UMI

A Bell & Howell Information Company
300 North Zeeb Road. Ann Arbor, MI 48106-1346 USA
313/761-4700 800/521-0600

UMI Number: 9605742

UMI

300 North Zeeb Road
Ann Arbor, MI 48103

ABSTRACT

AUTOMATIC VISUAL RECOGNITION USING PARALLEL MACHINES

by
Yui-Liang Chen

Invariant features and quick matching algorithms are two major concerns in the area of automatic visual recognition. The former reduces the size of an established model database, and the latter shortens the computation time. This dissertation, will discussed both line invariants under perspective projection and parallel implementation of a dynamic programming technique for shape recognition. The feasibility of using parallel machines can be demonstrated through the dramatically reduced time complexity.

In this dissertation, our algorithms are implemented on the AP1000 MIMD parallel machines. For processing an object with n features, the time complexity of the proposed parallel algorithm is $O(n)$, while that of a uniprocessor is $O(n^2)$. The two applications, one for shape matching and the other for chain-code extraction, are used in order to demonstrate the usefulness of our methods.

Invariants from four general lines under perspective projection are also discussed in here. In contrast to the approach which uses the epipolar geometry, we investigate the invariants under isotropy subgroups. Theoretically speaking, two independent invariants can be found for four general lines in 3D space. In practice, we show how to obtain these two invariants from the projective images of four general lines without the need of camera calibration.

A projective invariant recognition system based on a hypothesis-generation-testing scheme is run on the hypercube parallel architecture. Object recognition is achieved by matching the scene projective invariants to the model projective invariants, called *transfer*. Then a hypothesis-generation-testing scheme is implemented on the hypercube parallel architecture.

AUTOMATIC VISUAL RECOGNITION USING PARALLEL MACHINES

by
Yui-Liang Chen

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy

Department of Computer and Information Science

October 1995

APPROVAL PAGE

AUTOMATIC VISUAL RECOGNITION USING PARALLEL MACHINES

Yui-Liang Chen

Dr. DaoChaun D. Hung, Dissertation Advisor — Date
Assistant Professor of Computer and Information Science, NJIT

Dr. James A. M. McHugh, Committee Member, — Date
Associate Chairperson and Professor
of Computer and Information Science, NJIT

Dr. Frank Y. Shih, Committee Member, — Date
Associate Professor of Computer and Information Science, NJIT

Dr. Andrew Sohn, Committee Member, — Date
Assistant Professor of Computer and Information Science, NJIT

Dr. Nirwan Ansari, Committee Member, — Date
Associate Professor of Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Yui-Liang Chen

Degree: Doctor of Philosophy in Computer Science

Date: October 1995

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 1995

- Master of Science in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 1990

- Bachelor of Science in Mining Engineering,
  National Cheng Kung University, Taiwan, Taiwan, 1985

Major: Computer Science

Presentations and Publications:

D.C. Hung, Y.L. Chen, R. Chen and J. Cheng, "Text processing: Robust character recognition using calibrated text and diversified feature set," in *Proc. IEEE Int. Conf. Tools for Artificial Intelligence*, pp. 82-89, San Jose, CA, November 1991.

D.C. Hung and Y.L. Chen, "Invariant hypothetic feature in shape recognition," *in Proc. IEEE Regional Conference on Control Systems*, pp. 528-529, Newark, New Jersey, August 13-14, 1993.

Y.L. Chen, A. Sohn, and D.C. Hung, "Shape recognition using a MIMD parallel machine," *in Proc. 2nd Symposium on Pattern Recognition and Computer Vision*, Zurth, September 14, 1993.

Y.L. Chen, A. Sohn, and D.C. Hung, "Object recognition using parallel machine," *in Proc. Int. Symposium on Artificial Neural Networks*, Hsinchu, Taiwan, December 20-22, 1993.

BIOGRAPHICAL SKETCH
(Continued)

Y.L. Chen and D.C. Hung, ''Shape recognition using hypothetic feature,'' *in Proc. 10th Israeli Conference on Artificial Intelligence, Computer Vision, and Neural Networks*, Tel-Aviv, Israeli, December 27-28, 1993.

Y.L. Chen and D.C. Hung, ''Fundamentals of automated CAD-based vision system,'' *in Proc. 3rd Int. Conference on Advances in Pattern Recognition and Digital Techniques*, Calcutta, India, December 28-31, 1993.

Y.L. Chen and D.C. Hung, ''On the recognition of industrial parts,'' *in Proc. 10th IEEE Conference on Artificial Intelligence for Applications*, San Antonio, Texas, March 1-4, 1994.

W.T. Wong, Y.L. Chen and F.Y. Shih, ''A fully parallel algorithm for the extraction of chain and mid-crack codes of multiple contours,'' *in Proc. Int. Computer Symposium*, Hsinchu, Taiwan, December 12-15, 1994.

This dissertation is dedicated to
God, our heavenly Father

ACKNOWLEDGMENT

Doctoral research should be enjoyed and indeed this was enjoyable. This was partly due to helpful faculty and colleagues in the Department of Computer and Information Science, from whom the author received encouragement. The author is grateful to friends made during this research and would like to extend special thanks to them.

Firstly, the author would like to thank his research advisor, Professor DaoChaun Hung, for his advise and guidance during his six years of master's and doctoral research. Professor Hung's valuable time was instrumental to the author in processing this research. The author is proud to have co-authored with him for many conferences and journal papers.

Special thanks also goes to Professor Peter Ng, the chairperson of CIS, and the CIS department for the financial support given during the period of this research. The author appreciates the help and suggestions given by Professors Frank Shih, James McHugh, Andrew Sohn, and Nirwan Ansari for serving as members of the committee.

The author would like to thank the co-author of Chapter 4, Wai-Tek Wong. Without his contribution, this chapter would not have been possible. The author also wants to thank Fujitsu Parallel Computing Laboratory for allowing us to access the *AP1000* multicomputers.

Sincere thanks to the author's parents for their never ending support, understanding, and encouragement during the years of his advanced studies. The author is grateful to his wife, Yu-Hua, and his son, Daniel, for their love and consistent support in sharing their life with him.

Finally, the author gives heartfelt thanks to God, our Heavenly Father. It is He who made the completion of this dissertation possible.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES (Continued)

CHAPTER 1

INTRODUCTION

Invariant features and fast matching are two major concerns in automatic visual recognition. Invariant features reduce the size of a model database while fast matching speeds up the computations. In this dissertation, geometry invariants and parallel vision algorithms are to be discussed. New projective invariants extracted from four general lines are investigated both theoretically and practically. In addition, a parallel implementation of the dynamic programming matching algorithm is also developed. The significant results of parallel algorithms show the feasibility using parallel machines.

1.1 Motivation

Research in computer vision has been progressing dramatically over the last thirty years. One of its major goals is to develop time and space efficient methods for visual recognition. Two issues should be addressed in a successful recognition system. The first is to reach the high recognition rate, and the second is to achieve the real time requirement in recognition processes. The desired outcome is an approach which allows the attainment of both time and space efficiently.

The fundamental difficulty in recognizing objects from images is that the appearance of a shape depends on the viewpoint. This problem is entirely avoidable if the geometric description is unaffected by the imaging transformation. Such invariant descriptions can be measured directly from images without any prior knowledge of the position and orientation of the camera.

Geometric invariants are the extracted features which not only correctly represent the information of objects, but also meet minimum requirements in describing objects.

Invariant features reduce the size of the models needed to be stored in the database. Robustness is the major concern in invariance. Typically, robustness can be measured by the degree of freedom (DOF) in the processing. Three types of invariants, Euclidean invariants, affine invariants and projective invariants, associated with the DOF, are mainly considered in computer vision. Basically, Euclidean invariants are a subset of affine invariants, and affine invariants are a subset of projective invariants. Invariants under various transformation groups will be analyzed in the next section.

Parallel processing, which has progressed tremendously in the past decade, seems to be the general approach to providing the necessary power. In fact, the technological limits of how fast a serial processor can perform are being reached. Parallel processing has been suggested as the approach to providing the computational power needed for most computational-intensive problems. However, the utilization of parallel machines to solve the computer vision problems remains very limited. Most of the parallel vision algorithms have only dealt with problems regarding ideas, few of them have provided generic approaches to solving automatic recognition problems. Special parallel architectures of vision systems, and parallel algorithms on general parallel architectures for vision problems are both discussed in the latter section of this chapter.

In this dissertation, we will study how geometric invariants and parallel paradigms are being used to solve automatic visual recognition problems. The scope of this dissertation is described in the last section of this chapter.

1.2 Invariants Under Various Transformation Groups

Efficient object representation is a key to successful visual recognition. Invariance under a group of transformations is a desired component of any efficient representation. Relevant transformations include ridge transformations, similarity transformations, affine transformations and projective transformations, depending on the manner in which an image is formed.

1.2.1 Degree of Freedom (DOF)

Let D_δ denote the degree of freedom (DOF) of δ. Three different types of DOF are discussed as follows:

- D_s indicates the DOF of the structure of objects. Different objects have different D_s, e.g. $D_s = 4$ for a line in a 3D space, or $D_s = 2$ for a point on a 2D plane;
- D_t indicates the DOF of transformations. Different actions have different D_t, e.g. $D_t = 15$ for a projective transformation in a 3D space, or $D_t = 4$ for a similarity transformation on a 2D plane; and
- D_i indicates the DOF of isotropy transformations. We know the degree D_i from the matrix of isotropy transformation.

Thus, the number of invariants, N_i, under a transformation is given by [150]:

$$N_i = D_s - D_t + D_i \tag{1.1}$$

The preserved invariance under each group of transformations is discussed as follows:

1.2.2 Euclidean Groups

Ridge transformations and similarity transformations are considered for Euclidean invariants in both 2D to 2D and 3D to 3D transformations.

Rigid Transformations. In this case, only translations and rotations are performed. Displacements, including the angles between lines and the distances between points, are preserved. Since 2D translation has $D_t = 2$ and 3D translation has $D_t = 3$, and 2D rotation has $D_t = 2$ and 3D rotation has $D_t = 3$, the number of DOF of rigid transformation is $D_t = 4$ in 2D plane , and the transformation matrix T is

$$T = \begin{bmatrix} R & t \\ 0_3{}^T & 1 \end{bmatrix} \tag{1.2}$$

where R is a rotation matrix, t is a translation vector and 0_3 is $[0, 0, 0]^T$.

Similarity Transformations. A scaling factor is added into rigid transformations. One more DOF than in the rigid class is obtained. The distances between points are no longer preserved under similarity transformations. Instead of absolute distances, relative distances are preserved. Nevertheless, angularity is still preserved in this class. The transformation matrix is

$$T = \begin{bmatrix} R & t \\ 0_3{}^T & s \end{bmatrix} \tag{1.3}$$

where R is a rotation matrix, t is a translation vector and s is non-null scalar.

1.2.3 Affine Groups

Affine groups consist of geometric invariants under affine transformations.

Affine Transformations. Affine transformations preserve parallelism and the center of mass. The number of DOF is 12

$$T = \begin{bmatrix} A & v \\ 0_3{}^T & s \end{bmatrix} \tag{1.4}$$

where A is a non-singular 3x3 matrix, v is a 3D vector and s is a non-null scalar. T is non-singular defined up to a global scale factor.

1.2.4 Projective Groups

Projective groups consist of geometry invariants under projective transformations.

Projective Transformations. Homographies preserve collinearity, coplanarity and cross-ratio. The number of DOF is 15

$$T = \begin{bmatrix} A & v \\ \mu^T & s \end{bmatrix} \tag{1.5}$$

where A is a non-singular 3x3 matrix, μ and v are 3D vectors and s is a non-null scalar. T is non-singular defined up to a global scale factor.

1.3 Parallel Algorithms and Architectures

The processing of vision problems is usually time consuming. In order to speed up the processing time, parallel paradigms provide the solution for complex computational problems. Basically, two different approaches are investigated by researchers. One approach is to design new architectures and build new parallel machines for a specific purpose. The other approach is to develop some algorithms on general architectures and general purpose parallel machines.

A set of representative vision-related computational tasks was selected by principal investigators of the DARPA Image Understanding Program in 1986. The tasks of these benchmarks are given in Table 1.1 [177] [222]. According to these benchmarks, vision tasks are normally divided into three levels: a low level dealing with pixel arrays, an intermediate level dealing with geometry data, and a high level dealing with relational structures [211] [221] [39] [130] [206].

Low level tasks are mostly image processing algorithms. These algorithms, which are very regular in structure, involve data independent and local computations, and also involve pixel data. Available parallelism is normally on the pixel level [217] [127] [165] [62] [3] [105] [193]. Intermediate level algorithms perform computations on the output produced by low level algorithms. They involve more complex data structures, data dependent algorithms, symbolic processing, as well as varying degrees of parallelism which depends on the data and the nature of the computation [224] [24] [165] [178] [122] [1]. Finally, high level algorithms not only exhibit most of the properties of intermediate level algorithms, but also involve top-down processing in which knowledge based interpretation is performed. Therefore, the algorithms may involve accessing databases, performing enormous searches and may include other artificial intelligence algorithms

Table 1.1 The benchmarks of vision tasks.

Data Types	Tasks
pixel arrays	edge detection(including convolution, zero-crossing detection, and border following)
	connected component labeling
	Hough transform computation
geometric data	computation of convex hull, the Voronoi diagram, and minimal spanning tree of a set of points in the space
	visibility computation for a set of opaque triangle in 3-space
relational structures	finding subgraphs of a given graph that are isomorphic to another given graph
	finding the minimum-cost path between two vertices and edge-weighted group

[49] [174] [138] [33] [207]. In other words, for lower level processing, spatial decomposition of an image provides a natural way of generating parallel tasks. For higher level analysis operations, parallelism may also be based on other image characteristics and may be data dependent.

1.3.1 Parallel Architectures

Many new architectures have been designed or built that can be used for vision tasks:

The Image Understanding architecture (IUA). The IUA is made up of three levels, each having a particular type of processor [124] [220] [222] [223] [51]

- The Content Addressable Array Parallel Processor (CAAPP) consisting of 512 × 512 square grid array of custom 1-bit serial processors linked through a four way grid performs fine grained SIMD computing for low level vision tasks.

- The Intermediate Communications Associative Processor (ICAP) consisting of 64 × 64 square grid array of 16-bit processors performs MIMD computing for intermediate level vision tasks.
- The Symbolic Processing Array (SPA) is an ensemble of 64 powerful processors. Each processor is a 32-bit computer with at least 4 Mbytes of memory to perform MIMD computing for high level vision tasks.

The VisTA. Three architectures are pipelined into VisTA for a computer vision system [200].

- VisTA/1: a massive parallel sliding memory plane processor (SliM), a fine grained SIMD architecture proposed for low level vision.
- VisTA/2: a flexibly (tightly / loosely) coupled multiprocessor (FCM), a medium grained MIMD architecture proposed for intermediate level vision.
- VisTA/3: a flexibly coupled hypercube multiprocessor (FCHM), a coarse grained MIMD architecture proposed for high level vision.

1.3.2 Architecture-Independent Systems

Several architecture-independent systems have been investigated for different levels of computer vision tasks.

- Adapt is an architecture-independent language based on the split-and-merge model. It can provide different levels of architecture independence, depending on how much of the underlying architecture is hidden from the programmer [217] [218].
- The Partitionable SIMD/MIMD (PASM) Parallel Processing System can be dynamically reconfigured to operate as one or more independent SIMD and/or MIMD machines [193] [224].

1.3.3 Parallel Machines and Image Processors

General purpose parallel machines provide various basic image understanding features, while special purpose image processors provide certain image processing functions, such as the CLIP7A image processor [62] or the checkers player [140]. A survey of image processing LSIs in Japan can be found in [65]. Some popular parallel machines are often used for computing variant vision tasks:

- The connection machine provides 64 K physical processing elements, millions of virtual processing elements with its virtual processor mechanism, and general purpose, reconfigurable communication networks [206]. The recent version CM-5 operates in SPMD mode [110].
- The MasPar MP-1 provides 1 K processors operating in SIMD mode [111].
- The Warp Machine is a systolic array computer of linearly connected cells, each of which is a programmable processor capable of performing 10 million floating-point operations per second (10 MFLOPS) [3].

1.3.4 Parallel Algorithms

The benchmarks have been studied by general purpose architectures such as pyramid [1] [41], hypercube [160] [170] [97] [15], shuffle-shift [181], mesh [109] [18] [34] [78], reconfigurable mesh [106] and linear array in SIMD mode, and message-passing and shared memory [201] in MIMD mode. In addition, the idea mode, EREW PRAM [48] [49], was also used in vision tasks. Basically, three levels of vision tasks were separated by researchers:

Low level. The low level vision tasks dealing with the pixel arrays have been parallelized including: convolution [122] [92], boder following [129] [130], histogram [44] [106] [129] [130] [127] [92], edge detection [129] [130] [76], Hough transform [63] [129] [130] [178], Fourier transform [105], Wavelet transform [78], thinning [231], smoothing [24], component labeling [15] [154] [129] [130] [165] [52] [153] [92] [2] [201] [210], range data segmentation [160], length of curve [116], and markov random field [141] [108].

Medium level. Several parallel algorithms for intermediate vision tasks have been established including: convex hull [92] [142] [143] [165] [179], computing minimum convex containers of regions [2], computing nearest neighbors of pixels and regions [2], computing region moments [67], and feature extraction [171].

High level. Many high level vision applications have been implemented to exploit parallelism including: neural networks [5] [226], fuzzy system [97], character recognition [102] [203], template matching [34], motion estimation [39], clustering [170], stereo matching [118] [219] [109] [128], string matching [48] [49] Hidden Markov Model [138], model-based object recognition systems [215] [33], geometric hashing [174] [172] [173] [175] [110] [111] [26], and hypothesis testing [18] [207].

1.4 The Scope of this Dissertation

The scope of this dissertation which consists of model-based visual recognition, exploitation of parallelism, and distribution of invariants, is described below:

1.4.1 Model-based Visual Recognition

The state of the art feature extraction methods and matching algorithms are surveyed and analyzed. A new feature, *ACT*, is designed to reinforce the representation of images, and a hypothesis-generation-test scheme is proposed to match industrial objects.

1.4.2 Exploitation of Parallelism

Two implementations on the AP1000 parallel machine and its simulator, CASIM, are demonstrated for shape matching and chain coding. A backward dynamic programming algorithm is parallelized onto a linear array architecture. Both feature extraction and matching algorithms are implemented for shape matching. In addition, a divide-and-conquer parallel coding algorithm is also developed on a pyramid parallel architecture. The coding stage and the merging stage are implemented for chain coding. The timing results are demonstrated, and the computational complexity is analyzed.

1.4.3 Distribution of Invariants

Invariants from four general lines under perspective projection are represented. In contrast to other work which is done by epipolar geometry, we discuss the invariants by isotropy subgroups. Indeed, two independent invariants exist for four general lines in 3D space. We demonstrate how to obtain these two invariants from the projective image of a four general lines structure regardless of the reconstruction of the 3D structure.

We also describe the recognition system based on the hypothesis-generation-test scheme, using projective invariants. Instead of using the conventional machine to compute the complex projective invariant recognition system, the proposed system uses an equivalent parallel algorithm run on the hypercube parallel architecture. Object recognition is achieved by matching the scene projective invariants to the model projective invariants known as *transfer*. Then a hypothesis-generation-test scheme is implemented on a hypercube parallel architecture.

1.4.4 Future Prospects

As always, successful research and development unfolds new research areas and opportunities. While invariant approaches have achieved a certain level of maturity, further advances are expected along several lines of investigation.

First, for recognition, the projective invariants emerged parallel machines should demonstrate the real time performance, not only theoretically, but also practically.

Next, there are two main threads of development concerning parallel algorithms, namely processor usage and speedup. The optimality should be considered in all parallel algorithms.

Finally, there is also the scene so that the relatively isolated results of many researchers on various aspects of invariants analysis should be integrated into a single environment for scene description and recognition.

CHAPTER 2

MODEL-BASED VISUAL RECOGNITION

Surveys of model-based visual recognition systems are presented in this chapter. Two major components, namely feature extractions and matching algorithms, will be analyzed and surveyed. The former is classified into primitive descriptions and compound descriptions. The latter is described in several aspects, e.g. from the functional viewpoint and the measuring viewpoint. Subsequently, a case study for representing and classifying two-dimensional partially occluded shapes follow is described. An *analytic-circular-tag* (*ACT*), which groups a specified number of consecutive hypothetic points in the sequence, is used to provide a new scheme for the matching technique. A hypothesis-generation-test technique is used for contour matching. Finally, the experimental results will demonstrate that our algorithms can be used to match complex industrial parts.

2.1 Two Phases

Object recognition is one of the major tasks involved in computer vision. Since most practical recognition systems are model-based, their tasks are often to match models against an image in the scene. A growing number of studies have been conducted investigating various approaches to model-based recognition in computer vision. The body of literature generated within this developing field is both vast and scattered.

A model-based recognition system can be separated into a learning phase and a recognition phase, as illustrated in Figure 2.1. Two major components of the system are feature extractions and matching algorithms. In addition, model databases are constructed by storing the features which are extracted from model objects. Whenever we

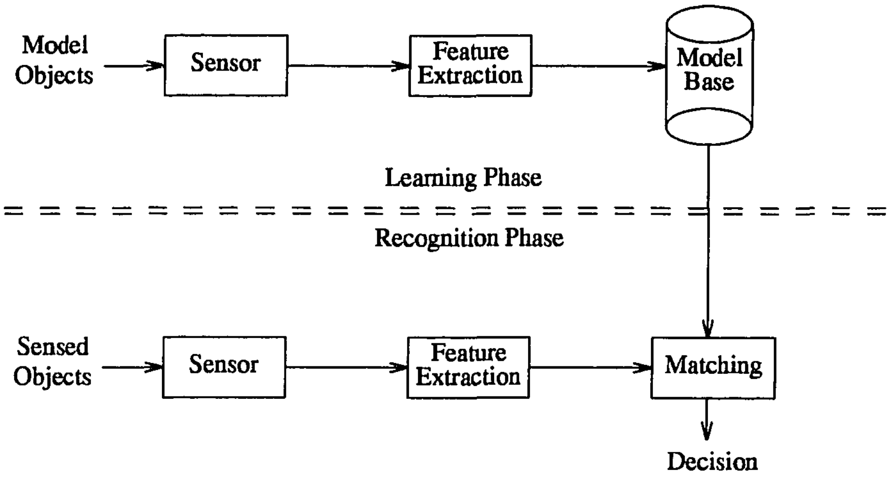


Figure 2.1 Components of a model-based recognition system.

obtain an object from the scene, we try to find the best match between the sensed object and the objects in the model bases. Some recent issues concerning this topic will be analyzed and surveyed in the next two sections. A distinct method is discussed and implemented in the following section an then some conclusions are made in the last section. Interested readers may find other surveys of model-based recognition systems in [199] [27] [37] [16].

2.2 Feature Extractions

Features play an important role in image analysis. In 2D and 3D spaces, position, orientation, pixel values, color, and texture can all be features of an object. This survey concentrates more on the geometric properties of features than on color or texture properties. The major descriptions of the geometric properties are positions and orientations. We categorize these features into primitive features and compound features, which are similar to the classifications by [198] [55] [71] [35].

Table 2.1 General geometric primitive features.

Feature	2-D	3-D
point	$p = [u,v]$	$P = [x,y,z]$
vector	$\vec{v} = p_1 - p_2$	$\vec{V} = P_1 - P_2$
line segment	$p = p_1 - t(p_1 - p_2)$ $0 \le t \le 1$	$P = P_1 - t(P_1 - P_2)$ $0 \le t \le 1$
edge fragment	$p = p_0 - t\vec{v}$ t: real number	$P = P_0 - t\vec{V}$ t: real number
curve	point set or control points	point set or control points
circular arc	3 points	3 points
surface		point set or control points
planar patch		3 or more points

2.2.1 Primitive Descriptions

Primitive descriptions are those characteristic properties directly obtained from images or their representations, such as contours of 2D objects and ridges of 3D objects. In general, points, line segments, curves and surface patches are all inherent in the position and orientation properties of images (see Table 2.1).

Points. Few researchers use all the points of the images or along the contours or ridges as the features for matching in a direct way [42]. However, most researchers are interested in the significant points which give a more efficient representation of the images. The most popular significant points are the curvature points, which are selected by thresholding the curvatures of points. Many different aliases for those high curvature points [53] were named by various researchers. They include segmentation points by Tsai [205], inflection points by [98] [99] [100], control points by [131] and landmark by [4] of 2D contour, or critical points on 3D surface [169]. Koch [114] [115] defined *corners* not only from the high curvature points but also from the end points of circular arcs. Perkins [161] defined *multisectors* by directed set points along the internal and external boundary.

Line Segment. A line segment was denoted as a straight line between two extreme points [131] [8] [7]. Polygon approximation [114] [17] [43] connected all adjacent segments to represent the object. Much information can be retrieved from polygon approximation, such as, the coordinates of the beginning point and end point of each segment, the length of the segment, the angle between two adjacent segments, the orientation of the segment, the inter-vertices distance, and so on. In addition, the sequence of the segments [166] was also an important part of the polygon approximation. Ayache [8] [7] chose the ten longest segments as *privileged segments.* Huttenlocher [98] [99] chose the segments from different scale spaces, where each segment was classified according to the degree of closure and the smoothness of the contour. Kalvin [107] used the length of the segment and the angle with the successive segment to be an *arc-length versus turning-angle* graph. Stein [195] used *gray coding* to represent the angle between the segment and the successive segment.

Curve. The choice of high curvature points was highly dependent on segmentation methods. To avoid the errors caused by segmentation, various approaches for constructing features were developed by other researchers. Curve representation which preserved the nature of the contour was one of the alternative ways. Unfortunately, the major problem in representing the curve was noise sensitivity. However, a smoothed curve could be obtained from the shortest path near the contour [182]. Knoll [113] sampled the contour into 10-pixel long curve segments by an automatic feature selection technique. Turney [208] [209] used 20-pixel long curve segments to be the *subtemplate.* Hong [94] chose the equally spaced points along the contour to construct the *arc-length versus the turning-angle* graph. Based on the *arc-length versus the turning-angle*, *shape signatures* were introduced by [228] [227]. Hong [93] also developed a canonical form for a shape based on the moment curve. Moreover, 3D curve moments were defined by selecting the Gaussian peaks [132]. Recently, Pao [157] designed a *scalable translation invariant rotation-to-shifting signature.*

Surface. The properties of surfaces and the relations between surfaces [156] were often used to represent an object. On the surface, curvature is one of the most important properties to identify characteristics of a surface. In range images, *Gaussian curvatures* and *mean curvatures* could be calculated from first and second fundamental forms [216]. An excellent idea of Stein's [197] was to use the *splashes* to represent the surface. Vemuri [214] computed the *principal direction* of the surface called *line of curvatures* using *normal curvatures*. Moreover, *jump edges* could be obtained from zero crossings of curvatures and the *crease edges* can be obtained from the local extrema of curvatures [196] [54]. On terrain images, both *Gaussian curvatures* and *mean curvatures* were used to select the points above the specified thresholds [68]. In addition, Radack [169] defined a *mean curvature* in three *critical points*.

Vector, edge fragment, circular arc and planar patch may be treated as subsets of the point, line segment, curve and surface, respectively. However, detailed discussion on these subjects is beyond the scope of this dissertation.

2.2.2 Compound Descriptions

Some primitive features may be structured into a compound feature which is often a local feature set. Then the transformation between two compound features can be determined (see Table 2.2). Basically, in 2D features, one degree of freedom in rotation and two degrees of freedom in translations are allowed. Three degrees of freedom in rotations and three degrees of freedom in translations are allowed in 3D features.

