Intelligent Robots and Computer Vision

David P. Casasent/Ernest L. Hall
Chairmen/Editors

Proceedings of SPIE—The International Society for Optical Engineering

Volume 521

Intelligent Robots and Computer Vision

David P. Casasent/Ernest L. Hall
Chairmen/Editors

Cooperating Organizations
Carnegie-Mellon University, Robotics Institute • Catholic University of Leuven (Belgium)
Massachusetts Institute of Technology, Artificial Intelligence Laboratory
University of Cincinnati, Center for Robotics Research • University of Rhode Island, Robotics Research Center
UCLA Computer Science Department

November 5-8, 1984
Cambridge, Massachusetts

Published by
SPIE—The International Society for Optical Engineering
P.O. Box 10, Bellingham, Washington 98227-0010 USA
Telephone 206/676-3290 (Pacific Time) • Telex 46-7053

SPIE (The Society of Photo-Optical Instrumentation Engineers) is a nonprofit society dedicated to advancing engineering and scientific applications of optical, electro-optical, and optoelectronic instrumentation, systems, and technology.

The papers appearing in this book comprise the proceedings of the meeting mentioned on the cover and title page. They reflect the authors' opinions and are published as presented and without change, in the interests of timely dissemination. Their inclusion in this publication does not necessarily constitute endorsement by the editors or by SPIE.

Please use the following format to cite material from this book:
Author(s), "Title of Paper," *Intelligent Robots and Computer Vision,* David P. Casasent, Ernest L. Hall, Editors, Proc. SPIE 521, page numbers (1985).

Library of Congress Catalog Card No. 84-052454
ISBN 0-89252-556-8

Printed in the United States of America.

INTELLIGENT ROBOTS AND COMPUTER VISION

Volume 521

Conference Committee

Chairmen: **David P. Casasent,** Carnegie-Mellon University; **Ernest L. Hall,** University of Cincinnati

Program Committee

Michael Brady, Massachusetts Institute of Technology; **Maurice Dunne,** Unimation Incorporated; **O. D. Faugeras,** INRIA, France; **Susan Hackwood,** AT&T Bell Laboratories; **Robert B. Kelley,** University of Rhode Island; **Allen Klinger,** University of California/Los Angeles; **Nelson Marquina,** Honeywell Systems & Research Center; **André Oosterlinck,** Catholic University of Leuven, Belgium

Session Chairmen

Session 1—Recognition for Intelligent Robots, **David P. Casasent,** Carnegie-Mellon University
Session 2—Image Processing for Robot Vision, **Ernest L. Hall,** University of Cincinnati; **William J. Wolfe,** Martin Marietta Denver Aerospace
Session 3—Artificial Intelligence in Image Understanding, Computer Vision, and Robotics, **Nelson Marquina,** Honeywell Systems & Research Center; **Allen Klinger,** University of California/Los Angeles
Session 4—Robot Path Planning and Visual Control, **Robert B. Kelley,** University of Rhode Island
Session 5—Tactile and Other Robotic Sensors, **André Oosterlinck,** Catholic University of Leuven, Belgium
Session 6—Three-dimensional Representation, Modeling, and Processing, **Michael Brady,** Massachusetts Institute of Technology; **Susan Hackwood,** AT&T Bell Laboratories
Session 7—Robotic Systems and Applications, **Maurice Dunne,** Unimation Incorporated

INTELLIGENT ROBOTS AND COMPUTER VISION

Volume 521

INTRODUCTION

This third international SPIE conference on intelligent robots and computer vision contains 55 papers by authors from 10 countries (including France, Germany, Belgium, Canada, Finland, Greece, China, and Turkey). The theme *intelligent robots* refers to reprogrammable, flexible, multifunctional manipulators. Intelligent robots require: advanced vision, touch and other sensors; advanced pattern recognition and image processing algorithms; new high speed processors; and the use of image understanding and artificial intelligence techniques. These topics and others, specific applications, and issues such as 3D data handling, were addressed in individual sessions.

Improved pattern recognition and image processing algorithms, architectures and modular designs are essential for smart robots. In pattern recognition, various feature extractors, plus log mapping, Wigner functions and moment invariants are described. In all cases, distortion-invariant pattern recognition is the concern. Different image processing functions for mobile robots, bin picking, and occluded object processing are addressed and perspective and multi-resolution processors are described.

In a session on image understanding (IU) and artificial intelligence (AI), automatic target recognition, autonomous robots, and the role of AI in robotics are addressed. Rule-based systems, models and primitives for AI and IU use are described. AI and IU methodologies are essential for control of advanced intelligent robots. Mobile robots are another major current topic. Various trajectory and task planning techniques and adaptive robot control systems were presented and described on this topic.

Camera, tactile and 3D sensors (vital for intelligent robots) were the subject of another session. Advances in each area and other sensor systems were detailed in these papers. 3D image processing, modeling, representation and data acquisition are addressed in another set of papers. The robot systems and applications session describes applications including measurement, assembly, inspection, detection of cracks, printed wiring board inspection.

We thank the various cooperating organizations for their help, Tony McPherson of the Charles Stark Draper Laboratory Inc. for arranging the post conference lab and facility tour of several robotics companies, our program committee, and the session chairmen and all authors.

It has again been a great pleasure to work on this annual meeting (the third in a continuing series) and the fourth major SPIE robotics conference.

