Complexity of Indexing: Efficient and Learnable Large Database Indexing *

Michael Werman and Daphna Weinshall

Institute of Computer Science, The Hebrew University of Jerusalem
91904, Jerusalem, Israel
{werman,daphna}@cs.huji.ac.il

Abstract. Object recognition starts from a set of image measurements (including locations of points, lines, surfaces, color, and shading), which provides access into a database where representations of objects are stored. We describe a *complexity theory of indexing*, a meta-analysis which identifies the best set of measurements (up to algebraic transformations) such that: (1) the representation of objects are linear subspaces and thus easy to **learn**; (2) direct indexing is **efficient** since the linear subspaces are of minimal rank. The index complexity is determined via a simple process, equivalent to computing the rank of a matrix. We readily re-derive the index complexity of the few previously analyzed cases. We then compute the best index for new cases: 6 points in one perspective image, and 6 directions in one para-perspective image; the most efficient representation of a color is a plane in $3D$ space. For future applications with any vision problem where the relations between shape and image measurements can be written down in an algebraic form, we give an automatic process to construct the most efficient database that can be directly obtained by learning from examples.

1 Introduction

Any recognition system starts with a finite list of n measurements computed on the image. The vector of n measurements defines the indexing space $\mathcal{R}^n$, and each image corresponds to a point in $\mathcal{R}^n$. Since objects appear in many different forms in different images (due to changes in viewing position, illumination, etc), each object is mapped to a collection of points. This collection, the set of all points corresponding to a feasible image of the object, is the object's *picture manifold* in the *indexing space*.

The particular form of an object's *picture manifold* is very important for object recognition, whether index-based or search-based:

Direct indexing: in the simplest approach each object's *picture manifold* is stored in the database. Now given a set of measurements computed from an image, which is a point in $\mathcal{R}^n$, we use the point as an index into a table,

* This research was supported the Israeli Ministry of Science under Grant 032.7568, and by the U.S. Office of Naval Research under Grant N00014-93-1-1202, R&T Project Code 4424341—01.

pointing to a cell where the identity of the observed object is written (e.g., geometric hashing [4]).

Search: If we have parametric representations of the picture manifolds, we can efficiently store those representations. Given a set of measurements computed from an image, we search the database to find the picture manifold which covers the measurements point in $\mathcal{R}^n$ (e.g., alignment [2]).

The first approach is the most time-efficient since no search is required. However, clearly the form of the *picture manifold* is critical here: if it is too complex, it may not be feasible to represent this picture manifold in memory. On the other extreme, the second approach is space-efficient since only a parametric representation of the picture manifold is stored. However, recognition requires search, and may therefore take too long to be useful.

For an optimal system we probably need to consider a hybrid of the 2 approaches to object recognition. In order to do that, we need to better understand the form of objects' *picture manifold*. Finally and most importantly, we need to be able to manipulate the measurements defining the indexing space, optimizing for simplicity and efficiency of the picture manifold at the same time. That is, we seek a transformation of the measurements such that:

Simplicity- the picture manifold is a linear subspace, a manifold which is both easy to represent and easy to learn from examples.

Efficiency- the picture manifold is of a low rank.

Indexing spaces where objects' picture manifolds are both simple and low dimensional can be used in recognition systems which are based on either one of the approaches described above. Defining and computing good indexing spaces is the subject of the present paper.

1.1 Previous work:

The concepts we study here are closely related to the concept of invariants, which is a subject of much interest in computer vision and pattern recognition. To analyze the picture manifold, we need to start with a relation between the vector of measurements and the object. This relation should not depend on variables of the imaging process, such as the camera orientation (viewing position) or the color of the illumination source; thus we need an invariant relation between image measurement and objects.

Invariant relations were divided in the literature into 2 types:

Model-free invariant: there exist image measurements which identify the object uniquely, so that their value is completely determined by the object regardless of the details of the imaging process. This is what is usually called an invariant in the literature. Such invariant relations do not exist for $2D$ images of general $3D$ objects, but for special classes of objects, such as planar or symmetrical objects, invariant relations may be found [6, 8].