Local focus. Lamdan [119] [120] obtained interest points or *footprints* from the *concavity entrances* or *basis region* which were the convex hull or the convex region of

Table 2.2 Compound features.

Feature	2-D	3-D
Local focus	2 unique points	3 unique, non-collinear points
	1 edge fragment	1 edge fragment and 1 non-collinear point
	1 segment and 1 point	1 segment and 1 point
	2 non-collinear, non-parallel segments	2 non-collinear, non-parallel segments
	1 unique normal	3 planar patches
		2 planar patches and 1 edge fragment
Graph	vertices and edges	vertices and edges
Hierarchy	nodes and arcs	nodes and arcs

the polygon approximation. *Footprint*, which was essentially used to trace the boundary of objects, was first defined by [107] and was also used by [94]. Huttenlocher [100] used three points as *class I features*, and oriented points as *class II features* for the alignment of matching objects. Bolles [19] defined the *local-feature-focus* method by *local-feature description* and *object description*, where the former included a cluster of holes and corners, and the latter included a boundary list. In addition, Bolles [20] located three concentrated edge features: *straight dihedrals*, *circular dihedrals* and *straight tangentials* by partitioning edges into two different planar subedges, circular arcs and straight lines. Also, both 2D and 3D versions of *super segments* were designed by grouping cardinality, *curvature angle*, and *torsion angle* [195] [197]. Moreover, Teh [204] used *L-structure* and *U-structure* to describe both global and local properties.

Graph. Fan [54] designed an *attributed graph* whose vertices contained the information of surface patches and edges, which contained binary information such as the occlusion and connectivity between surface paths. Lin [126] used the *precedence graph* to construct the CSG (constructive solid geometry) binary tree, where the vertices were quadratic surfaces and cubes with concave, convex or planar attributes. Chuang [40] proposed a directed labeled *V-E graph* from B-rep (boundary representation), where the labeled vertices presented topological entities, and the directed edges presented adjacent relationships. Flynn [59] designed a *relational graph* containing both view-independent and view-dependent information, where the former was from IGES's (initial graphics exchange specification) geometric primitives and polyhedral approximation, and the latter was from the synthetic view of objects.

Hierarchy. Ettinger [53] grouped the curvature points into a *sub-part* hierarchy. Camps [30] used five level abstract concepts of points, edges, triplets, holes, junction, and high-level combination with several constraints in order to predict the features called *prediction.* Hansen [79] designed a *strategy tree* including *level one features* and *corroborating evidence subtrees.* The *level one feature* might be a *dihedral edge* or a *dihedral arc* which had the strongest geometric information and was automatically selected by feature filters. The *corroborating evidence subtree* had either a *planar region* or a *curved surface* which was derived from the CAD model.

Others. A *distance profile* was constructed by [169] to obtain a *distance contour.* Borgefors [23] [22] [21] and Liu [131] used segments to construct *chamfer 3/4 distance image* and *polygon image.* Wang [216] designed a three layer structure, *curvature map,* including spatial coordinate, *Gaussian curvature* and *mean curvature.* Pipitone [163] used a class of feature extraction operators called *tripods* to recognize the objects. C-H Chen [35] proposed a *feature sphere* to obtain *tassel addresses* by applying the *tassel-assignment-function* to the *principle direction* of the feature. Jain [103] recovered from oversegmentation by merging the surface patches according to morphological information, patch information, and boundary information in range images. Park [158] used B-rep (boundary representation) and *design feature data* to select focus features. The

Fourier descriptors were used to present the contours of objects by many researchers [69] [125] [162] [230].

2.3 Matching Algorithms

Given a set of models that describe all aspects of the object to be recognized, the process of model-based recognition consists of matching features extracted from a given input image with those of models. The general problem of matching may be regarded as finding a set of features in the given image that matches a model's features approximately. Hypothesis-generation-test schema were used in most matching algorithms.

2.3.1 Hypothesis-Generation-Test Schema

Hansen [79] generated those matched *level one nodes* as hypotheses, and verified them by structural and pixel correction. Bolles [19] looked for the most consistent hypotheses and checked the boundary later for hypothesis verification. In addition, the larger circular arcs were selected to group a *local-feature-focus* for hypothesis generation, and a Z-buffer algorithm was used to compare with the predicate range image for hypothesis verification [20]. Park [158] used consistent matches of selected focus features to generate the hypotheses, and estimated the transformation in order to verify the hypotheses so that the parameters were refined. The final decision of the correct matches might be derived by some statistic methods [74] [121].

Hypotheses could be generated by the compatibility between two *privileged segments* [8] [7]. Three types of evidence, positive, neutral and negative were used by [20]. Ullmann [213] used the *subtracting out* technique to recognize the occluded objects by template matching. Fan [54] used *screeners* and Liu [131] used *boundary matching* to select the candidates. C-H Chen [35] used *relation attributes* to prune the list of model features, and *principle direction constraints* to select the model features. Stein [197] [195] clustered the mutually consistent as a model instance in the scene under geometric constraint, and retrieved model hypotheses from the hash table by indexing *super segments*. Knoll [113] indexed the hypotheses based on the similarities and differences between model types so as to group candidate models.

2.3.2 Functionality

In accordance with functionality, a description of the five most popular approaches for matching algorithms follows:

Modified Hough transform. Grimson [73] [Grim87b] used *Hough* clustering of constraints to prune the leaf of the *interpretation tree*. Stockman [198] used the *pose clustering* technique which accumulated the hypotheses by *Hough transform*. Perkins [161] aligned centers and the major axis of the model and image to match outer boundaries of the model and image in θ–s space. Turney [Turn85] determined the *saliency* of *subtemplates* by finding the constrained solution which made the corresponding hyperellipse at a tangent to the constraining hyperplane. A shape matching technique based on the straight line *Hough* transform was presented by Pao [157].

Interpretation tree. The range of possible pairings of features could be cast into the form of an *interpretation tree* [66], where each node defined a partial interpretation, the level of each ancestor defined a sensory feature, and the branch leading to each such node defined the corresponding model feature [73] [72]. The last branch was a wild card or null branch. Gaston [66] pruned the *interpretation tree* by a set of constraints from tactile images. Ettinger [53] voted for the hypotheses in accordance with the *sub-part* hierarchy from the *interpretation tree*. Fan [54] used the *graph matcher* which performed two-stage, depth-first explorations on the tree. Camps [30] used *branch-and-bound* search methods to find a path which carries the least cost in the interpretation tree. Park [158] used *depth-first tree* search algorithms to prune the tree space, according to relative sizes and angles.

Association graph. A set of matches could be represented by a graph, where the vertex is a pair of matches and the edge is pairwise consistent assignments [19]. The generation of hypotheses for a model in the scene was to find the maximal clique in the association graph [19], i.e., the largest completely connected subgraph [114] [115]. Davis [43] used an *augmented network* to represent the association graph. Lin [126] used a subgraph as the object matching in accordance with the geometric information. Chuang [40] transferred the *V-E graph* into a *shape graph* and found the *pattern graph* which was isomorphic to a subgraph of the *shape graph.*

Geometric hashing. Kalvin [107] first investigated *geometric hashing* for indexing and fast retrieval of the *footprints* in the model base. Wolfson [227] [228] and Hong [94] developed the *voting technique* for choosing the *substring* and *footprints.*

Alignment. Three pairs of non-collinear points always uniquely defined a 3D alignment transformation, which aligned the model so as to map the objects of the scene using the 2D affine transformation. A method to compute the alignment transformation was introduced by Huttenlocher [98] [99] [100]. The solutions of *affine transformation* between two 3D point sets were presented by Arun [6], while the solutions of *affine transformation* between two pairs of rigid planar patch objects were illustrated by Cyganski [42]. Lo [132] borrowed the *affine transformation* introduced by Cyganski [42] to estimate the positions in moments. A sequence of segments were transformed into the new matching segments [166]. The surface transformation constraints were used to eliminate the impossible interpretations on the candidate list [28]. Goldgof [68] used rigidity constraints to prune the search tree.

2.3.3 Measurements

Ansari [4] proved that any invariant function under a similarity transformation was a function of *sphericity*, which is a measure between two triangles. Hong [93] defined the dissimilarity between any pair of shapes and gave the definition of the *affine* dissimilarity function and the *affine* independent dissimilarity function by using *canonical forms* of moment curves. Koch [114] [115] measured the similarity by calculating the difference between the area of two polygons. Wang [216] evaluated the *curvature map* in each different layer. Jain [103] calculated the *evidence based* similarity measure according to the *evidence rulebase*. Oshima [156] computed the dissimilarity of the *candidate body* in order to get the consistency in it.

Least-squares method. Persoon [162] used the *least-squares method* to evaluate the matching between the *Fourier descriptors* of two curves. Lamdan [119] [120] derived the *affine transformation* using the *best least-squares match* method. Wolfson [227] [228] and Kalvin [107] used the *least-squares method* to confirm the final results after *geometric hashing*. The location of the model in the scene was estimated with a *least-squares* fit among the matched *landmarks* [4]. The *super segments* clustered as an instance of the model were evaluated by the *least-squares method* [195] [197]. Ayache [8] [7] and Knoll [113] used the *least-squares method* for hypothesis verification. Turney [208] [209] used *least-squares* to fit in s–θ space. Davis [43] used the *least-squares method* to obtain the tension between two templates. Schwartz [182] used the same method to find the Euclidean motion of the multi-dimensional curves.

Dynamic programming. A structural approach to shape recognition using attributed *string matching* with *merging* was proposed by Tsai [205]. Gorman [69] used *backward dynamic programming* to extract the shortest path from two matching curves from the *intersegment table*. In general, *string matching* had the same construct as the *dynamic programming* technique. Wolfson [228] [227] based the conversion of the curves into shape signature strings and used the application of *string matching* techniques to find long matching substrings. An efficient *string matching* algorithm was introduced to reduce complexity. Moreover, a *hopping dynamic programming* technique was described by Ansari [4] for the purpose of guiding the *landmark* matching through the compatibility table.

Stochastic labeling. Davis investigated a parallel relaxation labeling method to search for the acceptable combinations of matches from the association graph [43]. Barnard [10] drove relaxation labeling in motion and stereo visions. Bhanu [17] used the hierarchical stochastic labeling algorithm to label each model. Since these methods used relaxation labeling, they were computationally intensive.

2.4 A Case Study of Model-based Recognition

Shape recognition is an important task in computer vision. Object boundary is represented by its distinguishable characteristics (features) of which a computational description is used to synthesize a generalization. In accordance with the established description, an object is recognized by identifying correspondences between a scene and a given object, or so called model. This requires matching extracted features of the scene with those of the stored models. It is possible that the scene may be produced by the partially occluded or overlapped objects associated with a large amount of noise and distortion. These imperfections cause significant problems in identifying objects. To handle partial recognition problems, the computational description should be refined so as to include a certain degree of flexibility. In this case study, we investigate a translation-, rotation-, scaling-, occluding-, invariant feature called *analytic-circular-tag*. A complete model-based recognition system is described using *analytic-circular-tag*.

2.4.1 Hypothetic Angularities

Using extracted significant corners as breaking points, the contour is approximated by a polygon, which is represented by a set of significant corners, V_s, denoted as

$$V_s = \{ v_i \; ; \quad 1 \le i \le N \}. \tag{2.1}$$

Assume that these significant corners are ordered by their observation sequence. For a given significant corner v_i, two segments are formed by connecting its forward and backward significant corners v_{i-1} and v_{i+1}, and the associated angle can be calculated by

$$\theta_i = \tan^{-1}\left(\frac{\left|\vec{s}_i \times (-\vec{s}_{i-1})\right|}{\vec{s}_i \cdot (-\vec{s}_{i-1})} \right) \tag{2.2}$$

where $\vec{s}_i = (v_{i+1} - v_i)$. A hypothetic dimension of a given significant corner v_i vector is supplementarily defined as

$$\vec{h}_i = \frac{1}{\theta_\mathrm{i}} \hat{u}_h \tag{2.3}$$

where $\hat{u}_h$ is the unit vector of the hypothetic axis. The generated hypothetic vector, $\vec{h}_i$, has the following properties:

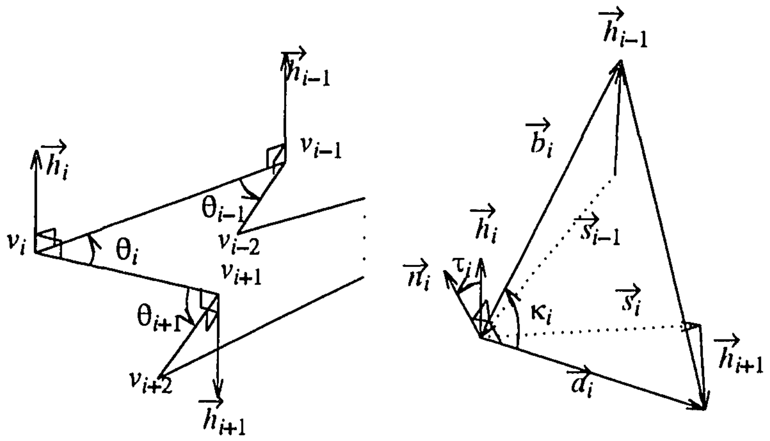
Property 2.1: The sharper the angle is, the longer the magnitude.

Property 2.2: The vector points to the positive or negative direction on the hypothetic axis when the angle is convex or concave, respectively.

Let v_{i-1}, v_i, and v_{i+1} be three consecutive significant corners and $\vec{h}_{i-1}$, $\vec{h}_i$, and $\vec{h}_{i+1}$ be their corresponding hypothetic vectors (as shown in Figure 2.2(a)). Then a plane can be formed by passing through three points, v_i, $v_{i-1} + \vec{h}_{i-1}$ and $v_{i+1} + \vec{h}_{i+1}$. Two hypothetic angularities κ_i and τ_i (as shown in Figure 2.2(b)), then, are defined as

$$\kappa_i = \tan^{-1}\left(\frac{\left|\vec{a}_i \times \vec{b}_i\right|}{\vec{a}_i \cdot \vec{b}_i} \right) \quad \text{and} \quad \tau_i = \tan^{-1}\left(\frac{\vec{h}_i \times \vec{n}_i}{\vec{h}_i \cdot \vec{n}_i} \right) \tag{2.4}$$

where $\vec{a}_i = \vec{s}_i + \vec{h}_{i+1}$, $\vec{b}_i = -\vec{s}_{i-1} + \vec{h}_{i-1}$ and $\vec{n}_i$ is the normal vector of the plane. By the nature of the angularities, it concludes that κ_i and τ_i are scaling-invariant, rotation-invariant, and translation-invariant.



(a) (b)

Figure 2.2 Example of hypothetic space.

2.4.2 Analytic Circular Tag

In the hypothetic feature space, the hypothetic points corresponding to high curvature points of the shapes are chained into a sequence which is defined as

$$^{m}\psi = \left\{ p_i = [\kappa_i\ , \tau_i\ ]\ ; \quad 1 \le i \le N \right\} \tag{2.5}$$

where m indicates the model and N is the number of extracted hypothetic points. Let $P_c = \{p_i\ ;\ c \le i \le c{+}n\}$ denote a hypothetic point set of cardinality n where n is an integer with $n < N$ and $1 \le c \le N$. The center of gravity g_c and the mean of given points r_c^2 can be estimated by

$$g_c \ = \ \frac{1}{n} \sum_{i=c}^{c+n} p_i \ \text{ and } \ r_c^2 \ = \ \frac{k_c}{n} \sum_{i=c}^{c+n} (p_i - g_c)^2\ , \tag{2.6}$$

respectively. In our experiments, we simply put the value of k_c as the invert of n. Now, let's draw a circle with a radius of r_c and place its center at g_c (see Figure 2.3). The artificially generated circle is denoted as *angular-circle* or *A-circle* for short. Obviously, the

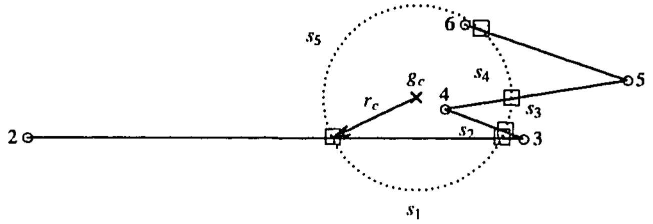


Figure. 2.3 Illustration of *A-circle* which is constructed by choosing cardinality 5 of hypothetic points is shown in a dotted line where *x-points* are denoted as squares.

A-circle intersects the segments $\{\overline{p_i p_{i+1}}\ ;\ c \le i \le c+n-1\}$ at a few positions, which are denoted as *crossing-points* abbreviated as *x-points*. In other words, the hypothetic point set P_c is represented by the *A-circle*, which is divided into a number of arcs $\{s_j;\ 1 \le j \le n'\ ,\ n' \le 2 \times n\}$, where

$$s_j \ = \ \{ \int_{\alpha_j}^{\alpha_{j+1}} r_c \cdot d\alpha \,|\, \alpha_j \text{ is the orientation of } \overrightarrow{g_c x_j}\ ,\ x_j \in x\text{-}points\}. \qquad (2.7)$$

Note that these arcs are traced counterclockwise, and those *x-points* are served as breaking points. To parametrize the arc s_j, several quantities are measured as follows.

- The *phase* α_j is the angular orientation of the lagging edge of the arc.
- The *amplitude* β_j is the angular period of the arc. That is, $\beta_j = \alpha_{j+1} - \alpha_j$.

Thus, a given *A-circle* can be expressed by its gravity, mean value and a sequence of quantities as illustrated here:

$$act_c \ = \ \{g_c\ ,\ r_c\ ,\ [\ \alpha_j\ ,\ \beta_j\ ]_{j=1}^{n'}\}. \qquad (2.8)$$

We called it an *analytic-circular-tag* (*ACT*).

Since random noise is inevitably encountered, most of the measurements must be scattered with a Gaussian distribution. This means that a tolerance must be allowed in order to make possible matches between the sample and the model, which is stored in the database.

2.4.3 Classification in Hypothetic Feature Space

The goal of the classification is to match one or several models in a template, while allowing a certain distortion and incompleteness. A process of hypothesis generation is used to build hypotheses between extracted features of the template and those of stored models. Then, shape matching is, then, performed by verifying the generated multiple hypotheses.

Hypothesis Generation. The task of hypothesis generation is to establish possible matching candidates between features of stored models and those of the unknown target. First of all, the hypothetic angularity features *ACTs* of both scene and models are established by the algorithm described in the previous section. Since noise is inevitably encountered, the angularities obtained by the aforementioned methodology should be compiled with a specified tolerance, for the sake of fair matching.

The hashing table technique is employed for storing the generated act_c. Let $Hash(\cdot)$ denote the hashing function which is used to sort these *ACTs*, and let key_k be the indexing key generated by $Hash(\cdot)$. Note that *ACTs* with the same indexing key will be grouped in the same entry. Let Ω^{key_k} be the entry associated with the kth indexing key,

$$\Omega^{key_k} = \left\{ {}^{r}act_p \ , \ {}^{r}act_q \ , \cdots , \ {}^{t}act_u \ , \ {}^{t}act_v \ , \cdots \right\} \tag{2.9}$$

where $Hash({}^{r}act_p) = Hash({}^{r}act_q) = \cdots = key_k$. Then the data base can be expressed as

$$\Omega = \left\{ \Omega^{key_k} \ ; \quad 1 \le k \le N_\Omega \right\} \tag{2.10}$$

where N_Ω is the maximum size of the entries in the hashing table. Based on their indexing keys, *ACTs* are stored in the hashing table (data base). We can then use the key to index where the act_c, (a member of an entry of the database), is stored. This means that, for a given indexing key, a list of *ACTs* can be retrieved.

Those *ACTs* of stored models whose hashing key is the same as $Hash({}_a act_j)$ can be retrieved from the hashing table. The resulting M hypotheses H are then divided into clusters with respect to their models. For the ith model, its M_i hypotheses can be expressed as

$$H_i = \{{}^i h_{1,j}\,,\ {}^i h_{2,j}\,,\ \cdots\,,\ {}^i h_{M_i,j}\} \tag{2.11}$$

with $M_i \leq M$ and $H_i \subseteq H$.

Hypothesis Verification. The purpose of verification is to distinguish between consistent hypotheses. Let ${}^m h_{p,q}$ be the hypothesis established by model ${}^m act_p$ and that of the target ${}_a act_q$. The least-squares error of ${}^m h_{p,q}$ is calculated by

$${}^m\delta_{pq} = \sum_{j=1}^{n'} \left\{ \left[\alpha_{p+j} - \alpha_{q+j} \right]^2 + \left[\beta_{p+j} - \beta_{q+j} \right]^2 \right\} \tag{2.12}$$

where n' is a number of arcs in *ACT*. A given hypothesis is considered to be consistent if its least-squares error is less than or equal to a specified threshold. A quantitative consistency of ${}^m h_{p,q}$, is then defined as

$$\gamma_{pq} = \frac{1}{1 + \rho^m \, {}^m\delta_{pq}} \tag{2.13}$$

where ρ^m is a scaling factor. It is obvious that $\gamma_{pq} \approx 100\%$ if ${}^m h_{pq}$ is a perfect match; or, on the other hand, if γ_{pq} is close to 0, it implies a bad match.

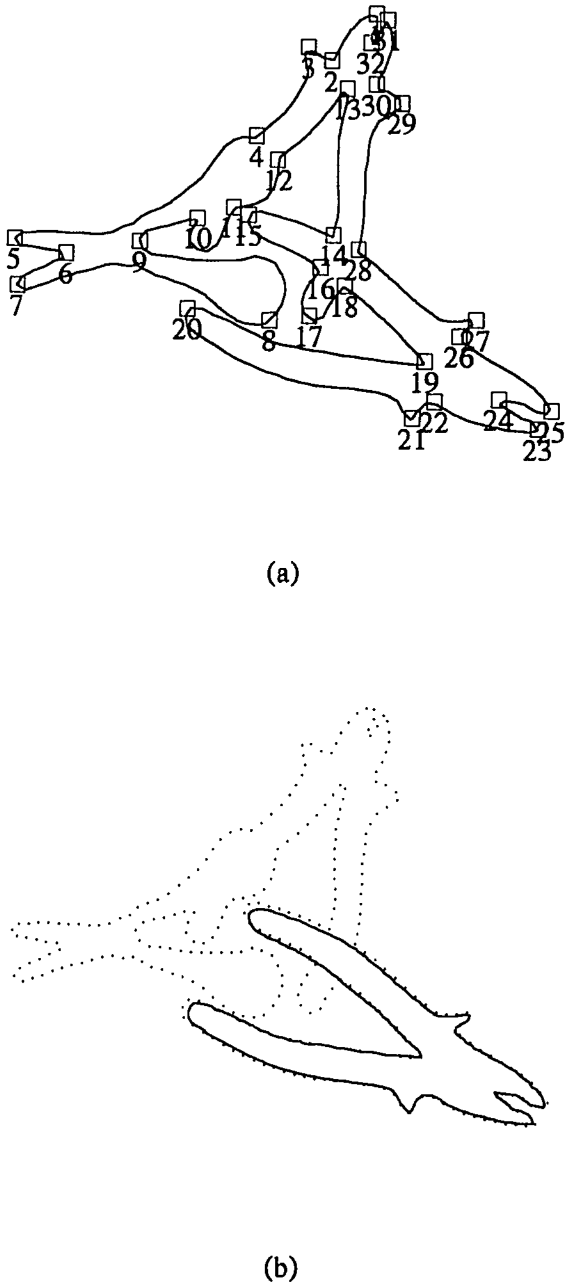
Geometrical Confirmation. Theoretically, one matched hypothesis is enough to identify the scene. However, it has to be considered that:

- *ACT* is a local feature and relies on its cardinality, and
- inevitable distortions have a great deal of influence on the consistency of a matching hypothesis.

For the sake of a confident classification, a geometrical confirmation is employed. Each consistent hypothesis is denoted as an *evidence* of a consistent matching. For a two-dimensional object, two *consistent pieces of evidence* (will be referred as H_{ev}) from distinct partitions are sufficient to establish a consistent geometrical transformation to map the matched model to the sample space. Thus, it is not necessary to check every hypothesis against the model. Alternatively, the remaining hypotheses of the model are examined; those that are geometrically consistent with both of the piece of *consistent evidences* are then denoted as *consistent instances* and are grouped into an additional set, H_{in}. A geometrical confirmation based on *consistent evidences*, H_{ev}, and *consistent instances*, H_{in}, is then performed. In our experiments, we rank the hypotheses by the radius size and pick up the piece of *evidence* by the order of radius size.

2.4.4 Experimental Results

The methodology described in this case study has been integrated within our vision system and tested on a large number of samples objects. For example, Figure 2.4 is acquired by the occlusion of three different models under certain rotations and scalings. By our methodology, the best match is verified as the consistent hypothesis sets. Considering the model wrench(w), the matching result is shown in Figure 2.4. After examining all significant corners involved in forming the H_{ev} and H_{in}, the remaining significant corners are constructed on a new input sample. Thus, all the components which formed the occluding sample can be identified iteratively.



(a)

(b)

Figure 2.4 Occluded scenes consist of various combinations of the models and the matched result of the model wrench.

2.5 Conclusions

The experimental results demonstrate that our *ACT*-based algorithm can be used to match complex industrial parts. Objects in interest may be partially occluded or overlapped. In our methodology, the hypothetic primitives κ_i and τ_i, play important roles. Points of high curvature are essential in describing objects, however, they may not represent the shape sufficiently for the purpose of classification. It has been shown that hypothetic angularities, as well as the *ACTs*, provide a powerful mechanism for recognizing objects. We make very few restricted assumptions about the objects in question. By using hypothetic angularities and *ACTs*, a sequence of consecutive significant corners is mapped into a distinct location in the hypothetic angularity space, in accordance with the mutual geometrical relationships among them. The mapping is invariant to rotation, translation, and the changing size.

CHAPTER 3

PARALLEL IMPLEMENTATION FOR SHAPE MATCHING

An implementation on the AP1000 parallel machine is demonstrated for both feature extraction and matching algorithms. This chapter will describe the implementation of a *parallel dynamic programming* contour-based shape matching algorithm that is suited for message-passing MIMD parallel machines. Our goal is to classify objects by features extracted from their boundaries in a process of automatic recognition. Different from traditional approaches, the proposed system uses parallelism instead of serial propagation. Our algorithm is implemented on a Fuiju *AP1000* parallel machine and tested by real images of industrial parts. For processing an object with n features, the time complexity of the proposed parallel algorithm is $O(n)$, while that of the uniprocessor is $O(n^2)$. In the case of an enormous number of models, the usage of processors is $O(1)$ when linear array is used. Our experimental results show that our method has a running time six times faster than that of the traditional one.

3.1 High-Level Vision

It is well known that, automatic recognition problems are major obstacles in computer vision. Because recognition algorithms are designed to imitate sophisticated human vision [139] [9], they tend to employ complex schema in solving all kinds of challenges. This implicates enormous computations that obstruct the applicability of a real-time performance in the computer vision process. Surmounting this obstacle has become a great challenge in the development of a computer vision system. In this chapter, these problems are overcome by the use of some parallel algorithms.

By consensus, parallel processing, which has progressed tremendously in the past decade, seems to be the approach to providing the necessary power. In fact, the technological limits of how fast a serial processor can perform are being reached. Parallel processing has been suggested as the approach to provide the computational power needed for most computational-intensive problems. However, the utilization of parallel machines to solve the computer vision problems remains limited. Most of the parallel vision algorithms have only dealt with problems regarding ideas, few of them have provided generic approaches to solving automatic recognition problems.

