LIBRARY
LIBRARY OF CONGRESS

Office of Business Enterprises
Duplication Services Section

THIS IS TO CERTIFY that the collections of the Library of Congress contain a book entitled **THREE-DIMENSIONAL MACHINE VISION,** Takeo Kanade, call number TJ 211.3.T47 1987, and that the attached photocopy of the front and back cover pages, Title page, Copyright Page, and Table of Contents pages are a true representation from that work.

THIS IS TO CERTIFY FURTHER, that work is marked with a Library of Congress Cataloging-In-Publication stamp that bears the date March 23, 1987.

IN WITNESS WHEREOF, the seal of the Library of Congress is affixed hereto on February 1, 2019.

Rosalina Delgado-Jones
Assistant Business Enterprises Officer
Office of Business Enterprises
Library of Congress

101 Independence Avenue, SE Washington, DC 20540–4917 Tel 202.707.5650 www.loc.gov;
duplicationservices@loc.gov

Three-Dimensional Machine Vision

Takeo Kanade

LIBRARY OF CONGRESS
00000781940

Kluwer Academic Publishers


THE LIBRARY OF CONGRESS
1800

THE KLUWER INTERNATIONAL SERIES IN ENGINEERING AND COMPUTER SCIENCE

ROBOTICS: VISION, MANIPULATION AND SENSORS

Consulting Editor

Takeo Kanade

Other books in the series:

Robotic Grasping and Fine Manipulation. M. Cutkosky.
ISBN 0-89838-200-9.

Shadows and Silhouettes in Computer Vision. S. Shafer.
ISBN 0-89838-167-3.

Perceptual Organization and Visual Recognition. D. Lowe.
ISBN 0-89838-172-X.

THREE-DIMENSIONAL MACHINE VISION

edited by

Takeo Kanade
Carnegie Mellon University

KLUWER ACADEMIC PUBLISHERS
Boston/Dordrecht/Lancaster

Distributors for North America:
Kluwer Academic Publishers
101 Philip Drive
Assinippi Park
Norwell, MA 02061, USA

Distributors for the UK and Ireland:
Kluwer Academic Publishers
MTP Press Limited
Falcon House, Queen Square
Lancaster, LA1 1RN, UNITED KINGDOM

Distributors for all other countries:
Kluwer Academic Publishers Group
Distribution Centre
Post Office Box 322
3300 AH Dordrecht, THE NETHERLANDS

Cover photograph by Bradley A. Hersch, Monroeville, Pennsylvania.

TJ211
.3
.T47
1987

LIBRARY OF CONGRESS
5
MAR 2 3 1987
COPY
CIP

Library of Congress Cataloging-in-Publication Data

Three-dimensional machine vision.

(The Kluwer international series in engineering and computer science. Robotics)
1. Robot vision. I. Kanade, Takeo.
TJ211.3.T47 1987 629.8'92 86-27599
ISBN 0-89838-188-6

Copyright © 1987 by Kluwer Academic Publishers

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher, Kluwer Academic Publishers, 101 Philip Drive, Assinippi Park, Norwell, MA 02061.

Printed in the United States of America

Table of Contents

Preface

A robot must perceive the three-dimensional world if it is to be effective there. Yet recovering 3-D information from projected images is difficult, and still remains the subject of basic research. Alternatively, one can use sensors that can provide three-dimensional range information directly. The technique of projecting light-stripes started to be used in industrial object recognition systems as early as the 1970s, and time-of-flight laser-scanning range finders became available for outdoor mobile robot navigation in the mid-eighties. Once range data are obtained, a vision system must still describe the scene in terms of 3-D primitives such as edges, surfaces, and volumes, and recognize objects of interest. Today, the art of sensing, extracting features, and recognizing objects by means of three-dimensional range data is one of the most exciting research areas in computer vision.

Three-Dimensional Machine Vision is a collection of papers dealing with three-dimensional range data. The authors are pioneering researchers: some are founders and others are bringing new excitements in the field. I have tried to select milestone papers, and my goal has been to make this book a reference work for researchers in three-dimensional vision.

The book is organized into four parts: 3-D Sensors, 3-D Feature Extractions, Object Recognition Algorithms, and Systems and Applications. Part I includes four papers which describe the development of unique, capable 3-D range sensors, as well as discussions of optical, geometrical, electronic, and computational issues. Mundy and Porter describe a sensor system based on structured illumination for inspecting metallic castings. In order to achieve high-speed data acquisition, it uses multiple light stripes with wavelength multiplexing. Case, Jalkio, and Kim also present a multi-stripe system and discuss various design issues in range sensing by triangulation. The numerical stereo camera developed by Altschuler, Bae, Altschuler, Dijak, Tamburino, and Woolford projects space-coded grid patterns which are generated by an electro-optical programmable spatial

light modulator. Kanade and Fuhrman present a proximity sensor using multiple LEDs which are conically arranged. It can measure both distance and orientation of an object's surface.

Having acquired range data, the next step is to analyze it and extract three-dimensional features. In Part II, Ponce and Brady present the Surface Primal Sketch. They detect, localize, and symbolically describe the significant surface changes: steps, roofs, joints, and shoulders. Vemuri, Mitiche, and Aggarwal represent 3-D object surfaces by patches which are homogeneous in intrinsic surface properties. Sugihara describes how the knowledge of vertex types in polyhedra can be used to guide the extraction of edges and vertices from light-stripe range data.

An important goal of 3-D vision systems is to recognize and position objects in the scene. Part III deals with algorithms for this purpose. After describing object surfaces by curves and patches, Faugeras and Hebert use the paradigm of recognizing while positioning for object matching by exploiting rigidity constraints. Oshima and Shirai have long worked on a system for recognizing stacked objects with planar and curved surfaces using range data. Their paper presents procedures for feature extraction and matching together with recent experimental results. The matching strategy used by Bolles and Horaud to locate parts in a jumble starts with a distinctive edge feature of an object and grows the match by adding compatible features. Grimson and Lozano-Pérez discuss how sparse measurement of positions and surface normals may be used to identify and localize overlapping objects. Their search procedure efficiently discards inconsistent hypothetical matches between sensed data and object model using local constraints.

Finally, Part IV contains three papers discussing applications and system issues of three-dimensional vision systems which use range sensing. Beeler presents a precision 3-D measurement system for replacing damaged or missing tiles on space shuttle vehicles. The acquired shape description of a

tile cavity is used to control a milling machine to produce a new tile. Nakagawa and Ninomiya describe two three-dimensional vision systems that use active lighting: one is for inspecting solder joints and the other for automatic assembly of electronic devices. The paper by Jarvis, arguing for a semantic-free approach to three-dimensional robotic vision, discusses various system issues in a real-time sensory-based robotic system, such as a hand-eye system.

In editing the book I have received great help from Richard Mickelsen and Magdalena Müller. Richard read all of the papers and brought consistency into the book. Maggie spent untiring effort for typing and perfecting the format. I am heartily grateful: without them, this book would not have been completed.

Takeo Kanade

PART I: 3-D SENSORS

A Three-Dimensional Sensor Based on Structured Light

J.L. Mundy
G.B. Porter III
General Electric Company
Corporate Research and Development
Schenectady, NY 12345

Abstract

This paper describes the design of a 3-D range sensor using structured light to provide multiple range points by triangulation. The algorithms and sensor hardware are described along with the supporting theory and design strategy.

1. Introduction

The formulation of three-dimensional descriptions of objects from two-dimensional perspective images is a major goal for machine vision research. In many cases, the images are formed using unconstrained lighting and camera position. The interpretation of such scenes relies on features such as shadows and intensity discontinuities as the primitives from which the object descriptions are formed. It is now recognized that this process cannot be carried out successfully without detailed *a priori* models of the objects and the properties of illumination and image formation. The nature of such models and the design of efficient matching algorithms are still the focus of intensive research efforts in image understanding. Existing techniques are not yet robust enough to form the basis for a practical industrial inspection or robot guidance system.

An alternative approach is to carefully control the illumination and position of the objects to be described. In a calibrated configuration, it is possible to derive an independent description of object surfaces without additional *a priori* knowledge. This description is in the form of three-

dimensional coordinate triples taken at intervals over the object surface. In many industrial applications this description is sufficient to determine product quality or object position for automated assembly. The various approaches for controlled illumination have often been presented under the name structured light or structured illumination.

This paper describes a sensor system that uses structured illumination to inspect metallic castings. The discussion focuses on optical design considerations and high speed processing techniques for converting intensity images into a range image. In addition, a number of examples of sensor performance are discussed.

2. Structured Light Principles

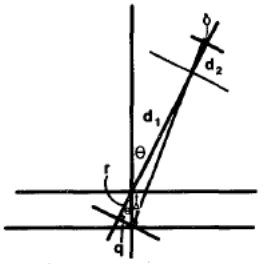
2.1. Simple Triangulation

The underlying principle of structured light methods is optical triangulation. This principle is illustrated with a single ray of light impinging a surface, as shown in Figure 1. The image of the ray is formed along an optical axis which is at an angle with the ray. This angle, θ, is known as the parallax angle. It follows that the image of the ray shifts in the image plane according to the position of the surface along the ray. This parallax shift can be used to calculate the surface location provided that the geometric configuration of the camera and ray are known. In practice, such a system can be calibrated by measuring a known object surface, such as a plane.

The relationship between the ray image and the surface position is given by

$$\Delta = \frac{\delta d_1}{d_2 sin\theta - \delta cos\theta}. \tag{1}$$

If the image field of view is limited so that the parallax angle can be considered constant, then the relation reduces to



$$\frac{q}{\delta} = \frac{d_1 + r}{d_2} \qquad q = \Delta \sin\theta \quad r = \Delta \cos\theta$$

$$d_2\,\Delta\sin\theta = \delta d_1 + \delta\Delta\cos\theta$$

$$\Delta(d_2 \sin\theta - \delta\cos\theta) = \delta d_1$$

$$\text{or} \quad \Delta = \frac{\delta d_1}{d_2 \sin\theta - \delta\cos\theta}$$

Figure 1: The geometry of parallax triangulation. θ is the parallax angle, Δ is the height of the surface above the reference position, and δ is the shift in the image of the beam.

$$\delta = \Delta \left[\frac{d_2}{d_1} \right] sin\theta \quad . \tag{2}$$

In this case, there is a linear relation between the ray image position and object surface position. This relation is conceptually useful, but not accurate enough for industrial measurement applications.

The object surface can be sampled by sweeping the ray over a range of angles within the field of view of the camera. If the triangulation relation of the ray image is calibrated for each ray position, it is possible to derive an array of surface positions within the field of view. This array is known as the range image. In this case, the range is defined as the distance from the unknown surface to a given reference surface such as a plane. The reflectance of the surface can be obtained by measuring the intensity of the ray image for each ray position.

2.2. More Complex Patterns

This simple approach to three dimensional measurement has been used in many industrial applications such as controlling the height of liquids and simple gauging instruments. However, the speed of such systems is limited by the rate at which the location of the ray image can be determined. A common sensor for this purpose is a solid state array camera which can operate at about 30 fields of view, or frames, per second; this corresponds to about 30 points per second of range information. This performance can be improved by using a one-dimensional image sensor or line array. These devices can be read out much faster, producing a range sample rate of several thousand points per second.