Model-based invariant: a relation which includes mixed terms, representing image measurements or model parameters. Model-based invariant relations exist for many interesting vision problems [9].

If a model-free invariant is used to construct the database (the indexing space), then the picture manifold of an object is a point. For example, geometric hashing [4] uses the class of planar objects composed of features, where model-free invariants exist and therefore the picture manifolds are points. If on the other hand a model-based invariant is used, the picture manifold may be a complex hypersurface in the indexing space. For example, Murase & Nayar [7] use a very general class of objects and illumination conditions, and therefore they have to represent picture manifolds which take on rather complex shapes.

Clearly, for object recognition using either direct indexing or search, it is better to use an indexing space where the picture manifolds are as simple as possible (for example, hyperplanes rather than quadrics), and have as low a rank as possible. Jacobs [1, 3] defined the concept of indexing space, and studied the complexity of picture manifolds. However, Jacobs did not address the manipulation of the indexing space. Achieving this goal, obtaining linear manifolds that are easy to learn from examples while maintaining low rank (efficiency), is a major contribution of our present study.

1.2 Description of our approach:

We start from a general vision problem, with one or more cameras, where the image measurements include points, lines, or surfaces, graylevels, color and texture. For each such problem we find an optimal set of transformed measurements, to define an indexing space where the picture manifolds are linear (hyperplanes) and of the lowest possible rank (among all linear picture manifolds). The lowest rank of the picture manifold is called the **index complexity** of the vision problem.

At the end we get the best linear relation between objects and image measurements. Objects are then represented by low rank hyperplanes. We advocate the use of linear representations of objects even at a modest cost of space efficiency for the following reasons: (1) Hyperplanes can be learned (or interpolated) robustly and efficiently from a small number of examples; this is NOT the case for general manifolds. Moreover, parametric representations of general manifolds may not even be computable (cf. [7]). (2) Using noisy measurements, it is easy to find the distance of a measurement point from a hyperplane (unlike general manifolds), which is the problem that an object recognition algorithm needs to solve.

Our approach allows us to find class constraints, on the permitted set of objects, such that the index efficiency is increased (and the rank of the picture manifold is decreased). With sufficient class constraints we can reduce the rank of the picture manifold to 0, achieving the most efficient indexing possible via model-free invariants. This was the goal of a few studies [6, 8]; these studies gave examples of class constraints but did not offer a general way to generate them, like we do here.

The rest of this paper is organizes as follows: In Section 2 we define the concept of index complexity, which is the rank of the object's picture manifold in the indexing space. We show how to compute the lowest complexity given a

particular vision problem. The lowest complexity of an index is simply the rank of a matrix, which we define and call the complexity-matrix. Many examples are discussed in Section 3. In Section 4 the addition of class constraints, for the purpose of reducing index complexity, is explained.

2 Complexity of indexing

A vision problem, describing a set of images, is defined by 3 types of variables:

1. $\mathbf{T}$ denotes the set of parameters describing the unknown **imaging variables**, including camera transformation and color of illumination.
2. $\mathbf{S}$ denotes the set of parameters describing the **object shape**, where shape denotes both geometrical and physical features (including color and shading).
3. $\mathbf{D}$ denotes the data - a set of **image measurements**.

Each image provides a number of relations between these variables, of the form $F(\mathbf{T}, \mathbf{S}, \mathbf{D}) = 0$; for example, an image of 1 point provides 2 such relations, one for the x coordinate and one for the y. The vision problem is fully described by the equations, relating these variables over the given images:

$$\mathcal{V} = \{F_l(\mathbf{T}, \mathbf{S}, \mathbf{D}) = 0\}_{l=1}^{M}$$

2.1 Database indexing:

Each vision problem $\mathcal{V}$ produces an ideal $\mathcal{I}$, which is the set of all the algebraic relations between the image measurables and the model shape. $\mathcal{I}$ is generated by a set of relations, obtained from the relations in $\mathcal{V}$ by eliminating the imaging variables $\mathbf{T}$. Thus:

$$\mathcal{I} = \{f_l(\mathbf{S}, \mathbf{D}) = 0\}_{l=1}^{L}$$

$L \leq M$. The ideal $\mathcal{I}$ includes precisely all true relationships between the model parameters $\mathbf{S}$ and the image measurements $\mathbf{D}$. The polynomials in $\mathcal{I}$ do not include the imaging variables $\mathbf{T}$. We will often not distinguish between the ideal and its set of generators. The ideal is the smallest set including the generators closed under addition and multiplication by polynomials in $\mathbf{S}$ and $\mathbf{D}$.

