 1975.

Hannah, Marsha Jo [1974], "Computer Matching of Areas in Stereo Images," Stanford A. I. Memo No. 239, July 1974.

Hoel,P.G. [1971], "Introduction to Mathematical Statistics," John Wiley and Sons, Inc., 1971.

Holland,S.W. [1975], "A Programmable Computer Vision System based on Spatial Relationships," Stanford Digital Systems Laboratory Technical Report #104, December 1975.

Kashioka, S., Ejiri, M., Sakamoto, Y. [1976], "A Transistor Wire-Bonding System Utilizing Multiple Local Pattern Matching Techniques," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-6, No. 8, August 1976, pp. 562-570.

Miyamoto, Eiichi and Binford, Thomas O. [1975], "Display Generated by a Generalized Cone Representation," Computer Graphics and Image Processing Conference, Anaheim, Ca., May 1975.

Moravec, Hans and Gennery, Donald [1976], "Cart Project Progress Report," Stanford A. I. internal memo, October 1976.

Sproull, Robert F. [1977], "Strategy Construction using a Synthesis of Decision Theoretic and Heuristic Techniques," Stanford Ph.D. Thesis, 1977.

Taylor, Russell H. [1976], "The Synthesis of Manipulator Control Programs from Task-level Specifications," Stanford Ph.D. Thesis, August 1976.

IIIT-Hyderabad - Library
CP22222