MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Memo No. 1353

February, 1992

Space Efficient 3D Model Indexing

David W. Jacobs

Abstract: We show that the set of 2D images produced by the point features of a rigid 3D model can be represented with two lines in two high-dimensional spaces. These lines are the lowest-dimensional representation possible. We use this result to build a system for representing in a hash table at compile time, all the images that groups of model features can produce. Then at run time a group of image features can access the table and find all model groups that could match it. This table is efficient in terms of space, and is built and accessed through analytic methods that account for the effect of sensing error. In real images, it reduces the set of potential matches by a factor of several thousand. We also use this representation of a model's images to analyze two other approaches to recognition: invariants, and non-accidental properties. These are properties of images that some models always produce, and all other models either never produce (invariants) or almost never produce (non-accidental properties). In several domains we determine when invariants exist. In general we show that there are an infinite set of non-accidental properties that are qualitatively similar.

Acknowledgements: This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology, and was funded in part by an Office of Naval Research University Research Inititative grant under contract N00014-86-K-0685, and in part by the Advanced Research Projects Agency of the Department of Defense under Army contract number DACA76-85-C-0010, and under Office of Naval Research contract N00014-85-K-0124.

1 Introduction

Object recognition systems typically search for matches between image features and model features that are consistent with some transformation of the model into the image. This search can require a great deal of computation, particularly in challenging domains such as the recognition of 3D objects from cluttered 2D images, and the identification of an object from a large data base of possible objects. One approach to handling this complexity is to decompose the recognition task so that as much of the work as possible is done on the models alone, at compile time, and on the image alone, in a bottom up process. The results of these two separate computations are then combined in a simple comparison.

Such an approach to recognition can take the form of indexing, combined with grouping. By indexing, we mean that the system places pointers to groups of model features in a hash table, at compile time. At run time, a group of image features accesses the hash table to find all the model groups that might match it. If we want efficient run-time processing, image groups will each access the table in only a single place. That means that the table must, in some form, represent the entire set of images that each model group could produce. This paper determines the most space-efficient possible method for representing a 3D model's point features in such a hash table. It also presents a method for analytically determining which entries to make in such a table.

Indexing may be used on its own, in which case the group size is usually made as small as possible to limit the number of model groups represented in the table, and the number of image groups that must be considered at run time. Alternately, indexing may be combined with grouping. Grouping selects a relatively small set of groups of model features to enter in the hash table. It then selects a small number of groups of image features in a model-independent, bottom up process. This avoids the combinatoric explosion that occurs when all possible image groups and model groups are considered. With effective grouping, large groups become desirable input to the indexing system, because they provide the greatest discriminatory power. That is, fewer model groups will match a large image group than a small image group. See [6] for further discussion of this point. In this paper we focus on the problem of building an indexing system appropriate for use with grouping.

The central problem of indexing is to determine the most economical possible representation for the set of image groups that each model group might produce. It is trivial to do indexing by making a different entry in the hash table for every different image group a model group can produce, but this would require excessive space. So instead we seek a

representation of images such that minimal space is required to describe all these images.

In 2D domains this has been done with invariant descriptions, that is, with descriptions of images that have the property that every image of a given model group produces the same description. Invariants have been found that can capture all the information in the image group that can be used to match it to model groups, while allowing us to represent each model group with a single entry in the indexing hash table. However, it has been shown that such invariants do not exist for the description of 3D models[3], [5], [6], [20]. Furthermore, [5], [6] show that to perform indexing of 3D models, each model must be represented by a 2D surface in a single index space. Unfortunately, large amounts of space are required to accurately represent such a 2D surface discretely. In this paper, however, we show that the images of a model group can be represented as a 2D surface that can be canonically decomposed into two 1D surfaces, represented in two orthogonal index spaces. This result makes the space requirements of indexing reasonable, because we can discretize two 1D surfaces using much less space than would be required to discretize one 2D surface.

Our results also provide a simple conceptualization of the recognition problem, which we use to analyze two other approaches to recognition. First, when we constrain our universe of objects to contain only some collections of 3D points, the question resurfaces as to whether invariant functions exist. We use our previously stated result to answer this question in several domains. We also consider a second, related approach to recognition. This approach uses non-accidental properties, which are defined as properties of an image that some models produce from all viewpoints, and other objects only produce from almost no viewpoints. A small set of non-accidental properties have been used. We show that they are a few instances of an infinite class of such properties, all of which are qualitatively similar.

Another problem left unanswered in previous 3D indexing systems was the analytic computation of the entries to make in the hash table. Previously, this has been done only by sampling the set of possible viewpoints of each model. In this paper we present an analytic method of building the indexing table, as well as an analytic method of accounting for the effects of sensing error. These results have allowed us to build a practical indexing system, which we demonstrate using real images. The resulting indexing system produces speedups of up to a factor of several thousand over brute force search.

2 Describing the Images a Model Produces

This section describes a compact, analytically determined representation for the set of all images that a general 3D model may produce. Models are assumed to consist of any arbitrary collection of ordered 3D point features. We ignore sensing error when describing the model because we account for error with regard to a particular image of the model, at lookup time. We are then able to describe all images of a model with two straight lines located in two orthogonal spaces. These lines can easily be derived analytically from the model. This representation is optimal in the dimensionality of the space required, and in introducing no false positive or false negative information.

In describing the images that a model produces we use the following novel model of projection. First, we assume that the object is imaged from an arbitrary viewpoint using orthographic projection with scale. Orthographic projection with scale is a common approximation to perspective projection. Next, we allow an arbitrary affine transform to be applied to the resulting image. Applying an arbitrary affine transform to an image is equivalent to viewing that image from an arbitrary position, assuming that this projection also is a scaled orthographic projection. Therefore, our projection model encompasses all images that a model might produce, as well as all images that a photograph of the model might produce.

There are several reasons for using this projection model. As we will see, it is mathematically convenient. But in addition, it allows us to build an indexing system that can recognize photographs of objects. This also suggests the hypothesis that human ability to interpret photographs is an epiphenomenon of the fact that, for computational reasons, our visual system does not make use of features of an image that vary under affine transforms (see section 7 for discussion of a related hypothesis). Finally, it may be easily shown that this model of projection is equivalent to multiplying 3D model points by an arbitrary two by three matrix, and then adding an arbitrary translation vector. This seems to be the simplest linear projection model from 3D to 2D (see [16, 24] for further discussion of this projection model).

We now show that under this model of projection, the set of images produced by any model is described by the cross product of two lines in two orthogonal spaces. To do this, we represent images as follows. As an image consists of 2D point features, we use the first three points to define an affine basis. That is, if we denote the image points: $(\mathbf{p_1}, \mathbf{p_2}, ..., \mathbf{p_n})$, let:

$$\mathbf{o} = \mathbf{p_1} \qquad \mathbf{u} = \mathbf{p_2} - \mathbf{p_1} \qquad \mathbf{v} = \mathbf{p_3} - \mathbf{p_1}$$

Then we may fully describe the locations of the remaining points using affine coordinates derived with respect to this basis. For example, we describe $\mathbf{p_4}$ with the parameters (α_4, β_4), where:

$$\mathbf{p_4} = \mathbf{o} + \alpha_4 \mathbf{u} + \beta_4 \mathbf{v}$$

Then an image is fully described by the parameters: $(\mathbf{o}, \mathbf{u}, \mathbf{v}, (\alpha_4, \beta_4), ...(\alpha_n, \beta_n))$. It is important to what follows that the affine coordinates of a point are left unchanged by any affine transform ([17]).