[11] describes a general elimination method to obtain the set of model-image relations from the vision problem using Gröbner bases (which is equivalent to Gauss elimination in linear systems).

2.2 Complexity of indexing:

We start by analyzing the simple case where the set of possible model-image relations $\mathcal{I}$ of our vision problem $\mathcal{V}$ includes a single relation $I : f(\mathbf{S}, \mathbf{D}) = 0$. The number of relations is often chosen to be one, as this requires the smallest number of constraints in the vision problem so that the imaging variables can be eliminated. Moreover, taking a larger set of equations can result in a combinatorial blowup. We define the complexity of I, a measure of the space and

time needed to implement the database index obtained from it. As explained in the introduction, the complexity of an index reflects the size of the smallest hyperplane that needs to be stored in the database, so that the object can be retrieved by the index. In the appendix we define and compute joint index complexity when multiple indices are available.

To begin with, we rewrite the invariant relation I in the following compact way, explicitly separating image variables $\mathbf{D}$ from shape variables $\mathbf{S}$:

$$f(\mathbf{S},\mathbf{D}) = \sum_{k=1}^{r+1} g_k(\mathbf{S}) * h_k(\mathbf{D}) = 0 \tag{1}$$

where g_k and h_k are polynomial functions of the shape $\mathbf{S}$ and the image $\mathbf{D}$ respectively. We call (1) the canonical representation of $f(\mathbf{S},\mathbf{D})$. Note that if $f(\mathbf{S},\mathbf{D})$ is algebraic, as we assume here, this decomposition always exists.

Definition 1 Index. The index, which includes all the image measurables in the invariant relation (1), is the r-dimensional vector:

$$[\frac{h_1(\mathbf{D})}{h_{r+1}(\mathbf{D})}, \ldots, \frac{h_r(\mathbf{D})}{h_{r+1}(\mathbf{D})}] \in \mathcal{R}^r \tag{2}$$

The indexing space is therefore $\mathcal{R}^r$. The manifold of all the possible images of object $\mathbf{S}$ is the hyperplane of rank $(r-1)$ defined by $\sum \frac{h_i(\mathbf{D})}{h_{r+1}(\mathbf{D})} \frac{g_i(\mathbf{S})}{g_{r+1}(\mathbf{S})} = -1$

In the simplest case where $r = 1$, we have:

$$f(\mathbf{S},\mathbf{D}) = h_1(\mathbf{D})g_1(\mathbf{S}) + h_2(\mathbf{D})g_2(\mathbf{S}) = 0 \quad \Rightarrow \quad \frac{h_1(\mathbf{D})}{h_2(\mathbf{D})} = -\frac{g_2(\mathbf{S})}{g_1(\mathbf{S})}$$

This means that the object $\mathbf{S}$ has a model-free invariant: the database is a 1-dimensional table, where each object is represented by a point (cell) in the table, and the cell's value is $-g_2(\mathbf{S})/g_1(\mathbf{S})$. The image provides a direct access to the table via $h_1(\mathbf{D})/h_2(\mathbf{D})$.

More generally, the index given in (2) is an element of $\mathcal{R}^r$. If $r = 1$ it is a model-free invariant; if $r > 1$ it is a model-based invariant as defined in [9]. The database now is an r-dimensional table, and a pointer to object $\mathbf{S}$ is stored in all the cells of the table that satisfy (1). In order to achieve efficient recognition, we would like to find a representation of the form (1) with the smallest r.