Another approach which can improve the range sample rate is to project a stripe of light rather than a single spot, as shown in Figure 2. Each row of

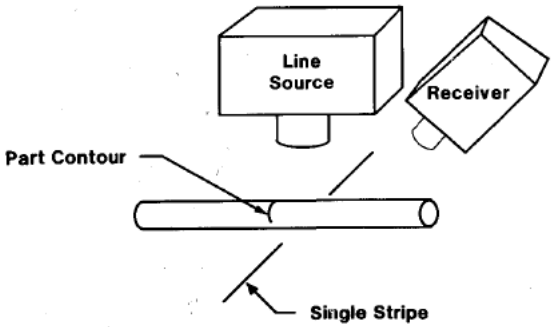


Figure 2: The projection of a single stripe on an object surface. The apparent location of the stripe in the image depends on the surface height due to parallax.

the image sensor is used to triangulate the position of the stripe along that row. This multiplies the throughput of range sensing by a factor

corresponding to the number of rows in the image sensor. If the sensor array can resolve 128 stripe positions, the range sample rate is about 5000 samples per second.

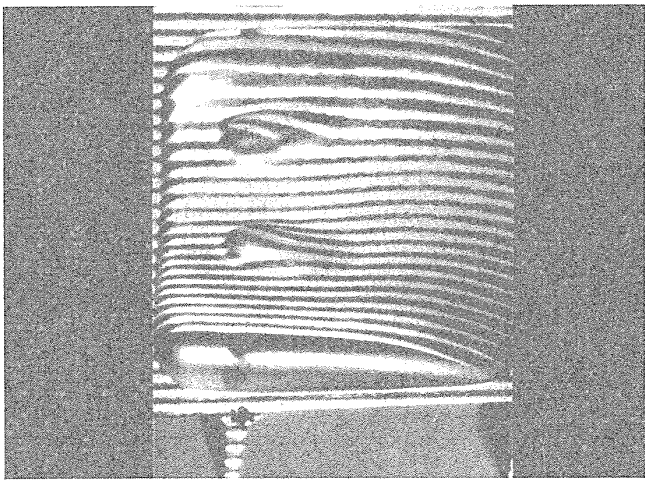


Figure 3: The projection of multiple stripes. The intricate surface profile of the casting is clearly revealed by the stripe contours.

Additional parallelism can be obtained by projecting more than one stripe, as shown in Figure 3. In this case, the details of the three-dimensional surface are clearly exposed. Again, it is possible to calibrate the relationship between each stripe position and the three-dimensional surface range. A major drawback to this method is that the stripes can be confused with each other. If the surface position is far enough from the reference position to shift the stripe image more than the stripe spacing, the range becomes ambiguous. The use of multiple stripes is attractive because, by virtue of its parallelism, it can produce a throughput of several hundred thousand range samples per second.

The ambiguity of the multiple stripes can be eliminated by projecting a number of stripe patterns in sequence. The set of stripes form a Grey code

sequence so that each surface point has a unique code value. Figure 4 illustrates such a set of patterns. An image point is recorded as illuminated

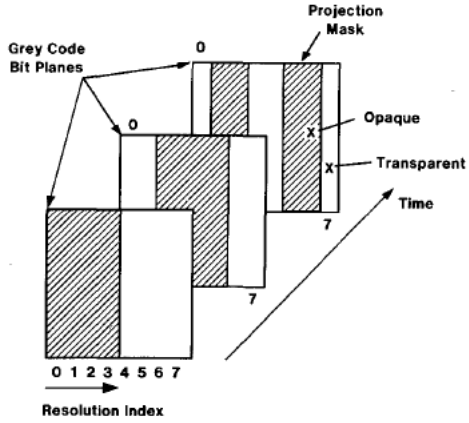


Figure 4: A set of Grey code patterns for time multiplex projection. The patterns define a unique binary code for each resolution element.

or unilluminated for each projected light pattern. The shift of each unique code from the reference position is determined and a range image is calculated. The number of stripe patterns required is proportional to the logarithm of the number of image points along each row. For example, with 256 image points in each row, 8 stripe code patterns are required. At 30 sensor image frames per second, this approach provides about 200,000 range points per second.

3. Limitations of the Structured Illumination Method

3.1. The Problem of Reflectance Variation

In the discussion above, it is assumed that the illumination ray has infinitesimal diameter, and the position of the ray image can be determined precisely. In practice, these ideal conditions cannot be met, and variations in surface reflectance produce errors in the range measurements. In this section the relation between reflectance variation and range error in the case of a single ray is analyzed.

The intensity cross section of the image of a laser beam is approximately a Gaussian function. That is,

$$I(x) = I_0 \, e^{-\left[\frac{(x-x_0)}{\sigma}\right]^2} \tag{3}$$

where x_0 is the beam center and I_0 is the intensity there. This function is shown in Figure 5(a). Note that the spot intensity function is circularly symmetric. For simplicity, this analysis will be carried out for the spot cross section, with the obvious extension for the symmetry. The analysis is, however, directly applicable to the practical case where a one-dimensional image sensor is used.

There are a number of methods for determining the location of the beam center in the image plane. The center location provides the parallax shift data necessary for the calculation of range values. The most common method is to compute the first moment of image intensity. If this is normalized by the zeroth moment, the result gives an estimate for the beam center, x_0.

For an ideal Gaussian, it follows that the ratio of the first intensity moment to the zeroeth moment is identically the beam center.

$$x_0 = \frac{\int_{-\infty}^{\infty} x e^{-\left[\frac{(x-x_0)}{\sigma}\right]^2}}{\int_{-\infty}^{\infty} e^{-\left[\frac{(x-x_0)}{\sigma}\right]^2}} \tag{4}$$

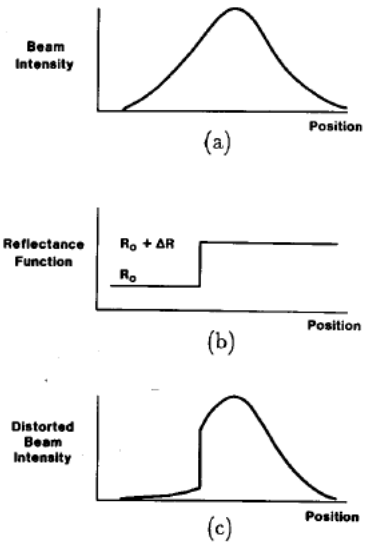


Figure 5: The effect of reflectance on the location of the center of a Gaussian beam. (a) The ideal Gaussian distribution; (b) A step in surface reflectance; (c) The resulting image intensity function. Note that it is no longer symmetrical about the beam center.

Actually, this result holds for any beam intensity cross section that is symmetric about x_0.

Now suppose that the incident beam is reflected from a surface that has variable reflectance, $R(x)$. The resulting beam image intensity is given now by

$$I(x) = P(x)\, R(x)\,, \tag{5}$$

where $P(x)$ is the incident illumination power density at the point x on the surface. The form of this modified intensity is plotted in Figure 5(c). The ratio of Equation (4) no longer corresponds to the center of the beam. This asymmetry introduces an error in the parallax shift measurement and, consequently, the resulting range is in error.

To get an idea of the magnitude of this error, a simple reflectance model is assumed. Suppose that the surface reflectance can be considered uniform everywhere except at the location x_r, where a step in reflectance from R_0 to $R_0 + \Delta R$ occurs as shown in Figure 5(b). To understand the effect of this variation in reflectance, let us examine a simple case in which the parallax angle is zero and the sensor is observing the surface at unit magnification. In this case, the coordinate systems of the sensor image plane and the object surface can be considered the same. The new location of the beam center is given by

$$x_0' \approx \frac{\Delta R \int_{x_r}^{\infty} x e^{-\left[\frac{(x-x_0)}{\sigma}\right]^2}}{R_0 \int_{-\infty}^{\infty} e^{-\left[\frac{(x-x_0)}{\sigma}\right]^2}} + x_0 . \tag{6}$$

It has been assumed that ΔR is small compared to R_0. Carrying out the integrations, it follows that

$$\frac{x_0' - x_0}{\sigma} \approx \frac{\Delta R}{2R_0\sqrt{\pi}} e^{-\left[\frac{x_r - x_0}{\sigma}\right]^2} . \tag{7}$$

Notice that the error in beam position is related to the location of the reflectance step within the beam cross section. If the step is far away from the beam center, then the error in beam location is small. The maximum error occurs when the reflectance step is right at nominal beam center. From this analysis we conclude that any nonuniformity in the surface reflectance will lead to a beam location error.

As another example, consider the case where the surface reflectance varies linearly over the region of interest. The resulting beam center error is given by

$$\frac{x_0' - x_0}{\sigma} \approx \frac{\sigma}{2R_0}\frac{dR}{dx}. \tag{8}$$

Here the error does not depend on the beam position, but only on the rate of change of surface reflectance and the beam spread, σ. In both of the examples, the width of the beam is the major factor that determines the magnitude of range error. Hence, the range errors can be reduced by using the smallest practical beam diameter.

To estimate the effect of these considerations on range accuracy, we will apply them to a typical case. Assume that the incident ray is an unfocused laser beam with a diameter of 1 mm and that the reflectance step is 10% of the incident energy. With the parallax angle at 30 degrees, the maximum range error is .05 mm.

The error can be much worse if the surface is metallic and has glint facets. High variations of reflected intensity due to glint are equivalent to radical variations in surface reflectance. It is possible to experience an intensity dynamic range of as much as 10^4 when imaging metallic surfaces with highly reflective facets. In this case, the measured beam center can be pulled far from the actual center if the glint source is at the periphery of the beam. As before, this problem can be reduced if the beam diameter is kept small.

3.2. Alternatives for Reducing the Effect of Reflectance

3.2.1. Normalizing the Reflectance

As mentioned above, the reduction in beam diameter is one important factor to consider for reducing reflectance effects. Another reasonable approach is to measure the reflectance and normalize the image intensity signal before calculating the parallax shift. This can be accomplished, in principle, by sampling the beam intensity over its cross section and computing the corresponding reflectance. However, in order to compute the

surface reflectance function using the beam, the beam center, which can be shifted from the expected position by parallax, must be known. In other words, a single beam does not provide enough information to determine both range and reflectance. The additional information necessary can be measured by introducing an independent beam and appropriate image analysis. If both beams are illuminating the same surface point, the reflectance effects can be separated from parallax shifts.

The next sections will discuss several methods of obtaining the equivalent of two independent surface measurements in order to reduce the effects of surface reflectance.

3.2.2. Time Multiplexed Beams

If two beams are time multiplexed, two sequential image frames are acquired. As a simple example of how normalization can be achieved, suppose that one beam is shifted slightly with respect to the other, as shown in Figure 6(a). If the two images are subtracted, the resulting difference is as shown in Figure 6(b). Note that the difference crosses zero at the point where the nominal beam intensities are equal.

The position of this zero crossing is not dependent on the surface reflectance, since at the crossing both beams are identically affected and the difference remains zero. There is error introduced in actually determining the zero crossing since the difference signal about the crossing must be examined in order to locate its position precisely. The beams do not experience identical reflectance variations on both sides of the zero crossing, so that methods for determining the location based on sampling or interpolation will be in error. This effect is discussed further in a later section.

An additional source of error is any relative motion of the sensor and the unknown surface. If a motion that alters the appearance of the surface to the two beams occurs during the time that the two measurements are made, then the correction for reflectance is invalid. Unfortunately, in the case of

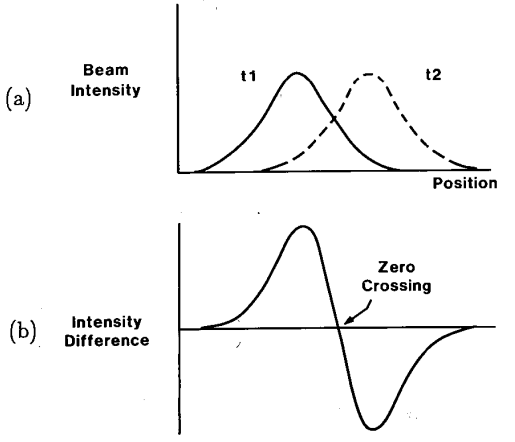


Figure 6: (a) The use of two time multiplexed beams; (b) The intensity difference between the two beams.

metal surfaces, only a small motion (or indeed, even a minor beam misalignment) can cause a significant error due to the faceted nature of metal surfaces.