The problem of automatic recognition can be reduced to the contour-based shape matching problem. In most cases, object contours are the most acquirable raw data, because they can be obtained easily by cameras in a well-illuminated controllable environment. In [36], a preliminary study of a contour-based shape matching implemented on a *AP1000* (a message-passing MIMD parallel machine), it was noticed that the computational complexity of this algorithm is lower than its sequential version, [69] just as we predicted. However, the major drawback is the inflexibility of the processing steps. In other words, this early work requires a fixed number of processing steps which cannot be manipulated dynamically.

The purpose of this chapter is to show a modified version of the *parallel dynamic programming* algorithm. We will demonstrate the capabilities of parallel dynamic programming algorithms to fulfill real-time requirements, and their flexibility in that they are independent of processing steps. In other words, the implemented parallel dynamic programming algorithm significantly reduces processing steps, while achieving a superior time performance.

Many parallel algorithms have been developed by various researchers in order to solve the contour-based shape matching problem. Cass [33] proposed an approach to solve the contour matching problem, implemented on a highly parallel SIMD computer.

This method, known as the *Transformation sampling* method, was developed to determine the optimal model image feature transformation through sampling the space of possible transformations. Tucker [207] demonstrated a parallel, 2D object recognition system based on hypothesis generation, and the estimation of supporting evidence for hypotheses in the parameter space. Hypothetic space clustering is used to rank hypotheses according to preliminary evidence and prior to verification.

More recently, Khokhar, Lin and Prasanna [109] presented a discrete relaxation technique to solve the image matching problems. The optimal cost was achieved when the implementation was on systolic architecture. Rigoutsos and Hummel [175] implemented the geometric hashing technique on Connection Machine. Two different versions of parallel algorithms, namely the connectionist algorithm and the hash-location broadcast algorithm, were implemented. Both algorithms achieved linear *speedup* during heavy loading. Dinstein [49] illustrated a parallel computation approach to 2D shape recognition. The approach used parallel techniques for contour extraction, parallel computation of normalized contour-based feature strings independent of scale and orientation, and parallel string matching algorithms. The complexity of the state-of-the-art parallel contour matching algorithms is illustrated in Table 3.1.

This chapter is organized as follows: In section 2, we show the definition of the optimal path in dynamic programming which serves as the basis for the parallel algorithm presented in section 3. Section 4 describes the implementation issues. In section 5, the objectives are carried out by some experimental results. The last section offers the chapter conclusion.

3.2 Preliminary Study

In general, contours are traced clockwise or counter-clockwise and are divided into a sequence of simple curve segments. Suppose that A is a curve containing n segments in a sequence, each of which can be patternized by one of the known descriptors, such as the

Table 3.1 The complexity of the state-of-the-art parallel contour-based shape matching algorithms

Algorithms	Time	Processors	Machines
String Matching [49]	$O(\log n)$	$O(n^2 / \log n)$	CREW PRAM
Discrete Relaxation [109]	$O(n^3)$	$O(n)$	Linear Array
Discrete Relaxation [109]	$O(n^{8/3})$	$O(n^{2/3})$	Mesh Array
Geometric Hashing [175]	$O(n + \log n)$	$O(n^4)$	CM-2
Transformation Sampling [33]	$O(n^2 \log^2 n)$	$O(n^3)$	CM-1
Hypothesis-Generation-Testing [207]			CM-1
Dynamic Programming [36]	$O(n)$	$O(n^2)$	AP1000

* n is the average number of segments of objects

Fourier descriptor [230] or the *footprint* [107], then parametrized descriptions can be represented as a sequence as well.

In a formal sense, two contours are stated as *matched* if and only if there exists a sequence of descriptors extracted from both contours with a high degree of similarity. Let $\Phi^A = \{ f_1^A, f_2^A, \cdots, f_n^A \}$ denote a set of parametrized descriptors and $F^A(i, N_i)$ be a subset of it starting at position i and following a subsequence of N_i descriptors.

Definition 3.1: Suppose that $f_i^m \in F^m(u, N_u) \subset \Phi^m$ and $f_j^s \in F^s(v, N_v) \subset \Phi^s$, a cost function is defined as the degree of matching between f_i^m and f_j^s calculated by one of the known similarity measures [69], [107], denoted as $d(i, j)$.

Definition 3.2: A path $P(u, v)$ mapping $F^m(u, N_u)$ onto $F^s(v, N_v)$ is defined as a sequence of consecutive (f_i^m, f_j^s) pairs, $\forall i \in [u, u + N_u)$ and $\forall j \in [v, v + N_v)$.

Definition 3.3: The accumulated cost function through a specific path $P(u, v)$ is defined by

$$C(u, v) = \sum d(i, j), \quad \forall (f_i^m, f_j^s) \in P(i, j). \tag{3.1}$$

Definition 3.4: For each sequence of $F^m(u, N_u)$ mapping to $F^s(v, N_v)$, there always exists one path $P^*(u, v)$ with the minimum accumulated cost $D(u, v)$. This specific path is called the *optimal path* which indicates the best fit between $F^m(u, N_u)$ and $F^s(v, N_v)$.

The problem of contour-based shape matching is then the same as finding the optimal path between two sequences of descriptors.

Dynamic programming has proven to be very useful for solving many matching problems in computer vision. Over the years as a methodology, quite a few approaches have been developed, such as 2D contour matching [69] [4] [70], stereo matching [155] and 3D curve matching [96]. Without loss of generality, we take the *backward dynamic programming* (*BDP*) [69] as an example (data flow illustrated as Figure 3.1). Fundamentally, the matching procedures can be summarized into the following steps:

1) Compute two sets of *Fourier descriptors* from curve segments extracted from both the known model and the unknown sample;
2) Calculate the cost functions { $d(i, j)$ } for all possible sample-to-model descriptor pairs;
3) Find the minimum accumulated $D(i, j)$ among all possible paths between (i, j) and (N_u, N_v) as a candidate for matching. For the BDP-algorithm, $D(i, j)$ can be

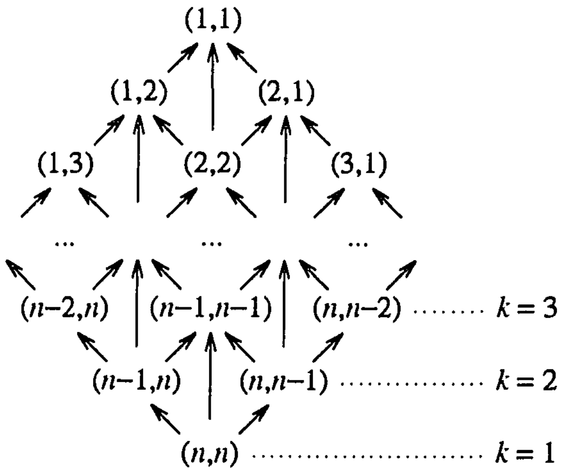


Figure 3.1 Illustration of the data flow of the *BDP*-algorithm, if $N_u = N_v = n$.

expressed in a recursion relation manner

$$D(i, j) = \min \begin{cases} d(i+1, j+1) + D(i+1, j+1) \\ d(i+1, j) + D(i+1, j) \\ d(i, j+1) + D(i, j+1) \end{cases}, \tag{3.2}$$

with given boundary conditions $D(i, N_v) = D(N_u, j) = 0, \quad 1 \le i \le N_u,\ 1 \le j \le N_v$;

4) Set $D(i, 0) = D(i, 1) + d(i, 1)$. If matching starts at ith segment, then the best matching is $D(i^*, 0)$ with

$$D(i^*, 0) = \min(D(i, 0), \forall i \in [1, N_u]). \tag{3.3}$$

Since the implementation of the fourth step is trivial, it will not be mentioned in the following discussion.

3.3 Iterative BDP-Algorithm in Parallelism

Some special purpose parallel algorithms and architectures for the performance of *dynamic programming* have been investigated by [225] [192]. Weste, Burr and Ackland [225] designed an integrated multiprocessing array to solve the simple optimal path problem using *forward dynamic programming*. Their paper described the architecture, algorithms, and design of the CMOS integrated processing array used for computing the dynamic time warp algorithm. Emphasis is placed on speech recognition applications because of the real-time constraints imposed by isolated and continuous speech recognition. Shu and Nash [192] designed a gated interconnection network to solve the shortest path problem using the *dynamic programming* technique. They have described a flexible communication capability that permits more effective usage of the traditional mesh connected SIMD arrays of the bit-serial type. However, none of these provide solutions for shape matchings.

In this section, an approach to performing BDP-algorithm is proposed through the use of multiple processors. To initiate the procedure, the input descriptors are loaded to corresponding processors and the boundary conditions are set. Supposing that the numbers of descriptors extracted from both a model and a sample are n, then, up to $n \times n$ processors are used in this study and each processor $PE(i,j)$ performs two distinct functions: *metric_of_similarity*() is used to calculate $d(i,j)$, and *minimum*() is used to measure $D(i,j)$.

By Eq. (3.2), the inputs of every $D(i,j)$ are dependent on three values $D(i+1,j)$, $D(i,j+1)$ and $D(i+1,j+1)$. This implies that these three values have to be evaluated prior to the computation of $D(i,j)$. In addition, the calculations of $D(i+1,j)$ and $D(i,j+1)$ also depend on $D(i+1,j+1)$. Assume that $PE(i,j)$ will be activated at the iteration k, then it is certain that $PE(i+1,j)$ and $PE(i,j+1)$ have to be activated at iteration $k-1$, while $PE(i+1,j+1)$ is activated at $k-2$. This chronographic relationship holds true while i and j vary independently. By going down the recursion relation of Eq.

Table 3.2 The degree of parallelism achieved while $k = 2n + 1 - i - j$.

k	(i,j)	degree of parallelism
1	(n,n)	1
2	$(n,n-1),(n-1,n)$	2
...	...	...
n	$(n,1),(n-1,2),\ldots,(2,n-1),(1,n)$	n
...	...	...
$2n$ -2	(2,1),(1,2)	2
$2n$ -1	(1,1)	1

(3.2), we learn that the BDP-algorithm starts from $D(n, n)$ at iteration 1. As the data propagate to their deterministic destination (shown as Figure 3.1), a subset of processors $\{ PE(i,j) \mid i + j = 2n + 1 - k \}$ is required to activate simultaneously at iteration k, and those processors belong to the same set from which they are called.

Definition 3.5: The *degree of parallelism* γ is defined as the number of processors executing in parallel.

To achieve the highest parallelism, all processors in the same level must be activated simultaneously. To simplify the notation, we use $D(i, j, k)$ to denote the linguistic expression "$D(i, j)$ is computed at the kth iteration." Similarly, $d(i, j)$ is replaced by $d(i, j, k)$. Then, Eq.(3.2) can be rewritten as

$$D(i, j, k) = \min\begin{cases} d(i+1, j+1, k-2) + D(i+1, j+1, k-2) \\ d(i+1, j, k-1) + D(i+1, j, k-1) \\ d(i, j+1, k-1) + D(i, j+1, k-1) \end{cases} . \tag{3.4}$$

In Table 3.2 shows the degree of parallelism achieved while $k = 2n + 1 - i - j$.

To correctly compute $d(i, j, k)$, f_i^m and f_j^s must arrive $PE(i, j)$ prior to the computation. Let $f_i^A(t)$ denote the descriptor f_i^A after it has propagated through the t processors,

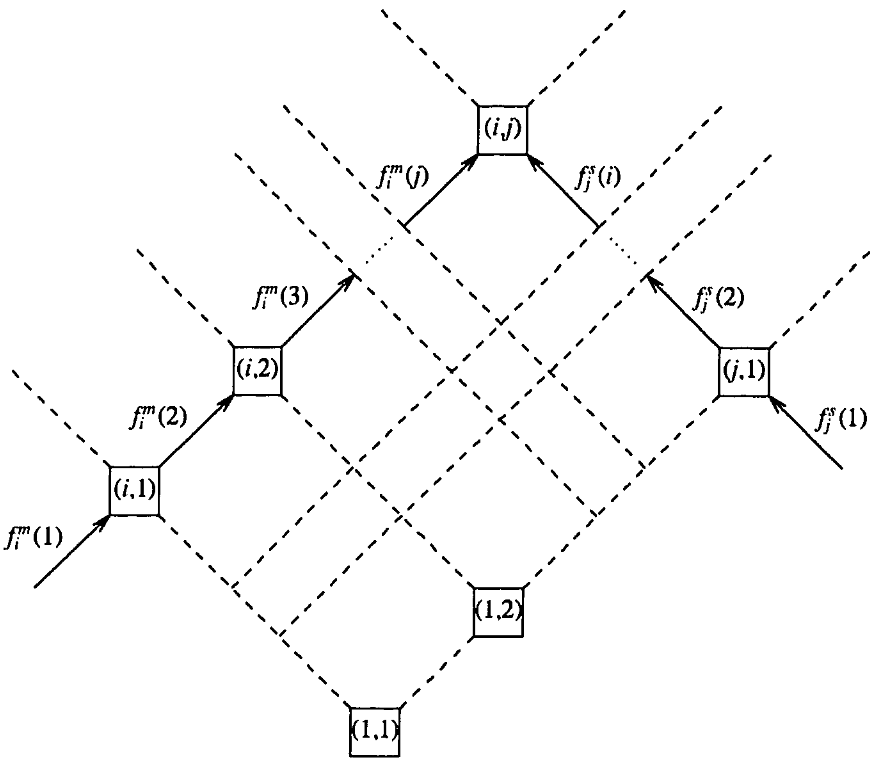


Figure. 3.2 The traversal of input descriptors in which the processors are denoted as boxes, while arrows indicate the directions of propagations.

column-wise or row-wise (illustrated in Figure 3.2). Then, $t \geq \max(i, j)$ must be satisfied. A detailed time schedule for propagating f_i^m is listed in Table 3.3. A similar time table for propagating f_j^s can be obtained by replacing (p, q) with (q, p). Based on data propagating schedules and processors activating schedules, a reinforced BDP-algorithm (called *parallel backward dynamic programming* or *PBDP* for short) is proposed in Figure 3.3.

Lemma 3.6. In Figure 3.3, the time complexity of the proposed PBDP-algorithm is $O(n)$ if a total of $O(n^2)$ processors is used.

Table 3.3 The propagation of f_i^m, $i = 1, \cdots, n$ where (p, q) denotes $PE(p, q)$.

k	$\{f_i^m\}$								
	n	$n-1$	$n-2$	$\cdots$	i	$i-1$	$\cdots$	2	1
1	(n, n)	–	–		–	–		–	–
2	$(n-1, n)$	$(n, n-1)$	–		–	–		–	–
3	$(n-2, n)$	$(n-1, n-1)$	$(n, n-2)$		–	–		–	–
·									
·									
·									
i	$(n+1-i, n)$	$(n+2-i, n-1)$	$(n+3-i, n-2)$	$\cdots$	(n, i)	–		–	–
·									
·									
·									
n	$(1, n)$	$(2, n-1)$	$(3, n-2)$	$\cdots$	(i, i)	$(i+1, i-1)$	$\cdots$	$(n-1, 2)$	$(n, 1)$
$n+1$	–	$(1, n-1)$	$(2, n-2)$	$\cdots$	$(i-1, i)$	$(i, i-1)$	$\cdots$	$(n-2, 2)$	$(n-1, 1)$
·									
·									
·									
$2n$	–	–	–		–	–		$(1, 2)$	$(2, 1)$
$2n+1$	–	–	–		–	–		–	$(1, 1)$

Proof : Basically, the PBDP-algorithm uses $n \times n$ processors; hence, it is $O(n^2)$. In general, it is reasonable to assume that the time complexities for both arithmetic computations and data communications are $O(1)$, and both *metric_of_similarity*() and *minimum*() are also $O(1)$. Therefore, the time complexity for the initialization stage is $O(1)$, and so does each *for all* loop. Let $T(n)$ be the running time required by the algorithm. Since total $2n - 1$ iterations are needed, we have $T(n) = O(2n - 1) = O(n)$. □

Recall that N_v is the number of features in a scene, and N_u is the total number of features in the model base. We present a refined version of a parallel algorithm, which is $O(N_u + N_v - 1)$ to find the matching between scene and models on a N_v linear array parallel computer. Our parallel *dynamic programming* algorithm is based on using the

```
PBDP ALGORITHM

input: Let $f_i^m$ be a descriptor in the model, and let $f_j^s$ be a descriptor
       in the sample, where $1 \le i, j \le n$. $D, d, a_1, a_2, b_1, b_2$, and $b_3$ are
       local variables in one of $n \times n$ processors.
output:The optimal evaluation value is put in $D[1,1]$.
begin
       1. Initialization: assign the boundary conditions.
          for all $1 \le i \le n$ pardo
              $a_2[i, n] = f_i^m, b_2[i, n] = 0, b_3[i, n] = 0$
          end pardo
          for all $1 \le j \le n$ pardo
              $a_1[n, j] = f_j^s, b_1[n, j] = 0, b_3[n, j] = 0$
          end pardo
       2. Iteration: evaluate the recursion relation.
          for $k = 1$ to $2n-1$ do
              for all $i + j = 2n + 1 - k$ pardo
                  $d[i,j] = metric\_of\_similarity\ ( a_1[i,j], a_2[i,j] )$
                  $D[i,j] = minimum\ ( b_1[i,j], b_2[i,j], b_3[i,j] )$
                  $D[i,j] = D[i,j] + d[i,j]$
                  $a_1[i-1,j] \longleftarrow a_1[i,j]$
                  $a_2[i,j-1] \longleftarrow a_2[i,j]$
                  $b_1[i-1,j] \longleftarrow D[i,j]$
                  $b_2[i,j-1] \longleftarrow D[i,j]$
                  $b_3[i-1,j-1] \longleftarrow D[i,j]$
              end pardo
          end do
end
```

Figure. 3.3 The PBDP-algorithm of the contour-based shape matching problem

generalized transitive closure operation described in [123] and [104]. From Eq. (3.4), all input data need to be propagated before the processing. A queue is designed for storing the needed data. If every processor has its own queue, assuming the input data are already inserted inside a queue, we can simply perform the processing by just looking at the queue itself. The data flow of our parallel algorithm is refined from Figure 3.1 and illustrated in Figure 3.4. In total, only $O(N_u + N_v - 1)$ steps are processed. In addition, the usage of processors is reduced to $O(N_u / (N_v + N_u - 1))$.

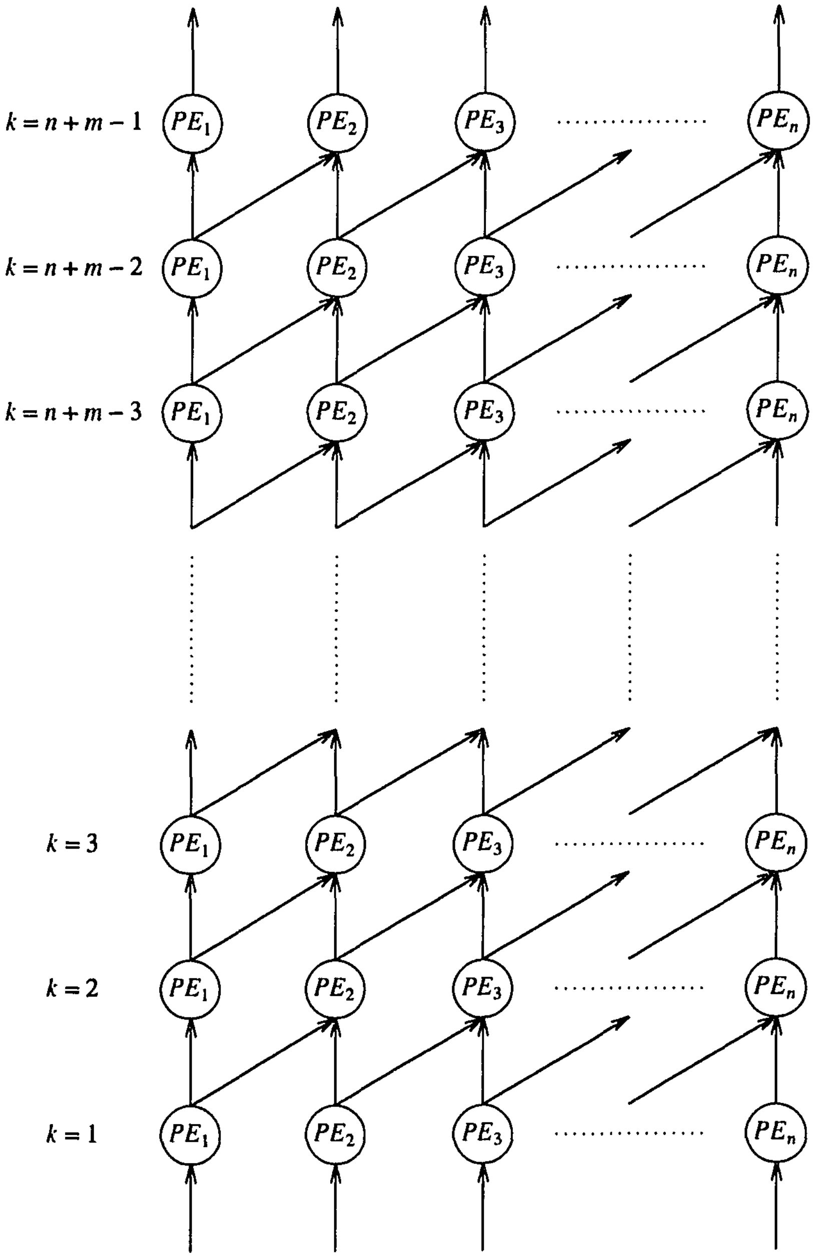


Figure. 3.4 The data flow between the processors.

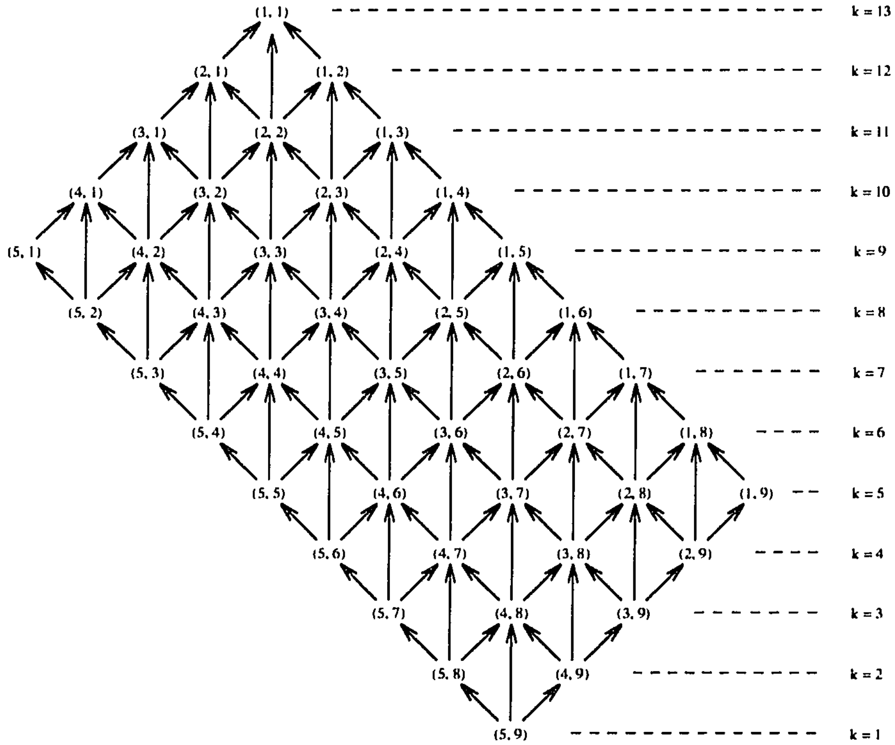


Figure 3.5 The iteration steps.

Example: A 5 ×9 table is shown in Figure 3.5 in which 5 processors are used. Figure 3.6 shows data processed in the queue of each processor. The parallelism at each iteration is:

k	1	2	3	4	5	6	7	8	9	10	11	12	13
degree of parallelism	1	2	3	4	5	5	5	5	5	4	3	2	1

The usage of processors is

$$usage = \frac{N_u}{N_v + N_u - 1} = \frac{9}{13}$$

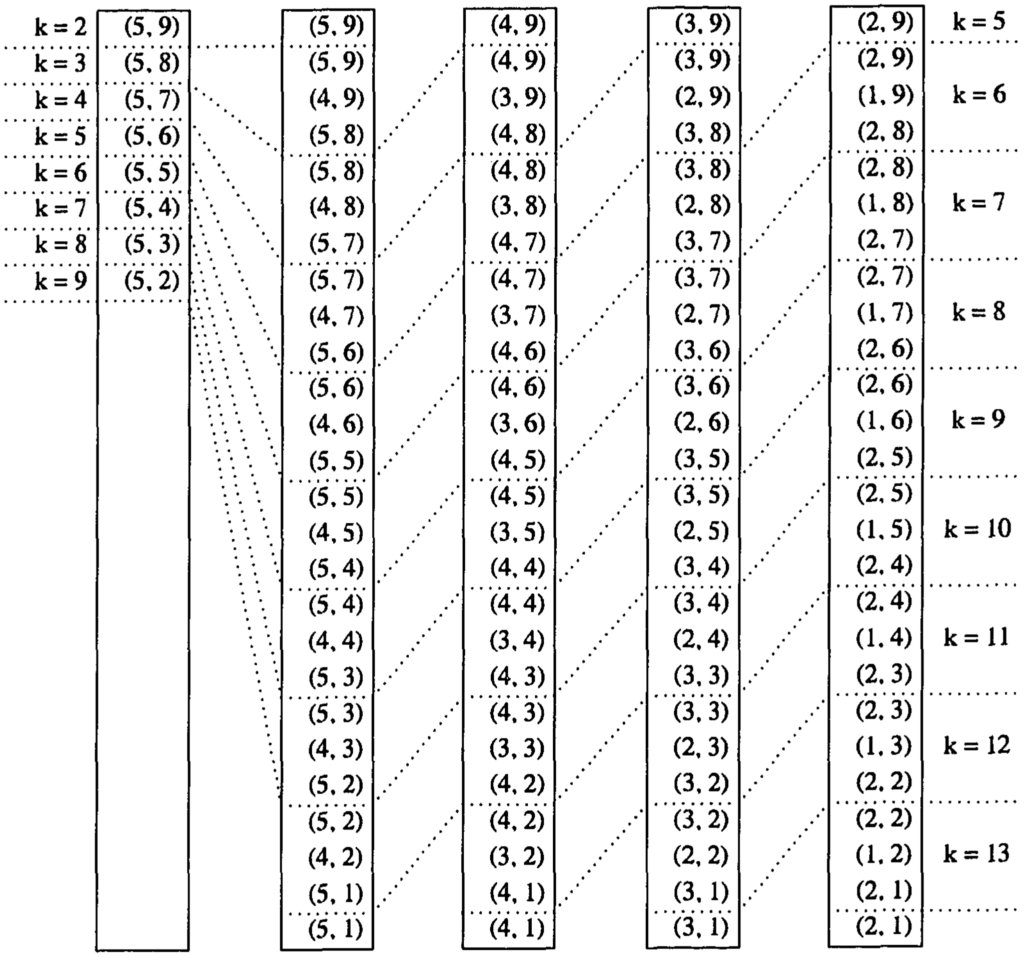


Figure 3.6 The queues and their processing data in each processor.

3.4 Implementation Issues on Message-Passing MIMD Multicomputers

In general, MIMD machines can be categorized into four classes; namely, shared memory multiprocessors, message passing multicomputers, data flow machines, and multithread machines [101]. *AP1000*, built by Fujitsu Co., is one of the message passing multicomputers. Its host server (known as *host*) and processing elements (known as *cells*) are connected with three types of networks named *B-Net*, *T-Net*, and *S-Net*.