Definition 2 Index Complexity. The index complexity $\mathcal{C}$ of a relation $f(\mathbf{S},\mathbf{D})$ is the rank of the smallest hyperplane that the relation defines. Thus if r is the smallest number of terms such that:

$$f(\mathbf{S},\mathbf{D}) = \sum_{k=1}^{r+1} g_k(\mathbf{S}) * h_k(\mathbf{D}) = 0 \tag{3}$$

then $\mathcal{C} = r - 1$.

When the index ideal $\mathcal{I}$ of a vision problem has a single relation in it, $\mathcal{C}$ is also the index complexity of the vision problem.

2.3 Computing index complexity:

We can always write the algebraic expression $f(\mathbf{S}, \mathbf{D}) = 0$ as a sum of multiplications; let

$$f(\mathbf{S}, \mathbf{D}) = \sum_{i=1}^{n} \sum_{j=1}^{m} m_{ij} s_i d_j = \mathbf{s} \cdot M \cdot \mathbf{d}^T = 0 \tag{4}$$

where s_i and d_j are distinct products of element of $\mathbf{S}$ and $\mathbf{D}$ respectively, $\mathbf{s} = [s_1, \ldots, s_n]$, $\mathbf{d} = [d_1, \ldots, d_m]$, and M is the $n \times m$ matrix whose elements are m_{ij}.

Definition 3. M is the **complexity-matrix** of a relation $f(\mathbf{S}, \mathbf{D}) = 0$.

Theorem 4. *The minimal representation of* $f(\mathbf{S}, \mathbf{D}) = \sum_{k=1}^{r+1} g_k(\mathbf{S}) * h_k(\mathbf{D})$ *has* $r + 1$ *terms where* $r + 1$ *is equal to the rank of the complexity-matrix* M.

Proof. The theorem follows from (4), the fact that elementary operations on the rows and columns of a matrix are algebraic operations, and that the rank of a matrix is the minimal number of outer products of vectors that sum to the matrix.

Algorithm to compute the complexity $\mathcal{C}$ and the corresponding index:

1. Compute the SVD decomposition of $M = U\Sigma V^T$; the rank of M is equal to the number of non-0 elements in the diagonal matrix Σ, and the complexity is $\mathcal{C} = \text{rank}(M) - 2$.
2. Since by construction $f(\mathbf{S}, \mathbf{D}) = \mathbf{s}U\Sigma V^T \mathbf{d}^T = \mathbf{g}(\mathbf{S}) \cdot \mathbf{h}^T(\mathbf{D})$ where $\mathbf{g}(\mathbf{S}) = [g_1(\mathbf{S}), \ldots, g_{r+1}(\mathbf{S})]$, $\mathbf{h}(\mathbf{D}) = [h_1(\mathbf{D}), \ldots, h_{r+1}(\mathbf{D})]$

   shape parameterization - $\mathbf{g}(\mathbf{S}) = \mathbf{s}U\sqrt{\Sigma}$

   most efficient index - $\mathbf{h}(\mathbf{D}) = \mathbf{d}V\sqrt{\Sigma}$

3 Examples

3.1 Reformulation of known results

We first reformulate the following known results:

1. The index complexity of an affine or perspective image of symmetrical points is -1, namely, the most efficient index does not depend on the model - it only verifies symmetry in the image.
2. The index complexity of a perspective image of 5 coplanar points is 0, in other words, there is a model-free invariant.
3. The index complexity of 5 points, projected with an affine camera, is 1.

3.2 New results

$\mathcal{C} = 3$ 6 points and a perspective camera:

We work out this example in detail because it shows the power and relative simplicity of our method. We start by using homogeneous coordinates to represent the $3D$ coordinates of the 6 points. Since we are working in $\mathcal{P}^3$, 5 points define a basis; we select the first 5 points to be a particularly convenient projective basis, leading to the following representation of the $3D$ shape of the points:

$$P_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} P_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} P_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} P_4 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} P_5 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} P_6 = \begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \\ W_1 \end{pmatrix}$$

Similarly we use homogeneous coordinates to represent the projected $2D$ coordinates of the 6 points. Since we are working in $\mathcal{P}^2$, 4 points define a basis; we select the first 4 points to be the projective basis, leading to the following representation of the image of the points:

$$p_1 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} p_2 = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} p_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} p_4 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} p_5 = \begin{pmatrix} a_0 \\ b_0 \\ c_0 \end{pmatrix} p_6 = \begin{pmatrix} a_1 \\ b_1 \\ c_1 \end{pmatrix}$$

Given any image of the 6 points, we can always compute the $2D$ projective transformation which will transform the points to the representation given above.