3.2.3. Wavelength Multiplexing

An alternative to time multiplexing two beams is to use multiple wavelengths. The beams are projected simultaneously, and each beam has a discernible color spectrum. The sensor separates the image into appropriate color spectra using wavelength filters. The resulting images can be subtracted to form a difference image as described in the time multiplexing approach. The advantage of wavelength multiplexing is that both images can be collected at the same time using two aligned sensors with color filters. This avoids errors due to relative motion of the surface during the time the measurements are made.

There is some risk in the assumption in that the reflectance of the surface is the same at the two beam wavelengths. A difference can cause an

erroneous shift in the location of the zero crossing and thus the range value. This effect does not present a problem for many industrial applications that involve metallic surfaces. Most metals have a reasonably flat spectral response over the range of wavelengths near the visible band. On the other hand, organic materials and specific color agents can cause a reflectance variation that produces a significant range error.

3.2.4. Other Multiplexing Possibilities

There are other physical phenomena which can be exploited to provide independent channels for reflectance normalization. For example, polarized light can be used to create two coincident beams. The image of the beams can be separated by polarizing filters. This method is not likely to be as robust as the wavelength technique since many materials cause a shift in the polarization. This shift causes crosstalk between the two channels and prevents proper normalization of the reflectance.

At present, it appears that time and wavelength multiplexing are the most promising methods. The next few sections discuss approaches to the normalization of reflectance variations using information from multiple beams.

3.2.5. Normalization Methods

Having established two separate channels of information using the multiple beam approach, the next issue is how to use the two intensity signals to normalize the effects of surface reflectance variations. There are many viable solutions, but the choice of an optimal one is still an open question. To determine the functional combination of the channels that leads to maximum accuracy in parallax shift detection, we must apply optimum signal theory analysis to a carefully honed model of the surface reflectance properties and the sensor system.

Two methods are presented that serve as a first approximation to satisfactory reflectance normalization. They use a simple model for the

spatial variation of surface properties to remove the influence of reflectivity on the normalized signal.

3.2.6. Difference Ratio

The simplest normalization is given by the ratio of the difference of the intensity of the two channels to the sum of the intensities. If the two intensities are given by I_1 and I_2, the normalized signal is

$$N(x) = \frac{I_1 - I_2}{I_1 + I_2}. \tag{9}$$

The normalized result, $N(x)$, is independent of surface reflectivity under the following model. Suppose that the image intensity is given by

$$I_1(x) = R(x)\, P_1(x)\,, \tag{10}$$

$$I_2(x) = R(x)\, P_2(x)\,. \tag{11}$$

Here $P_i(x)$ represents the incident beam power spatial distribution for beam i. The contributions of sensor geometry and sensitivity, as well as the effective numerical aperture of the camera, are included in the reflectance, $R(x)$.

It is easy to show that the expression for $N(x)$ leads to complete suppression of the spatial variation of $R(x)$. Using the expressions for $I_1(x)$ and $I_2(x)$, $N(x)$ is given by

$$N(x) = \frac{P_1(x) - P_2(x)}{P_1(x) + P_2(x)}. \tag{12}$$

Notice that $R(x)$ appears in the numerator and denominator of $N(x)$ and is thus canceled. The normalized signal depends only on the incident beam power distribution and the sensor properties.

Consider the simple case of two Gaussian beams which are shifted slightly with respect to each other. The power distributions for the two beams are

$$P_1(x) = e^{-\left[\frac{x-x_0}{\sigma}\right]^2}, \tag{13}$$

and

$$P_2(x) = e^{-\left[\frac{x-x_0-d}{\sigma}\right]^2}, \tag{14}$$

where d is the shift in beam centers. Using a Taylor series expansion about x_0, $P_1(x)$ and $P_2(x)$ become

$$P_1(x) = 1 - \left[\frac{x-x_0}{\sigma}\right]^2 + \frac{1}{2}\left[\frac{x-x_0}{\sigma}\right]^4 - \ldots, \tag{15}$$

$$P_2(x) = 1 + \left[\frac{2d}{\sigma}\right]\left[\frac{x-x_0}{\sigma}\right] + \frac{1}{2}\left[\left[\frac{2d}{\sigma}\right]^2 - 2\right]\left[\frac{x-x_0}{\sigma}\right]^2 + \ldots. \tag{16}$$

If we represent

$$P_2(x) = P_1(x) + D\,, \text{ where } D \approx -\left[\frac{2d(x-x_0)}{\sigma^2}\right], \tag{17}$$

then $N(x)$ is given approximately by

$$N(x) = \frac{P_1-P_2}{P_1+P_2} = \frac{-D}{2P_1+D} \tag{18}$$

or,

$$N(x) \sim \left[\frac{d}{\sigma}\right]\frac{(x-x_0)}{\sigma} + \left[\frac{d}{\sigma}\right]^2\left[\frac{(x-x_0)}{\sigma}\right]^2 + \ldots \tag{19}$$

assuming that the shift between the two beams is small compared to the beam spread, σ. Note that $N(x)$ vanishes at x_0 and is approximately linear at the zero crossing. The slope at x_0 depends on the ratio $\frac{d}{\sigma}$, which implies

that the accuracy of the zero location is improved either by a small beam radius σ, or by increasing the shift d. These conclusions only hold for small shifts since the Taylor series expansion is only accurate near x_0.

The assumption that the reflectance $R(x)$ is the same for both beams is quite reasonable for the time multiplex method. If wavelength multiplexing is used to produce separate beams, it is possible that a difference in reflectance for the two colors may result. This color variation introduces a reflectance dependency in the difference ratio normalization just discussed. The effect can be seen by assuming a different reflectance for each beam $R_1(x)$ and $R_2(x)$. Substituting into the normalization expression, we obtain

$$N(x) = \frac{R_1P_1 - R_2P_2}{R_1P_1 + R_2P_2}. \tag{20}$$

There is no obvious cancellation of the reflectance. In fact, the situation can become unsatisfactory if the reflectance in one wavelength band is much larger than the other. Suppose that $R_1 \gg R_2$. $N(x)$ becomes constant and no signal feature is computable to determine parallax shift. This problem can be reduced by using differential normalization.

3.2.7. Differential Normalization

Another form of normalization is given by the ratio of the spatial derivative of the intensity to the intensity itself. Consider this normalization function for a single illumination image intensity channel, $I(x)$.

$$N(x) = \frac{1}{I(x)} \frac{dI(x)}{dx} \quad . \tag{21}$$

Given that $I(x) = R(x)P(x)$, it follows that

$$N(x) = \frac{1}{P(x)} \frac{dP(x)}{dx} + \frac{1}{R(x)} \frac{dR(x)}{dx} \quad . \tag{22}$$

The first term is related only to the illumination pattern, while the second term gives the effect of reflectance.

The same normalization can be applied to the second illumination channel. By forming the difference of these normalized channel signals, the effect of reflectance variation may be further reduced. That is,

$$N(x) = N_1(x) - N_2(x)\,, \tag{23}$$

where $N_1(x)$ and $N_2(x)$ are the normalized signals for each beam. Expanding $N_1(x)$ and $N_2(x)$ we have

$$N(x) = \frac{1}{P_1(x)}\frac{dP_1(x)}{dx} - \frac{1}{P_2(x)}\frac{dP_2(x)}{dx} + \frac{1}{R_1(x)}\frac{dR_1(x)}{dx} - \frac{1}{R_2(x)}\frac{dR_2(x)}{dx}\,. \tag{24}$$

The effect of reflectance completely vanishes if $R_1(x) = R_2(x)$, as was the case for the difference ratio normalization form. The differential normalization goes further in that scaled reflectance forms are removed. For example, suppose that

$$R_1(x) = R_{10}\, f(x)\,, \tag{25}$$

$$R_2(x) = R_{20}\, f(x)\,. \tag{26}$$

We see that the reflectance terms in $N(x)$ cancel, leaving only a dependence on the illumination pattern.

There are reflectance variation effects that are not removed by differential normalization. For example, let

$$R_1(x) = R_{10} + r_1(x)\,, \tag{27}$$

$$R_2(x) = R_{20} + r_2(x)\,. \tag{28}$$

Then the effect of reflectance variation is not canceled by differential normalization unless

$$\frac{1}{R_{10}}\frac{dr_1(x)}{dx} = \frac{1}{R_{20}}\frac{dr_2(x)}{dx}\,. \tag{29}$$

It is unreasonable to expect that surface reflectance at different wavelengths will obey this relation for all materials. However, if the spatial variation of reflectance is small compared to the structured illumination pattern, the

effects of reflectance can be ignored. The reflectance spatial slope, $\frac{dR}{dx}$, is rarely significant except from metallic surfaces with glint. However, metallic surfaces are reasonably neutral in color, so the reflectance terms cancel.

There is one second order effect associated with differential normalization that produces an error in range measurement. In the analysis above, it was assumed that the rate of change of the beam position from the reference position is small. This assumption is not valid at points where the range is changing rapidly, such as at surface height discontinuities. The effect can be demonstrated by considering the location, x, of the beam image to be a function of surface height. In other words, the modulation function, $f(x)$, can be expressed as $f(x_n+g(z))$, where $g(z)$ is an approximately linear function of z, and x_n is the nominal image coordinate (where nominal means the location of the beam image when the surface is at the reference position).

The normalized form for $f(x)$ expands to two terms,

$$\frac{1}{f(x)}\frac{df(x)}{dx_n} = \frac{1}{f(x)}\frac{df(x)}{dx}\left[1+\frac{dg(z)}{dz}\frac{dz}{dx_n}\right]. \tag{30}$$

If the surface slope $\frac{dz}{dx_n}$ is small, then the expression reduces to the original form. The second term can be large in the vicinity of step profile changes. In practice, the surface slope measurement is restricted by the limited resolution of the optical and sensor elements. This effect can be seen in Figure 20, where the profile step measurements exhibit a small overshoot at the step location.

3.2.8. Other Forms of Normalization

One can approach the problem of normalization from the standpoint of optimal signal theory. Most classical results have been developed for additive noise, while the effects of reflectance appear in product form. This can be handled by considering the logarithm of intensity rather than

intensity itself. That is,

$$log\ I(x) = log\ R(x) + log\ P(x) \tag{31}$$

From this perspective, one can consider $log\ R(x)$ to be an additive noise term which interferes with an estimate of the signal, $log\ P(x)$.

This leads to consideration of classical noise elimination methods such as Weiner filtering. Such a filter is of the form

$$H(\omega) = \frac{P(\omega)}{P(\omega)+R(\omega)}. \tag{32}$$

The effectiveness of the filter depends on the degree to which the spatial frequency distribution of the reflectance variation is disjoint from that of the structured illumination.

At first, it would seem that this would be the likely case. For example, consider a periodic illumination pattern (stripes) which has a single distinct spatial frequency. The spatial frequency distribution corresponding to reflectance variations can be considered uniform, particularly for metallic surfaces with a rough finish. Thus, a notch filter can be constructed that only passes energy at the spatial frequency of the illumination pattern. This would eliminate most of the energy due to reflectance variations.

The fallacy in this line of reasoning is that the spatial frequency of the illumination pattern does not have a well defined distribution even if the projected pattern is a periodic set of stripes. The image of this pattern presents a stripe period which depends on the local surface slope. The spatial frequency can be higher or lower than the nominal stripe spacing depending on whether the surface is tilted toward or away from the sensor. For this reason, it is possible to encounter a band of spatial frequencies centered around the stripe frequency.