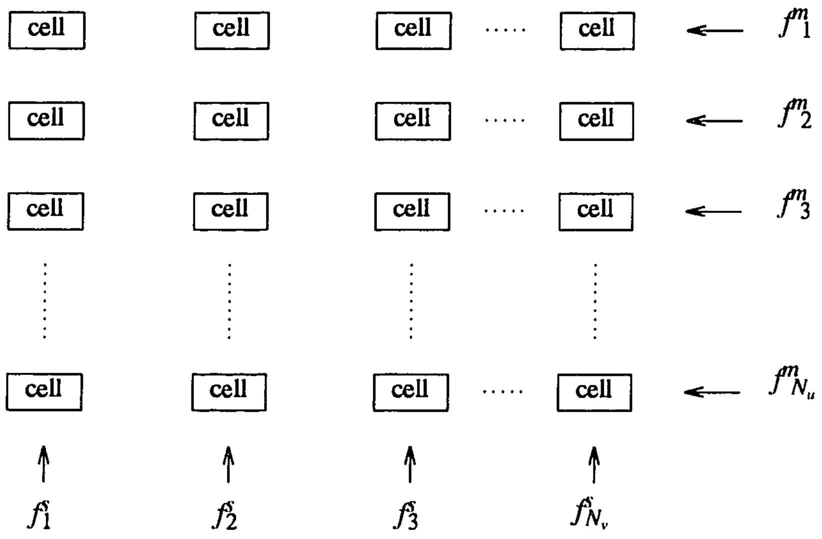


Figure 3.7 $N_u \times N_v$ *cells* are allocated as a mesh.

Through these three networks, the *host* and *cells* can communicate with each other by passing messages. Hence, parallel processing can be achieved.

In our implementation, we have chosen the *Fourier descriptor* [69] extracted from local boundary contour as the representation of the curve segment. The number of extracted *Fourier descriptors* is dependent on the number of segments obtained by the segmentation algorithm. Suppose that there are N_u segments obtained from the model and N_v segments obtained from the sample, then $N_u \times N_v$ *cells* are needed to be allocated as a mesh, as shown in Figure 3.7. If C coefficiences in the *Fourier descriptor* are used, then the *Fourier descriptor* is denoted as $f_i^A(c)$, where $c = -C,\cdots,-1,0,1,\cdots,C$.

Initially, all *Fourier descriptors* are calculated in parallel, using the rightmost column *cells* for the curve segments of the model and the bottom row *cells* for the curve segments of the sample. The *Fourier descriptors* are stored in a_1 and a_2 local variables, respectively. In addition, the boundary conditions of dynamic programming are considered here. In the bottom and the right *cells*, the local variables, b_1, b_2 and b_3, which are used for receiving the three "lower" neighbors' accumulated cost values are assigned the initial value 0.

At the iteration, the $d(i,j)$ values are calculated by the corresponding *Fourier descriptors* stored in a_1 and a_2. If $f_i^m(c)$ and $f_j^s(c)$ are two *Fourier descriptors*, then the *metric_of_similarity* function is defined as

$$d^2(i,j) = \sum_{c=-C}^{C} |f_i^m(c) - f_j^s(c)|^2 \tag{3.5}$$

to indicate the similarity between two descriptors. Three pre-receiving values stored in b_1, b_2 and b_3 are the inputs for *minimum* function. The result of the *minimum* function is added to the $d(i,j)$ value, and then distributed to its three "upper" *cells*. In addition, the *Fourier descriptors* of the model stored in a_1 are sent towards the next column's *cells*, and the *Fourier descriptors* of the sample stored in a_2 are sent towards the row's *cells* in the same fashion.

To implement the fourth step of the matching procedure, *T-Net* is simply used to obtain the minimum value of the two values located on the same row or the same column of *cells*. The final result can then be sent back to the *host*.

3.5 Timing Results

In our experiment, the *Fourier descriptors* are directly computed from chain codes. During the automation process, the images are obtained from cammer, and the chain codes of the boundary contours are extracted by image segmentation techniques. The test images represented by chain codes are illustrated in Figure 3.8.

The running time is determined by counting the intervals between sending the message to the *cells* and receiving the final message from the *cell* (1, 1) through the *host* computer. In general, the running time is contributed to by two aspects:

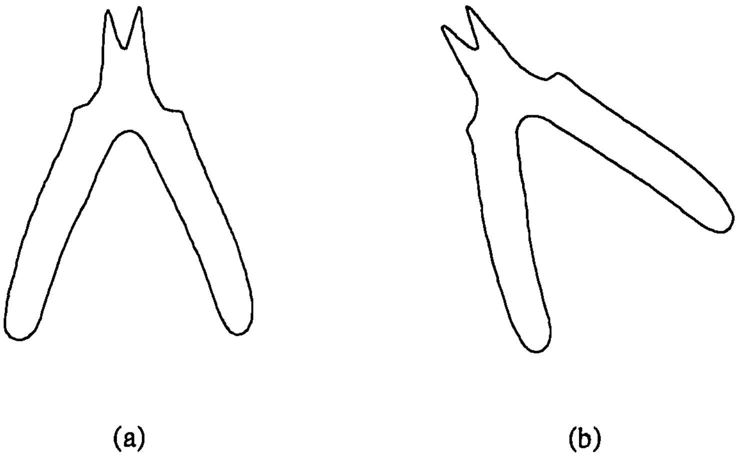


Figure 3.8 The test images represented by chain codes. (a) model. (b) sample.

- Communication time: depends on the message length and the frequency of transmission.
- Computation time: depends on the complexity of the algorithm.

For obtaining the parallel running time (T_p), we first allocate a sufficient number of *cells*, and then transmit and calculate the data according to the parallel dynamic programming algorithm described above. To obtain the sequential running time (T_s), only one *cell* is used to do the calculation. Therefore, the *speedup* can be estimated by

$$speedup = \frac{T_s}{T_p}. \tag{3.6}$$

In order to demonstrate the power of our implementation on *AP1000*, we subdivided the chain codes of contours into a different number of pieces of curve segments. In each experiment, we selected a different number of segments from both the model and the sample for testing. The length of the segment is in inverse proportion to the number of

Table 3.4 The running time and *speedup* of a different number of segments.

N_u	5	7	7	8	9	10	9	10	11	11	11	13	14
N_v	5	5	7	8	8	8	9	10	10	11	13	14	14
T_s	26.78	21.11	16.56	12.62	10.55	10.29	10.55	10.73	9.78	10.45	8.62	8.03	7.79
T_p	6.83	6.84	5.32	2.99	1.42	1.43	1.42	1.44	1.42	1.43	1.38	1.54	1.27
speedup	3.92	3.09	2.92	4.22	7.43	7.19	7.43	7.45	6.89	7.31	6.25	5.21	6.13

segments. The more segments necessary to interpret the contour, the shorter the length of the chain code of each segment. The length of the curve segment's chain codes affects both communication and computation times. A shorter length saves the computation time in the initial stage, but spends more communication time in the iterative stage because more iteratives are required. Suppose the average length of the chain codes of each segment is l and the average number of segments for each model and sample is n. If we calculate c coefficients in our application, it is often $n < l$ and $c < l$ in most cases.

The running time and *speedup* with a different number of segments are shown in Table 3.4. The average of the *speedup* is 5.8 which is approximately six times faster than the traditional computation for the automation. These results show the tremendous progress made in achieving the real-time requirement.

3.6 Conclusions

The purpose of this chapter was to present an implementation of the computer vision algorithm using parallel machines. The desired outcome of our research which was to achieve the real-time performance of the automation is demonstrated with the use of parallel mechanisms.

In this chapter we have addressed several issues of importance in the context of a contour-based shape matching algorithm designed on the parallel *AP1000* machine:

- Developing a modified version of parallel dynamic programming algorithm for solving the contour-based shape matching problem.
- Showing the flexibility of parallel dynamic programming in that it is independent of the processing steps.
- Achieving real-time performance using the parallel dynamic programming technique. A speed increase of approximately six times is reached.

Furthermore, the dynamic programming technique illustrated here is used for solving the optimal path problem; however, if the problem is changed, only a slight modification of the parallel algorithm is required.

CHAPTER 4

A PARALLEL CHAIN CODING ALGORITHM †

In this chapter a fully parallel algorithm for chain codes extraction, based on the table look-up approach is discussed [229]. The basic idea which uses a *divide-and-conquer* strategy is developed on a pyramid parallel architecture. There are two stages, *coding* and *merging*, which consist of the coding algorithm and this is implemented on the parallel machine *AP1000* using the parallel simulator *CASIM*. Experimental results demonstrate the correctness and the flexibility of our parallel coding algorithm.

4.1 Chain Coding

The freeman chain code [64], is widely-used for the representation of an object's contours in the image processing. The chain code moves along centers of two adjacent border pixels. Typically, the 8-connectivity of pixels can be used. In the case of the 8-connectivity of the chain code, the direction of the movement is encoded by a numbering scheme $\{i \mid i = 0, 1, \cdots, 7\}$, where $45i^\circ$ denotes an angle counting counter-clockwise from the positive x-axis.

The contour representation is accomplished by a starting point and a sequence of moves around the border pixels of an object, namely a *chain* which is firmly closed. A chain consists of the following four elements:

† This chapter was the joint work of Dr. Frank Shih, Wai-Tak Wong, and myself. Only the portions of the parallel algorithm, computational complexity analysis, and the experimental results are my contribution.

1) *Tail coordinates*: the coordinates of the starting point.

2) *Head coordinates*: the coordinates after the tracing of the chain is done.

3) *Code sequence string*: the sequence of codes which indicate the direction of movements.

4) *Contour type*: the type of contours being external or internal.

In the existing literature on coding algorithms, the contour-tracing sequential approach or the raster-scan single-pass approach is often used. The former of these traces the border pixels one-by-one and generates codes by considering the neighborhood allocation [50]. The latter runs in a raster-scan fashion to generate codes and then combines chains together [112] [191].

In contrast to the traditional approaches, we propose a fully parallel coding algorithm which is suited for the generation of chain codes by the adoption of only one corresponding look-up table. This proposed algorithm is a significant extension to previous work [191], so that it inherits the advantages of simplicity, efficiency and flexibility. The algorithm using a *divide-and-conquer* strategy [104] consists of the following two stages:

1) *Coding*. The input image is divided into subimages, and a *chain set* is extracted from each subimage.

2) *Merging*. The adjacent chain sets are merged into a new chain set. This is recursively applied working in a pyramid structure from bottom to top.

This chapter is organized as follows. In section 2, the parallel algorithm is introduced. In section 3, the coding stage is described. In section 4, the merging stage is presented. In section 5, the complexity of the parallel algorithm is analyzed. In section 6, experimental results are shown. Finally, in section 7, conclusions are drawn.

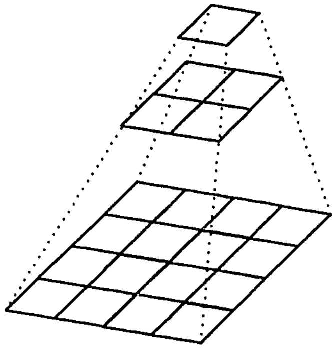


Figure. 4.1 An example of a pyramid architecture.

4.2 The Parallel Algorithm

The parallelism of the code extraction algorithm is introduced by partitioning an entire image into a number of subimages, in each of which a chain set is extracted, and by merging the adjacent chain sets hierarchically into a new chain set. A pyramid architecture [123] is well suited to the implementation of our algorithm. Suppose there are N nodes in the first (lowest) layer of the pyramid architecture shown in Figure 4.1. The higher the layer, the less the number of nodes is. On the first layer of the pyramid, a whole image is divided into N subimages corresponding to N nodes. On subsequent layers each chain set is a concatenation from several chain sets of its immediately lower layer. In other words, the coding stage is performed on the first layer and the merging stage is carried out on subsequent layers.

On the first layer, the object contours of each subimage are encoded into a chain set. The coded chains will be discussed in the next section. Two types of chains are:

1) *Open chain*: if a chain crosses over two subimages or more.

2) *Closed chain*: if a chain is located within a subimage.

The assumption is that there is no object partially located in the whole image. The following properties have been observed.

```
PCC ALGORITHM

input: Let $A_i^k$ be a chain set, where $k$ is the index of layers, $1 \le i \le N$,
       and $N$ is the number of processors on the first layer.
output:The final chain set $A^K$ on the top layer, $K$.
begin
    1. Coding stage:
       Divide the input image into $N$ subimages, $\{ s_i ; 1 \le i \le N \}$
       for all $1 \le i \le N$ pardo
           $A^1_i = coding\ ( s_i )$
       end pardo
    2. Merging stage:
       for $k = 1$ to $K - 1$ do
           for all $i$ in layer $k$ pardo
               $A^{k+1}_{i'} = merging\ (\{ A_i^k \ ;\ i' = parent(i) \})$
           end pardo
       end do
end
```

Figure 4.2 The Parallel Chain-Coding Algorithm in the Pyramid Architecture.

Property 4.1: On the subsequent layers, the closed chains remain closed. However, the open chains may either become closed or remain open.

Property 4.2: The open chains are merged only when they touch on adjacent boundaries.

Property 4.3: Eventually after merging all chains are closed.

Let *parent*(i) denote the parent of node i in the pyramid architecture. Consider a chain set in the (k+1)-th layer, denoted as $A^{k+1}_{i'}$, being merged with a number of chain sets in the k-th layer, A_i^k, $i' = parent(i)$. The parallel chain-coding algorithm (PCC) is stated in Figure 4.2. Typically, two stages, the coding stage and the merging stage, are processed in the PCC algorithm. Inside each stage, the parallel mechanism is applied.

00000001	00000010	00000100
00001000	00000000	00010000
00100000	01000000	10000000

Figure 4.3 The 8-bit index for different position.

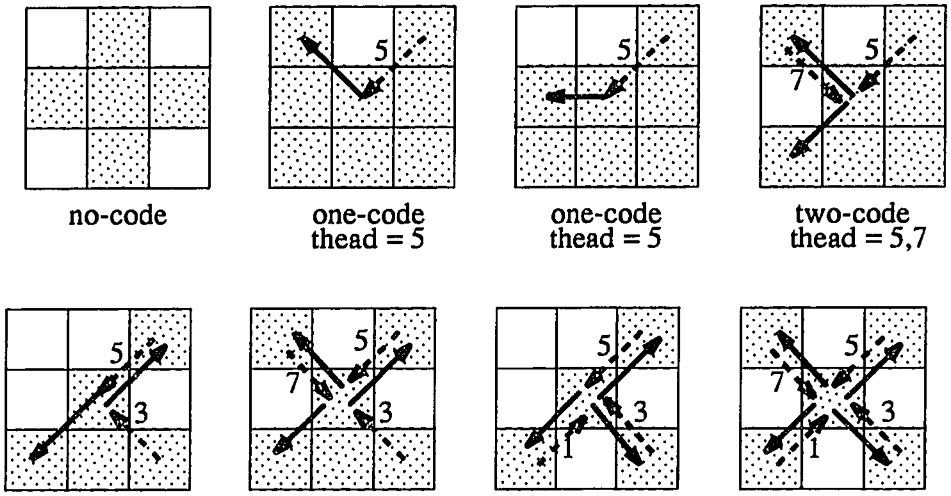
4.3 The Coding Stage

The idea of linking the chain-codes is simply to use a look-up table. Considering a 3×3 mask, we use 8-bit index,

$$I = I_0 I_1 I_2 I_3 I_4 I_5 I_6 I_7 \tag{4.1}$$

to indicate the positions of pixels, where only one of $I_i, i \in \{0,\cdots,7\}$ is "1", and the others are all "0". Each position is represented by an 8-bit index as shown in Figure 4.3. Then, the total index T is the convolution of this 3×3 mask on the processed image. The *codelink* and its associated *thead* present the directions of links, where the codelink is the direction from center pixel to other pixels and the thead is the direction from other pixels to center pixel. The thead is the former direction of the current codelink. Five encoding types of codelinks — no-code, one-code, two-code, three-code and four-code — associated with their theads are illustrated in Figure 4.4. A convolution calculates the total index of each permutation. According to the total index of permutation, a look-up table is set up for codes generation, as shown in Table 4.1. Then, a series of operations are applied to concatenate these individual codelinks to the related chains.

After the look-up table is established, the chain set, A_i^1, is generated by reference to the look-up table. From a codelink, we can determine the relative move in columns and



no-code

one-code
thead = 5

one-code
thead = 5

two-code
thead = 5,7

two-code
thead = 3,5

three-code
thead = 3,5,7

three-code
thead = 3,5,1

four-code
thead = 3,5,7,1

Figure 4.4 Examples of five types of codelinks using the chain code around the central pixel.

rows with respect to the current location. For example, code 0 indicates one-pixel move in x-axis (or column), and no move in y-axis (or row). The destination of the move has an important role when we search for the right link in the *check-head* or *check-tail* step which is discussed next. For each subimage s_i, A_i^1 is initialized as $\emptyset$, and is processed in a raster-scan fashion. If any object pixel is fetched, we can quickly obtain its codelinks simply by the look-up table. Then, the codelinks are joined to A_i^1. Three cases can occur during the process:

1) Link an existing chain.

2) Connect two existing chains.

3) Create a new chain.

Each case can be determined by checking the results from the check-tail and check-head steps which are described in the following paragraph. The process is repeated until the raster-scan is complete.

Table 4.1 The Look-Up Table

Total Index	Number of Codelinks	Thead	Codelink
0	0	Nil	Nil
1	1	7	3
2	1	6	2
3	1	7	2
4	1	5	1
5	2	5,0	1,3
...	...	...	...
130	2	6,3	7,2
131	2	7,3	7,2
132	2	5,3	7,1
133	3	7,5,3	7,3,1
134	2	6,3	7,1
...	...	...	...
251	0	Nil	Nil
252	1	5	4
253	1	5	3
254	0	Nil	Nil
255	0	Nil	Nil

In the check-head step, we match both *Head* coordinates and the first code to each chain with *Tail* coordinates and the thead of the codelink. In the check-tail step, we match both Tail coordinates and thead to each chain with the Head coordinates and the first code of the codelink. Two chains will be joined together into one chain if their codelinks are connected. Because a chain may be open or closed in the intermediate step, three types of concatenation exist:

- *Head concatenation* – concatenate the codelink with the matched chain.
- *Tail concatenation* – concatenate the matched chain with the codelink.
- *Head-and-tail concatenation* – concatenate the Tail's matched chain with the codelink, and then with the Head's matched chain.

If none of the chains satisfies the above conditions, a new chain is created in the chain set, A_i^1. Therefore, in each subimage s_i, the chain set, A_i^1, can be generated by the procedure shown in Figure 4.5. Since the 3×3 window operation is applied everywhere

```
                    Coding PROCEDURE
input: The row image s.
output:The chain set A consists of all chains in s.
begin
        while raster-scan each pixel on s do
        if an object pixel then
              Calculate the index value according to Figure 4.3.
              Obtain codelinks and theads according to Table 4.1.
              for each codelink do
                    Obtain the destination coordinates.
                    Apply the check-head step.
                    Apply the check-tail step.
                    Join to chain set A.
              end do
        end if
        end do
end
```

Figure 4.5 The Coding Procedure.

including the image boundary, the subimage is extended so as to have two more rows and columns of background elements.

4.4 The Merging Stage

After each chain set has been extracted from each subimage by an individual processor, the extracted chain sets need to be merged together. The final results of merging all the chain sets should be functionally identical to the one which is directly extracted from the input image. The rules for merging two chains, A_i and A_j can be defined as follows:

- The Head coordinates of A_i are matched to the Tail coordinates of A_j.
- The thead of A_j is matched to the first code of A_i.

Proposition 4.4: For two adjacent subimages, the open chains on the adjacent boundary should be a one-to-one correspondence, Head-To-Tail or Tail-To-Head.

Proof : Let A_i and A_j be two chain sets with the shared boundary, S. The open chains in A_i with Heads or Tails in S are denoted as L_i, and the open chains in A_j with Heads or Tails in S are denoted as L_j. Then, the number of open chains in L_i is equal to the number of open chains in L_j, and all the open chains of L_i will be merged to those of L_j. Otherwise, some open chains are kept open until the processing is finished, since one open end (Tail or Head) has no chance to merge with other chains in this continuous processing. However, we assume that there are no partial objects in the input image. All chains should be closed after processing. By contradiction, we say the open chains on the adjacent boundary should be one-to-one correspondence. Moreover, only Head can be connected to Tail. We say each pair of correspondence is either a Head-To-Tail connection or a Tail-To-Head connection. □

Definition 4.5: Given the two chain sets A_i and A_j, let $A_i \cup A_j$ be defined as the union chain set, which is the result of merging A_i and A_j if these two chain sets satisfy the proposition above.

Consider the chain set A which is the union of the chain sets from the immediately lower layer. Assume that the chains of A_i $(1 \le i \le m)$ have been consistently sorted by positions of Heads and Tails into L_i for those Heads and Tails located at the adjacent boundary. If the adjacent boundary is in the vertical direction of $x = x_1$, we only sort those chains with Heads and Tails which are at $x = x_1$ according to y coordinates. A sorted L_i contains only the chains corresponding to the neighbor's sorted L_j. A chain set $L_u = L_i \cup L_j$ is defined as merging chains associated with L_i and L_j. Then, a union chain set A_u of A_i and A_j is defined as

$$A_u = A_i - L_i + A_j - L_j + L_u \qquad (4.2)$$

The total chain set A is defined as

```
Merging PROCEDURE

input: The chain sets, $A_1, A_2, \cdots, A_m$.
output:The new chain set $A$ is generated from the source of inputs,
    $A_1, A_2, \cdots, A_m$.
begin
    1. Sort Heads and Tails according to the coordinates only for
       those on the adjacent boundary.
    2. Join Heads and Tails one to one starting from the smallest
       coordinate.
    3. Change the status of the chain (from open to closed) if necessary.
end
```

Figure 4.6 The Merging Procedure.

$$A = \bigcup_{\text{for all adjacent } A_i \text{ and } A_j} A_u \tag{4.3}$$

The procedure for merging several chain sets into a new chain set is shown in Figure 4.6.

4.5 Computational Complexity Analysis

The computational complexity is divided into two stages: the coding stage and the merging stage. The supposition is that there are n pixels in the image and m processors are used, and four connectivity is established between the neighbors of processors. For simplicity, the number of pixels is assumed to be the same as the number of processors, i.e., $n = m$.

In the coding stage, the image is divided into m subimages (i.e., only a single pixel exists in each subimage in the case of $n = m$). Since m processors are used, the chain set $A^1{}_i$ in the first layer can be obtained simultaneously from each subimage. Therefore, the complexity of the coding stage depends on the performance of the coding procedure. In the coding procedure, **while** loop and **for** loop are served as outer and inner loops in the procedure. In accordance with Table 4.1, at most three codelinks are obtained for each object pixel. Since every step in the **for** loop is evaluated in constant time, the complexity of the inner loop is $O(1)$. In the outer loop, since only one pixel exists in each subimage, the raster-scan is performed only once. Every step in the **while** loop is evaluated in constant time including the **for** loop, so the complexity of the coding procedure is $O(1)$, and the complexity of the coding stage is also $O(1)$.

In the merging stage, since four connectivity is established, $\log(m) - 1$ layers are needed for the merging stage. In other words, $\log(m) - 1$ iterations are performed for the outer **for** loop. In this loop, the number of the chain sets $A^k{}_i$ is monotonically decreased, so the number of processors, m, is always greater than the number of the chain sets in different layers. Therefore, the chain sets in the upper layer are always obtained from the chain sets in the lower layer simultaneously, through the merging procedure. The complexity of this stage then depends on the performance of the merging procedure. The first step of the merging procedure is sorting. In layer k, $2^k \times 2^k$ chains at most are possible in each chain set. However, $2^k \times 2^k$ processors are also available for each chain set. The best performance for sorting $2^k \times 2^k$ elements using $2^k \times 2^k$ processors is $O(k)$ with a very large constant factor in a special sorting network architecture. It is not known if $O(k)$ time is possible in some popular architectures such as hypercube, mesh or linear array. Nevertheless, the number of chains usually is much less than $2^k \times 2^k$. Constant time may be achieved if the number of chains is small. In the average case, assuming 2^k chains exist in each chain set, $O(k)$ time is achieved by the hypercube architecture. Other steps in the Merge procedure are all performed in constant time, so the complexity mainly depends on the first step which is $O(k)$ in the average case. The complexity is $O(\log^2(m))$ in the merging stage. Therefore, the overall complexity is $O(\log^2(m))$, also $O(\log^2(n))$ in this case.

Let us compare this to the same algorithm which is run on a single processor. The complexity of the coding stage is $O(n)$ and the complexity of the merging stage is $O(n\log(n))$. Therefore, the overall complexity of the sequential algorithm is then

$O(n\log(n))$, which is larger than the one using multiple processors. The speed up is equal to the complexity using the single processor divided by the complexity using multiprocessors, i.e., $O(n/\log(n))$ while n processors are used. If n is big, then linear speed up may be achieved.

4.6 Experimental Results

In our implementation, we choose an *AP1000* parallel computer and use a *CASIM* simulator to verify the correctness of our parallel algorithm. *CASIM*, the *AP1000* architectural simulator, simulates a parallel program for *AP1000* which is built by Fujitsu Laboratories Ltd. The information evaluated by *CASIM* is traced and listed during parallel processing. *CASIM* provides a variety of libraries for communications in parallel processing, and serves as a parallel language for multiprocessor systems.

In general, multiprocessor systems can be categorized into four classes: shared memory multiprocessors, message passing multicomputers, data flow machines, and multithread machines. *AP1000* is one kind of message passing parallel machine. Processing elements (PEs), cells, are connected with three networks called the broadcast network (B-Net), torus network (T-Net), and synchronization network (S-Net). In accordance with these networks, the communications between host and cells are enabled and parallel processing is achieved. The parallelism is highly dependent on the implemented program. The key point of programming on the *AP1000* is to generate the cell program.

We simulate the pyramid architecture using the communications of different networks between the host and cells. In the host program, the input image is subdivided into subimages according to the number of cells allocated. Then, each subimage is sent to each cell by B-Net. In the cell program, the procedures both in the coding stage and the merging stage are performed. At the coding stage, the associated subimage in each cell is processed by the coding procedure described above. At the merging stage, the resulting chain set from each cell is sent to its parent which is one of the cells located at the

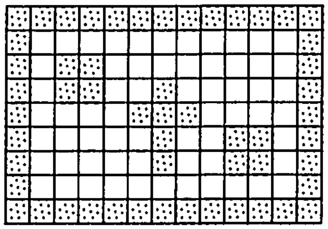


Figure 4.7 The input image with four objects and five contours.

next lower layer. In each parent's cell, once all chain sets are received from its associated cells, they are merged and become a new chain set. Finally, the resulting chain set is obtained at the cell of the top layer.

By simulating the pyramid architecture with four nodes in the first layer, the chain set of the final result in each node is shown. Multiple objects and the internal contour are involved in the input image which is shown in Figure 4.7. The result of chain coding is illustrated in Figure 4.8. A *C* language function *times()* is embedded in the program to compute the used CPU time. For a 64 × 64 input image with 8 contours, it costs 0.1 user time and 0.1 system time in a SUN4M machine.

4.7 Conclusions

This chapter has presented an implementation of a Freeman chain-coding algorithm using parallel machines by the adoption of look-up tables. We have also presented the important issues in the context of a fully coding algorithm using *CASIM*, the simulator of the *AP1000* parallel machine.