With this choice of coordinates, the relation between the $3D$ shape and the image measurements is the following:

$$\begin{pmatrix} \alpha & -\delta & -\delta & \delta \\ -\delta & \beta & -\delta & \delta \\ \alpha & \beta & \gamma & \delta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & X_1 \\ 0 & 1 & 0 & 0 & 1 & Y_1 \\ 0 & 0 & 1 & 0 & 1 & Z_1 \\ 1 & 1 & 1 & 1 & 1 & W_1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 & a_0 & a_1 \\ 0 & 1 & 0 & 1 & b_0 & b_1 \\ 0 & 0 & 1 & 1 & c_0 & c_1 \end{pmatrix}$$

Using the terminology of Section 2:

1. **T** denotes the set of 4 variables α, β, γ, δ, which represent the camera unknowns that we would like to eliminate.
2. **S** denotes the set of 4 variables X_1, Y_1, Z_1, W_1, - the $3D$ projective coordinates of the 6th point.
3. **D** denotes the set of 6 variables a_0, b_0, c_0, a_1, b_1, c_1; these are the image measurements - the projective coordinates of the points.

Using elimination we get 1 constraint in $\mathcal{I}$:

$$f(\mathbf{S},\mathbf{D}) = a_0 b_1 Z_1 Y_1 + c_1 {Y_1}^2 b_0 - c_1 {X_1}^2 a_0 - a_0 a_1 {Z_1}^2 + a_0 b_1 {Z_1}^2 - b_1 {Y_1}^2 b_0 + c_0 a_1 {Z_1}^2 - c_0 b_1 {Z_1}^2 - b_0 a_1 {Z_1}^2 + b_0 b_1 {Z_1}^2 + a_1 {X_1}^2 a_0 - a_1 Y_1 a_0 X_1 + a_1 Y_1 a_0 Z_1 + c_1 Y_1 b_0 Z_1 - Y_1 a_1 Z_1 c_0 + c_0 a_1 Z_1 X_1 - c_0 b_1 Z_1 Y_1 - b_0 a_1 Z_1 X_1 - a_1 Y_1 b_0 Z_1 - c_1 X_1 Y_1 b_0 + X_1 b_1 Y_1 b_0 + c_1 X_1 Y_1 a_0 + X_1 a_0 b_1 Z_1 - c_1 X_1 a_0 Z_1 + c_0 X_1 b_1 Z_1 - X_1 b_0 b_1 Z_1 + c_1 X_1 a_0 W_1 - c_0 a_1 Z_1 W_1 + c_0 b_1 Z_1 W_1 + a_0 a_1 Z_1 W_1 - a_0 b_1 Z_1 W_1 + b_1 Y_1 b_0 W_1 + b_0 a_1 Z_1 W_1 - b_0 b_1 Z_1 W_1 - c_1 Y_1 b_0 W_1 - a_1 X_1 a_0 W_1 - 2 X_1 b_1 Y_1 a_0 + 2 a_1 Y_1 b_0 X_1 = 0$$

Representing $f(\mathbf{S}, \mathbf{D})$ as a sum of multiplication as required in (4), we see that there are 10 shape monomials s_i, thus $n = 10$ and $\mathbf{s} = [X_1^2, X_1Y_1, X_1Z_1, X_1W_1, Y_1^2, Y_1Z_1, Y_1W_1, Z_1^2, Z_1W_1, W_1^2]$. There are 9 image monomials d_j, thus $m = 9$ and $\mathbf{d} = [a_0a_1, a_0b_1, a_0c_1, b_0a_1, b_0b_1, b_0c_1, c_0a_1, c_0b_1, c_0c_1]$.