This observation would suggest that it is not feasible to approach the reflectivity normalization problem from an optimal filtering point of view without a model for the surfaces that will be examined. At the very least, it

seems that a bandpass filter would be necessary and that the spatial response of the range sensor would be limited by the bandwidth of the filter.

An adaptive approach would be to form an estimate of the surface slope at each point of observation. It is not likely that this information will be available in advance. It may be possible to provide an iterative solution where a rough estimate is made of the surface range, and is used to compute surface slope. The slope is used to define a filter which provides a more refined estimate of surface range. This procedure can be repeated until the desired accuracy is achieved.

4. A Profile Sensor Based on Wavelength Multiplexing

4.1. Sensor Requirements

The basic principles for range sensing, described above, were exploited to implement a practical profile sensor to inspect metallic surfaces for small defects such as scratches and dents. In most cases, defect limits are related to the detailed profile of the defect. For example, maximum depth, radius of curvature, and lateral dimensions of a dent determine if it is unacceptable. All of these parameters can be computed from part surface profile measurements. The sensor must therefore perform a complete scan of the part surface. From this scan, an accurate measurement of the surface contour can be extracted.

As we have mentioned above, metallic surfaces can produce undesirable image conditions such as glint. These large excursions in surface reflectivity lead to a large dynamic range in image intensity. It has been shown [4] that the power reflected in a solid angle from metal varies exponentially as a function of incident and observation angles. This can lead to variations of several orders of magnitude in scattered light power over a small distance in the viewing plane. It follows that a sensor suitable for metal surface inspection must be tolerant to large and rapid variations in image intensity.

The scale of surface defects in the metal castings to be inspected is on the

order of .001". That is, the depth variations due to defects must be determined with this accuracy. The lateral resolution required is ideally on the order of .002" to allow adequate presentation of small scratches and pits. This determines the pixel size and consequently the processing rate. A typical casting has ten square inches of surface area. If the pixel area is 4×10^{-6} square inches, 4×10^{7} pixels are needed to cover the surface. In order to meet the inspection throughput requirements for a production environment, each casting must be processed in about 30 seconds. It follows that a rate of about 10^{6} pixels/sec must be achieved. The sensor to be described here will not achieve all of these goals, but they serve as an important guide in the selection of design approaches and implementation strategy.

4.2. Theory of Sensor Operation

In order to achieve the aggressive throughput requirement, it is necessary to project a parallel pattern of parallax shift features. It is desirable to have the pattern as dense as possible so that many range readings can be obtained in one image capture. The density of the pattern is limited by the problem of shift ambiguity. If the surface range varies by an amount which causes a shift in the pattern by more than one stripe spacing, then the range becomes ambiguous.

The desire for high throughput rules out the use of time multiplexed patterns since the range pixel rate is ultimately limited by the sensor frame rate. These considerations dictate the choice of wavelength multiplexing since the channels can be detected in parallel with separate sensor arrays for each color. In addition, it is difficult to provide more than two color channels due to problems in wavelength filtering and sensor alignment. Thus the stripe spacing must be large enough to avoid range ambiguity over the range depth of field. In the application described, the range interval is .05". Larger excursions in depth are accommodated by a manipulation

system which is guided by a nominal description of the part surface.

4.3. Sensor Normalization

The differential normalization method is used to minimize the effects of surface reflectance. In this design there are two wavelength channels, $P_1(x)$ and $P_2(x)$. The two patterns are projected in a complementary fashion which gives the best normalized signal amplitude. Assume that the patterns are complementary, that is

$$P_1(x) = s(x)\,, \tag{33}$$

$$P_2(x) = G\,(K - s(x)) \tag{34}$$

where G represents the different channel gain for $P_2(x)$, and K represents the overall illumination intensity. The normalized signal becomes,

$$N(x) = \left[\frac{1}{s(x)}\frac{ds(x)}{dx} + \frac{1}{R_1}\frac{dR_1}{dx}\right] - \left[\frac{-G}{G(K-S(x))}\frac{ds(x)}{dx} + \frac{1}{R_2}\frac{dR_2}{dx}\right]. \tag{35}$$

Assume for the moment that the reflectance terms cancel or are small compared to the illumination pattern term. Then, $N(x)$ is given approximately by

$$N(x) = \left[\frac{1}{s(x)} + \frac{1}{K - s(x)}\right]\frac{ds(x)}{dx}\,. \tag{36}$$

Also assume that the patterns are sinusoidal, which is the case in a practical optical projection system with limited spatial resolution. A suitable functional form for $s(x)$ is

$$P_1(x) = s(x) = 1 + m\cos x \tag{37}$$

where m is a modulation factor (it is assumed that the modulation factor is the same for both channels in this analysis).

$$P_2(x) = K - s(x) = (K-1) - m\cos x\,. \tag{38}$$

Substituting this into Equation (36) for $N(x)$ and choosing $K=2+\delta$, we obtain

$$N(x) = - \left[\frac{(2+\delta)m \sin x}{(1+m \cos x)(1+\delta-m \cos x)} \right] , \tag{39}$$

which simplifies to

$$N(x) = - \left[\frac{(2+\delta)m \sin x}{1-m^2 \cos^2 x + \delta(1+m \cos x)} \right] . \tag{40}$$

For m reasonably less than 1 and δ small, this expression reduces to $-2\, m \sin x$. For $m=1$ and δ small,

$$N(x) = - \frac{2 \sin x}{1 - \cos^2 x} = -2 \sec x . \tag{41}$$

The first case represents a low level of modulation of the illumination. The resulting normalized signal is a sinusoid having easily detected features from which to determine the lateral position of the stripe image. The extrema and zero crossing locations give four distinct features for each projected stripe.

The second case represents maximum modulation of the illumination where the dark interval goes completely to zero intensity. The resulting normalized signal provides two zero crossings for each stripe with no other distinct features. Thus it is desirable to use a value of m small enough to produce distinct extrema at $x=(2n+1)\pi/2$. A suitable value for m is 1/2 where reliable detection of the features for each pattern stripe is achieved.

Note that both of these cases have the same phase, $\sin x$, as long as δ is approximately zero. The phase of the peaks shifts slightly as δ becomes significant. For example, with $m=0.5$ and $\delta=0.1$, the phase of the peaks shifts about 1-1/3 percent. As we will see later, this is insignificant since only variations in phase with light level and time actually matter, and δ can be made relatively stable for both light level and time. Figure 7 shows the

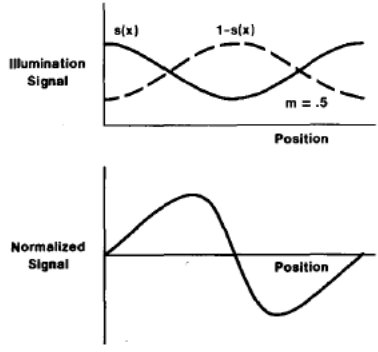


Figure 7: The illumination pattern intensity and the resulting normalized signal for a modulation factor of m = 0.5.

form of the illumination patterns and the normalized signal.

4.4. Sensor Optical Design

The ray diagram of the optical system is shown in Figure 8. The illumination projection system is conventional with the exception of the use of two wavelengths. The incandescent source is divided into two paths by the dichroic beam splitter, 5. The splitter has a wavelength cutoff at 800 nm. This divides the illumination into a visible component and a near infrared component. This choice leads to an equal division of effective power in the two wavelength bands. That is, when the wavelength distribution of the tungsten source and the detector response are taken into account, the detected signal from a neutral surface is equal for both channels. The two light beams are recombined by a patterned mirror, 13, providing a self-registered combination of the two wavelengths. The details of the mirror

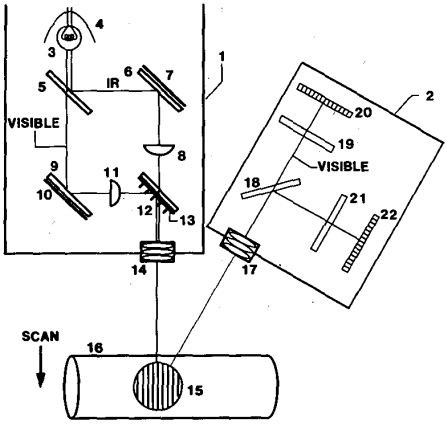


Figure 8: The optical system for the projection and detection of wavelength multiplexed patterns. The projector is 22 and the receiver 23.

configuration are shown in Figure 9. In the current implementation, the pattern is projected with a period that allows an unambiguous range depth of .05".

The receiver, 2, is composed of two linear photodiode arrays, 20 and 22. Each array has 512 detector elements spaced on .001" centers. The dichroic beam splitter, 18, splits the light into two images, one in the visible, the other in the near infrared. The receiver images at a magnification of .5X, resulting in an image of .002" for each detector element.

The overall sensor is shown in Figure 10. The internal views of the receiver and transmitter are shown in Figure 11. The optical design allows a nominal standoff distance of 3.5" from the sensor to the surface reference plane. The scan line covers a field of 1" with pixels on .002" centers. This scan is converted by special purpose hardware into an array of 128 distance

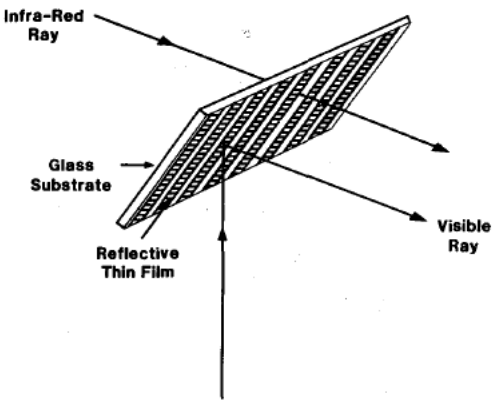


Figure 9: A detailed view of the patterned mirror principle. The two wavelength channels are automatically registered by the interleaved transmission and reflection of the deposited metal film pattern.

measurements on .008" centers. Since a one dimensional array is used, the sensor must be scanned mechanically along the axis orthogonal to the array. In this application, scanning is accomplished using a six-axis servo-controlled manipulator.

4.5. Normalization Signal Processing

The normalization ideas discussed earlier have been applied in the design of the sensor signal processing system. The functions to be described in this section have been implemented in special purpose hardware. The hardware implementation is necessary to achieve the design target of 60,000 range readings per second.

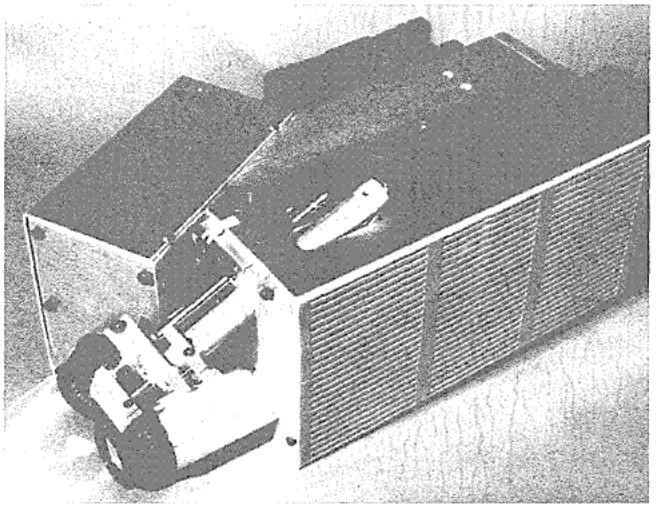


Figure 10: An overall view of the sensor. Note the parallax angle between the transmitter and receiver. There is an extendible mirror at the side for viewing at right angles to the normal optical axis of the sensor. This facilitates the observation of concave surfaces.