First Layer

Node 1

No.	Head Coord.	Tail Coord.	Thead	Type	Code Sequence String
1	(1,5)	(6,1)	4	Open	666644444
2	(7,1)	(1,4)	2	Open	00000122
3	(4,3)	(4,3)	0	Closed	2064

Node 2

No.	Head Coord.	Tail Coord.	Thead	Type	Code Sequence String
1	(6,1)	(13,4)	2	Open	4444444222
2	(13,5)	(7,1)	0	Open	666700000
3	(6,5)	(7,4)	0	Open	5

Node 3

No.	Head Coord.	Tail Coord.	Thead	Type	Code Sequence String
1	(7,9)	(1,5)	6	Open	0000006666
2	(1,4)	(6,9)	4	Open	222234444
3	(7,6)	(6,5)	5	Open	7

Node 4

No.	Head Coord.	Tail Coord.	Thead	Type	Code Sequence String
1	(7,4)	(7,6)	7	Open	3126
2	(13,4)	(7,9)	0	Open	22222000000
3	(11,6)	(11,6)	0	Closed	2064
4	(6,9)	(13,5)	6	Open	4444445666

Second Layer

Node 5 (from Node 1 and 2)

No.	Head Coord.	Tail Coord.	Thead	Type	Code Sequence String
1	(1,5)	(13,4)	2	Open	66664444444444444222
2	(13,5)	(1,4)	2	Open	666700000000000122
3	(4,3)	(4,3)	0	Closed	2064
4	(6,5)	(7,4)	3	Open	5

Node 6 (from Node 3 and 4)

No.	Head Coord.	Tail Coord.	Thead	Type	Code Sequence String
1	(13,4)	(1,5)	6	Open	222220000000000006666
2	(1,4)	(13,5)	6	Open	2222344444444445666
3	(7,4)	(6,5)	5	Open	31267
4	(11,6)	(11,6)	0	Closed	2064

Top Layer

Node 7 (from Node 5 and 6)

No.	Head Coord.	Tail Coord.	Thead	Type	Code Sequence String
1	(13,4)	(13,4)	6	Closed	22222000000000000666666664444444444444222
2	(1,4)	(1,4)	6	Closed	22223444444444456666666700000000000122
3	(4,3)	(4,3)	0	Closed	2064
4	(7,4)	(7,4)	5	Closed	312675
5	(11,6)	(11,6)	0	Closed	2064

Figure 4.8. The chain coding result of the input image from the CASIM parallel simulator.

CHAPTER 5

LINE INVARIANTS UNDER PERSPECTIVE PROJECTION

Most research regarding invariants under perspective projection employs the epipolar geometry which is established by a set of given points. In this chapter, we discuss the invariants among four general lines using isotropy subgroups. Theoretically speaking, two independent invariants exist for four general lines in 3D space. In this chapter, we will demonstrate that these two invariants can be obtained from the projective images of four general lines without reconstructing the 3D structure. That means that given two distinct images of four corresponding lines, then the corresponding geometry of these four line in any given third image can be estimated.

5.1 Perspective Projection

The problem addressed here is to identify solid objects from two ''distinct'' images which may be taken by different cameras and from varying positions. It is human visual perception which allows for the ability to identify objects from different images if a certain portion of their parts is revealed through these images. The problem involved here is the lack of intrinsic information in cameras, such as their focal lens, sampling ratios, and their optical characteristic, and the extrinsic information of cameras, such as their positions and orientations, the status of objects, and the illumination of the light source. By this we mean that objects are rendered by perspective projection, and the calibration among different cameras is unknown.

From a computer graphics point of view, an image of perspective projection [60] [159], can be illustrated by performing the following techniques:

- center of projection,
- inverse parametrization, and
- display transformation.

If the column matrix formulation is chosen, the center of projection is represented by a 4×4 homogeneous matrix; inverse parametrization is represented by a 3×4 homogeneous matrix; and display transformation is represented by a 3×3 homogeneous matrix. The complete perspective projection can thus be represented by the composition of these three matrices. If P is the point of solid objects, and p is its projective image, consider the equation

$$M P = \rho p \tag{5.1}$$

where M is a composited 3×4 camera matrix representing the perspective projection and ρ is a constant.

More specifically, three types of applications are mostly discussed in the problem of Eq. (5.1):

- In computer graphics, given P and M, p is computed;
- In object reconstruction, given p and M, P is computed;
- In camera calibration, given p and P, M is computed.

Unfortunately, in many cases, the information obtained from the scene is only 2D images. Neither the 3D space information of objects, nor the intrinsic and extrinsic information of the camera is available. It is very well known that recovering depth from a single image is not possible [29], but if we use more than one image the challenge becomes feasible.

Given two uncalibrated images, the epipolar geometry can be described by a fundmental matrix F (or an essential matrix E), which is obtained by different linear methods [133] [80] [81] or non-linear methods [134]. Let p_i and p'_i be corresponding points which are the projective points of P_i on two different projective planes. Typically, if several corresponding points p_i and p'_i are given, P_i is obtained by Eq. (5.1),

$$P_i = (MM^T)^{-1} M^T \rho_i p_i \tag{5.2}$$

and

$$P_i = (M'M'^T)^{-1} M'^T \rho'_i p'_i, \tag{5.3}$$

where M and M' are two camera matrices, and ρ_i and ρ'_i are constants. The fundamental matrix F corresponds to a pair of camera matrices M and M' if every point P_i in 3D space, then its corresponding image points p_i and p'_i satisfy the equation,

$$p'^T_i F p_i = 0. \tag{5.4}$$

Thus, the applications under perspective projection become feasible while both the 3D space information of objects and the calibration of the camera are unknown.

Projective invariants can also be found from four corresponding lines on a pair of images [88] [75]. However, most work is to find the invariants depending on epipolar geometry, which is obtained from some corresponding points by Eq. (5.4). It is not feasible to use some corresponding points to find the epipolar geometry first, and then use another four corresponding lines to find the invariants; therefore, the problem becomes clear, either we can use some corresponding lines to find the epipolar geometry, or we can find the invariants from the corresponding lines, independent of the epipolar geometry. Inspired by [232], the isotropy subgroup is another solution to finding the invariants independent of epipolar geometry.

In the following sections, we will investigate projective invariants from the isotropy subgroup of four general corresponding lines. In section 2, we will describe the isotropy transformation of lines. In section 3, we will discuss a complete four-line configuration of four general lines. In section 4, we will present a new projective invariant from four general lines. And section 5, we will conclude this chapter.

5.2 Isotropy Transformation of Lines

Lines are the most robust features extracted from projective images. While solid objects are projected onto a 2D projective plane, the edges of objects are preserved if they are visible from the viewer (or the center of projection). The edges of objects in the 2D projective plane can be obtained by simply applying any edge detector (such as the Canny Edge Detector [31]). Typically, the obtained edges are curved for natural objects or straight for analytical objects. However, it is sufficient to represent both curved edges and straight edges by pieces of line segments. Since pieces of line segments are the most robust features extracted from images, first, a definition of lines will be established, and that will be followed by some discussions on line properties.

Definition 5.1: A *line* is the extension of a line segment from both ends to infinity where the extending parts have the same slope as the line segment in both 3D and 2D space.

Definition 5.2: In the homogeneous coordinate system, the point P in 3D space is represented by $[P^{(1)}, P^{(2)}, P^{(3)}, P^{(4)}]^T$, and the point p in 2D image is represented by $[p^{(1)}, p^{(2)}, p^{(3)}]^T$.

Proposition 5.3: Two distinct points, P_i and P_j, lie on only the one line, L. Let $\vee$ denote an operator which connects two points by a line. Thus, $L = P_i \vee P_j$ represents

$$L = s[P_i^{(1)}, P_i^{(2)}, P_i^{(3)}, P_i^{(4)}]^T + t[P_j^{(1)}, P_j^{(2)}, P_j^{(3)}, P_j^{(4)}]^T. \qquad (5.5)$$

Proposition 5.5: Two distinct lines, L_i and L_j, meet at only the one point, P. Let $\wedge$ denote an operator which intersects two lines at a point. Thus, $P = L_i \wedge L_j$.

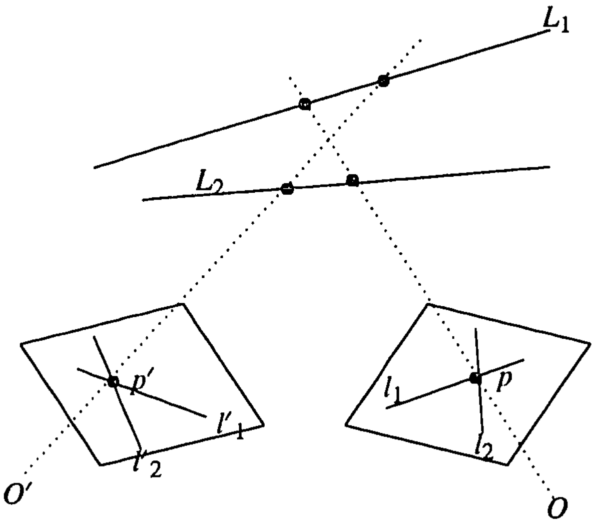


Figure 5.1 Two intersecting lines on two different projective planes may not intersect each other while they are in 3-D space. In most cases, the intersections, p and p' are not corresponding points.

Proposition 5.6: Two intersecting lines in 2D space may not intersect each other in 3D space.

Proposition 5.7: If two skewed lines in 3D space are projected onto any 2D projective plane, their projective lines intersect at one point only.

The illustration in Figure 5.1 shows that the observed intersections of two lines from two different images may not be the corresponding points. As a matter of fact, the corresponding points are usually too difficult to obtain from two different images. This chapter introduces an approach to finding projective invariants from lines.

Definition 5.8: An isotropy subgroup is the set of elements of a transformation subgroup which do not alter a configuration [232].

For example, a line is unaffected by a translation in the direction of the line.

Let L_{i}, where $i \in \{1, 2, 3, 4\}$, be four general lines in 3D space. That is, each line is skewed to the other three, and no two of these four lines are coplanar.

Lemma 5.9: Suppose that there is an isotropy subgroup for a configuration of four general lines. Under the action of the isotropy subgroup, the points on the line L_i need not be fixed, but the transformed points must still lie on L_i [75].

Corollary 5.10 (Isotropy subgroup theorem): By Lemma 5.9, we can obtain a one-dimensional transformation matrix, T, which guarantees that every point P_i on line L_i after transformation is still on L_i for all $i \in \{1, 2, 3, 4\}$. If a 4x4 matrix ,T, is considered, we may assign values μ, μ, ν, and ν to the diagonal of T, and zero to all other elements of T, that is

$$T = \begin{bmatrix} \mu & 0 & 0 & 0 \\ 0 & \mu & 0 & 0 \\ 0 & 0 & \nu & 0 \\ 0 & 0 & 0 & \nu \end{bmatrix}. \tag{5.6}$$

Corollary 5.11: Let T be the isotropy transformation matrix of four general lines, while P_i and P_j are two distinct points. One can show that

$$T(P_i \vee P_j) = TP_i \vee TP_j. \tag{5.7}$$

Proof : Let $L = P_i \vee P_j$. Using the *Isotropy subgroup theorem*, the left hand side of the equation, $T(P_i \vee P_j) = TL$ is L itself. Looking at the right hand side of the equation, since P_i and P_j are points on L, by *Isotropy subgroup theorem*, TP_i and TP_j are also points on L. Thus, $TP_i \vee TP_j = L$. It is evident that both sides represent the same line. □

Corollary 5.12: Let T be the isotropy transformation matrix of four general lines, while L_i and L_j are two distinct lines. One can show that

$$T(L_i \wedge L_j) = TL_i \wedge TL_j. \tag{5.8}$$

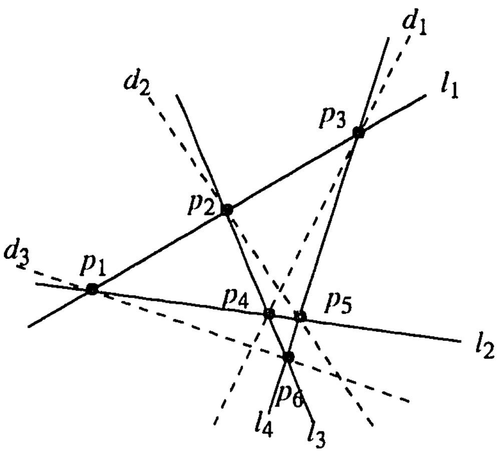


Figure 5.2 A complete four-line.

Proof : Let $P = L_i \wedge L_j$. Looking at the left hand side of the equation, since P is a point both on L_i and L_j, by the *Isotropy subgroup theorem*, $T(L_i \wedge L_j) = TP$ is a point both on L_i and L_j. By Proposition 5.4, we know $TP = P$. Using the *Isotropy subgroup theorem*, the right hand side of the equation, TL_i and TL_j are also L_i and L_j, respectively. Thus, $TL_i \wedge TL_j = L_i \wedge L_j = P$. It is evident that both sides represent the same point. □

5.3 Configuration of Lines

Suppose that a pair of projective images of L_{i}, $i \in \{1, 2, 3, 4\}$ exists then, we can find that l_i, $i \in \{1, 2, 3, 4\}$, are four projective lines of L_i on one image, and l'_i, $i \in \{1, 2, 3, 4\}$, are four projective lines of L_i on the other image. Four lines on a projective plane can construct various configurations. However, the most general case is called a complete four-line.

Definition 5.13: In a projective plane, the configuration consisting of four lines, l_i, $i \in \{1, 2, 3, 4\}$, six points, p_i, $i \in \{1, 2, 3, 4, 5, 6\}$, and three diagonal lines, d_i, $i \in \{1, 2, 3\}$, is called a complete four-line as shown in Figure 5.2.

In other words, no two of four projective lines are parallel in a complete four-line configuration. Therefore, we assume that two complete four-line configurations are formed by l_i and l'_i, $i \in \{1, 2, 3, 4\}$, on the images.

Since L_i, l_i and the center of projection O are coplanar, we may know that there are four planes, π_i, $i \in \{1, 2, 3, 4\}$, consisting of L_i and l_i; and all these four planes intersect at one point, O. Considering π_1, we also know that there are exactly three intersections with L_2, L_3 and L_4, and that they are denoted by $P_{2,1}$, $P_{3,1}$, and $P_{4,1}$. If we connect lines from $P_{i,1}$, $i \in \{2, 3, 4\}$, to p_1, p_2 and p_3 on l_1 as shown in Figure 5.3, these three lines are the projectors of projection, and meet at the center of projection, O.

If we properly assign the homogeneous coordinates $[1, 1, 1, 1]^T$ to O, π_i (containing L_i and l_i) can be represented by the following:

Proposition 5.14: For every plane π in 3D space, there are real constants $\pi^{(1)}$, $\pi^{(2)}$, $\pi^{(3)}$, and $\pi^{(4)}$, not all of which are zero, such that P lies on π if and only if the equation

$$\pi^{(1)}P^{(1)} + \pi^{(2)}P^{(2)} + \pi^{(3)}P^{(3)} + \pi^{(4)}P^{(4)} = 0 \tag{5.9}$$

is satisfied by the homogeneous coordinates $P = [P^{(1)}, P^{(2)}, P^{(3)}, P^{(4)}]^T$. The equation associated with the plane is the equation of π.

We may simply provide plane equations to π_1 by $\pi_1{}^{(1)} = 0$, π_2 by $\pi_2{}^{(2)} = 0$, π_3 by $\pi_3{}^{(3)} = 0$, and π_4 by $\pi_4{}^{(4)} = 0$. Thus, when considering the coordinates $[P^{(1)}_{i,1}, P^{(2)}_{i,1}, P^{(3)}_{i,1}, P^{(4)}_{i,1}]^T$ of $P_{i,1}$ on π_1, these can be simplified by $[1, P^{(2)}_{i,1}, P^{(3)}_{i,1}, P^{(4)}_{i,1}]^T$. Recalling the *Isotropy subgroup theorem*, since $P_{i,1}$ are on line L_i, after the isotropy transformation, the new points $TP_{i,1}$ still remain on line L_i for $i \in \{2, 3, 4\}$. Applying Eq. (5.6),

$$TP_{i,1} = \begin{bmatrix} \mu & 0 & 0 & 0 \\ 0 & \mu & 0 & 0 \\ 0 & 0 & \nu & 0 \\ 0 & 0 & 0 & \nu \end{bmatrix} \begin{bmatrix} 1 \\ P^{(2)}_{i,1} \\ P^{(3)}_{i,1} \\ P^{(4)}_{i,1} \end{bmatrix}, \tag{5.10}$$

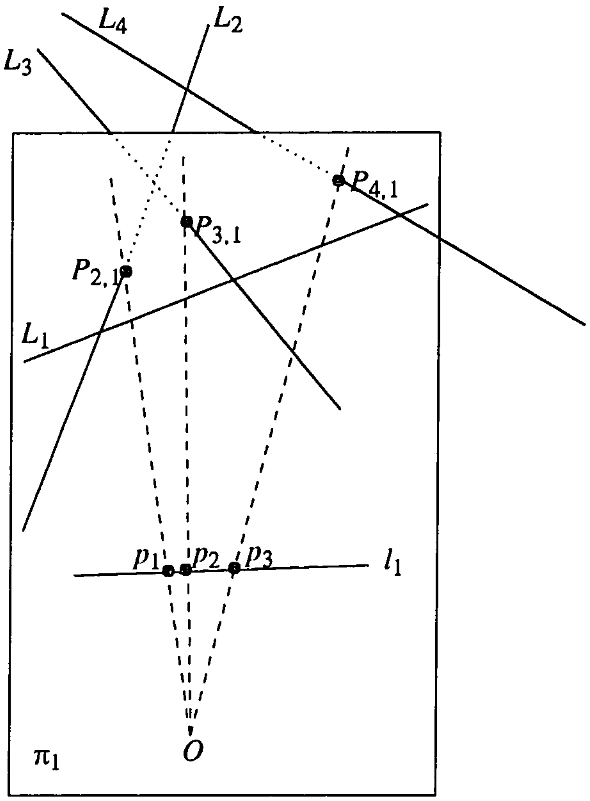


Figure 5.3 On π_1 plane, $P_{i,1}$ three intersections with L_i, $i \in \{2, 3, 4\}$, are connected to p_i, $i \in \{1, 2, 3\}$, on line l_1 by three dashed lines. Since these three dashed lines are the projectors, they meet at the center of projection, O.

the new coordinates of $TP_{i,1}$ are $[\mu, \mu P^{(2)}_{i,1}, \nu P^{(3)}_{i,1}, \nu P^{(4)}_{i,1}]^T$. In addition, the center of projection, O, is transformed to $O' = [\mu, \mu, \nu, \nu]$, and all points on L_1 are still on L_1 with the first coordinate equal to μ. Since the first coordinate of O', $TP_{i,1}$, $i \in \{2, 3, 4\}$, are all equal to μ, we say they are coplanar at plane π'_1, which contains L_1 and l'_1. Lines $O' \vee (TP_{i,1})$, $i \in \{2, 3, 4\}$, are the projectors of O′ which pass through points p'_1, p'_2, and p'_3.

Suppose that M and M' are camera matrices of two views, whose centers of projection are O and O', in accordance with Eq. (5.1), we have

$$MP_{2,1} = \rho_{2,1} p_1, \; MP_{3,1} = \rho_{3,1} p_2, \; MP_{4,1} = \rho_{4,1} p_3, \tag{5.11}$$

$$M' \, TP_{2,1} = \rho'_{2,1} p'_1, \; M' \, TP_{3,1} = \rho'_{3,1} p'_2, \; M' \, TP_{4,1} = \rho'_{4,1} p'_3, \tag{5.12}$$

on plane π_1,

$$MP_{1,2} = \rho_{1,2} p_1, \; MP_{3,2} = \rho_{3,2} p_4, \; MP_{4,2} = \rho_{4,2} p_5, \tag{5.13}$$

$$M' \, TP_{1,2} = \rho'_{1,2} p'_1, \; M' \, TP_{3,2} = \rho'_{3,2} p'_4, \; M' \, TP_{4,2} = \rho'_{4,2} p'_5, \tag{5.14}$$

on plane π_2,

$$MP_{1,3} = \rho_{1,3} p_2, \; MP_{2,3} = \rho_{2,3} p_4, \; MP_{4,3} = \rho_{4,3} p_6, \tag{5.15}$$

$$M' \, TP_{1,3} = \rho'_{1,3} p'_2, \; M' \, TP_{2,3} = \rho'_{2,3} p'_4, \; M' \, TP_{4,3} = \rho'_{4,3} p'_6, \tag{5.16}$$

on plane π_3, and

$$MP_{1,4} = \rho_{1,4} p_3, \; MP_{2,4} = \rho_{2,4} p_5, \; MP_{3,4} = \rho_{3,4} p_6, \tag{5.17}$$

$$M' \, TP_{1,4} = \rho'_{1,4} p'_3, \; M' \, TP_{2,4} = \rho'_{2,4} p'_5, \; M' \, TP_{3,4} = \rho'_{3,4} p'_6, \tag{5.18}$$

on plane π_4.

Definition 5.15: The planes, π_i, $i \in \{1, 2, 3, 4\}$, are called canonical planes. The planes, π'_i, $i \in \{1, 2, 3, 4\}$, are called the associated planes of π_i followed by an isotropy transformation.

Lemma 5.16: For any canonical plane π_i and its associated plane π_i', π_i consists of l_i and π_i' consists of l_i', and π_i and π_i' meet at one line L_i. If there is one line l_j intersecting l_i at point p_k, we can always find an intersection p'_k of two corresponding lines l'_i and l'_j, which satisfies the following equation

$$p'_k{}^T G p_k = 0, \tag{5.19}$$

where G is a 3×3 transformation matrix.

Proof : Since we can always find the point $P_{j,i}$ which is the intersection of L_j with π, by the *Isotropy subgroup theorem*, $TP_{j,i}$ and L_i are coplanar on π'. Suppose that M and M' are camera matrices of two views. By Eq. (5.1), we have

$$M\, P_{j,i} = \rho_{j,i}\, p_k, \tag{5.20}$$

and

$$M'\, TP_{j,i} = \rho'_{j,i}\, p'_k. \tag{5.21}$$

Since M and M' are 3×4 matrices and T is a 4×4 matrix, the equation $p'_k{}^T G p_k = 0$ is obtained and G is a 3×3 transformation matrix. □

Suppose that we have the corresponding complete four-lines on a pair of images, a 3×3 transformation for all six pairs of p_i and p'_i, $i \in \{1, 2, 3, 4, 5, 6\}$ exists. It is noted that these six pairs of intersections are not the corresponding points described by epipolar geometry.

5.4 Projective Invariants of Lines

Suppose that a pair of complete four-line configurations is given by a pair of images. When a fifth line is observed on the first image, a corresponding line needs to be found on the second image. In general, if the line l intersects a complete four-line with four intersections, x_i, $i \in \{1, 2, 3, 4\}$, as shown in Figure 5.4, it is most likely that the corresponding line l' will intersect a complete four-line with four intersections, x'_i, $i \in \{1, 2, 3, 4\}$. It is easy to find x'_i from x_i, but, the difficulty is that x'_i is not the corresponding point of x_i.

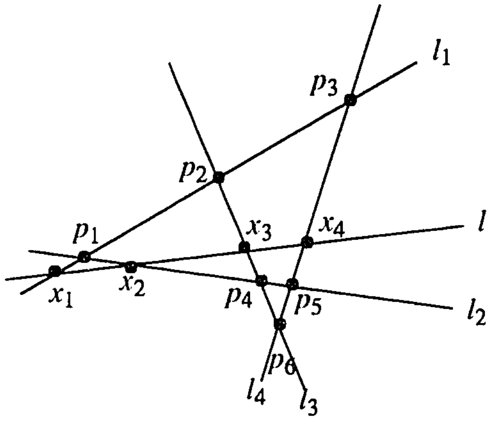


Figure 5.4 Given any line l, four intersections x_i, $i \in \{1, 2, 3, 4\}$, the complete four-line is obtained.

Fortunately, by isotropy subgroup transformation, x'_i can be obtained from x_i for all $i \in \{1, 2, 3, 4\}$.

Definition 5.17: For any four points P_1, P_2, P_3 and P_4 on the same line, the cross ratio (P_1, P_2, P_3, P_4) is defined by

$$\frac{(P_1 - P_2)}{(P_1 - P_3)} \cdot \frac{(P_4 - P_3)}{(P_4 - P_2)}. \tag{5.22}$$

Theorem 5.18 (Line invariant theorem): The cross ratios

$$(x_1, p_1, p_2, p_3) = (x'_1, p'_1, p'_2, p'_3), \tag{5.23}$$

$$(x_2, p_1, p_4, p_5) = (x'_2, p'_1, p'_4, p'_5), \tag{5.24}$$

$$(x_3, p_2, p_4, p_6) = (x'_3, p'_2, p'_4, p'_6), \tag{5.25}$$

and

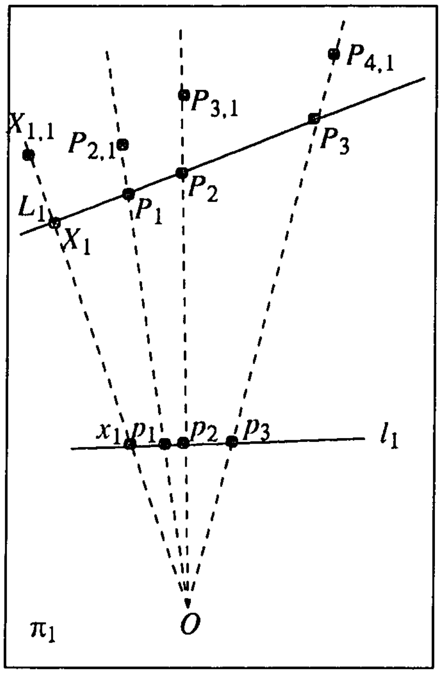


Figure 5.5 On π_1 plane, projectors Op_i and Ox_1 (four dashed lines) intersect with L_1 at P_i and X_1, $i \in \{1, 2, 3\}$.

$$(x_4, p_3, p_5, p_6) = (x'_4, p'_3, p'_5, p'_6), \tag{5.26}$$

are invariants under perspective projection.