We can now construct the 10×9 complexity matrix M. We use Gaussian elimination to decompose M as $M = UW$, where U is 10×5, and W is 5×9. Many matrices U satisfy these conditions, and we choose the "simplest":

$$M = \begin{bmatrix} 1 & -1 & 0 & -1 & 0 & 1 & 0 & -1 & 1 & 0 \\ 0 & -2 & 1 & 0 & 0 & 1 & 0 & 1 & -1 & 0 \\ -1 & 1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -1 & 0 & 0 & -1 & 0 & -1 & 1 & 0 \\ 0 & 1 & -1 & 0 & -1 & 0 & 1 & 1 & -1 & 0 \\ 0 & -1 & 0 & 0 & 1 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1/2 & 0 & 0 & -1/2 \\ 0 & 1 & -1 & 0 & 1/2 \\ -1 & -1/2 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 & -1/2 \\ 0 & -1/2 & 1 & 1 & 1/2 \\ 0 & 1/2 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 1/2 \\ 0 & 0 & -1 & 0 & -1/2 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & 0 \end{bmatrix}$$

Since the rank of M is 5, the complexity of the index is 3. From the above decomposition we get

shape parameterization: $\mathbf{g(S)} = \mathbf{s} \cdot U = [X_1^2 - X_1W_1, -2X_1Y_1 + 2Y_1Z_1, -X_1Z_1 + Y_1Z_1, -Y_1^2 + Y_1W_1, 2Z_1^2 - 2Z_1W_1]$

most efficient index: $\mathbf{h(D)} = \mathbf{d} \cdot V = [a_1a_0 - c_1a_0, 0.5a_1a_0 + a_0b_1 - 0.5c_1a_0 - b_0a_1 - 0.5b_0b_1 + 0.5b_0c_1, -a_0b_1 + c_1a_0 + b_0a_1 + b_0b_1 - c_0a_1 - c_0b_1, b_0b_1 - b_0c_1, -0.5a_1a_0 + 0.5a_0b_1 - 0.5b_0a_1 + 0.5b_0b_1 + 0.5c_0a_1 - 0.5c_0b_1]$

It is interesting to compare this result with the literature. Jacobs [3] showed recently that for 6 points in a single perspective image, the lowest rank picture manifold is a non-linear manifold of rank 3 in $\mathcal{R}^4$. We obtained the same rank 3, while using a different 4-dimensional measurement vector (a different indexing space). In this transformed space, the picture manifold is of the same low rank of 3 (provenly the lowest rank possible), and yet it is linear with all the benefits that come from linearity. In addition, our result was obtained automatically and relatively effortlessly, as we are using general and not problem-specific tools.

$\mathcal{C} = 3$: 4 points and a weak perspective camera:

The vision problem is a single image of 4 points, obtained by a weak perspective camera. The index complexity is 3. Thus the most efficient linear picture manifold is a $3D$ hyperplane in $\mathcal{R}^4$.

$\mathcal{C} = 3$ 6 lines and a para-perspective camera:

We took a vision problem of 6 lines projected para-perspectively (using an affine camera model) to a single image. We wrote down the equations relating the 6 unknown camera parameters, the 24 unknown parameters of the lines, and the 6 known projected directions in the image. Using elimination we obtained a single

index $f(\mathbf{S},\mathbf{D})$. The rank of the complexity matrix of the index is 5, thus the index complexity is 3.

For convenience of notation, we define the function $\Theta(a,b,c,d,e,f)$ as:

$$\Theta(a,b,c,d,e,f) = \begin{vmatrix} a & b & b \\ d & d & c \\ e & f & e \end{vmatrix} - \begin{vmatrix} a & a & b \\ c & d & c \\ e & f & f \end{vmatrix}$$

Using the analysis described above we computed the most efficient index:

$$\mathbf{h}(\mathbf{D}) = [\Theta(sl_1, sl_2, sl_3, sl_4, sl_5, sl_6),\ \Theta(sl_1, sl_2, sl_3, sl_5, sl_4, sl_6),\ \Theta(sl_1, sl_3, sl_5, sl_4, sl_6, sl_2),\ \Theta(sl_1, sl_3, sl_4, sl_2, sl_6, sl_5),\ \Theta(sl_1, sl_5, sl_4, sl_2, sl_6, sl_3)]$$

where sl_i is the projected slope of the i-th line.