4.6. Sensor Calibration

The first step is to remove inherent distortions and artifacts in the intensity signals derived from an A/D conversion of the solid state diode array signals. The array signals are corrected for gain and offset differences between diodes as well as a periodic noise term related to the geometric layout of the detector array known as pattern noise. The pattern noise term is difficult to remove since its amplitude depends non-linearly on the illumination intensity. In the current implementation, the solid state arrays exhibit a noise pattern that repeats with a period of four diodes.

A block diagram of the sensor calibration function is also shown in Figure 12. The gain and offset corrections for each diode are saved in a local RAM. The pattern noise correction is also stored in a RAM table and the correction value is accessed using an address formed by combining the least

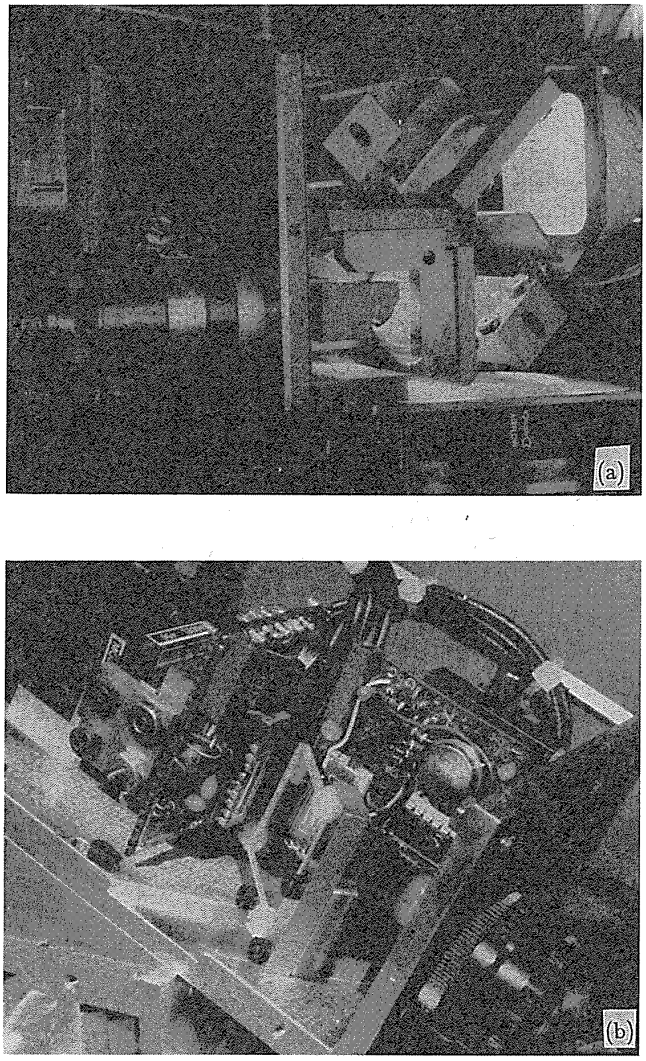


Figure 11: Internal views. (a) The transmitter; (b) The receiver. The projection lens for focusing the incandescent source is an aspheric design. The objective lenses are standard components. The detector arrays are 512 element photodiode arrays. The wavelengths are split by a dichroic filter at 800 nm.

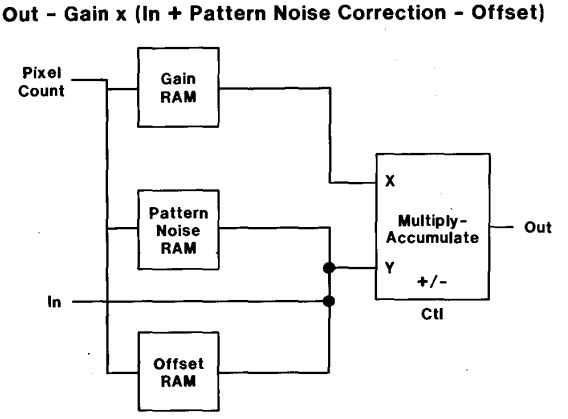


Figure 12: A block diagram of the sensor calibration and normalization process.

significant two bits of the diode location and the upper 7 bits of the signal amplitude. This allows correction to depend on the four diode period and the signal level, as required. The offset and pattern noise corrections are subtracted from each diode. The resulting signal is multiplied by the gain correction factor. The incoming signal is digitized to 10 bits and the corrected signal is carried to 12 bits which is the precision maintained throughout the remainder of the system.

4.7. Differential Normalization

As discussed earlier, reflectance normalization requires a computation of the form

$$\frac{1}{f(x)}\frac{df(x)}{dx}. \tag{42}$$

The first issue is the computation of the spatial derivative of the function $f(x)$. The approach here is to use one dimensional convolution with a mask selected to minimize the error in the estimate of the derivative. There has

been extensive effort applied to the problem of optimal estimators for linear operations on signals [1], [2]. The first derivative of the Gaussian with suitable spatial scale is often suggested. In the implementation described here, the derivative mask coefficients are completely programmable so that any derivative function may be used. In the experiments to be described later, the first derivative was calculated using a least mean square estimator based on a second order polynomial model for the local signal behavior.

The hardware implementation allows five signal values to be used in the calculation of the derivative. The coefficients for the derivative estimate are

$$\frac{\hat{d}}{dx} = \frac{1}{10}[-2 \quad -1 \quad 0 \quad 1 \quad 2]. \tag{43}$$

These coefficients are suitably scaled to allow fixed point computation with 12 bit accuracy. The hardware carries out the 5 point convolution in one microsecond.

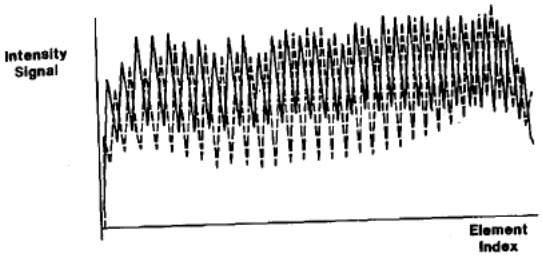
4.8. Signal Estimation

The signal itself is also processed with a linear estimator to obtain the denominator of the normalization form. The second order estimation for the value of the function, $f(x)$, is represented by the five point mask

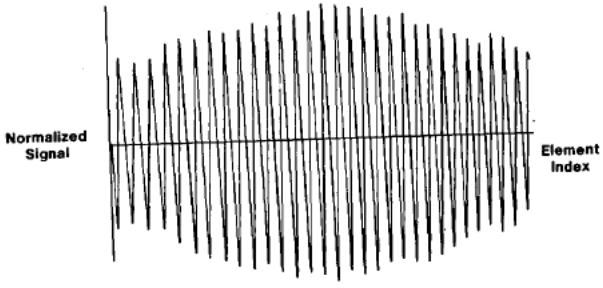
$$\hat{f} = \frac{1}{35}[-3 \quad 12 \quad 17 \quad 12 \quad -3]. \tag{44}$$

4.9. Experimental Normalization Results

The normalization process requires four convolution steps to be performed in parallel. The division of $\frac{df(x)}{dx}$ by $f(x)$ is actually done by retrieving the reciprocal of $f(x)$ from a ROM table and then multiplying. The result from each wavelength channel is combined by subtraction to produce the desired normalized signal. The actual signals obtained by viewing a flat neutral surface are shown in Figure 13. Figure 13(a) shows the two wavelength signals superimposed. The visible channel is solid, the infrared channel is shown dotted. The variation in the image signal



(a)



(b)

Figure 13: Experimental normalization results. (a) The two wavelength intensity channels. The infrared channel is shown dashed. (b) The normalized signal, $N(x)$. The variation in amplitude is due to lens distortions.

amplitude is due to spatial non-uniformity of the tungsten source and aberrations in the projection and receiver optics.

The resulting normalized signal is shown in Figure 13(b). The differential normalization is sensitive to the period of the illumination pattern since the period affects the magnitude of $\frac{df(x)}{dx}$. The lens used to project the illumination pattern introduces a small variation in the pattern period due to pincushion distortion. This appears in the normalized signal as a gradual variation in amplitude over the field of view with a peak in the center.

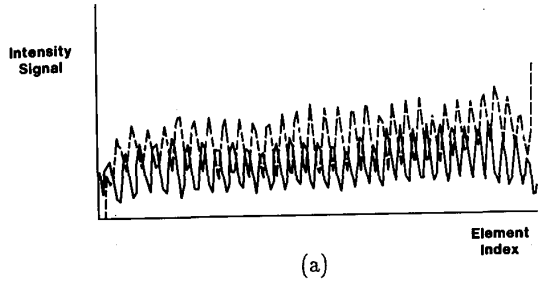
To illustrate the dynamic range of the normalization process, the same situation as in Figure 13 was repeated with the receiver optical aperture stopped down so that the signal level was reduced by a factor of 12:1. The illumination signals and the resulting normalized signal are given in Figure 14. Notice that there is only a minor variation in the normalized result for a decrease in image intensity of over an order of magnitude. (Note that the sensor intensity magnitude scales are not the same between Figures 13 and 14. The normalized signal scales are the same.)

4.10. Feature Detection and Selection

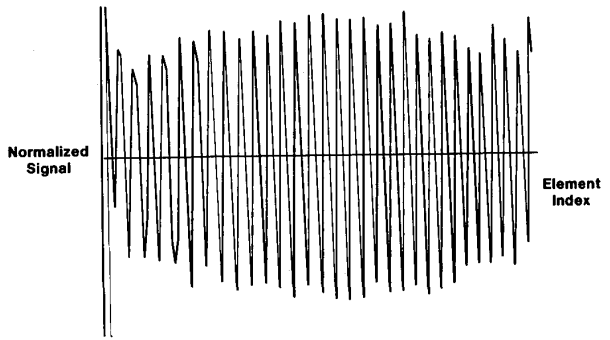
The next major step in the extraction of surface profile is the detection of the significant features in the normalized signal. The location of these features indicates the shift due to parallax, and thus is used to calculate surface depth.

In principle, each point in the normalized signal could be used to define a unique pattern location. If the normalization procedure were ideal, the value of $N(x)$ would map uniquely onto the corresponding location in the projected pattern. Thus the process of profile computation could be carried out for each pixel in the intensity sensor array.

In practice, this ideal condition is not realized due to many second order effects which lead to variations in the amplitude of the normalized signal



(a)



(b)

Figure 14: The same display as in Figure 13 except the receiver aperture is stopped down to reduce the signal level by 12:1.

that are not due to profile variations alone. The most stable features are the zero crossings and extrema of the approximately sinusoidal form of $N(x)$. The current design projects 32 stripes over the field of view so that each period of $N(x)$ contains 16 sensor array sample points. There are two extrema and two zero crossings in each period which are nominally separated by four sample intervals. This nominal spacing implies that 128 features are present in the field of view; thus 128 profile measurements can be obtained for each scan of the sensor array.

4.11. Peak Detection

The accurate location of peaks and valleys in the normalized signal is a problem which is encountered in many signal analysis applications. The approach here is to specify a number of properties that the signal extremum must satisfy and then implement a linear estimator for each property. This is done in such a way that the estimator vanishes if the property is satisfied. The final decision on the presence of the peak or valley is based on the sum of the absolute value of the estimators.

Three properties of a peak which should be satisfied at the maximum point of a signal, $f(x)$, are:

$$\frac{df}{dx}(x_m) = 0 \tag{p1}$$

$$\frac{df}{dx}(x_m+d) + \frac{df}{dx}(x_m-d) = 0 \tag{p2}$$

$$f(x_m+d) - f(x_m-d) = 0 \tag{p3}$$

These properties all reflect the basic idea that a signal is approximately symmetrical and constant in the vicinity of an extremum or inflection point. The conditions in (p2) and (p3) insure that the inflection point case is eliminated by requiring even symmetry in the distribution of the signal about the extremum location. The quantity, d, is a suitable distance on each

side of the extremum. In the current design, $d = 2$ pixels.