David P. Casasent
Carnegie-Mellon University

Ernest L. Hall
University of Cincinnati

INTELLIGENT ROBOTS AND COMPUTER VISION

Volume 521

Contents

原书模糊

INTELLIGENT ROBOTS AND COMPUTER VISION

Volume 521

Session 1

Recognition for Intelligent Robots

Chairman
David P. Casasent
Carnegie-Mellon University

Chord Distributions in Pattern Recognition: Distortion Invariance and Parameter Estimation

Wen-Thong Chang and David Casasent

Carnegie-Mellon University
Department of Electrical and Computer Engineering
Pittsburgh, Pennsylvania 15213

Abstract

The use of chord distributions in pattern recognition is discussed and efficient ways to compute such distributions are noted. New methods to achieve scale and in-plane rotational distortion-invariant multi-class recognition and estimates of the distortion parameters are described. 3-D out-of-plane rotational distortion-invariant methods are reviewed.

1. Introduction

Chord distributions are well-known features that describe the shape of an object and that are useful for object identification [1-3]. These features can easily be computed (optically or digitally) from the autocorrelation. In Section 2, we define the chord distribution and discuss different chord pdfs. These include an observation space $h(\ell_x,\ell_y)$ and a feature space h(r) and h(θ). New insight is provided into the local and global features produced by chord pdfs and the use of silhouette and boundary (profile) imagery. In Section 3, attractive properties of these chord distributions for scale and in-plane rotation invariance are discussed. A new use of such features for distortion-invariant multi-class object recognition and methods to extract the object's scale and orientation are advanced. In Section 4, methods to achieve 3-D object distortion-invariance (to out-of-plane rotations) are reviewed. The resultant feature extractor thus enables multi-class object classification in the presence of a wide variety of geometrical distortions.

2. Chord Features and Distributions

2.1 Definition. The conventional chord distribution h(r,θ) is a plot of the distribution of the lengths (r) and directions (θ) of all chords drawn between all pairs of points on the boundary of the object f(x,y). The two chord pdfs of most use are h(r) and h(θ), the pdfs of chord lengths r and directions θ. To most easily compute the various chord distributions, one can begin by forming the autocorrelation

$$b(x,y) \circledast b(x,y) = \iint b(x,y)\, b(x-\ell_x,\, y-\ell_y)\,dx\,dy = R(\ell_x,\ell_y) = h(\ell_x,\ell_y) \qquad (1)$$

of the boundary b(x,y) of an object. The autocorrelation describes the number of points of intersection for a given horizontal and vertical shift (ℓ_x,ℓ_y) between two shifted images of the object. The value of R at a given $(\ell_x,\ell y)$ thus precisely gives the number of chords with given horizontal and vertical projection lengths (ℓ_x,ℓ_y) [3-4].

To show this, we write $(\ell_x,\ell_y) = (r\cos\theta,\ r\sin\theta)$ where $r = (\ell_x{}^2+\ell_y{}^2)^{1/2}$ is the radial chord length and $\theta = \tan^{-1}(\ell_x/\ell_y)$ is the chord's angular orientation. Substituting into (1), we see that $R(\ell_x,\ell_y)$ contains information from which h(r,θ) can be obtained. From $h(\ell_x,\ell_y)$, the chord distribution h(r,θ) can be calculated. The chord pdfs h(r) and h(θ) are more useful and are most easily calculated from $h(\ell_x,\ell_y)$ by appropriately sampling the autocorrelation function. If the autocorrelation is sampled radially, we obtain

$$h(r) = \int h(\ell_x,\ell_y)\,r\,d\theta \qquad (2)$$

If we sample it angularly, we obtain

$$h(\theta) = \int h(\ell_x,\ell_y)\,dr \qquad (3)$$

2.2 Realization. These h(r) and h(θ) chord pdfs are the features we will use. To obtain (2) and (3) optically, we form $h(\ell_x,\ell_y)$ optically (typically from the Fourier transform of the power spectrum of the object) and sample this distribution using wedge and ring-shaped detector elements [4]. Such a detector unit exists (Figure 1) with 32 wedges in one-half of a circular plane and 32 rings in the other hall of the plane [5]. The autocorrelation function is symmetric and thus no loss of information results by sampling only half of the autocorrelation plane. In terms of chord distributions, the symmetry of the autocorrelation function arises because each chord in the image is counted twice as one traverses the boundary of the object. In one case, one end point of the chord is encountered first and then the other end point is encountered first. The first corresponds to a chord with projections

(ℓ_x,ℓ_y) and a length r. The symmetric case corresponds to a chord with projections $(-\ell_x,-\ell_y)$ and a direction $-\theta$ rather than $+\theta$. For similar reasons of symmetry, the orientation of the wedge and ring halves of the detector does not matter. The wedge outputs provide h(θ) (quantized to 32 θ values over 180°) and the ring outputs provide h(r) (quantized to 32 r values over the radius of the autocorrelation function). Figure 2 shows the general block diagram of our chord distribution feature generator using a wedge-ring detector (WRD).

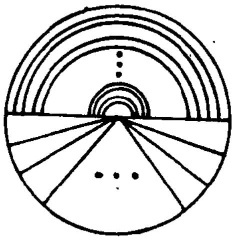

Figure 1. Simplified representation of a wedge-ring detector (WRD).