***C* = 2: The color at a point:**

Using the linear combination color model [5], the color of an object or a light source can be described as a linear combination of a set of basis color functions of wavelength λ. It has been argued that 3 basis functions can approximate well the color of natural objects and light sources. We use a model where the color spaces of objects and light sources are approximated by the span of 3 basis functions each. Our camera model has 4 different color sensors. Thus we have 4 measurements at each point (the output of the 4 sensors), and 6 unknowns - the 3 coefficients of the object color space and the 3 coefficients of the illuminant color space. With 4 equations we can eliminate the 3 illumination parameters, leaving us with a relationship between the measured colors (the sensors' output) and the reflectance properties (the color coefficients) of the object at a given point.

The index complexity of this relationship is 2, meaning that all the possible measured colors of a point (regardless of illumination color) sit in a plane in a 3-dimensional color space. The analysis also gives that there is no smaller subspace of the colors, which is insensitive to the illuminant color.

4 Adding class constraints

Once it has been shown that model-free invariants do not exist for unconstrained objects, attention had turned to characterizing the constraints (or classes of objects) which would lead to model-free invariants [6, 8], or in our language, lead to index complexity 0. The present analysis allows us to do so directly, as part of a more general question: what class constraints on objects reduce the index complexity (and thus make their recognition more efficient)? We determine sufficient and necessary conditions on class constraints to reduce index complexity, in particular to reduce it to 0 (implying the existence of model-free invariants).

We start from a relation in the indices set

$$f(\mathbf{S},\mathbf{D}) = \sum_{k=1}^{C+2} g_k(\mathbf{S}) * h_k(\mathbf{D}) = 0$$

where $g_k(\mathbf{S})$ and $h_k(\mathbf{D})$ are polynomial functions of the shape and image measurements respectively. Every class constraint of the form $\lambda(\mathbf{S}) = 0$, where $\lambda(\mathbf{S})$ divides some $\sum \alpha_i g_k(\mathbf{S})$, reduces the dimension of the best index by at least 1. Thus:

Theorem 5 class constraints:. *To reduce the index complexity of a vision problem from $\mathcal{C}$ to $p < \mathcal{C}$, the class constraints should provide at most $(\mathcal{C}-p)$ independent constraints of the form $\lambda_i(\mathbf{S}) = 0$, where each $\lambda_i(\mathbf{S})$ divides some $\sum \alpha_i g_k$ modulo the $\lambda_j(\mathbf{S})$, $j < i$.*

Clearly there is a trade off between index complexity, which is higher for more general (and less constrained) classes, and the density of the database, which is smaller for more general classes (as there are fewer types of such general objects).

Example: given 6 points and a perspective camera, we showed in Section 3.2 that the index complexity is 3, and we computed $\mathbf{g}(\mathbf{S})$ and $\mathbf{h}(\mathbf{D})$. It immediately follows that:

- If any of the parameters of the 6th point X_1, Y_1, Z_1 equals 0, the complexity goes down from 3 to 1; if any 2 of these parameters are equal, the complexity also goes down from 3 to 1. Thus *if 4 of the 6 points are coplanar, the index complexity of the system is 1.*
- If $X_1 = Y_1 = W_1$, the complexity is 0, namely, there is a model-free invariant.

Appendix: Handling multiple indices

Up to now we considered the case where the ideal of indices $\mathcal{I}$, obtained via elimination from the vision problem $\mathcal{V}$, has a single index function in it. Typically, however, the rank of the ideal is $L > 1$, namely, L independent indices can be computed from the image. We can compute the rank complexity of each index independently; thus object representation and recognition will require using L different (but minimal) indexing tables. Instead, we look for a single transformation of the measurement space into R^{Φ}, where each of the L relations defines a hyperplane of rank $\Phi - 1$. Now the index complexity (or the rank of the object manifold) is $\mathcal{C} = \Phi - L$. Even if $\mathcal{C}$ is larger than the individual index complexity of each of the L relations, we see two advantages to the joint approach:

- The table is sparser and therefore indexing into it is more robust.
- Object representation and recognition requires a single table.