4.12. Implementation of Symmetry Estimators

The estimators in (p1), (p2), and (p3) can all be calculated by convolutions, with the subtraction indicated in (p3) absorbed into the convolution coefficients. Using the same least mean square estimation methods as in the normalization, the masks are:

$$\text{p1} = [\,0 \;\; -3 \;\; -2 \;\; -1 \;\; 0 \;\; 1 \;\; 2 \;\; 3 \;\; 0\,]$$

$$\text{p2} = [\,-2 \;\; -1 \;\; 0 \;\; 1 \;\; 0 \;\; -1 \;\; 0 \;\; 1 \;\; 2\,]$$

$$\text{p3} = [\,-17 \;\; 5 \;\; 17 \;\; 19 \;\; 0 \;\; -19 \;\; -17 \;\; -5 \;\; 17\,]$$

These linear operators are also implemented with programmable coefficients so that alternative estimators can be tried. The convolution computation provides for up to nine coefficients. The computations are carried out in one microsecond using hardware multiplier circuits.

The scaled, absolute values of all three convolutions are added together to form the extremum signal. A peak or valley is indicated when this signal goes through a minimum. Ideally, the signal becomes identically zero at the extrema locations. The actual normalized signal is not perfectly symmetrical about the peak or valley so the properties in (p1) through (p3) do not all vanish exactly. However, the minimum location in the sum of the absolute values is a solid and reliable indication of the peak or valley location. In practice, the method works well, even for signals with barely discernable features.

4.13. Zero Crossing Detection

The detection of zero crossings is much more straightforward than for peaks and valleys. The approach here is simply to form an estimate of the normalized signal $N(x)$ and then take the absolute value of the estimate. It is seen that the resulting signal will go through a minimum at the location of

the zero crossing. This method has the added advantage that the resulting signal has the same form as that obtained for the extremum detection. Thus, the treatment of the signals later in the feature identification process can be implemented with identical circuits. The same linear estimator for signal value is used as in the normalization case. That is,

$$\hat{f}(x) = \frac{1}{35}[-3 \quad 12 \quad 17 \quad 12 \quad -3] \ . \tag{45}$$

4.14. Experimental Results

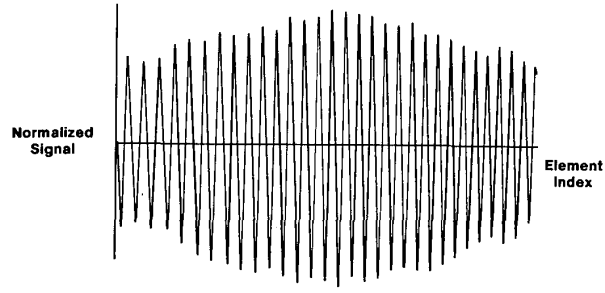
The results of the feature location operations are given in Figure 15; the conditions correspond to the setup for the normalized signal plot in Figure 13. Note that the detector signals for the extremum are interleaved with those of the zero crossing. The feature signals reach a minimum four times for each period of the normalized signal.

The effect of low image intensity is illustrated in Figure 16. Here the input light level has been reduced by stopping down the receiver objective. The normalized signal appears approximately the same as in Figure 15, but there are subtle differences which are clearly reflected in the feature signals. The positions of the minima are still quite stable and distinct. Note that the signal for the extrema feature is at a maximum when the zero crossing signal is at a minimum and vice versa. This property is exploited in the detection of the minimum in each signal.

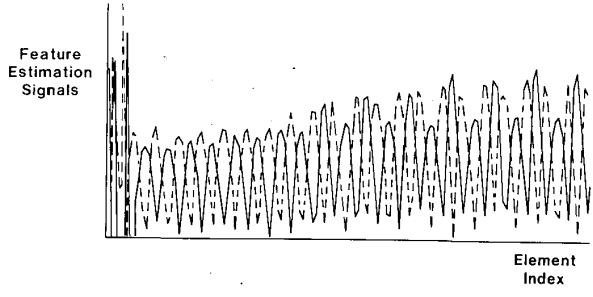
4.15. Detection and Interpolation of Feature Position

The location of peaks, valleys and zero crossings in the normalized signal has been transformed into the problem of detecting and precisely locating minima in the feature signals. A crude location of the minima is provided by starting a sequential search along one signal when its value falls just below the other. This idea is clarified by Figure 17, which shows an expanded section of the feature signal plot.

To be specific, suppose that the minimum of the zero crossing signal is



(a)



(b)

Figure 15: The feature location signals computed for the conditions of Figure 13. (a) The normalized signal of Figure 13(b); (b) The extremum and zero crossing signals. The extremum signal is solid and the zero crossing signal is shown dotted. The excursions at the left are due to an unilluminated portion of the detector array.

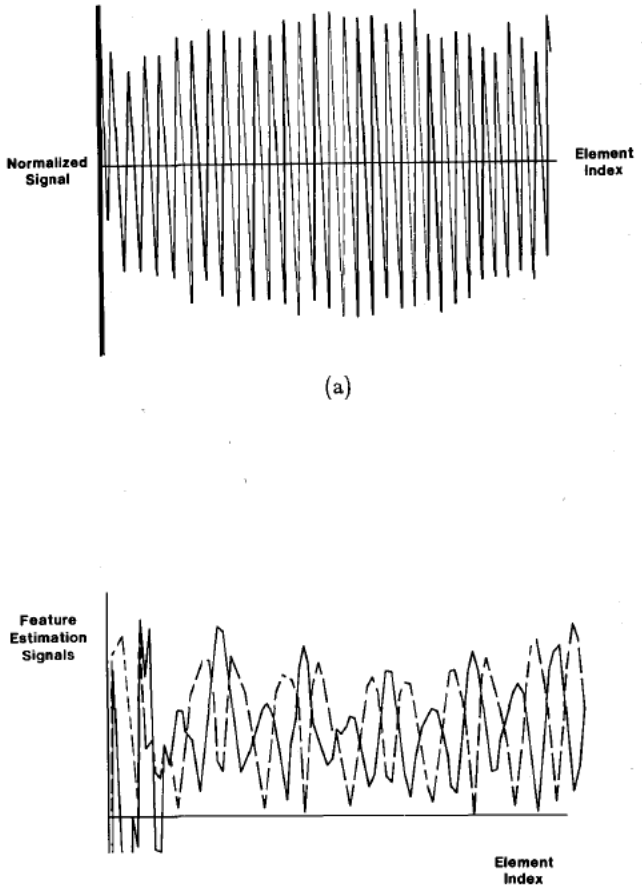


Figure 16: The same display as Figure 15 except the receiver aperture is stopped down as in Figure 14.

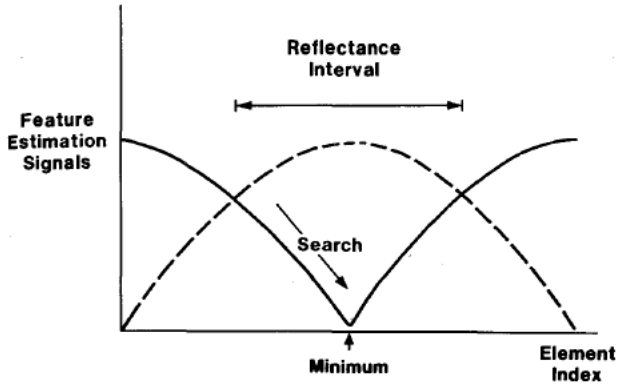


Figure 17: The procedure for locating the feature minimum. The search proceeds to the right from point A until the signal starts upward. This is taken as the starting point for the minimum interpolation.

desired. The zero crossing signal is scanned from left to right until it falls below the amplitude of the extremum signal. At this point a search is initiated for a minimum in the zero crossing signal. The minimum is assumed to be where the signal first reverses direction after falling below the extremum signal. The location of the minimum is thus known at the precision of the sampling interval of the feature signals, which in the current design is 512 samples per line.

The use of the complementary signal as a threshold to define the interval of search for the minimum in a signal is very robust since their amplitudes track together as the normalized signal degrades. There is, of course, the possibility that a signal could exhibit several minima over the interval where it is smaller than the complementary signal. This condition implies quite a high frequency phenomenon which should only occur in an extreme degradation of the normalized signal. Since there are only four sample intervals between each feature, a double minimum would require variation

which is dangerously close to the Niquist sampling condition and should not be considered valid.

The location of the minimum can be refined considerably by interpolating around the minimum using a variant of Newton's method. The equation for interpolation is obtained by a Taylor series expansion of the feature signal in the vicinity of the minimum. Representing the feature signal by f(x), the approximation is

$$f(x) \approx f(x_{min}) + \frac{df}{dx}(x_{min})(x - x_{min}) + \frac{1}{2}\frac{d^2f}{dx^2}(x_{min})(x - x_{min})^2 . \quad (46)$$

Now the minimum of $f(x)$ can be determined by setting

$$\frac{df}{dx} = 0 \quad (47)$$

or, in terms of the approximation,

$$\frac{df}{dx} \approx \frac{df}{dx}(x_{min}) + \frac{d^2f}{dx^2}(x_{min})(x - x_{min}) = 0 . \quad (48)$$

In this case x_{min} is the approximate minimum location determined by the search described above. This linear equation can be easily solved for x, the interpolated value of the minimum,

$$x = x_{min} - \frac{\frac{df}{dx}(x_{min})}{\frac{d^2f}{dx^2}(x_{min})} . \quad (49)$$

With this interpolation it is possible to refine the location of the minimum to subpixel accuracy. Under ideal conditions with uniform reflectance surfaces and sufficient illumination, the standard deviation of the feature location is on the order of $\frac{1}{16}$ of a sample interval. Under poor illumination conditions and observing a metallic surface, the standard deviation can increase to $\frac{1}{2}$ a pixel. This is an extreme case with glint and low surface reflectivity so that the image intensity variations exceed the dynamic range of the sensor array.

4.16. Mapping of Feature Location Into Profile

The precise location of illumination pattern features is now complete. The next step is to map the shifted location of the features into the surface profile which is related to the shift of the features from its nominal position. Since the mapping function is non-linear and learned through calibration, it would be difficult to compute directly, and so it is implemented using table lookup methods.

The features are classified into four groups corresponding to the unique positions on the sinusoidal normalized signal. The classes in order of occurrence in the signal are:

- Z+ = Positive going zero crossing
- P+ = Positive peak
- Z− = Negative going zero crossing
- P− = Negative peak

The classes repeat in this sequence over the image. There are 32 groups of each of these four features or a total of 128 individual features. The translation of feature location into range is performed using the feature map which contains four regions corresponding to the four feature types. Each region is 512 elements long and each address corresponds to the location of a feature on the input image. Thus, if a P+ feature was found at pixel 47 of the input image, then the corresponding surface position is stored in address 47 of the feature map. Both X and Z coordinates are stored; the Z value is used to compute the actual range data and the X position is used as the X position of that range value. The Z value stored is the range displacement of the calibration surface which would result in the identified feature appearing in the measured position of the input image. This value is obtained by actually measuring the position of each of the features on a calibration surface. This method has the advantage of accounting for the first order anomalies of the optics, with one calibration point for each feature. The X position is a pointer to an output slot corresponding to the X position of the center of the feature in the input image. When a particular feature is found

in the input image, its computed range or Z value is placed in the corresponding output or X position slot in the output range image.

The actual computation of the Z range is accomplished in three steps. First, an approximate (un-interpolated) Z value of the feature is retrieved from the address of the feature map corresponding to the input pixel location of the identified feature. In the example above, the value would be retrieved from address 47 of the feature map. The sign of the interpolator defined in Equation (49) is used to choose the adjacent pixel location in either the plus or minus direction. In our example, if the interpolator were negative, the value at pixel 46 would be retrieved. Finally, the interpolator magnitude is used to linearly interpolate between the two retrieved range values. The result is computed in fixed point arithmetic to twelve bits of resolution.