Proof : Once again, considering only on the plane π_1 as described in the last section, four projectors Ox_1 and Op_i, intersected L_1 at X_1 and P_i, $i \in \{1, 2, 3\}$ are observed in Figure 5.5. If $X_{1,1}$ and $P_{i,1}$ are the intersections of four lines L and L_i, $i \in \{2, 3, 4\}$ with the plane π_1, X_1, P_1, P_2 and P_3 can be represented by

$$X_1 = L_1 \wedge (O \vee X_{1,1}),\ P_1 = L_1 \wedge (O \vee P_{2,1}),$$

$$P_2 = L_1 \wedge (O \vee P_{3,1}), \text{and } P_3 = L_1 \wedge (O \vee P_{4,1}). \tag{5.27}$$

It is obvious that

$$(x_1, p_1, p_2, p_3) = (X_1, P_1, P_2, P_3). \tag{5.28}$$

Substituting Eq. (5.27) to Eq. (5.28), we obtain

$$(L_1 \wedge (O \vee X_{1,1}), L_1 \wedge (O \vee P_{2,1}), L_1 \wedge (O \vee P_{3,1}), L_1 \wedge (O \vee P_{4,1})) = \tag{5.29}$$

$$\frac{L_1 \wedge (O \vee X_{1,1} - L_1 \wedge (O \vee P_{2,1})}{L_1 \wedge (O \vee X_{1,1} - L_1 \wedge (O \vee P_{3,1})} \cdot \frac{L_1 \wedge (O \vee P_{4,1}) - L_1 \wedge (O \vee P_{3,1})}{L_1 \wedge (O \vee P_{4,1}) - L_1 \wedge (O \vee P_{2,1})} \tag{5.30}$$

If we multiply every term by the isotropy transformation matrix T, then (x_1, p_1, p_2, p_3) is equal to

$$\frac{T(L_1 \wedge (O \vee X_{1,1}) - T(L_1 \wedge (O \vee P_{2,1}))}{T(L_1 \wedge (O \vee X_{1,1}) - T(L_1 \wedge (O \vee P_{3,1}))} \cdot \frac{T(L_1 \wedge (O \vee P_{4,1})) - T(L_1 \wedge (O \vee P_{3,1}))}{T(L_1 \wedge (O \vee P_{4,1})) - T(L_1 \wedge (O \vee P_{2,1}))} \tag{5.31}$$

Recalling the *Isotropy subgroup theorem*, $TL_1 = L_1$, $O' = TO$. By Corollary 5.11 and Corollary 5.12. Eq. (5.31) becomes

$$\frac{L_1 \wedge (O' \vee X'_{1,1}) - L_1 \wedge (O' \vee P'_{2,1})}{L_1 \wedge (O' \vee X'_{1,1}) - L_1 \wedge (O' \vee P'_{3,1})} \cdot \frac{L_1 \wedge (O' \vee P'_{4,1}) - L_1 \wedge (O' \vee P'_{3,1})}{L_1 \wedge (O' \vee P'_{4,1}) - L_1 \wedge (O' \vee P'_{2,1})}, \tag{5.32}$$

when $X'_{1,1} = TX_{1,1}$, $P'_{2,1} = TP_{2,1}$, $P'_{3,1} = TP_{3,1}$, and $P'_{4,1} = TP_{4,1}$, respectively. Since $O' \vee (TX_{1,1})$ and $(O' \vee (TP_{i,1})$, $i \in = \{2, 3, 4\}$, are the projectors of the second image whose center of projection is O', we simply obtain $x_1{}'$ on l'_1 by Eq. (5.23). Similarly, points x'_2, x'_3 and x'_4 also can be obtained by Eqs. (5.24), (5.25) and (5.26), respectively. □

5.5 Conclusions

In this chapter, new invariants under perspective projection have been represented by four general lines. In contrast to other researchers who inquire into epipolar geometry, we investigated our projective invariants by isotropy subgroups. Firstly, we discussed that fact that the isotropy subgroup of four general lines can be represented by a 4×4 isotropy transformation matrix with only a diagonal containing values. Secondly, we used a 3x3 transformation matrix to describe the relationship between a pair of corresponding complete four-line configurations on two different images. Finally, given any fifth line across a complete four-line configuration with four intersections on one image, the corresponding line on the other image can be obtained by computing the intersections with the corresponding complete four-line. For applications of this, the reader should refer to the subsequent chapter.

CHAPTER 6

APPLICATIONS FOR RECOGNITION USING PROJECTIVE INVARIANCE

In this chapter, a projective invariant recognition system based on hypothesis-generation-test scheme will be described. Instead of using the conventional machine to compute the complex projective invariant recognition system, the system proposed here uses an equivalent parallel algorithm run on hypercube parallel architecture. Object recognition is achieved by matching the scene projective invariants to the model projective invariants, called *transfer*. A hypercube parallel architecture is proposed to implement the hypothesis-generation-test scheme. The projective invariance associated with the hypothesis-generation-test is used to enforce the matching constraints including match validity and geometry structure preservation. Experimental results on several projective images will be presented here to demonstrate the proposed approach.

6.1 Projective Invariance

Recently, an interest in the theory of projective invariants has emerged in vision to represent object shapes and their robustness in performing recognition tasks. In general, if P is a point in 3D space and p is its image, a perspective projection equation can be written as

$$MP = \rho p \tag{6.1}$$

where M is a composited 3×4 homogeneous matrix and ρ is a constant. Given a set of point correspondences of two or more images, a fundamental matrix F, (known as the essential matrix E), can be determined up to a collineation to represent the epipolar geometry [133] [80] [56] [134]. This supposes that some mathematical quantity, defined as a function of the scene geometry, may be computed, and that this quantity is unchanged under projective transformation in space. Such a quantity is called a projective invariant of the object geometry.

In the following subsections, two basic projective invariants and various extended invariants are discussed. Detailed proof of the following theorems is beyond the scope of this dissertation. Interested readers may refer to associated references.

6.1.1 The Cross Ratio

As is well known, the ratio of ratios of length, called the cross ratio, is invariant under perspective projection.

Definition 6.1: The cross ratio is defined as

$$(p_1, p_2, p_3, p_4) = \frac{(p_3 - p_1)(p_4 - p_2)}{(p_3 - p_2)(p_4 - p_1)}, \tag{6.2}$$

where p_i, $i \in \{1, 2, 3, 4\}$, represent points, and $p_i - p_j$ is the length between p_i and p_j.

Theorem 6.2: [150] [151] It is invariant under projective transformations in the sense that if T is any projective transformation from a projective line l_1 to a projective line l_2 as shown in Figure 6.1, then

$$(p_1, p_2, p_3, p_4) = (Tp_1, Tp_2, Tp_3, Tp_4). \tag{6.3}$$

In general, there are 24 cross ratios, six of which are numerically different. Letting $\tau = (p_1, p_2, p_3, p_4)$, other permutations are

$$\{\tau, \frac{1}{\tau}, 1-\tau, \frac{1}{1-\tau}, \frac{\tau-1}{\tau}, \frac{\tau}{\tau-1}\}, \tag{6.4}$$

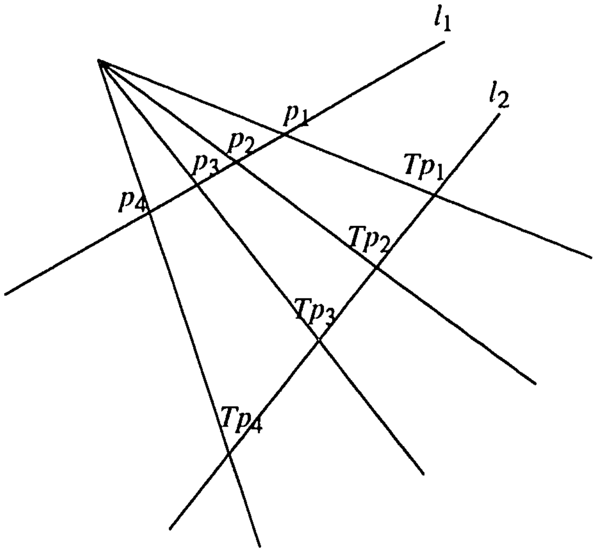


Figure 6.1 The cross ratios of sets of four points on the perspective projection are equivalent.

The cross ratio is the most popular projective invariant. A variety of cross ratios has been developed by researchers:

- The cross ratio of length from four collinear points [11] [12] (a pencil of four lines [144]) or conic [144].
- The cross ratio of vectors from a pencil of four planes [144] or four lines [90].
- The cross ratio of area from five coplanar noncollinear points [12] [148].
- The cross ratio of volume from six noncoplanar noncollinear points [12] [148].

6.1.2 The Coplanarity

Another useful invariant is the coplanarity. From observation, a set of coplanar points after a projective transformation is also on the same plane. Indeed, an arbitrary projective transformation of the plane can be represented as a non-singular linear transformation. The fundamental theorem of plane projectivity is as follows:

Theorem 6.3: [150] [151] If p_i and p_i', $i \in \{1, 2, 3, 4\}$ are two sets of four coplanar points in space, with no three points in either set being linearly dependent, there exists a non-singular linear transformation T such that $Tp_i = \rho_i p_i'$, $i = 1, ..., 4$, here the ρ_i are scalars; and the matrix T is uniquely determined apart from a scalar factor.

The mapping (homography) of points with points is called a collineation. The term collineation implies that any projective transformation of the plane preserves the collinearity of a set of points. The projective transformation matrix, T, requires eight independent parameters to define a unique mapping. Since each point in the plane provides two Cartesian coordinate equations, it is necessary to find four point correspondences between two projectively transformed planes in order to define the transformation matrix uniquely.

6.1.3 The State-of-the-Art Investigations

In most cases, the fundamental matrix, F, is computed from points matches between the perspective images. Then, invariants of the 3D configuration are computed either from the recovered 3D structure [56] [81] or from image measurements together with the epipolar geometry [90] [144] [185] [75]. However, some other approaches to computing invariants only from the elements of the projected configuration without the epipolar geometry are also presented by [232]. A summary of projective invariants is illustrated in Table 6.1.

Recently, four groups of researchers have been studying this subject. Furthermore, some independent researchers are also investigating new projective invariants. [32] [147] [180] [117] [164] [194].

INRIA. Much excellent work has been developed by the researchers in *INRIA*. Basically, two different approaches to invariances are been investigated by INRIA researchers.

Table 6.1 A summary of projective invariants. Let n_v denote the number of views and n_i denote the number of invariants.

Types	Structures	n_v	n_i	References
cross ratios	6 points	2	3	[75]
	6 points (4 being coplanar)	2	2	[232] [45]
	5 points (4 being coplanar)	2	0	[232]
	4 points + 2 epipoles	2	1	[183] [188]
	5 points + 1 epipole	2	2	[56]
	4 lines	2	2	[90] [75]
	4 lines (2 of 4 being coplanar)	2	1	[75]
	4 lines (2 pairs of coplanar lines)	2	1	[75]
	4 points + 1 lines	2	1	[75]
	3 points + 2 lines	2	2	[75]
	2 points + 3 lines	2	3	[75]
	4 coplanar points + 1 line	2	2	[232]
recovered 3D coordinates	6 points	2	3	[81] [90]
	7 points	2	6	[164]
cross ratio + coordinates	6 points	2	3	[13]
relationship between 3D and 2D	6 points	3	10	[167] [168]

Some standard projective bases are chosen by the Faugeras' group [56] [75] [134] [135] [57] [176] [137] [167] [136] [46] [47] [58] so as to reconstruct the epipolar geometry. Given five point correspondences in two images, the standard projective bases for 3D points are

$$[1,0,0,0]^T,\ [0,1,0,0]^T,\ [0,0,1,0]^T,\ [0,0,0,1]^T, \text{ and } [1,1,1,1]^T. \qquad (6.5)$$

The standard projective bases for 2D points are

$$[1, 0, 0]^T,\ [0, 1, 0]^T,\ [0, 0, 1]^T,\ [1, 1, 1]^T, \text{ and } [\alpha, \beta, \gamma]^T. \tag{6.6}$$

Thus, the camera matrix M in Eq. (6.1) can be decomposed into

$$M = \rho_1 M_1 + \rho_2 M_2, \tag{6.7}$$

where

$$M_1 = \begin{bmatrix} \alpha & 0 & 0 & 0 \\ 0 & \beta & 0 & 0 \\ 0 & 0 & \gamma & 0 \end{bmatrix} \text{ and } M_2 = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix}. \tag{6.8}$$

Thus, the center of projection (COP) and the epipoles of both images can be easily determined by the decomposed matrices.

Different from the approach of the Faugeras' group, the method used in Mohr's group [145] [146] [95] [25] is formulated globally as a least squares estimation method which does not need epipolar geometry. Eq. (6.1) is usually written in the following way, hiding the scaling factor, using the non homogeneous coordinates of image points:

$$p^{(1)} = \frac{M^{(11)}P^{(1)} + M^{(12)}P^{(2)} + M^{(13)}P^{(3)} + M^{(14)}P^{(4)}}{M^{(31)}P^{(1)} + M^{(32)}P^{(2)} + M^{(33)}P^{(3)} + M^{(34)}P^{(4)}} \tag{6.9}$$

$$p^{(2)} = \frac{M^{(21)}P^{(1)} + M^{(22)}P^{(2)} + M^{(23)}P^{(3)} + M^{(24)}P^{(4)}}{M^{(31)}P^{(1)} + M^{(32)}P^{(2)} + M^{(33)}P^{(3)} + M^{(34)}P^{(4)}}, \tag{6.10}$$

where $M^{(ij)}$ is the i row, jth column element of the camera matrix, M.

Hartley's Group. [80] [81] [86] [82] [83] [84] [90] [88] [89] [87] [91] Here, image correspondences are used to compute the fundamental matrix F. We suppose that two camera matrices M and M' are represented by $(I|0)$ and $(\hat{M}|t)$, where I is a 3x3 identity matrix, 0 is a 3x1 vector with all element 0, $\hat{M}$ is a 3x3 matrix, and t is a 3x1 vector. Then a factorization $F = [t]_{\times}\hat{M}$ is established. The object is constructed up to a collineation. Given a set of six points, a coordinate system may be selected in which the first five points have coordinates

$$[1,0,0,0]^T,\ [0,1,0,0]^T,\ [0,0,1,0]^T,\ [0,0,0,1]^T, \text{ and } [1,1,1,1]^T. \tag{6.11}$$

The coordinates of the sixth point give rise to three independent projective invariants of the six points.

Mundy and Zisserman's Group. [151] [45] [152] [232] [61] [14] For six 3D points in general position there are three functionally independent scalar projective invariants

$$I_1 = \frac{|l_{3561}||l_{3542}|}{|l_{3564}||l_{3512}|},\ I_2 = \frac{|l_{3562}||l_{3142}|}{|l_{3512}||l_{3642}|},\ I_3 = \frac{|l_{3564}||l_{5612}|}{|l_{3561}||l_{5642}|}, \tag{6.12}$$

where $l_{abcd} = [P_a, P_b, P_c, P_d]$ and $|\cdot|$ is the determinant.

Shashua's Group. [183] [184] [185] [189] [188] [187] [190] [186] Here, an invariance relation is captured by a single equation relating image points correspondences across two or more views and the homographies of two arbitrary virtual planes.

Theorem 6.4: [188] Let A, E be the two projective homographies from one image to the other image due to having two distinct planes in space. Then for any corresponding pair p and p' coming from a point P in space, the following equation

$$p' \approx Ap + kEp, \tag{6.13}$$

indicating the scalar k, called projective depth, is a projective invariant.

6.2 Transfer Technique

Since object geometry is represented by a set of corresponding features taken from two views, the projected geometry can be constructed in any third view. In other words, the equivalence class of images of an object undergoing rigid transformation can be captured by any two of its images. The advantage of this approach is that the actual reconstruction of the 3D structure is not required. The geometric relation between objects and their views is used for performing recognition. Therefore, recognition is achieved if the reprojected image is successfully matched against the third image. The problem of predicting a target view from a set of model views using a limited number of corresponding points is called *transfer* [11] [12] [13] [77].

The traditional approach to transfer based on the epipolar geometry obtains the point in a target image by intersecting epipolar lines computed between image pairs. Typically, the 3D structure together with 3x4 projection matrices are computed, and then the imaged point is obtained by projecting the 3D point onto a target image [202] [146] [85] [89]. However, the main drawback is that the process fails when epipolar lines are coincident. For example, numerical instabilities arise when the centers of projection of the three cameras are nearly collinear, or, equivalently, when the object rotates around nearly the same axis for all views [151] [58] [45] [12] [11].

The other approach to transfer based on the relative structure of the object determines the location of object points in target views. From *Theorem 6.4*, a simple method avoiding instability is investigated by [183] [184] [185] [188]. According to two of the same planes in space, given the homographies A and E between first view and second and given the homographies A' and E' between the first view and a target view, the obvious use of Eq. (6.13) is that

$$p' \approx Ap + kEp \tag{6.14}$$

$$p'' \approx A'p + kE'p, \tag{6.15}$$

where p, p', p'' are corresponding points. Given p, p' we can first recover k, and then, by substitution in the second equation, the location of p'' is obtained.

In addition to the geometry expression, the algebraic functions are developed so as to express the transfer. In general cases, a trilinear relationship between three perspective views has been shown to exist [187] [58]. In case where two of the views are orthographic and the third is perspective the function is reduced to a bilinear form [185]. In the case where all three views are orthographic, the function reduces further to a linear form [212]. In the case of calibrated cameras, a quadratic function is equivalent to intersecting epipolar lines [11] [12] [13] [150] [190]. If line correspondences are used, the algebraic function is also designed by [88].

6.3 Matching Method

Without losing a sense of generality, we take the transfer method described above by Shashua [183] [184] [185] [188] in order to demonstrate our matching scheme. However, other methods may also be adopted for our matching scheme with very minor efforts. The matching scheme can be realized by storing two model views, and, with the ''help'' of corresponding points between model views and any target view. Thus the object is reprojected onto a target viewing position. The geometric source of variability then can be factored out during the recognition process by matching a few points between the model of an object and the input image.

Problems arise with respect to the relationship between model views and a target view. If there is no ''help'' to determine the corresponding points between model views and a target view, the problems become critical. Let us suppose that n points across three views are extracted. We can easily determine the corresponding points between two model views. However, if no further information is used, the correspondences between model views and a target view are unknown. We need to try all possible pairs and verify each transfer by matching the transformed model with a target. Let us suppose that an m point structure is constructed for the projective invariance. It means that up to $C(n,m)$ pairs are possible correspondences; the complexity of such a scheme is not very attractive since it is approximately $O(n^m)$ in the case of a pair of correspondences to be computed. In most cases, m is much less than n.

In order to reduce this complexity and to allow simultaneous processing of all known models, we propose a hypothesis-generation-test scheme. Different from geometric hashing [227] [228] and alignment [98] [99] [100] which can only deal with Euclidean transformation and affine transformation, our scheme can recognize objects under projective transformation. Instead of storing all possible pairs of geometric relationship in a model base, we simply store the coordinates of corresponding points. Suppose that n points are extracted, only $4n$ coordinates need to be stored for each model. Unlike geometric hashing and alignment, they need to store $O(n^2)$ data for rigid transformation, $O(n^3)$ for similarity transformation and $O(n^4)$ for affine transformation.

The goal of the hypothesis-generation-test scheme is to match in a template, one or several models while allowing different transformations (ignoring incompleteness in this case). A process of hypothesis generation is used to build hypotheses between extracted features of the template and those of stored models. A hypothesis test is, then, performed by verifying generated multiple hypotheses.

6.3.1 Hypothesis Generation

The task of hypothesis generation is to establish possible matching candidates between features of stored models and those of the unknown target. Randomly picking up a set of m points from a target view, we can compare it to other $C(n, m)$ sets of m points from a single model. The projective invariant between models is established by *Theorem 6.4* captured in Eq. (6.13). There are projective depths, k computed from every m point structure between two model images and k' computed from every m point structure between the target image and one model image. In total, we have $C(n, m)$ pairs of k and k' projective depths. Since noise is inevitably encountered, the projective invariants obtained by the aforementioned methodology should be compiled with a specified tolerance for the sake of a fair matching.

Those of stored models whose projective depths are the same as the projective depths between model and target can be retrieved from a model base to generate the hypotheses. Since two model images are stored for the same objects, then two different projective depths, k_1' and k_2' between models and target are obtained. The difference, δ, between models and target can be measured by the following equation:

$$\delta = k^2 - k_1' * k_2' \tag{6.16}$$

The δ of all projective depth pairs is greater than some certain threshold δ_t. It is selected as the hypotheses, h_j, $j \in \{1, ..., M_i\}$, where M_i is the number of hypotheses generated for the ith model. The resulting M hypotheses H are divided into clusters with respect to their models. For the ith model, its M_i hypotheses can be expressed as

$$H_i = \{{}^ih_1 \ , {}^ih_2 \ , \cdots , {}^ih_{M_i}\} \tag{6.17}$$

with $M_i \leq M$ and $H_i \subseteq H$.

6.3.2 Hypothesis Testing

The purpose of verification is to distinguish between consistent hypotheses. Let ${}^ih_p$ be the hypothesis established by model ${}^ik_p$ and that of target k'. The least-squares error of ${}^ih_p$ is calculated by

$${}^i\delta_p = \sum_{j=1}^{m}\left\{\left[q_j'' - p_j''\right]^2 + \left[r_j'' - p_j''\right]^2\right\} \tag{6.18}$$

where p_j'' is the corresponding points in a target image, and,

$$q_j'' \approx A_1'p_j + kE_1'p_j, \quad \text{and} \tag{6.19}$$

$$r_j'' \approx A_2'p_j' + kE_2'p_j', \tag{6.20}$$

where p_1 and p_2 are the corresponding points in model images. A given hypothesis is considered to be consistent if its least-squares error is less than or equal to a specified threshold. A quantitative consistency of ${}^mh_p$, is then defined as

$$\gamma_p = \frac{1}{1 + \rho^m \delta_p} \tag{6.21}$$

where ρ^m is a scaling factor. It is obvious that $\gamma_p \approx 100\%$ if ${}^m h_p$ is a perfect match; or if γ_p is close to 0, then a bad match is implied.

6.4 Parallel Approach

The general framework for our approach to model-based recognition is that of the parallel hypothesis generation and test. Whereas serial algorithms traditionally use some sort of tree-based search [66] [73] [72] may be effectively replaced by a parallel approach to reduce the enormous computational cost of generating and testing hypotheses. We show ways to manage this cost through a judicious choice of representations and strategies in order to combine evidence so that within some limit, the time to find a solution is unaffected by the number of hypotheses generated.

Models are often represented as some groups depicting the spatial relationships between points. The models used here are represented simply as a set of n points assigned one point per processor. This representation was chosen over a single processor per point representation so that each point could, in parallel, participate independently in matching and verification. A further consequence of this representation is that by directly relating the processing resource to the complexity of the object, the time to perform operations is made the same for all objects.

As previously shown, hypotheses are generated wherever a close relationship of projective invariant pairs is found. Conceptually one may view this process as one of broadcasting each image point and allowing model points that match sufficiently well to generate a hypothesis. In implementation, however, the association of target points with corresponding model points is accomplished by a parallel search procedure that is independent of the number of elements involved. Thus, the time for hypothesis generation is essentially reduced.

Hypotheses are represented by a simple data structure assigned one-per-processor containing the corresponding points and the projective invariants required to bring the model into alignment with the particular target point. The total number of potential hypotheses is bounded by the combination of extracted points, n, and constructed points, m.

Many thousands of hypotheses may need to be verified, and for competitive matching to be successful, we need to verify as many as possible at a time. Hypotheses selected for instantiation create instances by copying the points from their respective model processor sets. A projective transformation specified for each hypothesis transforms model points from a model-centered coordinate system into the target coordinate system. Each projected point can then compute how well it matches a corresponding point in the scene by searching selected regions of the scene for confirming evidence.

Points are extracted from the scene using the same methods as were used to construct models. These points will be matched to the points in the instances. A quantitative consistency is computed for each of these potential matches, based on how closely the orientation and position of the scene point matches the parameters of the instance point. The final quantitative consistency for each instance point is the maximum quantitative consistency found by matching against any of the scene points it has examined. The confidence for the model instance as a whole is found by summing up individual point match quantitative consistency.

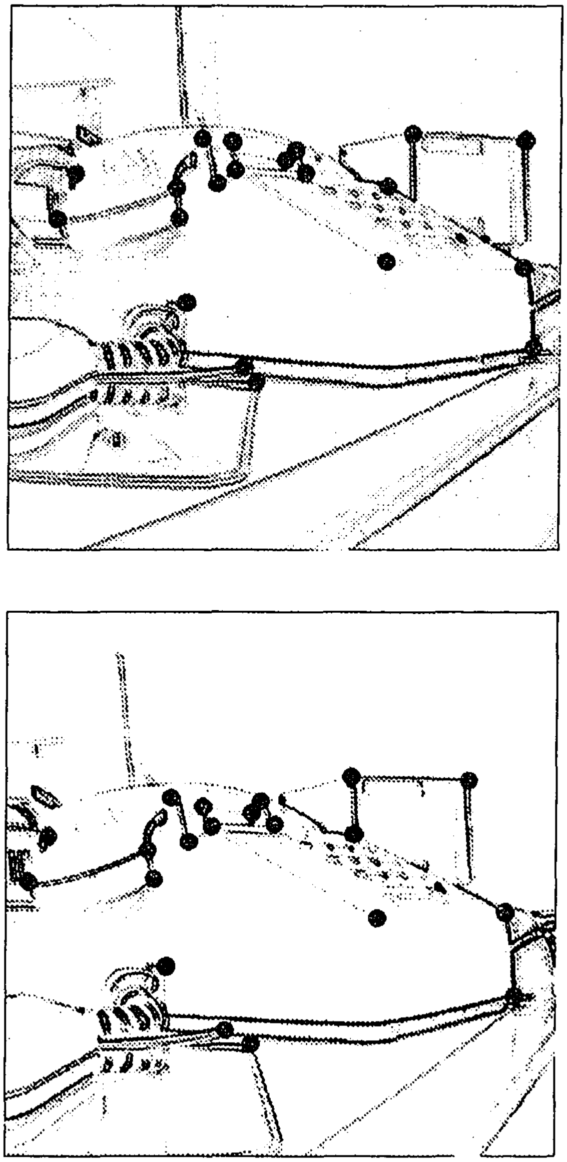


Figure 6.2 Two model images.

6.5 Experimental Results

For the purpose of experimentation, a different type of *transfer* is considered and the robustness of the approach to the hypothesis-generation-test scheme is also considered. A large number of real images were selected and an intensive experimental work was carried out in order to test the robustness and the accuracy of the hypothesis-generation-test scheme as well as the efficiency in the parallel approach.

In order to show the robustness of our approach, insteading of using the projective depth proposed by Shashua [188], we chose the other *transfer* technique proposed by Faugeras and Robert [58]. They present a *transfer* technique which can predict image features in one image from image features in two other images. Therefore, the predicting points can be computed from two model images by the following equation:

$$p'' = F'p \wedge F''p' \tag{6.22}$$

where

$$pF'p'' = 0 \text{ and } p'F''p'' = 0 \tag{6.23}$$

and F' and F'' are fundamental matrices. In accordance with this *transfer* technique, we use some real images to show how much computation is needed if the correspondence is unknown, and how we can overcome this problem by our proposed approach.

On an indoor scene, figures 6.2 and 6.3 illustrate the performances of the matching approach that have been proposed. The displacements among three images are mainly horizontal translations toward to the left side. The image size used for matching is 435 × 435 and the corresponding points related to the high curvature points are selected by hand. Figure 6.2 shows the set of 20 corresponding points of two model images, and Figure 6.3 shows the set of 20 corresponding points of the target image. The coordinates of the extracted high curvature points related to these images are listed in Table 6.2.

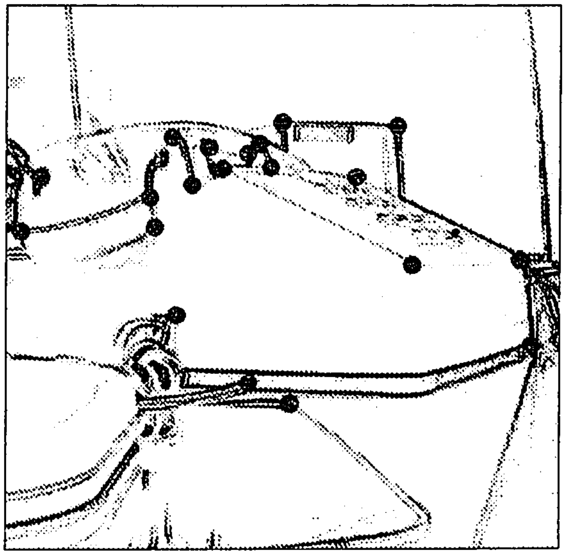


Figure 6.3 A target image.