Due to the model of projection we use, we may ignore the first three of these parameters. To see this, we note that, except in degenerate cases, there exists an affine transform that will map any three image points to any other three image points. Therefore, under the type of projection that we consider, if a model can produce the image, $(\mathbf{o}, \mathbf{u}, \mathbf{v}, (\alpha_4, \beta_4), ...(\alpha_n, \beta_n))$, it can also produce the image $(\mathbf{o'}, \mathbf{u'}, \mathbf{v'}, (\alpha_4, \beta_4), ...(\alpha_n, \beta_n))$ for any choice of $(\mathbf{o'}, \mathbf{u'}, \mathbf{v'})$, by combining the affine transform that maps $(\mathbf{o}, \mathbf{u}, \mathbf{v})$ to $(\mathbf{o'}, \mathbf{u'}, \mathbf{v'})$ with the affine transform that was part of the projection that produced the original image. Therefore, the parameters $(\mathbf{o}, \mathbf{u}, \mathbf{v})$ provide no information about whether a model could produce an image.

The remaining image parameters form what we will call an *affine space*. An image with n ordered points is mapped into a point in a $2(n-3)$-dimensional affine space by finding the affine coordinates of the image points, using the first three as a basis. We divide the affine space into two orthogonal subspaces, an *α-space*, and a *β-space*. The *α-space* is the set of α coordinates of the image's affine coordinates, and the β-space is similarly defined. The affine space is then equal to the cross product of the *α-space* and the *β-space*, and each image corresponds to a point in each of these two spaces. The previous paragraph states that the images that a model can produce are fully described by the locus of points these images map to in affine space. We now show, that for any model, these images map to the cross product of lines in *α-space* and *β-space*.

First we must note that our model of projection can be decomposed into one part, which captures the effects of viewing direction, and a second part, which is an affine transform of the projected image. We can think of orthographic projection as projecting the model along some viewing direction into a plane. Following this, orthographic projection also allows rotation in the plane, translation, and scale, but we can ignore these parts of the transformation by folding them into the affine transformation that we allow on the image, following the orthographic projection. To find the parts of affine space corresponding to

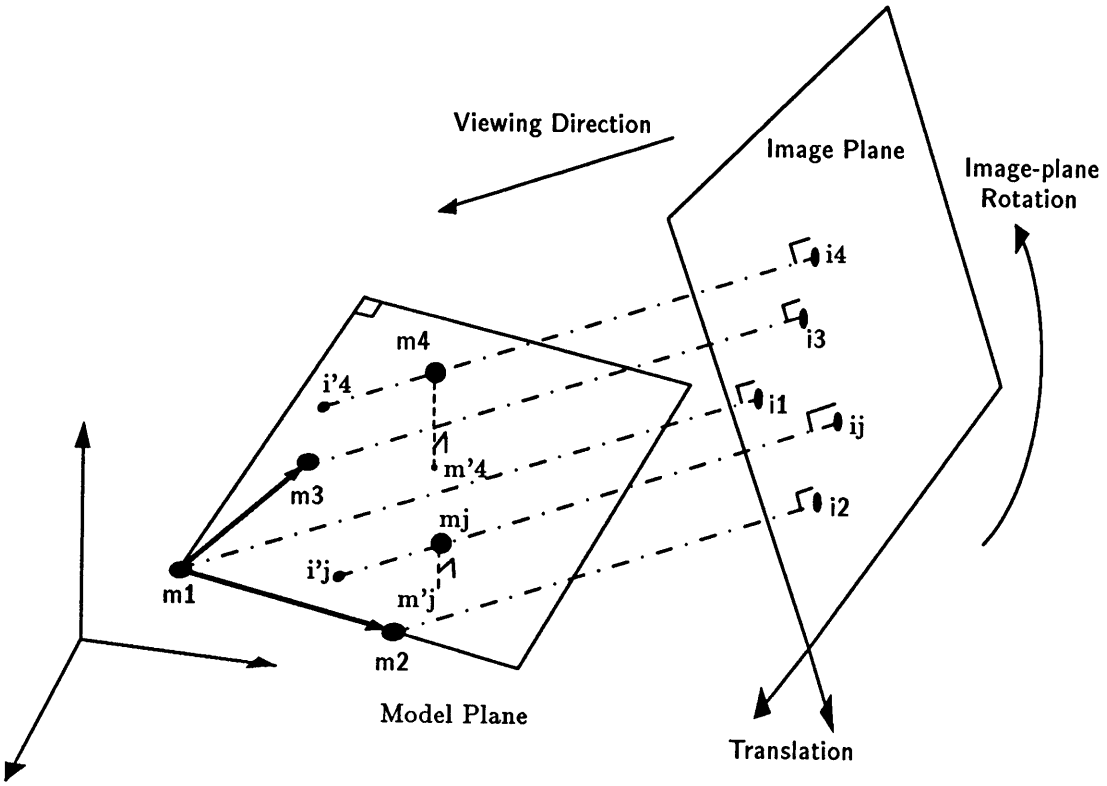

Figure 1: The image points $\mathbf{i_1}$, $\mathbf{i_2}$, $\mathbf{i_3}$, $\mathbf{i_4}$, and $\mathbf{i_j}$ are the projections of the model points $\mathbf{m_1}$, $\mathbf{m_2}$, $\mathbf{m_3}$, $\mathbf{m_4}$, and $\mathbf{m_j}$, before the affine transform portion of the projection is applied. The values of the image points depend on the pose of the model relative to the image plane. In the viewing direction shown, $\mathbf{i}'_4$ and $\mathbf{m_4}$ project to the same image point. $\mathbf{m}'_4$ is in the model plane, directly below $\mathbf{m_4}$. Note that $\mathbf{i_4}$ has the same affine coordinates as $\mathbf{i}'_4$.

images of a particular model, we only need to consider the sets of affine coordinates that a model may produce in an image as the viewing direction varies. The subsequent affine transformation leaves the affine coordinates unchanged.

We now assume the model consists of at least five points. Call the plane determined by the first three model points, the *model plane*. If we project the fourth model point, $\mathbf{m_4}$, perpendicularly into the model plane, we call this point $\mathbf{m}'_4$. Since $\mathbf{m}'_4$ is in the plane of the first three model points, we can discuss its affine coordinates with respect to these three model points. We call these affine coordinates (a_4, b_4). Similarly, for the j'th model point, $\mathbf{m_j}$, we define $\mathbf{m}'_\mathbf{j}$ and (a_j, b_j) (see figure 1). Without loss of generality, assume the model plane is $z = 0$. So z_4 is the height of $\mathbf{m_4}$ above this plane, and z_j is the height of $\mathbf{m_j}$ above the model plane. We define $r_j = \frac{z_4}{z_j}$.