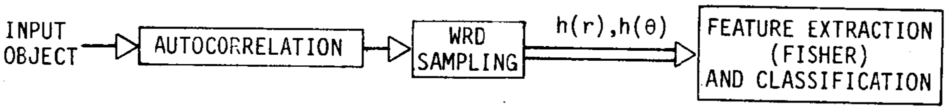

Figure 2. Simplified Block Diagram of a chord distribution pattern recognition system.

2.3 Boundary, Silhouette and Gray-Level Objects. Different chord distributions result depending on the type of input object. For a boundary or edge image (case A), the distribution produced is of the number of edge or boundary pixels (i.e., the number of chords). This is the conventional chord distribution. For a silhouette image (binary with all ones on the object and with zeroes on the background), the distribution produced (case B) is the same as case A, but weighted by the common area of overlap of the two images for the given (ℓ_x,ℓ_y) shift. If the shift is large, corresponding to long chords, the weighting will be small. However, if the shift is small, corresponding to short chords, the weighting will be large. Thus, this weighted chord distribution that results for the case of a silhouette object (case B) emphasizes short chords more than long chords. The chord distribution in case A will be more susceptible to noise in the interior of the object (internal pixels of value 1 result in many new chords being produced in case A, whereas in case B zero internal pixels cause a loss of chords but a much lower percent change results than in case A). When the chord distribution in case A is computed from the autocorrelation or power spectrum (as in Sections 2.1 and 2.2), it is much simpler to calculate than by other methods which have great difficulty when applied to a non-continuous boundary. However, each missing boundary pixel in case A will still result in a loss in the number of chords counted.

The weighted chord distribution (case B) emphasizes short chords. These correspond to local object features (whereas long chords correspond to global object features). Since local object features are useful for discrimination between object classes (inter-class), we expect the weighted chord distributions to provide superior object discrimination. Long chords, corresponding to global object features, are more useful for intra-class object recognition (within one object class, in the face of various object distortions). The performance of weighted chord distribution features in the presence of noise in the input is expected to be superior to the use of conventional chord features. In a boundary image (case A) with N pixels on the boundary, each noise pixel on the object produces N new chords and each missing boundary pixel (due to noise) causes N chords to be removed from the distribution. With N^2 total chords, each noise pixel thus changes the total h by a factor 1/N. In case B, each weighting function is on the order of N^2 (this is more true for short chords than long chords) and thus each noise pixel produces a change in h by a factor of only $1/N^2$ (this is a considerable improvement, since N is usually quite large). For the same reason that the change in h for short chords is less susceptible to noise, it will also be less susceptible to small differences in the object's shape (due to distortions). but changes due to sufficiently different objects are still retained.

The dynamic range of the chord features in cases A and B appears to be comparable. Since use of the boundary image (case A) whitens the image's spectrum and results in a sharper correlation function compared to the broader correlation pattern that results in case B, wedge-ring detection in case B is much simpler. Case B is clearly preferable from noise considerations, its inter-class discrimination is clearly enhanced and its intra-class recognition should be retained. Since all chords are available (and more easily detectable in case B), one can use the preferable chord features (short or long, local or global) for a given problem.

If the gray-levels of the object and its internal structure are reliable, then the chord distribution for the gray-level image (case C) is most useful. The distribution in case B is one level of a general chord distribution. The distribution in case C is a higher-level of generalized chord distribution [4]. In this case, the chord distribution for all internal chords or internal object points is provided. Algorithms such as (1) with the boundary object b(x,y) replaced by the full object f(x,y) provide such features with no increase in computational load for optical systems (digital systems can achieve simplified correlations when operating on binary imagery).

3. Scale and Rotation-Invariant Chord Processor

3.1 Insight. The chord pdf h(r) is invariant to in-plane rotation of the object. This is obvious since the in-plane rotation of an object does not alter its radial distribution. The chord pdf h(θ) simply shifts with in-plane rotations. This follows directly since $h_1(\ell_x,\ell_y) = h(r\cos\theta, r\sin\theta)$ changes to $h_2(\ell_x,\ell_y) = h[r\cos(\theta+\theta_0), r\sin(\theta+\theta_0)]$ for rotation of the input object by θ_0, i.e. in (r,θ) space, $h_2(r,\theta) = h_1(r,\theta+\theta_0)$. Thus in-plane object rotations rotate $h(\ell_x,\ell_y)$ and translate h(θ). The chord pdf h(θ) is invariant to scale distortions of the object whereas h(r) scales (rather than shifts) with an input scale change α. The invariance of h(θ) with scale is obvious. For a scale change α in the input object, the h(r) distribution scales proportional to α and h(αr) is obtained. As long as half of the correlation plane is sampled in θ and r, the above remarks remain valid [due to the symmetry of the autocorrelation and due to the cyclic shift nature of h(θ)]. Table 1 summarizes these properties.