In the case where epipolar geometry is established, this situation is called weak-calibration. In our experiment, we also assumed this situation; however, the fundamental matrices, F' and F'' between models and target image need to be obtained first. Many different linear and non-linear methods [133] [80] [56] [134] were proposed to computer the fundamental matrix. We simply used a linear method with the first eight corresponding points in Table 6.2, and computed the following fundamental matrices:

$$F' = \begin{bmatrix} -0.0000063580 & 0.0003104377 & -0.0425432562 \\ -0.000332528 & 0.0000316359 & 0.0517105751 \\ 0.0455858041 & -0.062250261 & 1.0000000000 \end{bmatrix} \tag{6.24}$$

Table 6.2 Some extracted points from two model images and the target image.

index	model-1		model-2		target	
	x	y	x	y	x	y
1	38	172	18	216	11	181
2	51	134	35	179	27	137
3	141	238	128	283	133	249
4	135	170	117	215	116	179
5	133	147	113	190	112	152
6	153	106	131	149	132	106
7	164	143	145	185	145	144
8	175	108	156	156	160	113
9	180	129	164	173	170	129
10	218	124	194	162	190	121
11	226	115	202	151	199	109
12	234	134	212	171	208	129
13	186	289	175	333	191	302
14	195	302	195	343	223	318
15	297	205	293	245	320	207
16	298	142	275	178	276	138
17	317	102	273	133	216	95
18	407	106	365	135	306	95
19	409	210	395	241	404	204
20	413	274	402	306	412	270

and

$$F'' = \begin{bmatrix} 0.0000071126 & -0.0000093008 & 0.004384167 \\ 0.0000212907 & 0.0000041199 & -0.0139154639 \\ -0.0079641321 & 0.0101197113 & 1.0000000000 \end{bmatrix} . \tag{6.25}$$

By the *transfer* technique [58], we can predict eight corresponding points if the permutation of each corresponding point is known. The corresponding points of predicting points and target points are listed in Table 6.3. However, the permutation is unknown, so the least squared errors of different permutations must be calculated. The least squared

Table 6.3 The errors between predicting points and target points.

index	predict		target		error
	x	y	x	y	
1	10	181	11	181	1
2	26	136	27	137	2
3	132	249	133	249	1
4	115	178	116	179	2
5	112	152	112	152	0
6	132	106	132	106	0
7	144	143	145	144	2
8	160	113	160	113	0

errors of some different permutations are shown in Table 6.4. In this table, we show only the right permutation getting low errors, while others get very high errors.

In the case where the correspondence between images is unknown, we show the power of our matching method to recognize an object. A slight modification of our hypothesis-generation-test scheme is needed to fit different *transfer* techniques. Since the projective depth is not used in our experiment, we simply generated all pairs of hypotheses. Only the verification step is considered here. In a hypercube architecture, we broadcast each extracted point from target image to each processor. Inside each permutation, we computed the least squared error with the prediction point, and summed up the errors together in parallel. If the summation of least-squared errors is larger than some threshold, we indicate this permutation being matched.

Typically, assuming n points are used, originally $n!$ computations are needed. If a hypercube architecture is used, only $O(\log(n))$ is needed for each summation. The overall complexity is reduced to more than n times of the original approach.

Table 6.4 The errors associated with different permutations.

predict	target	δ
1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8	8
1 2 3 4 5 6 7 8	1 2 3 4 5 6 8 7	2348
1 2 3 4 5 6 7 8	1 2 3 4 5 7 6 8	3132
1 2 3 4 5 6 7 8	1 2 3 4 5 7 8 6	3538
1 2 3 4 5 6 7 8	1 2 3 4 5 8 6 7	3608
1 2 3 4 5 6 7 8	1 2 3 4 5 8 7 6	1674
1 2 3 4 5 6 7 8	1 2 3 4 6 5 7 8	5040
1 2 3 4 5 6 7 8	1 2 3 4 6 5 8 7	7380
1 2 3 4 5 6 7 8	1 2 3 4 6 7 5 8	5188
1 2 3 4 5 6 7 8	1 2 3 4 6 7 8 5	9046
1 2 3 4 5 6 7 8	1 2 3 4 6 8 5 7	5664
1 2 3 4 5 6 7 8	1 2 3 4 6 8 7 5	7182
1 2 3 4 5 6 7 8	1 2 3 4 7 6 5 8	2264
1 2 3 4 5 6 7 8	1 2 3 4 7 6 8 5	6122
1 2 3 4 5 6 7 8	1 2 3 4 7 5 6 8	5240
1 2 3 4 5 6 7 8	1 2 3 4 7 5 8 6	5646
1 2 3 4 5 6 7 8	1 2 3 4 7 8 6 5	7382
1 2 3 4 5 6 7 8	1 2 3 4 7 8 5 6	3930
1 2 3 4 5 6 7 8	1 2 3 4 8 6 7 5	7658
1 2 3 4 5 6 7 8	1 2 3 4 8 6 5 7	6140
1 2 3 4 5 6 7 8	1 2 3 4 8 7 6 5	10782
1 2 3 4 5 6 7 8	1 2 3 4 8 7 5 6	7330
1 2 3 4 5 6 7 8	1 2 3 4 8 5 6 7	9116
1 2 3 4 5 6 7 8	1 2 3 4 8 5 7 6	7182

6.6 Conclusions

This chapter is a general introduction to the projective object recognition method using *transfer* which is elegant, simple, and of an algorithmic nature. We have given a systematic demonstration of how to use it on examples relevant to computer vision, points in projective planes, and have derived the related hypothesis-generation-test schema. We have then used these schema to show the analysis of the projection of points according to a *transfer* technique which could be done in a common framework.

In order to make this hypothesis-generation-test scheme more practical, we need a more detailed investigation of how our parallel approach can be implemented on real parallel machines. We have started doing this on *nCube* and *AP1000* parallel machines. In addition, we also need to investigate the sensitivity to errors of the projective invariants which are necessary in order to establish a correspondence between images obtained from various viewpoints.

CHAPTER 7

CONCLUSION

7.1 Summary of Results

The results of this dissertation about model-based visual recognition, exploitation of parallelism, and distribution of invariants is presented below.

Model-based Visual Recognition. The state of the art feature extraction methods and matching algorithms are surveyed and analyzed. A new feature, *ACT*, is designed to reinforce the representation of images, and a hypothesis-generation-test scheme is proposed as matching algorithms. Objects in interest may be partially occluded or overlapped. In our methodology, the hypothetic primitives κ_i and τ_i, play important roles. Points of high curvature are essential in describing objects, but they may not represent the shape sufficiently for the purpose of classification. It has been shown that hypothetic angularities, as well as the *ACTs*, provide a powerful mechanism for recognizing objects. We make very few restricted assumptions about the objects in question. By using hypothetic angularities and *ACTs*, a sequence of consecutive significant corners is mapped into a distinct location in the hypothetic angularity space in accordance with the mutual geometrical relationships among them. The experimental results demonstrate that our *ACT*-based algorithm can be used to match complex industrial parts.

Exploitation of Parallelism. Two implementations on the AP1000 parallel machine and its simulator, CASIM, are demonstrated for shape matching and chain coding. A backward dynamic programming algorithm is parallelized onto a linear array architecture. Both feature extraction and matching algorithms are implemented for shape matching. In addition, a divide-and-conquer parallel coding algorithm is also developed on a pyramid parallel architecture. The coding stage and the merging stage are implemented for chain coding. The timing results are demonstrated, and the computational complexity is analyzed.

In the context of a contour-based shape matching algorithm designed on the parallel *AP1000* machine this paper:

- Develops a modified version of parallel dynamic programming algorithm for solving the contour-based shape matching problem.
- Shows the flexibility of parallel dynamic programming in that it is independent of processing steps.
- Achieves real-time performance using the parallel dynamic programming technique. A speed increase of approximately six times is achieved.

In the implementation of a Freeman chain-coding algorithm using parallel machines we adopt a look-up table. We have also presented important issues in the context of a fully coding algorithm using *CASIM*, the simulator of *AP1000* parallel machine. For a 64 × 64 input image with 8 contours, it costs 0.1 of user time and 0.1 of system time in a SUN4M machine.

Distribution of Invariants. Invariants from four general lines under perspective projection are represented. In contrast to other work based on epipolar geometry, we discuss the invariants by isotropy subgroups. Indeed, two independent invariants exist for four general lines in 3D space. We show how to obtain these two invariants from the projective image of a four general lines structure, regardless of the reconstruction of the 3D structure.

We also describe the recognition system based on a hypothesis-generation-test scheme using projective invariants. Instead of using the conventional machine to compute the complex projective invariant recognition system, the proposed system uses an equivalent parallel algorithm run on hypercube parallel architecture. Object recognition is achieved by matching the scene projective invariants to the model projective invariants known as *transfer*. Then a hypothesis-generation-test scheme is implemented on the hypercube parallel architecture.

7.2 Future Direction of Research

As always, successful research and development opens up new research areas and opportunities. While invariant approaches have achieved a certain level of maturity, advances are expected along several lines of investigation.

First, for recognition, the projective invariants emerged parallel machines should demonstrate real time performance not only theoretically but practically.

Next, there are two main threads of development concerning parallel algorithms, namely processor usage and speedup. Optimality should be concerned in all parallel algorithms.

Finally, there is also the scene for which the relatively isolated results of many researchers on various aspects of invariants analysis should be integrated so as to create a single environment for scene description and recognition.

REFERENCES

[1] N. Ahuja and S. Swamy, ''Multiprocessor pyramid architectures for bottom-up image analysis,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 6, no. 4, pp. 463-475, July 1984.

[2] H.M. Alnuweiri and V.K. Parasnna Kumar, ''Optimal image algorithms on an orthogonally-connected memory-based architecture,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 350-355, Atlantic City, New Jersey, June 16-21, 1990.

[3] M. Annaratone, E. Arnould, T. Gross, H.T. Kung, M. Lam, O. Menzilcioglu, and J.A. Webb, ''The warp computer: Architecture, implementation, and performance,'' *IEEE Trans. Comput. (T-C)*, vol. C-36, no. 12, pp. 1523-1538, December 1987.

[4] N. Ansari and E.J. Delp, ''Partial shape recognition: A landmark-based approach,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 12, no. 5, pp. 470-483 , May 1990.

[5] M.A. Arbib, ''Schemas and neural networks for sixth generation computing,'' *J. of Parallel and Distributed Computing*, vol. 6, pp. 185-216, 1989.

[6] K.S. Arun, T.S. Huang, and S.D. Blostein, ''Least-squares fitting of two 3-D point sets,'' *IEEE Trans. Patt. Anal. and Mach. Intel. (T-PAMI)*, vol. 9, no. 5, pp. 698-700, September 1987.

[7] N. Ayache and O.D. Faugeras, ''A new method for the recognition and positioning of 2-D objects,'' *in Proc. 7'th IEEE Int. Conf. Patt. Recogn. (ICPR-84)*, pp. 1274-1277, Montreal, Canada, July 30 - August 2, 1984.

[8] N. Ayache and O.D. Faugeras, ''HYPER: A new approach for the recognition and positioning of two-dimensional objects,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 8, no. 1, pp. 44-54, January 1986.

[9] D.H. Ballard and C.M. Brown, *Computer Vision*, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[10] S.T. Barnard and W.B. Thompson, ''Disparity analysis of images,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 2, no. 4, pp. 333-340, July 1980.

[11] E.B. Barrett and P.M. Payton, ''General methods for determining projective invariants in imagery,'' *Comput. Vision, Graphics, Image Processing: Image Understanding (CVGIP)*, vol. 53, no. 1, pp. 46-65, January 1991.

[12] E.B. Barrett, M.H. Brill, N.N. Haag, and P.M. Payton, ''Some invariant linear methods in photogrammetry and model-matching,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-92)*, pp. 122-128, Champaign, Illinois, June 15-18, 1992.

[13] E. Barrett, G. Gheen, and P. Payton, ''Representation of three-dimensional object structure as cross-ratios of determinants of stereo image points,'' *in Proc. Applications of Invariance in Comput. Vision*, pp. 47-68, Ponta Delgada, Azores, Portugal, October 9-14, 1993.

[14] P.A. Beardsley, A. Zisserman, and D.W. Murray, ''Navigation using affine structure from motion,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 2, pp. 85-96, Stockholm, Sweden, May 2-6, 1994.

[15] K.P. Belkhale and P. Banerjee, ''Parallel algorithms for geometric connected component labeling on a hypercube multiprocessor,'' *IEEE Trans. Comput. (T-C)*, vol. C-41, no. 6, pp. 699-709, June 1992.

[16] P.J. Besl and R.C. Jain, "Three-dimensional object recognition,'' *ACM Computing Surveys*, vol. 17, no. 1, pp. 75-154, 1985.

[17] B. Bhanu and O.D. Faugeras, ''Shape matching of two-dimensional objects,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 6, no. 2, pp. 137-156, March 1984.

[18] S.D. Blostein and T.S. Huang, ''Implementation of an MHT-based object detection algorithm on a 2-D processor mesh,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 512-515, Atlantic City, New Jersey, June 16-21, 1990.

[19] R.C. Bolles and R.A. Chain, ''Recognizing and locating partially visible objects: The local-feature-focus method,'' *Int. J. of Robotics Res.*, vol. 1, no. 3, pp. 57-82, 1982.

[20] R.C. Bolles and P. Horaud, "3DPO: A three-dimensional part orientation system,'' *Int. J. of Robotics Res.*, vol. 5, no. 3, pp. 3-26, Fall 1986.

[21] G. Borgefors, ''An improved vision of the chamfer matching algorithm,'' *in Proc. 7'th IEEE Int. Conf. Patt. Recogn. (ICPR-84)*, pp. 1175-1177, Montreal, Canada, July 30 - August 2, 1984.

[22] G. Borgefors, ''Distance transformations in digital images,'' *Comput. Vision, Graphics, Image Processing (CVGIP)*, vol. 34, pp. 344-371, 1986.

[23] G. Borgefors, ''Hierarchical chamfer matching: A parametric edge matching algorithm,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 10, no. 6, pp. 849-865, November 1988.

[24] G. Borgefors and G.S. di Baja, ''Parallel smoothing and decomposition of digital shapes using a multiresolution structure,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 1, pp. 745-748, Atlantic City, New Jersey, June 16-21, 1990.

[25] B. Boufama, D. Weinshall, and M. Werman, ''Shape from motion algorithms: a comparative analysis of scaled orthography and perspective,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 2, pp. 199-204, Stockholm, Sweden, May 2-6, 1994.

[26] O. Bourdon and G. Medioni, ''Object recognition using geometric hashing on the Connect machine,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 596-600, Atlantic City, New Jersey, June 16-21, 1990.

[27] J. Brady and N. Nandhakumar, ''Recent progress in the recognition of objects from range data,'' *in Proc. 9'th IEEE Int. Conf. Patt. Recogn (ICPR-88)*, pp. 85-92, Rome, Italy, November 1988.

[28] R.A. Browse, ''Feature-based tactile object recognition,'' *IEEE Trans. Patt. Anal. and Mach. Intel. (T-PAMI)*, vol. 9, no. 6, pp. 779-786, November 1987.

[29] J.B. Burns, R.S. Weiss, and E.M. Riseman, ''The non-existence of general-case view invariants,'' *in Geometric Invariance in Comput. Vision, J.L. Mundy and A. Zisserman Ed., Chapter 6*, pp. 120-131, The MIT Press, Cambridge, Massachusetts, 1992.

[30] O.I. Camps, L.G. Shapior, and R.M. Haralick, ''PREMIO: An overview,'' *in Proc. Workshop on Direction in Automated CAD-Based Vision*, pp. 11-21, Computer Society Press, Los Alamitos, California, 1991.

[31] J. Canny, ''A computational approach to edge detection,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 8, no. 6, pp. 679-698, November 1986.

[32] S. Carlsson, ''The double algebra: An effective tool for computing invariants in computer vision,'' *in Proc. Applications of Invariance in Comput. Vision*, pp. 145-164, Ponta Delgada, Azores, Portugal, October 9-14, 1993.

[33] T.A. Cass, ''A robust parallel implementation of 2-D model-based recognition,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-88)*, pp. 879-884, Ann Arbor, Michigan, June 5-9, 1988.

[34] C. Chakrabarti and J.F. JaJa, ''A parallel algorithm for template matching on an SIMD mesh connected computer,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 362-367, Atlantic City, New Jersey, June 16-21, 1990.

[35] C.H. Chen and A.C. Kak, ''A robot vision system for recognizing 3-D objects in low-order polynomial time,'' *IEEE Trans. Syst., Man, Cybern. (T-SMC)*, vol. SMC-19, no. 6, pp. 1535-1563, November/December 1989.

[36] Y.L. Chen, A. Sohn, and D.C. Hung, ''Object recognition using parallel machine,'' *in Proc. Int. Symposium on Artificial Neural Networks*, Hsinchu, Taiwan, December 20-22, 1993.

[37] R.T. Chin and C.R. Dyer, ''Model-based recognition in robot vision,'' *ACM Computing Surveys*, vol. 18, no. 1, pp. 67-108, 1986.

[39] A.K. Choudary, M.K. Leung, T.S. Huang, and J.H. Patel, ''Parallel implementation and evaluation of motion estimation system algorithms on a distributed memory multiprocessor using knowledge based mappings,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 337-342, Atlantic City, New Jersey, June 16-21, 1990.

[39] N. Choudhary and J.H. Patel, *Parallel Architectures and Parallel Algorithms for Integrated Vision System*, Kluwer Academic Publishers, Massachusetts, 1990.

[40] S.H. Chuang and M.R. Henderson, ''Three-dimensional shape pattern recognition using vertex classification and vertex-edge graphs,'' *Computer-Aided Design*, vol. 22, no. 6, pp. 377-387, July/August 1990.

[41] P. Clermont and B. Zavidovique, ''Communication control in a pyramid computer application to region labeling,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 551-555, Atlantic City, New Jersey, June 16-21, 1990.

[42] D. Cyganski and J.A. Orr, ''Applications of tensor theory to object recognition and orientation determination,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 7, no. 6, pp. 662-673, November 1985.

[43] L.S. Davis, ''Shape matching using relaxation techniques,'' *IEEE Trans. Patt. Anal. and Mach. Intel. (T-PAMI)*, vol. 1, no. 1, pp. 60-72, January 1979.

[44] L.S. Davis, L.T. Chen, and P.J. Narayanan, ''Connection machine vision - Replicated data structure,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 299-304, Atlantic City, New Jersey, June 16-21, 1990.

[45] S. Demey, A. Zisserman, and P. Beardsley, ''Affine and projective structure from motion,'' *in Proc. British Machine Vision Conf. (BMVC-92)*, pp. 49-58, Leeds, United Kingdom, September 22-24, 1992.

[46] R. Deriche, Z. Zhang, Q.T. Luong, and O.D. Faugeras, ''Robust recovery of the epipolar geometry for an uncalibrated stereo rig,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 1, pp. 567-576, Stockholm, Sweden, May 2-6, 1994.

[47] F. Devernay and O.D. Faugeras, ''Computing differential properties of 3-D shapes from stereoscopic images without 3-D models,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-94)*, pp. 208-213, Seattle, Washington, June 21-23, 1994.

[48] I. Dinstein and G.M. Landau, ''Using parallel string matching algorithms for contour based 2-D shape recognition,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 451-455, Atlantic City, New Jersey, June 16-21, 1990.

[49] I. Dinstein, G.M. Landau, and G. Guy, ''Parallel (Pram Erew) algorithms for contour-based 2D shape recognition,'' *Patt. Recogn.*, vol. 24, no. 6, pp. 929-942, 1991.

[50] K. A. Dunkelberger and O. R. Mitchell, ''Contour tracing for precision measurement,'' *Proc. IEEE Inter. Conf. Robotics and Automation*, pp. 22-27, 1985.

[51] R. Dutta and C.C. Weems, ''Parallel dense depth from motion on the image understanding architecture,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-93)*, pp. 154-159, New York City, New York, June 15-17, 1993.

[52] H. Embrechts, D. Roose, and P. Wambacq, "Component labeling on a MIMD multiprocessor," *Comput. Vision, Graphics, Image Processing: Image Understanding (CVGIP)*, vol. 57, no. 2, pp. 155-165, March 1993.

[53] G.J. Ettinger, "Large hierachical object recognition using libraries of parameterized model sub-parts," *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-88)*, pp. 32-41, Ann Arbor, Michigan, June 5-9, 1988.

[54] T. Fan, G. Medioni, and R. Nevatia, "Recognizing 3-D objects using surface descriptions," *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 11, no. 11, pp. 1140-1157, November 1989.

[55] O.D. Faugeras and M. Hebert, "The representation, recognition, and locating of 3-D objects," *Int. J. of Robotics Res.*, vol. 5, no. 3, pp. 27-52, Fall 1986.

[56] O.D. Faugeras, "What can be seen in three dimensions with an uncalibrated stereo rig?," *in Proc. 2nd European Conf. on Comput. Vision (ECCV-92)*, pp. 563-578, Santa Margherita Ligure, Italy, May 19-22, 1992.

[57] O.D. Faugeras, "Cartan's moving frame method and its application to the geometry and evolution of curves in the Euclidean, affine and projective planes," *in Proc. Applications of Invariance in Comput. Vision*, pp. 11-46, Ponta Delgada, Azores, Portugal, October 9-14, 1993.

[58] O.D. Faugeras and L. Robert, "What can two images tell us about a third one?," *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 1, pp. 485-492, Stockholm, Sweden, May 2-6, 1994.

[59] P.J. Flynn and A.K. Jain, "CAD-based computer vision: From CAD models to relational graphs," *IEEE Trans. Patt. Anal. and Mach. Intel. (T-PAMI)*, vol. 13, no. 2, pp. 114-132, February 1991.

[60] J.D. Foley, A. Van Dam, S. Feiner, and J. Hughes, *Computer graphics: Principles and practice, 2nd Ed.*, Addison-Wesley, Reading, Massachusetts, 1990.

[61] D.A. Forsyth, J.L. Mundy, A. Zisserman, and C.A. Rothwell, "Using global consistency to recognise Euclidean objects with an uncalibrated camera," *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-94)*, pp. 502-507, Seattle, Washington, June 21-23, 1994.

[62] T.J. Fountain, K.N. Matthews, and M.J.B. Duff, "The CLIP7A image processor," *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 10, no. 3, pp. 310-319, May 1988.

[63] N.D. Francis, G.R. Nudd, T.J. Atherton, D.J. Kerbyson, R.A. Packwood, and J. Vaudin, "Performance evaluation of the hierarchical Hough transform on an associative M-SIMD architecture," *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 509-511, Atlantic City, New Jersey, June 16-21, 1990.

[64] H. Freeman, "Computer processing of line-drawing data," *Comput. Surveys*, vol. 6, no. 1, pp. 57-96, 1974.

[65] T. Fukushima, ''A survey of image processing LSIs in Japan,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 394-401, Atlantic City, New Jersey, June 16-21, 1990.

[66] P.C. Gaston and T. Lozano-Perez, ''Tactile recognition and localization using object models: The case of polyhedra on a plane,'' *IEEE Trans. Patt. Anal. and Mach. Intel. (T-PAMI)*, vol. 6, no. 3, pp. 257-266, May 1984.

[67] D.C. Gerogiannis and S.C. Orphanoudakis, ''Efficient embedding of interprocessor communication in parallel implementations of intermediate level vision tasks,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 368-372, Atlantic City, New Jersey, June 16-21, 1990.

[68] D.B. Goldgof, T.S. Huang, and H. Lee, ''A curvature-based approach to terrain recognition,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 11, no. 11, pp. 1213-1217, November 1989.

[69] J.W. Gorman, O.R. Mitchell, and F.P. Kuhl, ''Partial shape recognition using dynamic programming,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 10, no. 2, pp. 257-266, March 1988.

[70] J. Gregor and M.G. Thomason, ''Dynamic programming alignment of sequences representing cyclic patterns,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 15, no. 2, pp. 129-135, February 1993.

[71] W.E.L. Grimson and T. Lozano-Perez, ''Model-based recognition and localization from sparse range or tactile data,'' *Int. J. of Robotics Res.*, vol. 3, no. 3, pp. 3-35, Fall 1984.

[72] W.E.L. Grimson, ''Recognition of object families using parameterized models,'' *in Proc. 1'st Int. Conf. Computer Vision (ICCV-87)*, pp. 93-101, London, England, June 1987.

[73] W.E.L. Grimson, ''On the recognition of curved objects,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 11, no. 6, pp. 632-643, June 1989.

[74] W.E.L. Grimson and D.P. Huttenlocher, ''On the verification of hypothesized matches in model-based recognition,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 13, no. 12, pp. 1201-1213, December 1991.

[75] P. Gros, ''How to use the cross ratio to compute projective invariants from two images,'' *in Proc. Applications of Invariance in Comput. Vision*, pp. 107-126, Ponta Delgada, Azores, Portugal, October 9-14, 1993.

[76] C. Guerra and S. Hambrusch, ''Parallel algorithms for line detection on a mesh,'' *J. of Parallel and Distributed Computing*, vol. 6, pp. 1-19, 1989.

[77] G.D. Hager, ''Real-time feature tracking and projective invariance as a basis for hand-eye coordination,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-94)*, pp. 533-539, Seattle, Washington, June 21-23, 1994.

[78] R.W. Hall, S. Kucuk, and M. Hamdi, ''Wavelet transform embeddings in mesh architectures,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-93)*, pp. 596-597, New York City, New York, June 15-17, 1993.

[79] C. Hansen and T.C. Henderson, ''CAGD-based computer vision,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 11, no. 11, pp. 1181-1193, November 1989.

[80] R.I. Hartley, ''Estimation of relative camera positions for uncalibrated cameras,'' *in Proc. 2nd European Conf. on Comput. Vision (ECCV-92)*, pp. 579-587, Santa Margherita Ligure, Italy, May 19-22, 1992.

[81] R. Hartley, R. Gupta, and T. Chang, ''Stereo from uncalibrated cameras,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-92)*, pp. 761-764, Champaign, Illinois, June 15-18, 1992.

[82] R. Hartley, ''Camera calibration using line correspondences,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 361-366, Washington, District of Columbia, April 18-21, 1993.

[83] R. Hartley, ''Invariants of lines in space,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 737-744, Washington, District of Columbia, April 18-21, 1993.

[84] R. Hartley, ''Cheirality Invariants,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 745-753, Washington, District of Columbia, April 18-21, 1993.

[85] R.I. Hartley, ''Euclidean reconstruction from uncalibtated views,'' *in Proc. Applications of Invariance in Comput. Vision*, pp. 237-256, Ponta Delgada, Azores, Portugal, October 9-14, 1993.

[86] R. Hartley and R. Gupta, ''Computing matched-epipolar projections,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-93)*, pp. 549-555, New York City, New York, June 15-17, 1993.

[87] R.I. Hartley, ''Self-calibration from multiple views with a rotating camera,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 1, pp. 471-478, Stockholm, Sweden, May 2-6, 1994.

[88] R.I. Hartley, ''Projective reconstruction from line correspondences,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-94)*, pp. 903-907, Seattle, Washington, June 21-23, 1994.

[89] R.I. Hartley, ''An algorithm for self calibration from several views,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-94)*, pp. 908-912, Seattle, Washington, June 21-23, 1994.

[90] R.I. Hartley, ''Projective reconstruction and invariants from multiple images,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 16, no. 10, pp. 1036-1041, October 1994.

[91] R.I. Hartley and R. Gupta, ''Linear pushroom cameras,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 1, pp. 555-566, Stockholm, Sweden, May 2-6, 1994.

[92] D. Helman and J. JaJa, ''Efficient image processing algorithms on the scan line array processor,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 17, no. 1, pp. 47-56, January 1995.

[93] J. Hong and X. Tan , ''The similarity between shapes under affine transformation,'' *NYU Robotics Research Report 133* , December 1987.

[94] J. Hong and H. J. Wolfson, ''An improved model-based matching method using footprints,'' *in Proc. IEEE Int. Conf. Patt. Recogn. (ICPR-88)*, pp. 72-78, Rome, Italy, November 1988.

[95] R. Horaud, F. Dornaika, B. Boufama, and R. Mohr, ''Self calibration of a stereo head mounted onto a robot arm,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 1, pp. 455-462, Stockholm, Sweden, May 2-6, 1994.