In addition to the determination of surface position (X and Z), the feature map also contains the address of the range of input pixels over which the reflectance of the corresponding range pixel should be computed. This information is passed to an averaging circuit which will be described next.

4.17. Acquisition of Surface Reflectance

The formation of surface profile readings is carried with a spatial resolution of four intensity samples (pixels) on the average. Provision is also made to acquire a measure of surface reflectance for each profile sample. In this design the reflectance is taken to be proportional to the sum of the intensities of each wavelength channel.

The intensity sum is averaged over the interval corresponding to each profile feature. This interval is illustrated in Figure 17, which shows a series of feature detection signals. The computation for R_i, which corresponds to profile sample z_i, is

$$R_i = \frac{2}{d_1+d_2} \sum_{j=i-\frac{d_1}{2}}^{j=i+\frac{d_2}{2}} (s_{1j} + s_{2j}) \ , \tag{50}$$

where s_{1j} and s_{2j} are the intensity signals in channels 1 and 2, respectively, d_1 is the number of pixels to the left of the current feature to the previous feature, and d_2 is the number of pixels to the right of the current feature to the next feature. The index j corresponds to the sensor sample index, while the index i is the profile feature location. There is not a fixed relationship between i and j because the profile feature location depends on the surface height, z.

The sum of the intensity channels does not exactly measure the surface reflectance since other factors can influence the intensity in each channel. For example, if the surface is not neutral, one of the wavelength channels will be smaller than the other and the sinusoidal pattern of illumination will not cancel. To see this effect, consider the following forms for $s_1(x)$ and $s_2(x)$:

$$s_1 = 1 + m \cos x \,, \tag{51}$$

$$s_2 = 1 + \delta K - (m + \delta m) \cos x \,. \tag{52}$$

The quantity δK represents the difference in optical transmission and δm represents the difference in modulation factor between channel 2 and channel 1. The sum of these two signals is

$$R(x) = 2 + (\delta K - \delta m \cos x) \,. \tag{53}$$

The reflectance measure should be constant (2), but instead is modulated with a sinusoidal component, $\delta m \cos x$ which is proportional to the imbalance in the channel modulation characteristics, and a constant offset, δK, which is proportional to the imbalance in the channel transmission. These two factors correspond to a first order approximation of the dynamic and static lumped response characteristics of the optics, the part surface, and the electronics.

In practice, for the application of the sensor to metallic surfaces, there is not a significant difference in the reflectance for the channels. The

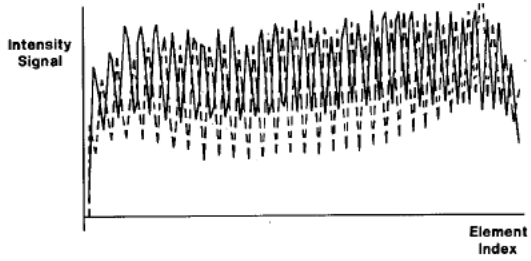
difference that does exist is reasonably uniform over the surface and can be removed by calibrating the gain constants for the channels to give equal intensity signals.

4.18. Experimental Results for Line Scans

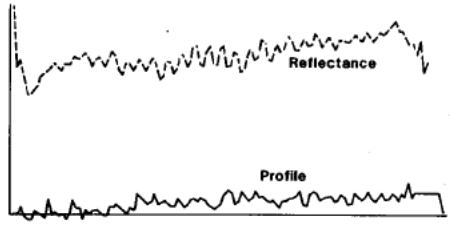
Continuing the example of Figures 13 and 15, the results of profile and reflectance are now presented in Figure 18. The uppermost plot shows the original sensor intensity data for each channel. The profile readings should be nominally zero since these data were taken on the reference surface. Note that there is a small sinusoidal component in the reflectance signal. The standard deviation in profile is approximately .00015".

The corresponding result with the receiver objective stopped down is shown in Figure 19. Note that the reflectance signal is about one order of magnitude less than in Figure 18. The standard deviation in profile has increased to about .0005". The systematic slope to the profile data is caused by the change in optical characteristics of the receiver when the objective f number is increased. The rays are confined more to the center of the lens. As a consequence, the effective focal length of the lens is changed slightly, which produces a small change in image magnification. This change in feature scale is interpreted as a slope in surface height. Under normal operation, the lens settings are not changed and the optical characteristics of both the transmitter and receiver are incorporated into the profile mapping function.

Another example of sensor performance is shown in Figure 20. Here the sensor is measuring an adjustable step on a metal surface. The step is increased in steps of .002" for each scan. The overshoot in profile is due to the effect of surface slope on the differential normalization process. The nominal scatter in the profile readings is quite small in regions away from the step. In this example the deviations are on the order of .0005".

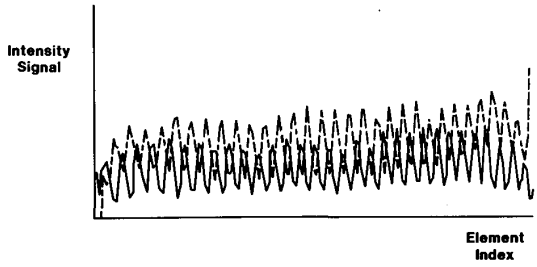


(a)

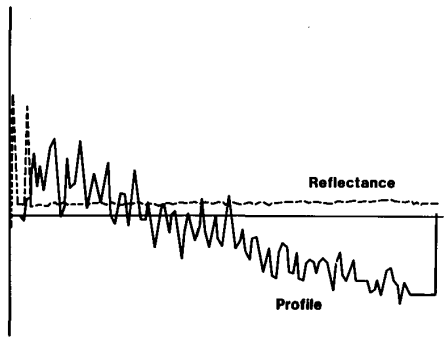


(b)

Figure 18: (a) The sensor intensity data of Figure 13; (b) The reflectance and profile results for the conditions in Figure 13. The dotted curve shows reflectance on a relative scale. The profile data is shown as a solid curve. Full scale is about .008".



(a)



(b)

Figure 19: The same conditions as Figure 18 except the receiver aperture is stopped down to reduce the signal level. Note that the reflectance signal is about 1/12 of that in Figure 18.

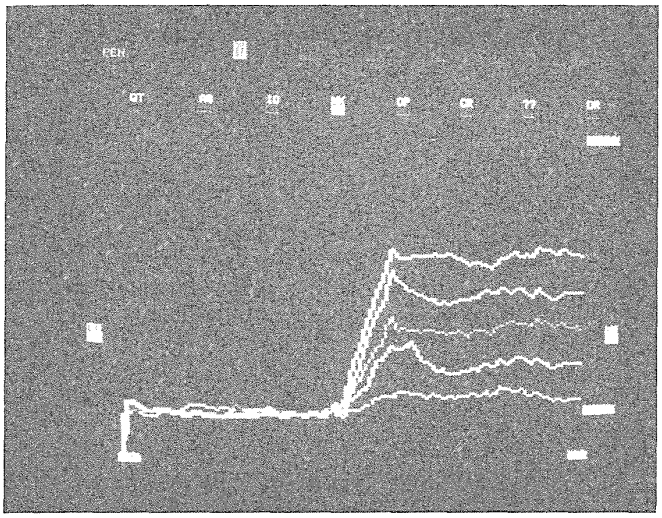


Figure 20: The measurement of profile steps on a metal surface. Each trace corresponds to a .002" increase in surface height.

4.19. Two Dimensional Profile Scanning

The sensor as described provides a single line of profile data. The data provide a profile sample every four sensor sample points, which corresponds to .008" using the average magnification of .5X. This magnification produces a sensor pixel image corresponding to .002" x .002" on the sample surface. The profile readings are computed using a normalized signal which is first averaged over two sensor lines for a physical domain of .008" x .004". This is again averaged for two lines which yields a square profile pixel of .008" x .008".

The input sensor rate is designed to be as high as 10^6 sensor pixels per second. After the indicated sampling and averaging, the resulting profile generation rate is 62,500 profile readings per second. This rate corresponds to an area coverage of four square inches per second, which is quite acceptable for inspection purposes.

In practice, this rate was not fully achieved. The sensor A/D conversion

module limits the input sample rate to 800,000 samples per second. In addition, the low reflectance of many casting surfaces requires that the sensor integrate the incident intensity for longer than the time corresponding to the maximum line scan rate. The extra integration is necessary to achieve enough signal to noise ratio in the sensor data to allow computation of the normalized signal. For dark, oxidized metal surfaces, the maximum scan is limited by this effect to about one inch per second. Profiles can be measured at scan rates up to about three inches per second for moderately reflective surfaces.

Another limitation is imposed by the multi-axis manipulator, which provides the other direction of scan along the part surface. In the current system the six manipulator axes are divided between the sensor and the part, as shown in Figure 21. For best results, the sensor should have its optical

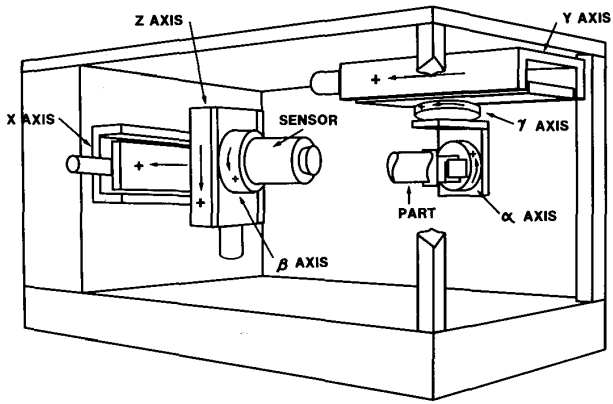


Figure 21: The configuration of manipulator axes for part surface scanning. The inertia is reduced by splitting the axes between the part and sensor manipulation.

axis aligned with the normal to the object surface. If the the surface is highly curved, this condition requires large axial accelerations. The massive construction of the manipulator, required to maintain global positional

accuracy, limits the maximum axial accelerations that can be achieved, so that for curved surfaces the scan rate is also limited to one inch per second.

4.20. Two Dimensional Surface Measurements

A typical surface scan is shown in Figure 22(a). This scan is taken of the surface of an airfoil casting as shown in Figure 22(b). The region corresponding to the profile surface is indicated by the arrow. A small step (.002") has been introduced in the data by moving the manipulator during the scan sweep. This step provides a calibration for the scale of the profile data. Another example is given in Figure 23, which shows the profile displayed as an intensity plot. The figure shows a small flaw which is about .025" across and .005" deep.

In Figure 24 a series of scans are taken on a recessed portion of a casting. The casting is shown in Figure 24(a) with the two regions to be scanned illustrated with arrows. A scan of the edge of the pocket is shown in Figure 24(b) as an intensity plot. The corresponding perspective view is in Figure 24(c). The lateral resolution of the sensor is illustrated by a scan of a portion of the raised serial number on the casting. The characters "80" are clearly legible in the profile intensity plot of Figure 24(d). The double images in some of the plots are a result of scanning back and forth over the area and recording profile images for both directions.

The effect of surface glint is illustrated by a scan made of a small metallic washer shown in Figure 25(a). The washer surface has many small glint sources and leads to false excursions in the profile data. The resulting profile is shown in an intensity plot in Figure 25(b) and the profile format in Figure 25(c). The profile noise here has a peak-to-peak variation of about .003".