We now show that for any affine coordinates (α_4, β_4), there is a viewpoint in which the projection of $\mathbf{m_4}$ has those affine coordinates. We then express the affine coordinates of the remaining projected model points as a function of (α_4, β_4). Some point, $\mathbf{i'_4}$, in the model plane has affine coordinates (α_4, β_4). If we form a line including $\mathbf{i'_4}$ and $\mathbf{m_4}$, this line describes a viewing direction from which $\mathbf{m_4}$ and $\mathbf{i'_4}$ project to the same image point, $\mathbf{i_4}$. Since $\mathbf{i'_4}$ is coplanar with the first three model points, it has the same affine coordinates when viewed from any direction, since affine coordinates of planar points are not changed by any affine transformation, and viewing a planar object from an arbitrary viewpoint is equivalent to applying an affine transform. So $\mathbf{i_4}$ has affine coordinates (α_4, β_4).

A line parallel to the viewing direction will also pass through $\mathbf{m_j}$, and intersect the model plane at a point we call $\mathbf{i'_j}$. The affine coordinates of $\mathbf{i_j}$, the image of $\mathbf{m_j}$, are the same as the affine coordinates of $\mathbf{i'_j}$ in the model plane. Since the line connecting $\mathbf{m_j}$ to $\mathbf{i'_j}$ is parallel to the line connecting $\mathbf{m_4}$ to $\mathbf{i'_4}$, the triangle $\mathbf{m_4 m'_4 i'_4}$ will be similar to the triangle $\mathbf{m_j m'_j i'_j}$, and scaled by a factor of r_j. In particular, this means that: $(\mathbf{i'_4} - \mathbf{m'_4}) = r_j(\mathbf{i'_j} - \mathbf{m'_j})$, and therefore:

$$(\alpha_j, \beta_j) = (a_j, b_j) + \frac{((\alpha_4, \beta_4) - (a_4, b_4))}{r_j}. \tag{1}$$

This equation describes all image parameters that these five points may produce. For any image, this equation will hold. And for any values described by the equation, there is a corresponding image that the model may produce, since, from above we know that for any values (α_4, β_4), there is a view of $\mathbf{m_4}$ that produces these values. Taking the α component of these equations:

$$\alpha_j = a_j + \frac{(\alpha_4 - a_4)}{r_j}$$

we have equations that describe a line in α-space. We may derive a similar set of equations in β-space. These equations are independent. That is, for any set of α coordinates that a model may produce in an image, it may still produce any feasible set of β coordinates. There are also degenerate case. If some of the model points are coplanar, than some of the r_j are infinite, and the line is vertical in those dimensions. If all the model points are coplanar, the affine coordinates of the projected model points are invariant, and each model is represented by a point in affine space. If the first three model points are colinear, then the line is undefined.

Notice that for any line in α-space, there is some model whose images are described by that line. It is not true that there is a model corresponding to any pair of lines in α-space and β-space because the parameters r_j are the same in the equations for the two lines. This

means that the two lines are constrained to have the same directional vector, but they are not further constrained.

In the absence of image error, indexing could proceed in the following way. Form two hash tables which discretize α-space and β-space. For each model, compute the corresponding line in each of these spaces, and make an entry in each bucket that intersects one of these lines. At run time, given an image, determine the points in α-space and β-space that correspond to that image. Look in the appropriate bucket in each of the two spaces, and intersect the results. If the buckets are made sufficiently small, this will produce exactly the set of models that could produce the image.

Representing each model group's images with two 1D surfaces is an important improvement over using a single 2D surface, because it requires much less space to discretely represent a pair of 1D surfaces than to discretely represent a 2D surface. There is a run-time price that must be paid for this space, when we intersect the results of two separate lookups. However, this cost is negligible in the actually system we have built. We note that the dimensionality of the surfaces used to represent a model group's images is now the best that can be done, since [6] has shown that in a single index space, a 2D surface is required to represent these images, and it is not possible to represent a 2D surface as the cross-product of any countable number of countable sets.

3 Invariants and Non-Accidental Properties

Before describing this indexing system further, we will consider some implications of this view of the recognition problem. In the error free case, we have almost entirely reduced the recognition problem to a very simple form, in which recognition is the problem of determining which points fall on which lines. To demonstrate the usefulness of this, we will consider two influential approaches to recognition from this point of view.

A number of recognition systems have been based on *invariants*. In the context of recognition, an invariant is a function of the image that has the following property: if f is an invariant function, then for any model, m, if i_1 and i_2 are images of m, then $f(i_1) = f(i_2)$. That is, an invariant is a property that is true of all images of a model, and hence does not vary under the transformation that turns a model into an image.

Our formulation of the recognition problem makes it easy to prove a number of results about invariants. These results will only apply to our model of projection. However, we note that considering our projection model is equivalent to assuming that any invariant

function will be both invariant for the orthographic projection of a set of 3D models, and will also be invariant for the orthographic projection of 2D images of these models.

2D recognition systems have long implicitly relied on the descriptions of 2D objects that do not vary as the object is rotated or translated in the plane. More recently, invariants have been used for the recognition of planar models from arbitrary, 3D views ([28], [17], [18], [9], [10], [25]). Recently, [3], [5], [6], and [20] have proven that there are no non-trivial invariants when models may consist of arbitrary collections of 3D point features. These proofs took the following form. Given any two models, M_1 and M_2, a set of intermediate models, P_1, P_2 ... P_n were constructed. By this construction, M_1 and P_1 produce a common image, hence any invariant function would have to have the same value on any images these models produce. Similarly, P_1 and P_2 produce a common image, and so on, until M_1 and M_2 are linked by this series of intermediate models. Hence, the invariant function must produce the same value for images of any two models, and is trivial.

We can now show that there are no invariant functions of 3D models with a proof that requires only two intermediate models. Suppose model M_1 corresponds to the two lines, A_1 and B_1 in α-space and β-space respectively. Similarly, suppose model M_2 corresponds to A_2 and B_2. Then there are an infinite number of lines that intersect both A_1 and A_2. Choose one of these, A_1'. Choose B_1' as any line that is parallel to A_1', and intersects B_1. Then there is a model, P_1 that corresponds to the lines (A_1', B_1'). P_1 has an image in common with M_1, since A_1 intersects A_1', and B_1 intersects B_1'. We may then construct P_2 and its lines, (A_2', B_2'), so that B_2' intersects B_1' and B_2, and so that A_2' passes through the point where A_1' and A_2 intersect. So P_2 will have an image in common with P_1 and M_2. Therefore, any invariant function must have the same value for any image of any of the four models.

We may also use our previous results to examine other questions about the occurrence of invariants. As [20] points out, there may be invariant functions for a particular set of 3D models, if the models can be divided into non-trivial equivalence classes, where two models are equivalent if they have an image in common, or are both equivalent to another model. From our previous work, it becomes easy to form these equivalence classes for a particular set of models, because we can tell that two models produce a common image if their corresponding lines in α-space and in β-space intersect.