[96] M. Hospital, H. Yamada, T. Kasvand, and S. Umeyama, ''3D curve based matching method using dynamic programming,'' *in Proc. 1'st Int. Conf. Computer Vision (ICCV-87)*, pp. 728-732, London, England, June 1987.

[97] T.L. Huntsberger and P. Ajjimarangsee, ''Parallel self-organizing feature maps for unsupervised pattern recognition,'' *Int'l. J. General Systems*, vol. 16, no. 4, pp. 357-372, 1990.

[98] D.P. Huttenlocher and S. Ullman, ''Object recognition using alignment,'' *in Proc. 1'st Int. Conf. Computer Vision (ICCV-87)*, pp. 102-111, London, 1987.

[99] D.P. Huttenlocher and S. Ullman, ''Object recognition using alignment,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 370-380, Los Angels, California, February 1987.

[100] D.P. Huttenlocher and S. Ullman, ''Recognizing solid objects by alignment,'' *in Proc. of the DARPA Image Understanding Workshop*, April 1988.

[101] K. Huwang, *Advanced computer architecture with parallel programming, preliminary edition*, McGraw-Hill, New York City, New York, 1993.

[102] S.Y. Hwang and S.L. Tanimoto, ''Parallell coordination of image operators on shared-memory architecture,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 343-349, Atlantic City, New Jersey, June 16-21, 1990.

[103] A.K. Jain and R.L. Hoffman, ''Evidence-based recognition of 3-D objects,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 10, no. 6, pp. 783-802, November 1988.

[104] J. JaJa, *An Introduction to Parallel Algorithms*, Addison Wesley, Reading, Massachusetts, 1992.

[105] L.H. Jamieson, P.T. Muller, JR., and H.J. Siegel, ''FFT algorithms for SIMD parallel processing systems,'' *J. of Parallel and Distributed Computing*, vol. 3, pp. 48-71, 1986.

[106] J.W. Jang, H. Park, and V.K. Parasanna, ''A fast algorithm for computing a histogram on reconfigurable mesh,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 17, no. 2, pp. 97-106, February 1995.

[107] A. Kalvin, E. Schonberg, J.T. Schwartz, and M. Sharir, ''Two-dimensional, model-based, boundary matching using footprints,'' *Int. J. of Robotics Res.*, vol. 5, no. 4, pp. 38-55, Winter 1986.

[108] Z. Kato and J. Zerubia, ''Multiscale Markov random field models for parallel image classification,'' *in Proc. 4'st Int. Conf. Computer Vision (ICCV-93)*, pp. 253-257, Berlin, Germany, May 11-14, 1993.

[109] A. Khokhar and V.K. Prasanna Kumar, ''Parallel algorithms for stereo and image matching,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 1057-1062, San Diego, California, January 26-29, 1992.

[110] A. Khokhar and V.K. Prasanna, ''Scalable data parallel geometric hashing: Experiments on MasPar MP-1 and on connection machine CM-5,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 851-860, Washington, District of Columbia, April 18-21, 1993.

[111] A.A. Khokhar, V.K. Prasanna Kumar, and H.J. Kim, ''Scalable geometric hashing on MasPar machines,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-93)*, pp. 594-595, New York City, New York, June 15-17, 1993.

[112] S.D. Kim, J.H. Lee, and J.K. Kim, ''A new chain-coding algorithm for binary images using run-length codes,'' *Comput. Vision, Graphics, Image Processing (CVGIP)*, vol. 41, pp. 114-128, 1988.

[113] T.F. Knoll and R. Jain, ''Recognizing partially visible objects using feature indexed hypotheses,'' *IEEE J. Robotics Automat.*, vol. RA-2, no. 1, pp. 3-14, March 1986.

[114] M.W. Koch and R.L. Kashyap, ''A vision system to identify occluded industrial parts,'' *in Proc. IEEE Int. Conf. Robotics and Automation (ICRA-85)*, pp. 55-60, March 1985.

[115] M.W. Koch and R.L. Kashyap, ''Using polygons to recognize and locate partially occluded objects,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 9, no. 4, pp. 483-494, July 1987.

[116] S.R. Kulkarni, S.K. Mitter, T.J. Richardson, and J.N. Tsitsiklis, ''Local versus nonlocal computation length of digitized curves,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 16, no. 7, pp. 711-718, July 1994.

[117] K.N. Kutulakos and C.R. Dyer, ''Efficient indexing techniques for model based sensing,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-94)*, pp. 323-330, Seattle, Washington, June 21-23, 1994.

[118] A.F. Laine and G.C. Roman, ''A parallel algorithm for incremental stereo matching on SIMD machines,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 484-490, Atlantic City, New Jersey, June 16-21, 1990.

[119] Y. Lamdan, J.T. Schwartz, and H.J. Wolfson, ''Object recognition by affine invariant matching,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-88)*, pp. 335-344, Ann Arbor, Michigan, June 5-9, 1988.

[120] Y. Lamdan, J.T. Schwartz, and H.J. Wolfson, ''On recognition of 3-D objects from 2-D images,'' *in Proc. IEEE Int. Conf. Robotics and Automation (ICRA-88)*, pp. 1407-1413, Philadelphia, Pennsylvania, April 1988.

[121] C.Y. Lee and D.B. Cooper, "Structure and motion from region correspondences and affine invariants," *in Proc. of the DARPA Image Understanding Workshop*, pp. 707-711, Washington, District of Columbia, April 18-21, 1993.

[122] S.Y. Lee and J.K. Aggarwal, "Parallel 2-D convolution on a mesh connected array processor," *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 9, no. 4, pp. 590-594, July 1987.

[123] F.T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays.Trees.Hypercubes*, Morgan Kaufmann, San Mateo, California, 1992.

[124] S.P. Levitan, C.C. Weems, A.R. Hanson, and E.M. Riseman, "The UMass image understanding architecture," *in Parallel computer vision*, Leonard Uhr ed., Academic Press, Boston, Massachusetts, 1987.

[125] C.C. Lin and R. Chellappa, "Classification of partial 2-D shapes using fourier descriptors," *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 9, no. 5, pp. 686-689, September 1987.

[126] W.C. Lin and T.W. Chen, "CGS-based object recognition using range images," *in Proc. 9'th IEEE Int. Conf. Patt. Recogn (ICPR-88)*, pp. 99-103, Rome, Italy, November 1988.

[127] W. Lin and V.K. Prasanna Kumar, "Efficient histogramming on hypercube SIMD machines," *Comput. Vision, Graphics, Image Processing (CVGIP)*, vol. 49, pp. 104-120, 1990.

[128] W.M. Lin and V.K.P. Kumar, "Parallel algorithms and architectures for discrete relaxation technique," *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-91)*, pp. 514-519, Lahaina, Maui, Hawaii, June 3-6, 1991.

[129] J.J. Little, G. Blelloch, and T. Cass, "Parallel algorithms for computer vision on the connection machine," *in Proc. 1'st Int. Conf. Computer Vision (ICCV-87)*, pp. 587-591, London, England, June 1987.

[130] J.J. Little, G.E. Blelloch, and T.A. Cass, "Algorithmic techniques for computer vision on a fine-grained parallel machine," *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 11, no. 3, pp. 244-257, March 1989.

[131] H. Liu and M.D. Srinath, "Partial shape classification using contour matching in distance transformation," *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 12, no. 11, pp. 1072-1079, November 1990.

[132] C. Lo and H. Don, "Pattern recognition using 3-D moments," *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 1, pp. 540-543, Atlantic City, New Jersey, June 16-21, 1990.

[133] H.C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, no. 10, pp. 133-135, September 1981.

[134] Q.T. Luong, R. Deriche, O. Faugeras, and T. Papadopoulo, "On determining the fundamental matrix: Analysis of different methods and experimental results," *INRIA Technical Report, RR-1894*, 1993.

[135] Q.T. Luong and O.D. Faugeras, ''Determining the fundamental matrix with planes: Instability and new algorithms,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-93)*, pp. 489-494, New York City, New York, June 15-17, 1993.

[136] Q.T. Luong and O.D. Faugeras, ''A stability analysis of the fundamental matrix,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 1, pp. 577-588, Stockholm, Sweden, May 2-6, 1994.

[137] Q.T. Luong and T. Vieville, ''Canonic representations for the geometries of multiple projective views,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 1, pp. 589-599, Stockholm, Sweden, May 2-6, 1994.

[138] W.D. Mao and S.Y. Kung, ''An object recognition system using stochastic knowledge source and VLSI parallel architecture,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 1, pp. 832-836, Atlantic City, New Jersey, June 16-21, 1990.

[139] D. Marr, *Vision : a computational investigation into the human representation and processing of visual information*, W.H. Freeman, San Francisco, California, 1982.

[140] B. D. Marsh et al., ''The Rochester checkers player - Multimodel parallel programming for animate vision,'' *Comput.*, vol. 25, no. 2, pp. 12-19, February 1992.

[141] E. Memin, F. Heitz, and F. Charot, ''Efficient parallel multigrid relaxation algorithms for markov random field-based low-level vision applications,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-94)*, pp. 644-648, Seattle, Washington, June 21-23, 1994.

[142] R. Miller and Q.F. Stout, ''Geometric algorithms for digitized pictures on a mesh-connected computer,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 7, no. 2, pp. 216-228, March 1985.

[143] R. Miller and Q.F. Stout, ''Convexity algorithms for parallel machines,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-88)*, pp. 918-924, Ann Arbor, Michigan, June 5-9, 1988.

[144] R. Mohr, L. Morin, and E. Grosso, ''Relative positioning with uncalibrated cameras,'' *in Geometric Invariance in Comput. Vision, J.L. Mundy and A. Zisserman Ed., Chapter 22*, pp. 440-460, The MIT Press, Cambridge, Massachusetts, 1992.

[145] R. Mohr, B. Boufama, and P. Brand, ''Accurate projective reconstruction,'' *in Proc. Applications of Invariance in Comput. Vision*, pp. 257-276, Ponta Delgada, Azores, Portugal, October 9-14, 1993.

[146] R. Mohr, F. Veillon, and L. Quan, ''Relative 3D reconstruction using multiple uncalibrated images,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-93)*, pp. 543-548, New York City, New York, June 15-17, 1993.

[147] T. Moons, L.V. Gool, M.V. Diest, and E. Pauwels, ''Affine reconstruction from perspective image pairs obtained by a translating camera,'' *in Proc. Applications of Invariance in Comput. Vision*, pp. 297-316, Ponta Delgada, Azores, Portugal, October 9-14, 1993.

[148] J.L. Mundy, R.P. Welty, M.H. Brill, P.M. Payton, and E.B. Barrett, ''3-D model alignment without computing pose,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 727-735, San Diego, California, January 26-29, 1992.

[150] J.L. Mundy and A. Zisserman, ''Introduction - Towards a new framework for vision,'' *in Geometric Invariance in Comput. Vision, J.L. Mundy and A. Zisserman Ed., Chapter 1 (including reference list)*, pp. 1-39, The MIT Press, Cambridge, Massachusetts, 1992.

[151] J.L. Mundy and A. Zisserman, ''Appendix - Projective geometry for machine vision,'' *in Geometric Invariance in Comput. Vision, J.L. Mundy and A. Zisserman Ed., Chapter 23*, pp. 463-519, The MIT Press, Cambridge, Massachusetts, 1992.

[152] J.L. Mundy and A. Zisserman, ''Repeated structures: Image correspondence constraints and 3D structure recovery,'' *in Proc. Applications of Invariance in Comput. Vision*, pp. 89-106, Ponta Delgada, Azores, Portugal, October 9-14, 1993.

[153] D. Nassimi and S. Sahni, ''Finding connected components and connected ones on a mesh-connected parallel computer,'' *SIAM J. Comput.*, vol. 9, no. 4, pp. 744-757, November 1980.

[154] C.J. Nicol, ''A systolic architecture for labeling the connected components of multi-valued images in real time,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-93)*, pp. 136-141, New York City, New York, June 15-17, 1993.

[155] Y. Ohta and T. Kanade, ''Stereo by intra- and inter scanline search using dynamic programming,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 7, no. 2, pp. 139-154, March 1985.

[156] M. Oshima and Y. Shirai, ''Object recognition using three-dimensional information,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 5, no. 4, pp. 353-361, July 1983.

[157] D.C.W. Pao, H.F. Li, and R. Jayakumar, ''Shapes recognition using straight line Hough transform: Theory and generalization,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 14, no. 11, pp. 1076-1089, November 1992.

[158] H.D. Park and O.R. Mitchell, ''CAD based planning and execution of inspection,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-88)*, pp. 858-863, Ann Arbor, Michigan, June 5-9, 1988.

[159] M.A. Penna and R.R. Patterson, *Projective geometry and its applications to computer graphics*, Prentice-Hall, Englewood Cliffs, New Jersey, 1986.

[160] P. Perez, M.A. Abidi, and R.C. Gonzalez, ''Experimental evaluation of hypercube-based range analysis tools,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 584-590, Atlantic City, New Jersey, June 16-21, 1990.

[161] W.A. Perkins, ''Simplified model-based part locator,'' *in Proc. 5'th IEEE Int. Conf. Patt. Recogn. (ICPR-80)*, pp. 260-263, 1980.

[162] E. Persoon and K.S. Fu, ''Shape discrimination using fourier descriptors,'' *IEEE Trans. Syst., Man, Cybern. (T-SMC)*, vol. SMC-7, no. 3, pp. 170-179, March 1977.

[163] F. Pipitone and W. Adams, ''Tripod operators for recognizing objects in range images; rapid rejection of library objects,'' *in Proc. IEEE Int. Conf. Robotics and Automation (ICRA-92)*, pp. 1596-1601, Nice, France, May 1992.

[164] J. Ponce, D.H. Marimont, and T.A. Cass, ''Analytical methods for uncalibrated stereo and motion reconstruction,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 1, pp. 463-470, Stockholm, Sweden, May 2-6, 1994.

[165] V.K. Prasanna Kumar and D.I. Reisis, ''Image computations on meshes with multiple broadcast,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 11, no. 11, pp. 1194-1202, November 1989.

[166] K.E. Price, ''Matching closed contours,'' *in Proc. 7'th IEEE Int. Conf. Patt. Recogn. (ICPR-84)*, pp. 990-992, Montreal, Canada, July 30 - August 2, 1984.

[167] L. Quan, ''Invariants of 6 points from 3 uncalibrated images,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 2, pp. 459-470, Stockholm, Sweden, May 2-6, 1994.

[168] L. Quan, ''Invariants of six points and projective reconstruction from three uncalibrated images,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 17, no. 1, pp. 34-46, January 1995.

[169] G.M. Radack and N.I. Badler, ''Local matching of surfaces using a boundary-centered radial decomposition,'' *Comput. Vision, Graphics, Image Processing (CVGIP)*, vol. 45, pp. 380-396, 1989.

[170] S. Ranks and S. Sahni, ''Clustering on a hypercube multicomputer,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 532-536, Atlantic City, New Jersey, June 16-21, 1990.

[171] C. Reinhart and R. Nevatia, ''Parallel linear feature extraction,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 1049-1055, San Diego, California, January 26-29, 1992.

[172] I. Rigoutsos and R. Hummel, ''Scalable parallel geometric hashing for hypercube SIMD architectures,'' *NYU Robotics Research Report 553*, January 1991.

[173] I. Rigoutsos and R. Hummel, ''On a parallel implementation of geometric hashing on the Connection machine,'' *NYU Robotics Research Report 554*, April 1991.

[174] I. Rigoutsos and R. Hummel, ''Implementation of geometric hashing on the connection machine,'' *in Proc. Workshop on Direction in Automated CAD-Based Vision*, pp. 76-84, Computer Society Press, Los Alamitos, California, 1991.

[175] I. Rigoutsos and R. Hummel, ''Passively parallel model matching: Geometric hashing on the connection machine,'' *Comput.*, vol. 25, no. 2, pp. 33-42, February 1992.

[176] L. Robert and M. Hebert, ''Deriving orientation cues from stereo images,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 2, pp. 377-388, Stockholm, Sweden, May 2-6, 1994.

[177] A. Rosenfeld, ''A report on the DARPA integrated image-undestanding architecture workshop,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 298-301, Los Angels, California, February 1987.

[178] A. Rosenfeld, J. Ornelas, JR., and Y. Hung, ''Hough transform algorithms for mesh-connected SIMD parallel processors,'' *Comput. Vision, Graphics, Image Processing (CVGIP)*, vol. 41, pp. 293-305, 1988.

[179] S. Sarkar and K.L. Boyer, ''Optimal, efficient, recursive edge detection filters,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 1, pp. 931-936, Atlantic City, New Jersey, June 16-21, 1990.

[180] H.S. Sawhney, ''3D geometry from planar parallax,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-94)*, pp. 929-934, Seattle, Washington, June 21-23, 1994.

[181] H. Schomberg, ''A transputer-based shuffle-shift machine for image processing and reconstruction,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 445-450, Atlantic City, New Jersey, June 16-21, 1990.

[182] J.T. Schwartz and M. Sharir, ''Identification of partially obscured objects in two and three dimensions by matching noisy characteristic curves,'' *Int. J. of Robotics Res.*, vol. 6, no. 2, pp. 29-44, Summer 1987.

[183] A. Shashua, ''Projective structure from two uncalibrated images: Structure from motion and recognition,'' *AI Memo 1363*, Artificial Intell. Lab, MIT, September 1992.

[184] A. Shashua, ''Projective depth: A geometric invariant for 3D reconstruction from two perspective/orthographic views and for visual recognition,'' *in Proc. 4'st Int. Conf. Computer Vision (ICCV-93)*, pp. 583-590, Berlin, Germany, May 11-14, 1993.

[185] A. Shashua, ''On geometric and algebraic aspects of 3D affine and projective structures from perspective 2D views,'' *in Proc. Applications of Invariance in Comput. Vision*, pp. 127-143, Ponta Delgada, Azores, Portugal, October 9-14, 1993.

[186] A. Shashua, ''Algebraic functions for recognition,'' *AI Memo 1452*, Artificial Intell. Lab, MIT, January 1994.

[187] A. Shashua, ''Trilinearity in visual recognition by alignment,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 1, pp. 479-484, Stockholm, Sweden, May 2-6, 1994.

[188] A. Shashua, ''Projective structure from uncalibrated images: Structure from motion and recognition,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 16, no. 8, pp. 778-790, August 1994.

[189] A. Shashua and N. Navab, ''Relative affine structure: Theory and application to 3D reconstruction from perspective views,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-94)*, pp. 483-489, Seattle, Washington, June 21-23, 1994.

[190] A. Shashua and S. Toelg, ''The quadric reference surface: applications in registering views of complex 3D objects,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 2, pp. 407-416, Stockholm, Sweden, May 2-6, 1994.

[191] F.Y. Shih and W.T. Wong, ''A new single-pass algorithm for extracting the Mid-Crack Codes of Multiple Regions,'' *J. Visual Communication and Image Representation*, vol. 3, no. 3, pp. 217-224, 1992.

[192] D.B. Shu and G. Nash, ''The gated interconnection network for dynamic programming,'' *in Concurrent Computations - Algorithms, Architecture, and Technology, Chapter 32, Ed. S.K. Tewksbury, B.W. Dickinson and S.C. Schwartz*, pp. 645-658, Plenum Press, New York City, New York, 1988.

[193] H.J. Siegel, L.J. Siegel, F.C. Kemmerer, P.T. Mueller, JR., H.E. Smalley, JR., and S.D. Smith, ''PASM: A partitionable SIMD/MIMD system for image processing and pattern recognition,'' *IEEE Trans. Comput. (T-C)*, vol. C-30, no. 12, pp. 934-947, December 1981.

[194] G. Sparr, ''A common framework for kinetic depth, reconstruction and motion for deformable objects,'' *in Proc. 3rd European Conf. on Comput. Vision (ECCV-94)*, vol. 2, pp. 471-482, Stockholm, Sweden, May 2-6, 1994.

[195] F. Stein and G. Medioni, ''Efficient two dimensional object recognition,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 1, pp. 13-17 , Atlantic City, New Jersey, June 16-21, 1990.

[196] F. Stein and G. Medioni, ''Recognition of 3-D objects from 2-D groupings,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 667-674, San Diego, California, January 26-29, 1992.

[197] F. Stein and G. Medioni, ''Structural indexing: Efficient 3-D object recognition,'' *IEEE Trans. Patt. Anal. Machine Intell (T-PAMI)*, vol. 14, no. 2, pp. 125-145, February 1992.

[198] G. Stockman, ''Object recognition and localization via pose clustering,'' *Comput. Vision, Graphics, Image Processing (CVGIP)*, vol. 40, pp. 361-387, 1987.

[199] P. Suetens, P. Fua, and A.J. Hanson, ''Computational strategies for object recognition,'' *ACM Computing Surveys*, vol. 24, no. 1, pp. 5-61, March 1992.

[200] M.H. Sunwoo and J.K. Aggarwal, "VisTA: An image understanding architecture," *in Parallel architecture and algorithms for image understanding, V.K. Prasanna Kumar ed.*, pp. 121-154, Academic Press, Boston, Massachusetts, 1991.

[201] R.E. Suorsa and B. Sridhar, "A parallel implementation of a multisensor feature-based range-estimation method," *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-93)*, pp. 379-385, New York City, New York, June 15-17, 1993.

[202] R. Szeliski and S.B. Kang, "Recovering 3D shape and motion from image streams using non-linear least squares," *DEC Technical Report, CRL 93/3*, March 1993.

[203] Y.Y. Tang, X. Cheng, L. Tao, and C.Y. Suen, "Parallel regional projection transformation (RPT) and VLSI implementation," *Patt. Recogn.*, vol. 26, no. 4, pp. 627-641, 1993.

[204] C.H. Teh and R.T. Chin, "Two dimensional CAD-based object recognition," *in Proc. 9'th IEEE Int. Conf. Patt. Recogn (ICPR-88)*, pp. 382-384, Rome, Italy, November 1988.

[205] W.H. Tsai and S.S. Yu, "Attributed string matching with merging for shape recognition," *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 7, no. 4, pp. 453-462, July 1985.

[206] L.W. Tucker and G.G. Robertson, "Architecture and applications of the connection machine," *Comput.*, vol. 21, no. 8, pp. 26-38, August 1988.

[207] L.W. Tucker, C.R. Feynman, and D.M. Fritzsche, "Object recognition using the connection machine," *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-88)*, pp. 871-878, Ann Arbor, Michigan, June 5-9, 1988.

[208] J.L. Turney, T.N. Mudge, and R.A. Volz, "Recognizing partially occluded parts," *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 7, no. 4, pp. 410-421, July 1985.

[209] J.L. Turney, T.N. Mudge, and R.A. Volz, "Recognizing partially hidden objects," *in Proc. IEEE Int. Conf. Robotics and Automation (ICRA-85)*, pp. 48-54, March 1985.

[210] A. Tyagi and M. Bayoumi, "Systolic array implementation of image segmentation by a directed split and merge procedure," *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 491-493, Atlantic City, New Jersey, June 16-21, 1990.

[211] L. Uhr, ed., *Parallel computer vision*, Academic Press, Boston, Massachusetts, 1987.

[212] S. Ullman and R. Basri, "Recognition by linear combinations of models," *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 13, no. 10, pp. 992-1006, October 1991.

[213] J.R. Ullmann, ''Analysis of 2-D occlusion by subtracting out,'' *IEEE Trans. Patt. Anal. and Mach. Intel. (T-PAMI)*, vol. 14, no. 4, pp. 485-489, April 1992.

[214] B.C. Vemuri and J.K. Aggarwal, ''Localization of objects from range data,'' *in Proc. IEEE Int. Conf. Comput. Vision and Patt. Recogn. (CVPR-88)*, pp. 893-898, Ann Arbor, Michigan, June 5-9, 1988.

[215] H. Voorhees, D.M. Fritzsche, and L.W. Tucker, ''Exploiting data parallelism in vision on the connection machine system,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 617-622, Atlantic City, New Jersey, June 16-21, 1990.

[216] W. Wang and S.S. Iyengar, ''Efficient data structures for model-based 3-D object recognition and localization from range images,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 14, no. 10, pp. 1035-1045, October 1992.

[217] J.A. Webb, ''Steps toward architecture - Independent image processing,'' *Comput.*, vol. 25, no. 2, pp. 21-31, February 1992.

[218] J.A. Webb, ''Overcoming the barriers to architecture-independent image processing,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 1037-1048, San Diego, California, January 26-29, 1992.

[219] J.A. Webb, ''Implementation and performance of fast parallel multi-baseline stereo vision,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 1005-1010, Washington, District of Columbia, April 18-21, 1993.

[220] C.C. Weems, D. Rana, D.B. Shu, and J.G. Nash, ''An overview of architecture research for image understanding at university of Massachusetts,'' *in Proc. IEEE Int. Conf. Patt. Recogn (ICPR-90)*, vol. 2, pp. 379-384, Atlantic City, New Jersey, June 16-21, 1990.

[221] C.C. Weems, ''Architectural requirements of image understanding with respect to parallel processing,'' *Proc. IEEE*, vol. 79, no. 4, pp. 537-547, April 1991.

[222] C.C. Weems, E. Riseman, A. Hanson, and A. Rosenfeld, ''Preliminary results from the DARPA integrated image-understanding benchmark exercise,'' *in Parallel architecture and algorithms for image understanding, V.K. Prasanna Kumar ed.*, pp. 399-419, Academic Press, Boston, Massachusetts, 1991.

[223] C. Weems, M. Herbordt, M. Scudder, J. Burrill, R. Lerner, and T. Williams, ''Status and current research in the image understanding architecture effort,'' *in Proc. of the DARPA Image Understanding Workshop*, pp. 269-283, San Diego, California, January 26-29, 1992.

[224] F. Weil, ''Dynamic intelligent scheduling and control of reconfigurable parallel architectures for computer vision/image processing,'' *J. of Parallel and Distributed Computing*, vol. 13, pp. 273-285, 1991.

[225] N. Weste, D.J. Burr, and B.D. Ackland, ''Dynamic time warp pattern matching using an integrated multiprocessing array,'' *IEEE Trans. Comput. (T-C)*, vol. C-32, no. 8, pp. 731-744, August 1983.

[226] P. Wohl and T.W. Christopher, ''MIMD implementation of neural networks through pipelined, parallel communication trees,'' *in Proc. IEEE Int. Conf. Tools for AI*, pp. 82-89, San Jose, California, November 1991.

[227] H.J. Wolfson, ''On curve matching,'' *in Proc. IEEE Comput. Society Workshop on Comput. Vision*, pp. 307-310, Miami Beach, Florida, November 1987.

[228] H.J. Wolfson, ''On curve matching,'' *IEEE Trans. Patt. Anal. Machine Intell. (T-PAMI)*, vol. 12, no. 5, pp. 483-489, May 1990.

[229] W.T. Wong, Y.L. Chen and F.Y. Shih, ''A fully parallel algorithm for the extraction of chain and mid-crack codes of multiple contours,'' *in Proc. Int. Computer Symposium*, Hsinchu, Taiwan, December 12-15, 1994.

[230] C.T. Zahn and R.Z. Roskies, ''Fourier descriptors for plane closed curves,'' *IEEE Trans. Comput. (T-C)*, vol. C-21, no. 3, pp. 269-281, March 1972.

[231] T.Y. Zhang and C.Y. Suen, ''A fast parallel algorithms for thinning digital patterns,'' *Commun. ACM*, vol. 27, no. 3, pp. 236-239, March 1984.

[232] A. Zisserman and S.J. Maybank, ''A case against epipolar geometry,'' *in Proc. Applications of Invariance in Comput. Vision*, pp. 69-88, Ponta Delgada, Azores, Portugal, October 9-14, 1993.