Table 1. Properties of R(r) and h(θ) distributions

PARAMETER	Feature	Distribution Property	Amplitude Effects
Rotation, θ_0	h(r)	Invariant	None
	h(θ)	Shifts $\propto\theta_0$	None
Scale,	h(r)	Scales $r \rightarrow \alpha r$	α^{-3}
	h(θ)	Invariant	α^{-3}
Translation	h(r)	Invariant	None
	h(θ)	Invariant	None

Table 1 also notes the effects on the amplitudes of the h(r) and h(θ) features. We now detail the origin of these variations. We consider first the effect of a scale change (by a factor of α) in the input object on the amplitudes of h(r) and h(θ). First, we consider the observation space $h(\ell_x,\ell_y)$. The image f(x,y) with scale α = 1 produces h_1. This relates to h_2 for $\alpha \neq 1$ as detailed below. From (1),

$$h_1(\ell_x, \ell_y) = \iint f(x,y)\ f(x + \ell_x, y + \ell_y)dxdy \qquad (4)$$

For the scaled object (scale factor α)

$$h_2(\ell_x,\ell_y) = \iint f(\alpha x,\alpha y)\ f[\alpha(x + \ell_x),\ \alpha(y + \ell_y)]dxdy \qquad (5)$$

Changing variables (u,v) = (αx,αy), we obtain

$$h_2(\ell_x,\ell_y) = (1/\alpha^2) \iint f(u,v)\ f(u + \alpha\ell_x,\ v+\alpha\ell_y)dudv = (1/\alpha^2)h_1(\alpha\ell_x,\alpha\ell_y). \qquad (6)$$

From (6), we see that h_2 is a scaled version of h_1 with the amplitudes scaled by $(1/\alpha^2)$.

Now we consider the effect of scale changes on the h(r) and h(θ) distributions. For the h(r) distribution, we find, from (2),

$$h_1(r) = \int f(r\cos\theta, r\sin\theta) r d\theta \ . \tag{7}$$

For a scaled object (scale factor α),

$$h_2(r) = (1/\alpha^2) \int f(\alpha r\cos\theta, \alpha r\sin\theta) r d\theta = (1/\alpha^3) \int f(\alpha r\cos\theta, \alpha r\sin\theta) \alpha r d\theta = (1/\alpha^3) h_1(\alpha r) \tag{8}$$

Thus, from (7), we find a scale change (by α) between the $h_1(r)$ and $h_2(r)$ distributions and an amplitude scale factor $(1/\alpha^3)$. For $h(\theta)$, the effect of a scale change α is simply

$$h_2(\theta) = (1/\alpha^3) h_1(\theta) \quad , \tag{9}$$

i.e., only an $(1/\alpha^3)$ amplitude factor.

The distribution and amplitude effects of θ_0 and α distortions summarized in Table 1 and detailed above are valid for continuous data and continuous r and θ sampling. Finite r and θ sampling is expected to change the exact results somewhat. Specifically, due to sampling, an exact ratio of α^3 is not expected. Furthermore, the scale change from h(r) to h(αr) can be quite difficult to uncover since the distribution for one scale may lie in 11 rings and the distribution for another scale can easily lie in 6 or 8 rings. Thus, the h(θ) distribution is the most useful one for general (α plus θ_0) distortions. The h(r) scale r changes linearly (to αr) and is thus not a simple shift. When the effect of a finite number of r samples is included, the h(r) effect with α is nonlinear. If we scale the h(r) distribution in r by α, the ratio α^3 then exists between the h(r) for a scaled object and the original h(r) scaled in r by α. Thus, the distribution and amplitude effects of scale are coupled as just detailed. Specifically, this means that the amplitude ratio is α^3, but it is this for different r and αr points in the distribution (not the same r points).

By $g(x,y) = f(\alpha x,\alpha y)$, we describe both the position and value of the pixels. Specifically new pixel (x,y) is old pixel (αx,αy) (i.e. $\alpha > 1$ corresponds to a scale decrease) and the value of the old and new pixel are the same. Our above formulae for amplitude effects proportional to α^{-3} thus apply for binary silhouette images (analogous formulae for gray-scale images can be derived and used if the input data is gray-scale. In such cases, with $\alpha < 1$, we have a larger image with more pixels and more intensity per pixel, since the object is closer and received intensity is proportional to range squared). For binary silhouette images and $\alpha < 1$, the new image is larger. Thus, for a given (ℓ_x, ℓ_y) shift, we obtain more overlap, larger correlation values, more weighting and more chords. Our new h_2 will have larger amplitudes (more chords) than h_1 and this agrees with $h_2 = \alpha^{-2} h_1 > h_1$ predicted.

3.2 Distortion - Invariant (α and θ_0) Pattern Recognition. The insight provided in Section 3.1 and the distortion effects summarized in Table 1 are most useful in devising a new pattern recognition feature extractor (invariant to scale α and in-plane rotation θ_0 distortions). We consider 3 distortion cases separately below and summarize our results in Table 2. From Table 1, we note that the $h(\theta_0)$ distribution is the most useful one in general (since it provides invariance to scale automatically and to rotations if shifted versions of h(θ) are tested; and since the ratio of h(θ) and a reference $h_R(\theta)$ provides an estimate of α, whereas the best shift of h(θ) provides an estimate of θ_0). For only scale distortions, h(θ) is best, and for only rotation distortions, h(r) is best for classification (since these features are invariant to the indicated distortions).

3.2.1 In-Plane Rotations. For the case when θ_0 is the only distortion present, we compare the h(r) distribution $h_R(r)$ for all references R. This provides an estimate of the object class R. Next, for the best reference R (obtained from the h(r) and $h_R(r)$ comparisons), we compare h(θ) and $h_R(\theta)$ for various shifts θ_0 in $h_R(\theta)$. From the $h_R(\theta + \theta_0)$ and h(θ) comparisons, we obtain a verification of our initial class estimate R and an estimate of θ_0. A combination of both h(r) and h(θ) tests thus provides the best class R estimates.