Another question that arises for a restricted set of models is whether there is an invariant function that will produce no false positive matches. We can think of an invariant function as assigning values to models, because it assigns the same value to every image of a model.

An invariant function leads to no false positives if the function maps a model and an image to the same value only when there really is a transformation that will cause the model to produce that image.

To answer this question we consult our representation of a model's images as planes in affine space, that is, we take the cross product of the model's pair of lines in α-space and β-space. We can see that there is an invariant function with no false positives for a specific set of models if and only if no two planes that correspond to two models intersect without completely coinciding. If this condition is met, we may construct an invariant function that assigns a common value to all the images that fall on a model's plane of images, and assigns a different value to any two images that lie on the planes of different models. Then, any model that can produce one image with a particular value of the invariant function will be able to produce exactly the set of images that have that value of the invariant function. If, on the other hand, two planes intersect and do not coincide, then all images that either plane contains must have the same value for any invariant function, but neither object can produce all these images, so false positives will occur.

From this result we can see that a specific set of models can have an invariant function with no false positives only if it is a measure 0 subset of the set of all possible models. There is a correspondence between the set of all possible models and the set of all planes in affine space, subject to the restriction that for each plane, the directional vectors of the two lines it produces when projected onto α-space or β-space must be the same. Any set of such planes that do not intersect is a measure 0 subset of the set of all such planes. For example, if we have a restricted set of models corresponding to a set of non-intersecting planes in affine space, this means that any point in affine space can belong to only one of these planes, although it belongs to uncountably many planes that correspond to some model.

An especially interesting special case is that of models containing five points. This is the smallest group that can produce invariants, because in general any four model points can appear as any four image points. Most systems based on invariants have used the smallest possible model groups, in order to limit the number of possible model groups they must consider (this is true of [17], [10], [25], for example). A set of model groups of five points will each produce a pair of lines in 2D α-space and β-space. Furthermore, recall that these two lines will have the same slope, which means that two different models will produce lines that are either parallel in both spaces, or that intersect in both spaces. Therefore, an invariant function for groups of five points is possible only when all lines produced by all

models are parallel. For example, if one model produces lines not parallel to the others, it will have an image in common with each of them, implying that the invariant function must be constant over all images produced by all models. The lines produced by all models will be parallel only when r_5, (see equation 1), is the same for all models, that is, when the ratio of the height above the model plane of the fourth point to the height of the fifth point is always the same.

We now turn to the analysis of some non-accidental properties. Lowe[19] first used these in his recognition system, SCERPO, and Biederman[1] has based his GEON approach to recognition on these properties. A non-accidental property is a property of an image such that some models only produce images with that property, while for other models either no or almost no images they produce have that property. The work of Lowe and Biederman has been based on identifying a small number of such properties, such as parallelism, colinearity, or symmetry. Our above work now allows us to see that there is an infinite class of such properties, and that there is nothing about the inherent geometry of models which makes one such property qualitatively different from another.

First, we illustrate this in the simplest case of colinearity. We will show that there are an infinite set of properties that are qualitatively similar to colinearity. That is, if we ignore the distribution of models in our universe of possible models, there is nothing we can say about colinearity that we can not also say about an infinite number of other properties. Suppose we have an image group with five points, and the first point and last three points are colinear. This is equivalent to saying that α_4 and α_5 of the image's affine coordinates both equal 0. That is, any image which corresponds to the point $(0,0)$ in α-space, and to any point in β-space, has those four points colinear. If we look at this from the perspective of non-accidental properties, we say that either the four points are colinear in the model that produced them, in which case they always appear colinear, or they are not really colinear, in which case there is only a measure 0 chance that they would appear as colinear. We would then infer that we should match these colinear image points with only similarly colinear model points.

From our new perspective, we would say that there are two possibilities. Either the model that produced this image is planar, and always produces images with the values $(0,0)$ in α-space, or the model corresponds to a line in α-space that passes through the origin. These two views of the situation are equivalent. However, we may now see that there is nothing special about the coordinates $(0,0)$ in α-space. We could say the same thing about any other coordinates. For example, we get a similar non-accidental property when

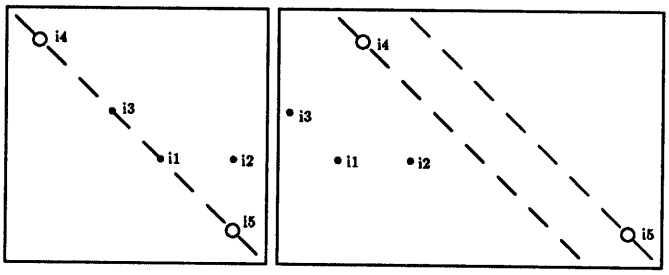

Figure 2: If an image corresponds to the point $(0,0)$ in α-space, that means that the last two points in the image are colinear with the first and the third points. This is shown on the left, where the first three points are shown as dots, and the second two points, shown as open circles, must lie somewhere on the dashed line. If the image corresponds to $(2,3)$ in α-space, the points must fall on the two lines shown on the right. In both cases we have an equivalent non-accidental property.

we consider an image with affine coordinates $(2,3)$ in α-space (see figure 2). Again, either the object that produces such an image is planar, and always produces such an image, or the object corresponds to a line in α-space that passes through the point $(2,3)$. So the α-coordinates $(2,3)$, and any other pair of α-coordinates, define a new non-accidental property, and there is no qualitative difference between these properties and colinearity that can be inferred from the fundamental geometry of objects.

More generally, other non-accidental properties, such as parallelism or skew symmetry, can also be thought of as describing regions in affine space. The non-accidentalness of these properties is equivalent to saying that a planar object that produces these properties from one viewpoint will always produce them, while a non-planar object that can produce the property will be equivalent to a pair of lines that intersects the property in only a measure 0 portion of its extent. If two properties correspond to similarly shaped regions in affine space, then there will be no qualitative difference between them.

This argument does not imply that non-accidental properties are not useful, or that the non-accidental properties that have been proposed are not more useful than others. It simply suggests that the special perceptual saliency of some non-accidental properties, such as colinearity, lies with the particular nature of objects that occur in the world, and can not be explained by some qualitatively different characteristic of the geometry of these properties.

4 Handling Error

Our previous discussion provides us with a basis for building an indexing system in the absence of image error. We now consider what happens when we allow bounded error. We assume that there is some uncertainty in the location of each image point, which can be bounded by a disc of radius ϵ. To account for this error we have two possible choices. First, we could attempt to represent in the lookup table the set of all images that a particular model could produce, allowing for error. Then at run-time when confronted with a particular image, we would look in the table at a single location to find the models that could produce that image. This approach has serious problems, however. We found that in the absence of error, the images a model can produce can be described without reference to the location of the first three image points. We were therefore able to ignore them, and represent images in affine space. However, as we will see below, the effect of error on these affine parameters depends a great deal on the location of the three image points used as a basis. Therefore, a description of the images a model produces, with error, would have to include a description of the locations of the three basis points. This would result in a model's images forming a higher-dimensional surface. Moreover, error turns each image into a volume of possible images in any space that represents the images. We could not discretely represent this with a reasonable amount of memory. This issue is discussed further in [6] and [11].