We propose the following algorithm to compute the joint index complexity.

1. Obtain the **joint complexity matrix**.
   - Write the L relations as sums of multiplications, as in (4), with n_l distinct products of elements of $\mathbf{S}$ (shape) and m_l distinct products of elements of $\mathbf{D}$ (image measurements), for $1 \leq l \leq L$.
   - Let $\tilde{s}_i$ denote all the distinct products of elements of $\mathbf{S}$ (shape), which appear in any of the L relations, $i = 0..N$ and $N \leq \sum_l^L n_l$.

- For each relation, rewrite (4) using $\tilde{s}_i$ (we can always do this by adding terms whose coefficients are 0).
- Construct the individual complexity matrix M_l for each relation. The size of M_l, the complexity matrix of the l-th relation, is $N \times m_l$.
- Concatenate the L matrices M_l from left to right, giving us the joint complexity matrix M of size $N \times \sum_1^L m_l$.

Note the asymmetrical role of rows and columns here: the row variables define the elements of the joint indexing table, and thus should be the same for all indices; the column variables define the elements of the index, and thus can (and should) vary for different indices.

2. Compute the SVD decomposition of the joint complexity matrix $M = U\Sigma V^T$. Let Φ denote the rank of M, then $\Phi - L$ is the joint complexity of the indices.
3. The shape parametrization is $\mathbf{g}(\mathbf{S}) = \tilde{\mathbf{s}}U\sqrt{\Sigma}$, where $\tilde{\mathbf{s}} = [\tilde{s}_1, \ldots, \tilde{s}_N]$.
4. Because of the way M was constructed, we can find a matrix V_l such that the individual complexity matrix of the l-th relation can be decomposed as $M_l = U\Sigma V_l^T$. The most efficient index is $\mathbf{h}^l(\mathbf{D}) = \mathbf{d}_l V_l \sqrt{\Sigma}$.

The result of using this algorithm on 7 points in a perspective image can be seen in this volume [10].

References

1. D. T. Clemens and D. W. Jacobs. Space and time bounds on indexing 3-D models from 2-D images. *IEEE T-PAMI*, 13(10):1007–1017, 1991.
2. D. P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. *IJCV*, 5:195–212, 1990.
3. D. W. Jacobs. Matching 3-D models to 2-D images. *IJCV*, 1995. in press.
4. Y. Lamdan and H. Wolfson. Geometric hashing: a general and efficient recognition scheme. In *Proc. 2nd ICCV*, pages 238–251, Tarpon Springs, FL, 1988. IEEE, Washington, DC.
5. L. T. Maloney and B. Wandell. A computational model of color constancy. *JOSA*, 1:29–33, 1986.
6. Y. Moses and S. Ullman. Limitations of non model-based recognition schemes. In G. Sandini, editor, *Proc. 2nd ECCV, Lecture Notes in Computer Science*, volume 588, pages 820–828. Springer Verlag, 1992.
7. H. Murase and S. K. Nayar. Visual learning and recognition of 3-D objects from appearance. *IJCV*, 1995. in press.
8. C.A. Rothwell, D.A. Forsyth, A. Zisserman, and J.L. Mundy. Extracting projective structure from single perspective views of 3d point sets. In *Proc. 4th ICCV*, pages 573–582, Berlin, Germany, 1993. IEEE, Washington, DC.
9. D. Weinshall. Model-based invariants for 3D vision. *IJCV*, 10(1):27–42, 1993.
10. D. Weinshall, M. Werman, and A. Shashua. Duality of multi-point and multi-frame Geometry: Fundamental Shape Matrices and Tensors. In *Proc. 4th ECCV*, Cambridge, UK, 1996.
11. M. Werman and A. Shashua. Elimination: An approach to the study of 3D-from-2D. In *Proc. ICCV*, June 1995.
12. M. Werman and D. Weinshall. Complexity of indexing: Efficient and learnable large database indexing. TR 95-7, Hebrew University, 1995.