3.2.2 Scale Changes. For the case of an α distortion alone, we compare h(θ) for the test input vs. $h_R(\theta)$ for all references R. We must compare $h(\theta)/h_R(\theta)$ for each θ. The reference R for which this ratio is constant for all θ provides the class estimate R. The ratio $h(\theta)/h_R(\theta)$ provides an estimate of α also. To confirm our R and α estimates, we form h(r) and $h_R(\alpha r)$ for the initial R and α estimates. Agreement of h(r) and $h_R(\alpha r)$ confirms our initial estimates. Combining both the h(θ) and h(r) tests again yields better estimates.

3.2.3 Combined Scale (α) and rotation (θ_0) Distortions. When both α and θ_0 distortions are present (the most general case), analysis relies on h(θ) and is more complex. We form $h(\theta)/h_R(\theta + \theta_0)$ for all R and all shifts θ_0. When this ratio is constant for all θ, the corresponding R, α and θ_0 estimates are obtained. The ratio provides the α estimate.

Table 2 Scale α and In-Plane Rotation θ_0 Invariant Multi-Class Pattern Recognition

CASE	Procedure	Remarks	Results
(A) Rotation θ_0 Only	Compare $h(r)$ and $h_R(r)$	$h(r)$ is Rotation Invariant	Class R Estimate
	Compare $h_R(\theta + \theta_0)$ and $h(\theta)$	$h(\theta)$ shifts with θ_0	Confirms R Estimate Provides θ_0 Estimate
(B) Scale α only	Compare $h(\theta)/h_R(\theta)$ for each θ	Constant Ratio Provides R Ratio Provides α Estimate	Class R and Scale α Estimates
	Compare $h(r)/h_R(\alpha r)$	Confirms above estimate	Confirms R and α Estimates
(C) Rotation θ_0 and Scale α	Compare $h(\theta)/h_R(\theta + \theta_0)$ for all R and all shifts θ_0	Constant Ratio Provides R and θ_0. Ratio gives α	Initial Estimates of R, θ_0, α
	Compare $h(r)/h_R(\alpha r)$	Confirm above Estimates	Confirm R and α Estimates

As a check, we form $h(r)/h_R(\alpha r)$ for the initial R and α estimates. From the constancy of the ratio, we verify our R and α estimates. Forming $h(r)/h_R(\alpha r)$ initially for all α, is more computationally intensive and thus the order chosen appears best. This is also the most general case.

4. Out-Of-Plane Distortions ϕ_0

For α and θ distortions, we require one $h(r)$ and $h(\theta)$ distribution per class R for our training set. To accommodate out-of-plane distortions ϕ , we use several training set images per object class and from all $h(r)$ and $h(\theta)$ features select those with the largest Fisher ratio F (from training set data). We then form a linear discriminant functions $\underline{w}$ that maximizes F for a multi-class feature set. An input test feature vector $\underline{c}$ (chord distribution) is projected onto $\underline{w}$ and the projection value determines the input object class. This algorithm [4] has demonstrated perfect performance in selected image distortion tests.

5. Summary.

Chord distributions $h(r)$ and $h(\theta)$ have been shown to be easily computed from the autocorrelation of the input object and WRD (radial and angular) sampling. Using the various properties (Table 1) of $h(r)$ and $h(\theta)$, a new multi-class pattern recognition system for scale and in-plane rotational distortions was advanced (Table 2). Combined with our prior out-of-plane rotational distortion work (Section 4), this feature space can provide full 3-D object distortion invariance and estimates of the distortion parameters (orientation and scale) of the object.

Acknowledgements

The support of this research by AFOSR (Grant AFOSR-79-0091 and F49620-83-C-0100) is gratefully acknowledged.

References

1. G. Tenery, IEEE Trans. Mil. Electron. ME-7, 196 (1963).
2. D. J. H. Moore and D. J. Parker, Pattern Recognition 6, 149 (1974).
3. S. P. Smith and A. K. Jain, Comput. Graphics Image Process, 20, 1699 (1982).
4. D. Casasent and Wen-Thong Chang, Applied Optics, Vol. 22, 2037, July 15, 1983.
5. H. Kasdan and D. Meade, Proc. Electron. Opt. Syst. Des., 248 (1975).

3-D OBJECT RECOGNITION FROM A SINGLE IMAGE

Fernand S. Cohen
Jean-Francois P. Cayula

Department of Electrial Engineering
University of Rhode Island, Kingston, Rhode Island 02881

Abstract

A conceptually **new algorithm** for 3-D object recognition and shape estimation from a single image is presented. Here **complex** 3-D objects are viewed as concatenation of simple surfaces, essentially planar, cylindrical, and **spherical**. This paper addresses the problem of recognizing these different surfaces along with estimating their shape parameters from a single image. The surfaces are assumed to be Lambertian *illuminated with a point source at infinity, and we allow more than one surface to exist in the image.* Surface classification and recognition relies on exploiting the contours of constant image intensity associated with each surface. By Lambert's law the image intensity for a plane is just a constant; for a cylinder the contours are lines parallel to the axis of the cylinder, whereas for a sphere they are concentric circles (ellipses). The image is partitioned into small square blocks. In each the data patch could either *be classified as planar or nonplanar (cylindrical). That involves looking at whether or not the ratio of the* 2 eigenvalues of the scatter matrix associated with the contours is close to 1. For nonplanar (cylindrical) blocks the angle θ associated with the direction of the lines is computed. Any nonplanar block is classified as cylinder or sphere by considering the distribution of the angles θ in a 3x3 neighborhood centered around the block. Again based on the classification of the blocks (surface type) as well as their direction (θ), the *image is segmented into connected regions. Once a surface region is extracted, shape estimation is achieved.*