So instead, we choose to consider error at run-time. This means our lookup table is built without taking account of error, as described above. Then, for a particular image, we determine all sets of affine coordinates that are compatible with that image and with our error model. We access the lookup table in two volumes, instead of at two points, and intersect the results to find all the models that are consistent with the image.

First, we consider the following problem. Given four image points, $\mathbf{i_1}, \mathbf{i_2}, \mathbf{i_3}, \mathbf{i_4}$, and error bounded by ϵ, what affine coordinates might the fourth image point have had with respect to the other three, before the error was introduced? In [14] and [11] we solved this problem, showing that the the answer was any set of affine coordinates, (α, β) satisfying the equation:

$$||(\mathbf{i_1} + \alpha(\mathbf{i_2} - \mathbf{i_1}) + \beta(\mathbf{i_3} - \mathbf{i_1})) - (\alpha_4(\mathbf{i_2} - \mathbf{i_1}) + \beta_4(\mathbf{i_3} - \mathbf{i_1}) + \mathbf{i_1})||$$

$$\leq \epsilon(|1 - \alpha - \beta| + |\alpha| + |\beta| + 1)$$

where (α_4, β_4) stand for the affine coordinates of $\mathbf{i_4}$ with respect to $(\mathbf{i_1}, \mathbf{i_2}, \mathbf{i_3})$, as usual. In

most cases this equation describes an ellipse in a 2D affine space.

For each image point but the first three, we use this equation to determine separately the set of α and β values consistent with that point. Combining all these bounds produces two independent rectanguloids in α-space and β-space. We know that before error corrupted the image, the α and β coordinates of the image must have fallen inside these rectanguloids. This method is conservative, in that it overstates the effects of error in two ways. In the above equation, the effect of error on the α and β coordinates is not independent. But we separately determine the ranges of possible α and β values that can occur, without accounting for the fact that each α value is consistent with only some of the β values, and vice versa. Secondly, we treat the error as if it had an independent effect on each pair of affine coordinates. However, error in the three basis points has a related effect on the affine coordinates of every other point. In effect, we enclose the true volume of affine space consistent with error inside the smallest possible rectanguloid that has sides parallel to the axes of the affine space. While this may produce some unnecessary matches, it can not cause us to miss any correct matches.

5 Determining the Projected Model

Our indexing system produces matches between groups of image features and groups of model features. While there are standard techniques for verifying such hypotheses, our representation of models lends itself to a particularly simple method of determining the projection of the model into the image.

A match produced by the indexing system provides us with lines in α-space and β-space that represent the model, and a pair of points in these spaces for the image. By finding the locations on the lines that are closest to these two points, we find the error-free set of affine coordinates that best fit our image. We then can use any one of these coordinates to determine the affine coordinates of any point on the projected model, since, as we pointed out above, all further pairs of affine coordinates are functions of any single pair of affine coordinates.

Let us illustrate this with an example. Suppose indexing matches model points $\mathbf{m_1}$, $\mathbf{m_2}$, $\mathbf{m_3}$, $\mathbf{m_4}$, $\mathbf{m_5}$, $\mathbf{m_6}$ to image points $\mathbf{i_1}, \mathbf{i_2}, \mathbf{i_3}, \mathbf{i_4}, \mathbf{i_5}, \mathbf{i_6}$, and suppose based on this match we wish to project model points $\mathbf{m_1}, \mathbf{m_2}, ...\mathbf{m_n}$ into the image. At compile time, we use the points $\mathbf{m_1}, \mathbf{m_2}, \mathbf{m_3}$ as a basis, and compute the two lines in the affine spaces that describe all images of the model when the image of these three points are used as a basis. (This is

done at compile time for all possible basis triples). Call these two lines L_1 and L_2. These lines are in $(n-3)$-dimensional affine spaces, $(\alpha_4 ... \alpha_n)$ and $(\beta_4 ... \beta_n)$, because they represent the locations of $n-3$ points using the first three points as a basis. Our six matched image points map to two points in the 3-dimensional affine spaces $(\alpha_4, \alpha_5, \alpha_6)$ and $(\beta_4, \beta_5, \beta_6)$. Call these points $\mathbf{p_1}$ and $\mathbf{p_2}$. By projecting L_1 and L_2 into these lower dimensional spaces, we get lines that describe the possible images that the first six model points can create. By finding the point on the projection of L_1 closest to $\mathbf{p_1}$ we find the α coordinates of the image of the model that best match the image points. Similarly, we find the appropriate β values. These values determine locations on L_1 and L_2 that tell us the affine coordinates of all the model points in the image that will best fit the matched image points. Without explicitly computing the viewing direction we have computed the appearance of the model, when seen from the correct viewing direction. (A different method must be used if the matched model points are coplanar, because in that case their affine coordinates provide no information about viewing direction).

In addition to determining the effects of the viewing direction on the image, we must also allow for the effects of the affine transformation portion of the projection. However, once we have determined the affine coordinates of all the projected model points, it is straightforward to apply a least squares method to find the affine transformation that optimally aligns the image points with the projected model points.

6 Experiments with an Indexing System

6.1 The Recognition System

Our indexing system is designed to match relatively large groups of image points to equally large groups of model points. Such an approach requires a bottom-up grouping process to control the combinatorics of forming all possible large groups of image and model points ([6] discusses this issue at length). While we are currently interfacing this system with a particular grouping system, our purpose here is to examine the performance of the indexing system alone. So that our results will be independent of the deficiencies of any specific grouping system, we have tested the indexing system with somewhat ideal groups, in which automatically located features are formed into groups by hand.

We begin model building by running an edge detector[4] on many images of the object. We find corner features by making straight line approximations to the edges[21], and locating a corner where nearby lines have a stable intersection point, when extended. We

then form by hand groups of three to five points that are formed by a set of convex lines (see [13] and [12] for discussion of the value of convex groups). The convexity of the group orders the points, although it does not tell us which point comes first. So a different group is formed for each possible starting point. To allow for the effects of occlusion, we also form groups in which any one of the points is omitted, as long as the group still contains at least three corners. Corners that appear in these groups are matched by hand between the images used to build a model.

Since three to five points do not provide sufficient information to discriminate between models, we then form all pairs of these groups. Each pair of groups gives us an ordered set of points. For each set of points, we calculate the lines in α-space and β-space to which they correspond. Each image that contains all the corners appearing in the group is used to calculate two points, one in α-space and one in β-space, that describe the ordered group. We then fit lines to these points, to determine the set of all images that the group could produce.

We then compute which buckets of discretized α-space and β-space these lines intersect, and make an entry in each bucket, pointing to the appropriate group of model points.