1. Introduction

Pioneering work on the inference of shape from shading was done by Horn and his co-workers [1,2,3]. They have used the reflectance map, which shows scene radiance as a function of the surface gradient and the distribution of light sources, to extract 3-D surface information from image data. Their work is a conceptually basic and extremely important approach to scene understanding, but unfortunately is not directly applicable to the robot vision problem which imposes the demands of real-time processing and the ability to handle very noisy, highly variable image data. Cooper and his co-workers [4,5] nicely bypassed that problem by modelling the surface image by 2D quadric polynomial where the approximation is constrained by the 3-D object surface shape. An asymptotically Bayesian classifier is then used for classifying a patch of the image into one of the candidate surfaces. It is assumed here that the patch is a piece of a single surface. Once a patch is classified the 3-D surface shape parameters location and orientation are estimated by fitting lines and ellipses to thresholded data.Because of the polynomial representation and curve fitting, the 3-D surface recognition and estimation problem is computationally simple and applicable to noisy data. Haralick and his coworkers [6] also considered the problem of estimating shape from shading using the Facet approach. In the single image case the problem was formulated as a nonlinear optimization problem which may result in multiple solutions and ambiguous result.

This paper presents a new approach on 3D part recognition and shape estimation from a single image where the contours of constant image intensity are exploited and used for both surface classification segmentation as well as for shape estimation. As a significant percentage of manufactured parts can be approximated by a concatenation of simple surfaces [7], such as planes, cylinders, and spheres, we concentrate here on the problem of classifying and estimating the shape of these surfaces from a single image. We assume that the 3-D surfaces are Lambertian illuminated by a point source at infinity. As any robot vision problem requires real-time processing, the image is partitioned into small square windows or blocks, that could be processed in parallel with appropriate architecture. Locally (on the block level), the block is classified as planar or nonplanar depending on whether or not the contours of constant image intensity are parallel lines. This is followed by a classification of a nonplanar block into cylinder or sphere depending on how the contours directions presented by an angle θ vary from one block to the next on a finite neighborhood. Again as the neighborhood is finite, all nonplanar blocks could be processed in parallel with simple appropriate architecture.

This paper makes the following contributions:

1) The total amount of computation required for classifying a block into planar, cylinder, or spherical is essentially that for classifying a block into a planar or nonplanar and is of the order of N^2 additions (see Appendix B) for a NxN block. This is due to the fact that the decision involved in classifying a nonplanar block does not involve the data in each block in the neighborhood, but rather depends on the θ-distribution in the block neighborhood, and there the computation involved is very small;

2) only the nature of the contours are to be exploited for classification of a single surface patch, as well as for shape estimation. This is achieved at no real added amount of computation. In contrast, as mentioned in [4] it is impossible to extract all the 3-D surface location and orientation from a single 3-D surface location and orientation from a single 2-D quadric polynomial approximation to an image patch, only

single surface classification is possible. Once a patch is classified, then shape estimation is carried through fitting lines and ellipses to thresholded data as explained in [5,8];

3) blocks of small size (16x16) are still classified correctly because we exploit the local as well as the global structure. This has an immediate influence on the processing time when the algorithm is implemented in hardware. Again as no global structure is exploited in [4], the window has to be big enough so that locally a spherical patch doesn't look like a cylindrical or planar patch, and a cylindrical patch doesn't look like a plane (see section (3.2, 3.3);

4) the same approach is extendable for the case where the image has mixed surfaces. As in the single surface patch case, a block is classified by exploiting the local as well as the global structure. By adding the natural constraint that regions that correspond to the same surface are connected, and that the contours vary smoothly we can easily cluster the blocks into connected regions of the same surface type;

5) the paper shades on the importance of image contours as an efficient way for modelling, coding, and synthesizing images of 3-D manufactured parts (see Appendix B).

Finally we should emphasize that when the assumption about the 3-D surface being Lambertian, or the illumination being a point source is not met, the approach developed in this paper appears to be robust. Object boundary and edge detection should be incorporated in the segmentation process and will prove to solve the problem that the segmentation algorithm in the present form can't solve (see section 7). The same approach can also be applied to cones (the incorporation of boundary information and the approach applied to cones are treated in a forthcoming paper).

2. Image formation and image contours

The assumption here is that 3-D surfaces are essentially Lambertian and are illuminated by a point source at infinity. The 3-D structure of the different surfaces are inferred from a single 2-D image by exploring and modelling the contours of constant image intensity associated with each surface. By Lambert's law, the image intensity of a point in the image space is given by

$$I = I_o C \cos \theta \qquad (1)$$

θ = Angle between normal to the surface at a given point and the incident ray.

C = a measure of surface reflectance camera to surface distance.

From (1) we can easily see that for a planar surface I is just a constant. For a cylindrical surface one can show [5,8] that the contours of constant image intensity are straight lines parallel to the cylinder axis, whereas for a spherical surface they are concentric ellipses with constant minor-to-major axis ratio. The center of the sphere lies on a line joining the ellipses centers.

3. Surface recognition -- the classification problem

3.1 Overview

If the image consists of a single surface patch, then surface recognition and shape estimation is achieved by first classifying the patch, and then estimating the parameters of the surface using the patch data. Surface patch classification is achieved by first dividing the image into small windows or blocks that could be processed in parallel with appropriate architecture to achieve speed. On the window level (i.e. locally) the patch in each window can only be classified as cylinder or plane depending on whether or not the contours of constant image intensity are parallel lines. The reason it is impossible locally to differentiate between a spherical patch and a cylindrical patch is that locally the elliptic (circular) contours look like parallel lines. Globally (i.e., by considering all the windows), however, this could be achieved by noting that for a cylinder the angle of orientation of the lines θ will be the same in all the windows, whereas for a sphere it will vary smoothly from -90^o to 90^o along the elliptic (circular) contours. Any window (i,j) is classified as being a cylinder or a sphere depending on whether the θ-distribution on a 3 x 3 neighborhood centered about (i,j) is more consistent with a θ-distribution of a sphere or a cylinder. We consider a finite number of most likely masks for the sphere and the cylinder. A penalizing function (cost) cost(i,j|class) based on a distance measure (in the θ-space) is computed on the 3x3 neighborhood, and the window is classified according to a least cost decision rule.

The image is first classified as a planar or nonplanar surface (cylindrical or spherical). This step relies on a majority rule decision, that is, if the majority of the windows are classified as planar, and since we know we are dealing with a single surface patch, then the whole image patch is classified as planar, and vice versa. If the patch is classified as nonplanar, we compute the total cost associated with all the blocks that are classified as cylinders under cylindrical surface assumption, and compare it to that under the spherical assumption.

3.2 Block-classification--local information

Block classification into planar or nonplanar surface depends on whether the patch data is essentially a constant, or whether the contours of constant image intensity are parallel lines. In order to detect that we use a combination of constrained line fitting and scatter matrix analysis similar to that used in [5,9]. The basic idea is to pick a pair of thresholds, and look at the pixels with image intensity falling in (α,β) as a data set $D_{\alpha\beta} = \{(i,j): \alpha<X_{ij}<\beta\}$ (with X_{ij} being the image intensity of pixel (i,j). This data set constitutes a contour $C_{\alpha\beta}$ of constant image intensity. Next we form the scatter matrix S

$$S_k = \sum_{(i,j)\varepsilon D_{\alpha\beta}}^{N_k} (U_{ij}-U_{\alpha\beta})\ (U_{ij}-U_{\alpha\beta})^t,\ U_{ij} = (i,j),\ U_{\beta} = \frac{1}{N_k} \sum_{(i,j)\varepsilon D_{\alpha\beta}}^{N_k} U_{ij} \tag{2}$$

and compute its 2 eigenvalues and corresponding eigenvectors. If $\lambda_1 >> \lambda_2$, this implies that the data lies on a line with slope given by the orientation of the eigenvector associated λ_1. However, if λ_1 and λ_2 are comparable, this means that the data lies more on a plane than on a line. We have also allowed for a third class (undefined) to account for the case where the cloud doesn't have a pronounced direction. The undefined blocks will be reclassified when considering neighboring blocks. When the block is classified as cylinder the angle θ of orientation of the cylinder axis is obtained by computing the orientation of the biggest eigenvector. The angle θ has been quantized to 8 levels: 90°, 65°, 45°, 20°, 0°, -20°, -45°, -65°.

There are obvious problems associated with arbitrarily picking the thresholds. If we plot the intensity versus the distance for a cylindrical patch in a direction perpendicular to the cylinder axis for most of the source directions the function is nonlinear. The loci is that of an ellipse [8]. An example is shown in Figure 1 , where the cylinder is assumed to be in the (x,y) plane, and the light source in the (y,z) plane but far away from the object. Now if we arbitrarily pick two thresholds (α,β)
$(\alpha-\beta=\Delta)$, then we might wind up with a data set (in the interval δ_1) which form a data cloud which is too wide. Consequently, the eigenvalues of the scatter matrix will be comparable, and the patch will be incorrectly classified as plane. A way around that is to consider an interval δ which is far removed from the maximum image intensity contour. This has the effect of avoiding the highest curvature region where the contours of constant image intensity are wide (i.e., consider data in δ_2 instead of δ_1). Though this solves the problem for the cylinder, it has its drawbacks when dealing with a plane. When dealing with real objects, the image intensity for a planar surface is never constant. An appreciable amount of noise is superimposed on the image and it is seldom the case that the noise is white. This means that when the thresholds are removed from the highest image intensity region, we are essentially amplifying the effect of the noise, and if it happens that the noise has a favored direction (nonwhite) there, we might end up with the misclassification of a plane. A natural solution, therefore, is to consider more than one interval and form the total scatter matrix. Each such interval should have a minimum predefined number of pixels so that the direction of the cloud is well defined.