To build a model of the object's line segments we match by hand corners that are at the end points of line segments that appear in a number of images of the model. Then, for every triple of corners used as a basis for one of the groups entered into the lookup table, we form lines in the two affine spaces representing the possible affine coordinates of the corners of all the model's line segments. We use a new hash table to store these pairs of lines for easy access.

At run time, we take a new picture of the model, along with occluding objects. We form groups from the corners of this picture, just as we did for the images used to build the model. These groups may be missing some of the corners that appear in the modeled groups, due to occlusion or due to failure in the corner finder. Next, as before, we form all pairs of these groups. We have some freedom in how we order the points in these pairs of groups before using them for lookup. In building the model, each point in a group was used as a starting point for that group. That point, and the next two, were used as a basis for computing the affine coordinates of the remaining points. Since we may use any point in the image group as a starting point, we select the point which gives us the most stable set of basis points. Then, we use this ordered set of image points to compute two rectanguloids in the affine spaces, as described above, and find all matching sets of model points. Since we can only represent a finite portion of the affine spaces, it is possible that

the rectanguloids produced by a group will fall outside the bounds of this portion of affine space. We ignore such groups, and in fact, groups that produce large affine coordinates are likely to be unstable, providing poor candidates for matching.

We perform this indexing for each pair of image groups. Ordering these pairs so that we start with the pair that produces the fewest matches, we then perform verification on each match until the object is found. We use the method described above to determine the projection of the model's line segments for each match. We then search the image for line segments that are near the model's projected line segments, and that have roughly the same orientation, in order to determine the fraction of the model which the image can explain. We stop when a sufficiently good hypothesis is found.

Note that in general, when determining the appearance of a model from a given viewpoint, we should eliminate lines that are not visible from that viewpoint. To avoid the need for this and simplify our verification system, we have taken all images from a single aspect of the object. That is, we restricted our viewpoint to about a quarter of the viewing hemisphere, in which all the same set of points and lines were visible.

6.2 Experiments

In experimenting with the above recognition system, we have used the following values for various parameters. We allowed image error of five pixels in indexing. The index table represents all affine coordinates between -25 and 25. Each dimension of the table is divided into 100 intervals. We make the buckets of uniform size for affine coordinates between 0 and 1, and then increase the bucket size linearly for coordinates above 1 or below 0. We do this because error has a greater effect on images with higher affine coordinates, at approximately this rate. A projected model line is matched to an image line if the image line is no more than 10 pixels from the model line, and if the orientations of the lines differ by no more than $\frac{\pi}{10}$. A hypothesis is accepted if it accounts for at least 50% of the model's lines.

We have performed two sets of experiments. In one, we build models and recognize them, as described above. In the second, we perform indexing using randomly generated models and images. This provides a means of more carefully measuring the discriminatory power of our system.

Figure 3 shows the edges found in two pictures of a telephone, and 15 of the corners that are located from these edges. The corners are numbered for reference. We form groups containing the following sets of corners: ((0 1 2 3 4) (1 2 3 13) (1 0 9 10) (14 15 16 18)

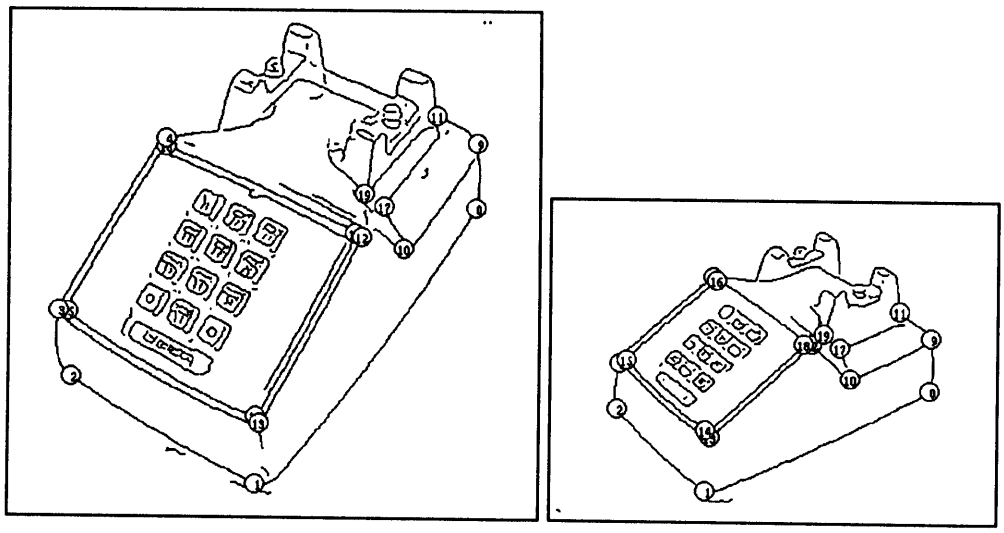

Figure 3: Edges from two of the pictures used to build a model of the phone. Circles indicate the location of automatically found corners that were then selected by hand to be in the model. They are numbered for reference.

(11 9 10 17) (12 13 3 4) (11 17 19))), and for verification we use line segments with the following corners as end points: ((0 1) (1 2) (2 3) (3 4) (0 9) (9 10) (9 11) (11 17) (10 17) (3 13) (13 12) (12 4) (14 15) (15 16) (16 18) (18 14))). Out of these primitive groups, we formed pairs of model groups for entry in the index table, as described above. There were an average of 472 table entries for each pair of groups.

Figure 4 shows pictures containing the telephone, and the first hypothesis of the recognition system as to the location of the phone. For both pictures, the first hypothesis is correctly accepted. In all, the system was tested on three scenes. Fourteen groups of corners were formed from these images. Indexing for two of the groups required going outside the bounds of the lookup table, so these groups were ignored. Indexing produced a correct match in for all of the remaining twelve groups. Five groups contained seven corners each, and seven groups contained six corners each. The groups with seven corners produced a total of two incorrect matches in addition to the correct matches. Each group could potentially be matched to any of 1,386 groups represented in the lookup table that contained seven points. So indexing reduced the number of incorrect matches by a factor of 3,465. The groups with six corners produced an average of 16.7 incorrect matches, compared to the 2,931 groups in the table they could match. This implies an average speedup of a factor of 244. We can see that indexing can produce tremendous savings in time in this domain, particularly as larger groups are formed, and when we may choose among several groups,

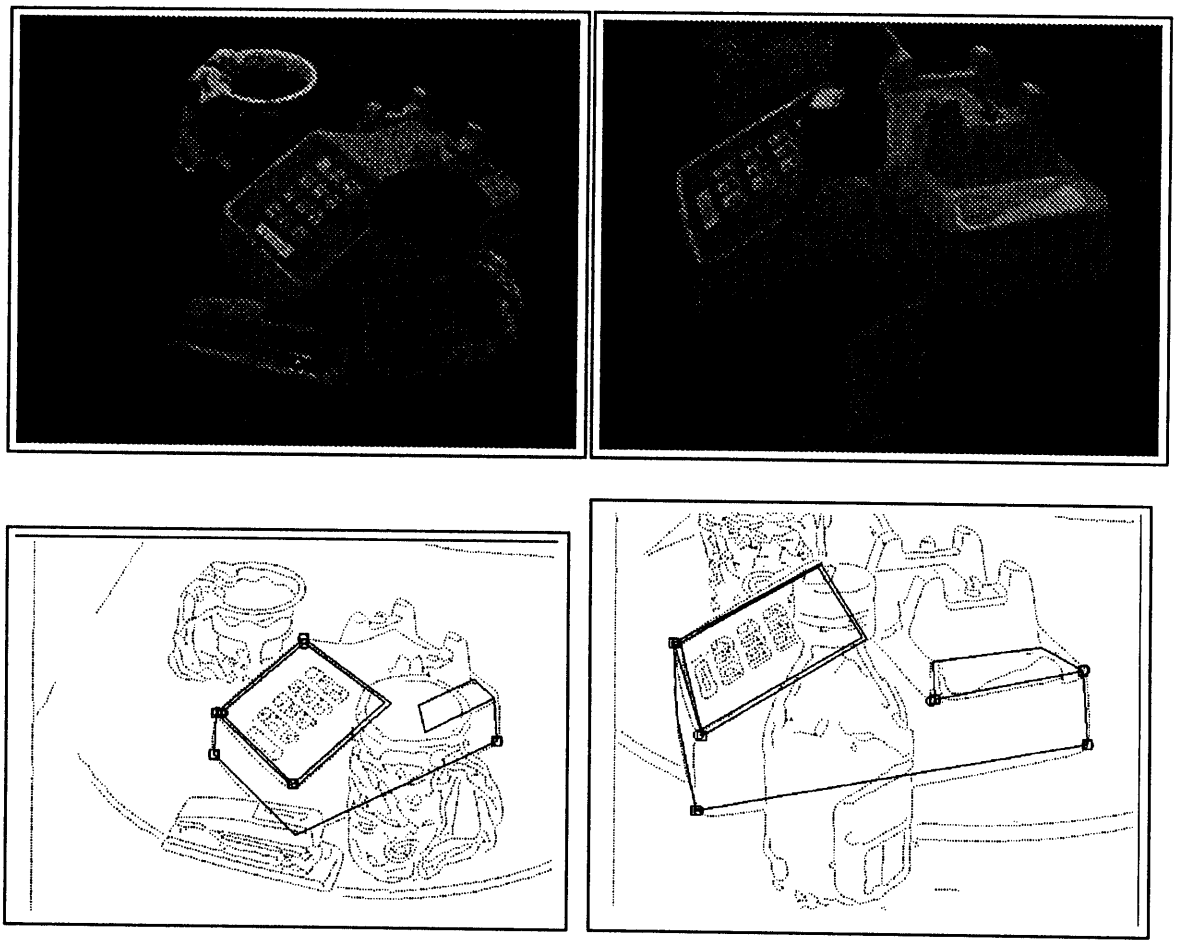

Figure 4: On the top are two scenes containing the phone. Underneath each scene is the first hypothetical projection of the model considered by the recognition system. Both are correct. In the hypotheses, edges are shown as dotted lines, projected line segments appear as lines, circles represent the image corners in the match, and squares show the location of the projected model corners.

Number of Points	Error	Images out of Bounds	Speedup
6	5	.7	170
7	5	.4	7,300
8	5	.01	127,000 <
6	7	1.1	90
7	7	.9	850
8	7	.04	127,000 <
6	9	1.44	49
7	9	.9	340
8	9	.02	7,500

Figure 5: Experiments with randomly generated models and images. Points are divided into two groups, and all orderings of these groups are considered. Error is in number of pixels. We give the percentage of images for which error would require them to access the index table outside its boundaries. These images were ignored. Speedup indicates the ratio of total possible matches to matches produced. "X <" indicates that no matches were found, out of X possible matches.

using the most distinctive group first.

For the synthetic experiments, we generated random model groups by selecting points at random in a cube, and we generated random image groups by selecting points in a square, in which the minimum and maximum distance between the points differed by no more than a factor of 10. We then matched different orderings of these groups using our indexing system, using the same parameters as above. This allowed us to estimate the likelihood that an image group will match a model group by chance, as we vary the size of the group, and the amount of the error. For each set of values we generated 85 model groups, and for each model group we generated 100 image groups. The results are summarized in figure 5. For error of five pixels, these results are similar to our results with real images. They indicate how dramatically speedups increase with group size.

7 Previous Work

Our work is based on a general approach to recognition that links grouping and indexing. In this approach, a bottom-up process forms salient groups in the images, and then matches

them to only those model groups that could have produced them. This approach was first used by [19], and has since been taken in: [13], [22], [12], [27], [6], and [7].

A number of recent indexing systems for recognizing planar objects from arbitrary 3D views have been based on invariant functions of the image [28], [17], [18], [9], [10], [25]. For example, as we have noted, the affine coordinates of a planar model are invariant under affine transformations. [17],[18] use this invariance in their system. [27] describes an indexing system that makes use of invariant descriptions of planar faces of objects, or of otherwise restricted classes of objects for the recognition of 3D models. Invariants allow these systems to represent the image groups that a model group may produce using a single point in index space.

However, as previously mentioned, [3], [5], [6], and [20] have recently shown that there are no non-trivial invariants when models may consist of arbitrary collections of point features. [20] includes further, related results.

[6] describes an indexing system for general 3D models. In that paper, it is proven that in this domain each model must be represented by a 2D surface if a single lookup table is used. (We get around that result in this paper by using two, orthogonal index spaces). The system described in that paper suffered from two problems. First, there was no analytic method known for building or accessing the table. This meant that sampling was used both to determine which images a model could produce, and to determine which portions of the table were consistent with a particular image and the error bounds. To perform sufficient sampling required excessive computation both at compile time and at run time. Second, it took a good deal of space to represent these 2D surfaces. That system required over 5,000 table entries to represent a model group of five points, while our current system requires about 472 table entries for groups containing mostly six or seven points. This comparison may be misleading however. For one thing, space constraints required the system described in [6] to discretize the index space less finely than one would like. Each dimension of that table was divided into 40 parts, compared to 100 parts in the current system. Also in general, space requirements grow as group size grows. Finally, building the lookup table with sampling undoubtedly resulted in many table entries being missed.

[23] had previously used a similarly constructed lookup table to determine the object pose implied by a match between a pair of image vertices and a pair of model vertices. They also built their table by sampling the images a model produces, doing this by sampling the viewing sphere. Their table required approximately 2,500 table entries per vertex pair. Their discussion of interpolation methods to reduce space requirements suggests that they

also found space to be a problem.

[18] also describes a system that samples the viewing sphere, in this case at 10 degree intervals, and then creates a separate model for each view. Then, a 2D indexing scheme based on invariants is used. [2] also proposes an indexing scheme based on sampling the viewing sphere. This method is distinguished by taking careful account of the effects sampling has on error in order to limit the correct matches missed. The previously discussed systems do not account for the fact that from some viewpoints, a small change in viewpoint can produce a large change in the image produced by the model.