3.3 Nonuniform histogramming

The gray scale image is first normalized, and the gray levels are mapped onto a (0 to 64) or (0 to 128) scale. Then the image is partitioned into 16x16 squares, and a histogram is computed for each window. Let n(i) be the number of pixels with gray level I (i.e., histogram value of the interval (I-1,I)), and let N be the area under the histogram curve. N here is a discrete variable which can take the values of 0,...,256 for a 16x16 window and serves to control the width of the stripes. We fix N and subdivide the gray level scale (0 to 127) into intervals, each of which has an area of at least N. Let's assume that we have M (M can take 1,...,128)such intervals denoted by $\{(s_I,f_I)\}$, I = 1, ...M. Here s_I and f_I denote the starting and ending point of the interval with $s_1 = 0$, and $s_I = f_I - 1$. The next f_I should staisfy the following inequalities

$$\sum_{j=s_I}^{f_I} n(j) \geq N\ ,\ \text{and} \sum_{j=s_I}^{f_I-1} n(j) < N, \tag{3}$$

and is picked accordingly.
This procedure is repeated until all the pixels are assigned to the intervals. For each interval (s_I,f_I) the coordinates of the points $\{(i,j)\}$ which have an image intensity within the (s_I,f_I) interval form a data cloud. We compute the scatter matrix associated with the data for each cloud and sum all the scatter matrices to form the total scatter matrix S

$$S = \sum_{I=1}^{M} S_I, \tag{4}$$

and compute the eigenvalues and eigenvectors of S. Block classification is then based on the ratio of the eigenvalues of S. The advantage of superimposing clouds is shown in Figure 2. Each arrow represents the eigenvector associated with the largest eigenvalue for each cloud. For a plane (Figure 2a) the eigenvectors are distributed randomly in the (x,y) plane, whereas for a cylinder, they are clustered is a small portion of (x,y) plane (Figure 2b). This means the superposition of the eigenvectors for the plane leads to a resultant eigenvector that lies anywhere within a sphere of radius r_o (Figure 2c), whereas for a cylinder the resultant eigenvector has a large magnitude (much bigger than r_o) and a definite direction (Fig.2b). It follows therefore that basing the decision using superimposed clouds is far more reliable than just using only one cloud (interval). Finally we should note that computing the orientation of the eigenvector associated with the biggest eigenvalue of S is exactly similar to MSE constrained line fitting.

3.4 Global information and cost computation

3.4.1 Global information Up to that point all the blocks are classified as cylinders, undefined or planes. The next stage is to classify each nonplanar block as a cylindrical surface, a spherical surface, or undefined depending on how neighboring blocks have been classified and depending on the angle θ in each cylindrical block in the neighborhood. Here undefined means that the cost incurred under cylinder assumption is the same as under sphere. For speed we limit ourselves to a 3 x 3 neighborhood centered about the block to be classified. The pattern observed is matched with predefined templates for the sphere and the cylinder. A penalizing (cost) function is computed for each match and the best match is determined based on a minimum cost decision rule. For a cylindrical surface the angle θ should ideally be the same in all blocks in the 3x3 neighborhood. Hence for the cylindrical case we match the observed pattern to the following template.

θ	θ	θ
θ	θ	θ
θ	θ	θ

θ is the angle most often found in the 3x3 window.

For a sphere, the angle θ will vary smoothly from -90° to 90° along the elliptic (circular) contours. We have considered a set of 16 templates to account for the most likely patterns encounterd for a spherical surface (see Appendix A on the choice of these templates). The templates are depicted in Figure 4 for the case where the point source is on top of the sphere - - hence circular image contours. A cost is computed for each match of the observed pattern with each one of the 16 templates. The least cost is then compared to that for the cylindrical surface and the block is classified according to a least cost decision. Note that we can compute the cost associated with all the blocks at the same time with special purpose hardware. In the 3x3 neighborhood if some of the blocks are missing or have been classified as plane or undefined they don't contribute in the cost computation.

3.4.2 Cost Computation The cost is computed based on a distance measure in the θ-space. Any block in the 3x3 neighborhood is matched with its corresponding one in the template. If the angles are the same, no cost is incurred. Otherwise, a cost proportional to how many quantization levels apart the two angles are, is incurred. The justification for that measure is that for a spherical surface, θ changes smoothly from one block to a neighboring one and is therefore not likely to be very different from the block at the center. This is even more so when dealing with a cylinder. To illustrate the idea, compute the cost incurred when we match the pattern in (a) with the template in (b). The cost incurred in each block is given in (c)

plane	20°	UND
-20°	0°	0°
-20°	-20°	-45°

(a)

pattern

20°	20°	20°
-20°	0°	0°
-20°	-20°	-65°

(b)

template

0	0	0
0	0	0
0	0	1

(c)

Total cost = 1

3.5 Classification of a nonplanar single surface patch

If an image of a single surface is classified as nonplanar (i.e., most of the blocks are classified as nonplanar), then the patch is classified as cylinder or sphere by computing the total cost associated with all the blocks under cylinder assumption and comparing it to that under the sphere assumption.

$$\sum_{(i,j)\varepsilon C} \text{cost } (i,j|\text{Cyl}_{\theta \text{ max}}) \overset{\text{cylinder}}{\underset{\text{sphere}}{\gtrless}} \sum_{(i,j)\varepsilon C} \text{cost } (i,j|\text{ sphere}) \qquad (5)$$

cyl $_{\theta \max}$ denotes that cylinder with orientation angle $\theta_{\max}$, and $\theta_{\max}$ is the angle most accounted for within the blocks classified as cylinders (C).

3.6 Magnification problem

There is a problem associated with using fixed masks when classifying nonplanar blocks into either cylinders or spheres: the masks are not magnification invariant. This means that if the sphere size is large compared to the block size as shown in Figure , then the blocks that are far removed from the center of the sphere might be classified as cylinders. This is due to the fact that the bandwidth of the contours is big and therefore any block based as 3x3 neighborhood will look like a cylinder. One way of remeding this problem is to consider a larger neighborhood, but this has the disadvantage of slowing down the process. An alternative is to use a "blown-up" mask configuration as shown below.