[18] also proposes another indexing scheme for 3D model recognition. In this approach, each model is represented at a single point, and each image accesses the index table along a line. This approach assumes the same projection model as [16, 24], which we have noted is equivalent to ours. Accessing the table along a line, however, has the disadvantages of adding to run-time cost in order to save space, and it is not clear how to accurately account for error in such a scheme. Also, this approach applies only when image and model groups contain exactly five points.

In addition to work on indexing, our system is related to the linear combinations idea of [24]. In this paper, a method of projection equivalent to ours is used, and it is shown that any image of an object is a linear combination of two independent images of that object, which is also easily seen from our result. In doing this, [24] shows that the set of x and y image coordinates produced by a model will each occupy a three-dimensional linear subspace of a higher dimensional space describing all possible images. The focus of [24] is on using this result to determine the projection of the model from a new viewpoint, in a manner similar to that described here. For that purpose, the dimensionality of the set of images produced by a particular object was not important. However, the results of [24] are not directly useful for an indexing system, because the high dimensionality of the linear subspace described would require excessive storage space in an index table.

[16] have produced related results for motion analysis. Their analysis shows that using two views of a set of points, we can recover the affine structure of a 3D object. That is, we recover that part of the structure that is not altered by an arbitrary 3D affine transformation. They note that this result allows them to predict the appearance of the object from new views, given the location of four known points in this new view. This result is related to ours in that we make use of the same underlying mathematics to construct the lines representing a model, and to use these lines to predict new views of the model.

Finally, we have suggested that there is significant mathematical convenience in not

distinguishing between an image of an object, and an image of a picture of the object, and that this may explain human ability to interpret pictures. [8] has suggested a related hypothesis. That work suggests that projective invariants of planar objects, such as the cross ratio, are used in image interpretation. Such invariant properties are preserved when a planar model is viewed with perspective projection. When a picture of a model is viewed, two perspective projections have occurred, and so the projective invariants are still preserved.

8 Conclusions

We have presented a general method for representing 3D models of point features in an indexing table in order to quickly match them to 2D images. In terms of the dimensionality of the space required, this method is optimal. We also present analytic methods of building and accessing the table which ensure that all correct matches are found. Together, we use these results to build a fast and robust recognition system.

In addition, we feel that this work is valuable because it reduces recognition to an extremely simple geometric problem. We show that at its core, the problem of recognizing an object in the absence of error is equivalent to determining which lines a point falls on. When error is present, recognition is equivalent to finding which lines intersect a volume. We use this view of recognition to produce a novel analysis of non-accidental properties, and to answer some outstanding questions about invariants.

References

[1] Biederman, I., 1985. "Human Image Understanding: Recent Research and a Theory". CVGIP (32):29-73.

[2] Breuel, T., 1990. "Indexing for Visual Recognition from a Large Model Base." MIT AI Memo 1108.

[3] Burns, J., Weiss, R., and Riseman, E., 1990. "View Variation of Point Set and Line Segment Features." *DARPA IU Workshop*:650-659.

[4] Canny, J., 1986. "A Computational Approach to Edge Detection." *IEEE Trans. PAMI*, 8(6):679-698.

[5] Clemens, D. and Jacobs, D., 1990. "Model-Group Indexing for Recognition." *DARPA IU Workshop*:604-613.

[6] Clemens, D. and Jacobs, D., 1991. "Space and Time Bounds on Model Indexing." *IEEE Trans. PAMI*,13(10):1007-1018.

[7] Clemens, D., 1991. *Region-Based Feature Interpretation for Recognizing 3D Models in 2D Images.* MIT AI TR–1307.

[8] Cutting, J., 1986. *Perception with an Eye for Motion.* Cambridge, MA, USA: MIT Press.

[9] Cyganski, D. and Orr, J., 1985. "Applications of Tensor Theory to Object Recognition and Orientation Determination." *IEEE Trans. PAMI*, 7(6):662–673.

[10] Forsythe, D., Mundy, J., Zisserman, A., and Brown, C., 1990. "Invariance: A New Framework for Vision." *IEEE Proc. ICCV*:598-605.

[11] Grimson, W., Huttenlocher, D., and Jacobs, D., 1991. "Affine Matching with Bounded Sensor Error: A Study of Geometric Hashing and Alignment." MIT AI Memo 1250.

[12] Huttenlocher, D. and Wayner, P., 1991. "Finding Convex Edge Groupings in an Image." *IEEE Proc. CVPR*:406-412.

[13] Jacobs, D., 1989. "Grouping for Recognition." MIT AI Memo 1177.

[14] Jacobs, D., 1991. "Optimal Matching of Planar Models in 3D Scenes." *IEEE Proc. CVPR*:269-274.

[15] Kalvin, A., Schonberg, E., Schwartz, J., and Sharir, M., 1986. "Two-Dimensional, Model-Based, Boundary Matching Using Footprints." *Int. J. of Robotics Research*, 5(4):38-55.

[16] Koenderink, J. and van Doorn, A., 1991. "Affine Structure from Motion." *J. of the Optical Soc. of America*, 8(2):377-385.

[17] Lamdan, Y., Schwartz, J. and Wolfson, H., 1988. "Object Recognition by Affine Invariant Matching." *IEEE Proc. CVPR*:335-344.

[18] Lamdan, Y. and Wolfson, H., 1988. "Geometric Hashing: A General and Efficient Model-Based Recognition Scheme." *IEEE Proc. Robotics and Automation*:238-249.

[19] Lowe, D., 1985. *Perceptual Organization and Visual Recognition.* The Netherlands: Kluwer Academic Publishers.

[20] Moses, Y. and Ullman, S., 1991. "Limitations of Non Model-Based Recognition Schemes." MIT AI Memo 1301.

[21] Pavlidis, T. and Horowitz, S., 1974. "Segmentation of Plane Curves." *IEEE Trans. on Comp.*, C(23):860-870.

[22] Sha'ashua, A. and Ullman, S., 1988. "Structural Saliency: The Detection of Globally Salient Structures Using a Locally Connected Network." *IEEE ICCV*:321-327.

[23] Thompson, D. and Mundy, J., 1987. "Three-Dimensional Model Matching from an Unconstrained Viewpoint." *IEEE Proc. Robotics and Automation*:208-220.

[24] Ullman, S. and Basri, R., 1991. "Recognition by Linear Combinations of Models." *IEEE Trans. PAMI*,13(10):992-1007.

[25] Van Gool, L., Kempenaers, P., and Oosterlinck, A., 1991. "Recognition and Semi-Differential Invariants." *IEEE Proc. CVPR*:454-460.

[26] Wallace, A., 1987. "Matching Segmented Scenes to Models Using Pairwise Relationships Between Features." *Image and Vision Computing*, 5(2):114-120.

[27] Wayner, P., 1991. "Efficiently Using Invariant Theory for Model-based Matching." *IEEE Proc. CVPR*:473-478.

[28] Weiss, I., 1988. "Projective Invariants of Shape." *DARPA IU Workshop*:1125-1134.