The sensor has been applied to the inspection of casting surfaces for small pits and scratches. Extensive tests have indicated that defects as small as .020" across by .003" deep can be reliably detected for gradually curved

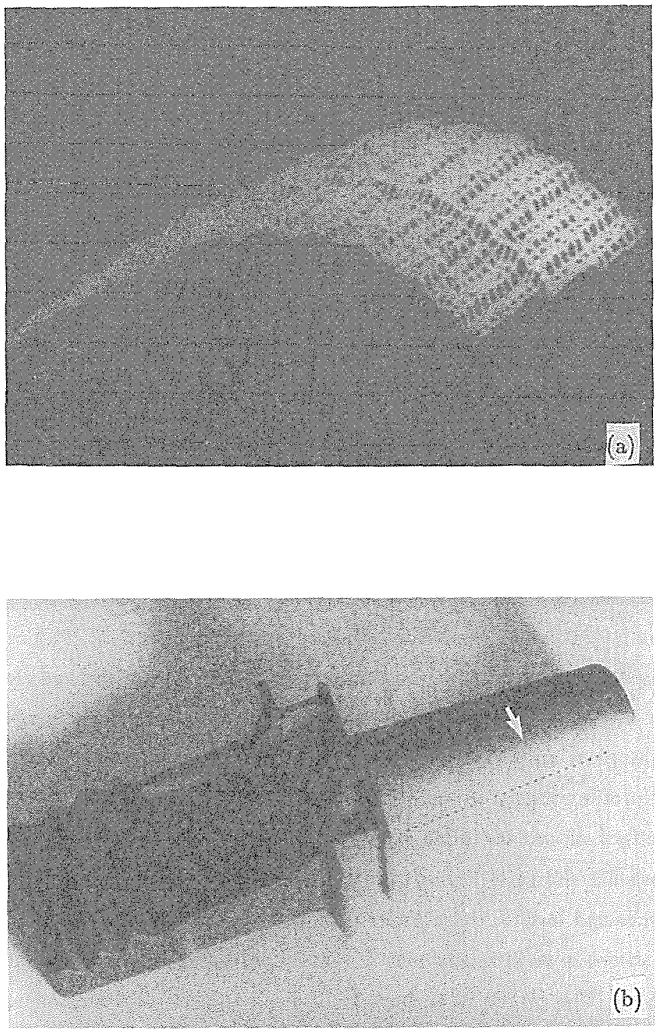


Figure 22: A two-dimensional scan of an airfoil surface. (a) The profile shown in perspective. The sensor was shifted .002" to show the scale. (b) The airfoil surface. The arrow indicates the area of the profile scan.

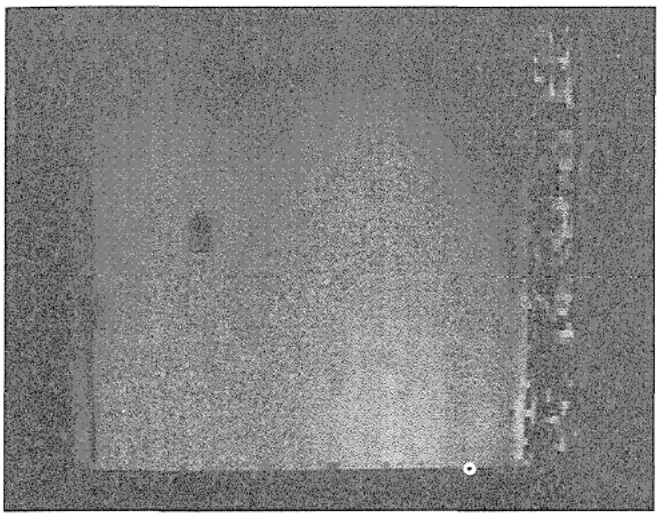


Figure 23: A small flaw displayed as an intensity plot. The flaw is about .025" across and .005" deep.

surfaces. The case of highly curved surfaces is still a problem because of the difficult mechanical problems in maintaining the sensor axis normal to the surface.

5. Future Directions

The effects of reflectance are diminished the most by projecting highly localized patterns. In the simple case of a single beam of illumination, this corresponds to reducing the beam spread as much as possible. It is therefore relevant to consider what optical phenomena ultimately limit the size of the beam.

5.1. Limits on Beam Spread

The effects of diffraction impose a final limit on the size to which a beam can be focussed. The main parameters are the wavelength of the light, λ, and the numerical aperture of the objective, NA. The definition of numerical aperture is given in Figure 26. Essentially, NA is a measure of the

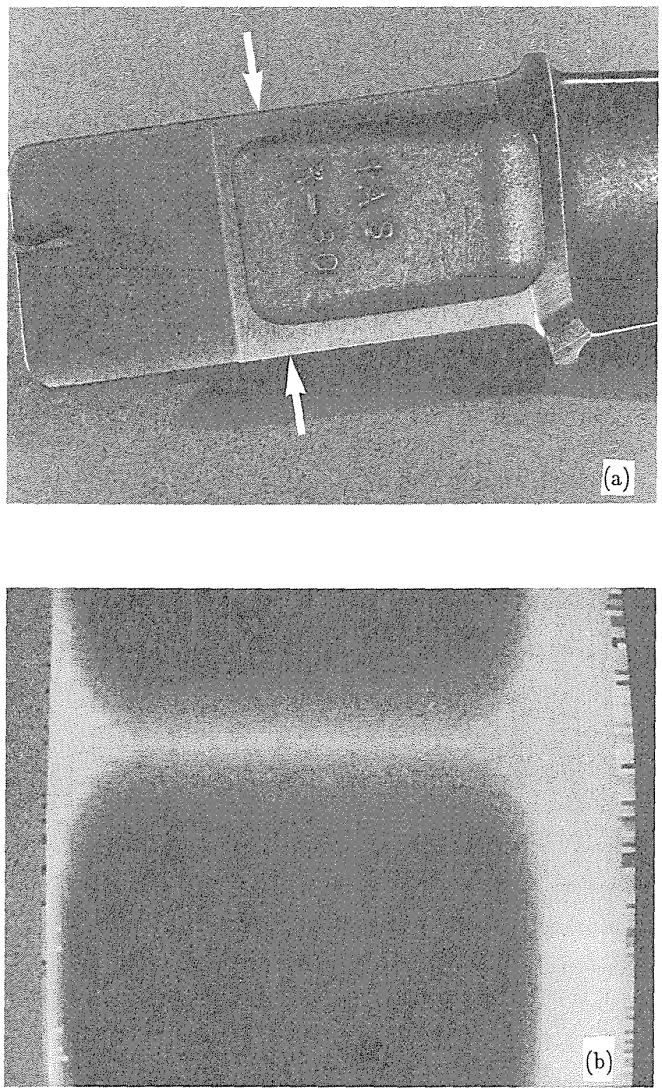


Figure 24: A scan of a recessed casting pocket. (a) A view of the casting with arrows indicating the location of the scans; (b) The profile of the edge of the pocket as an intensity plot. The surface was scanned in two directions which causes the repetition.

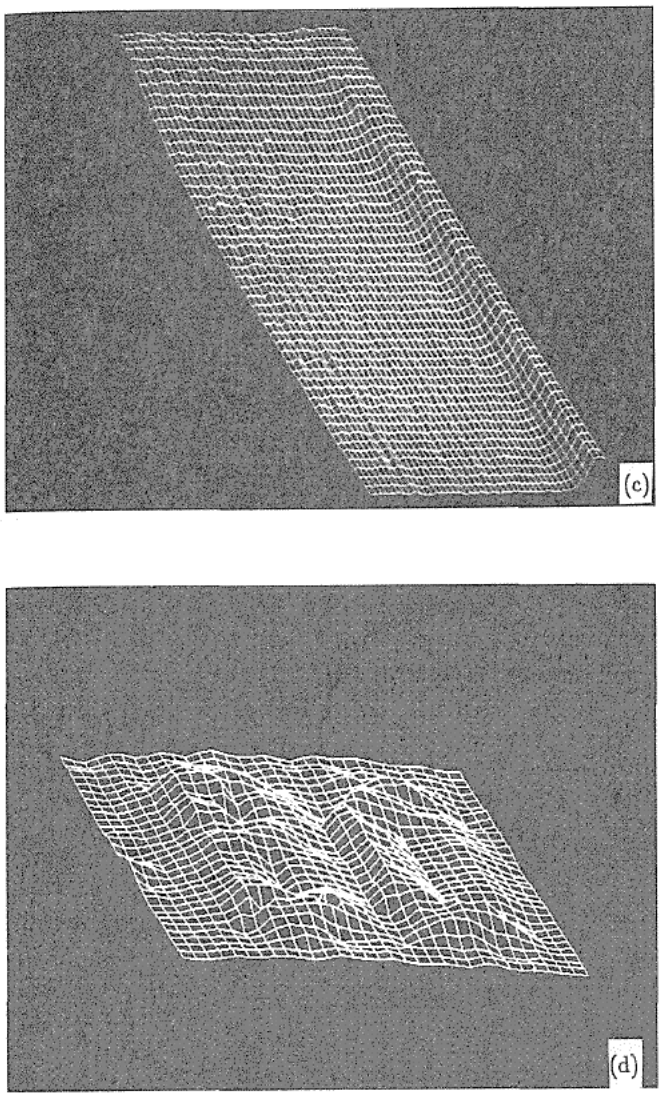


Figure 24: (c) The data in (b) shown as perspective; (d) Profile data in perspective showing the characters "80".

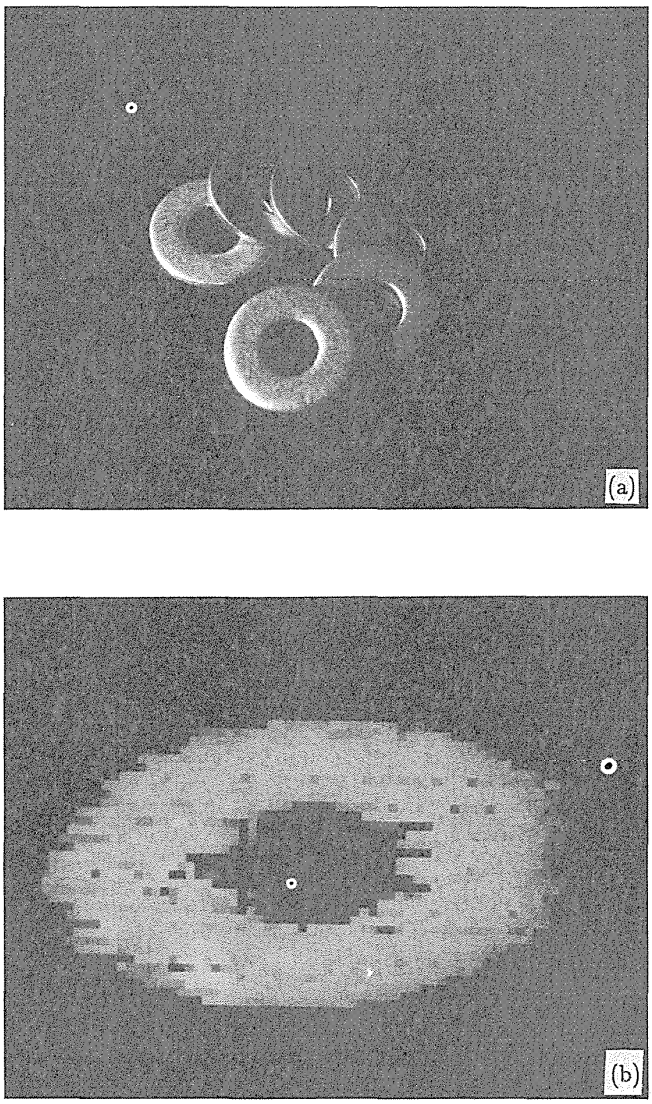


Figure 25: A scan of a metallic washer. (a) The washer; (b) The washer profile as intensity;

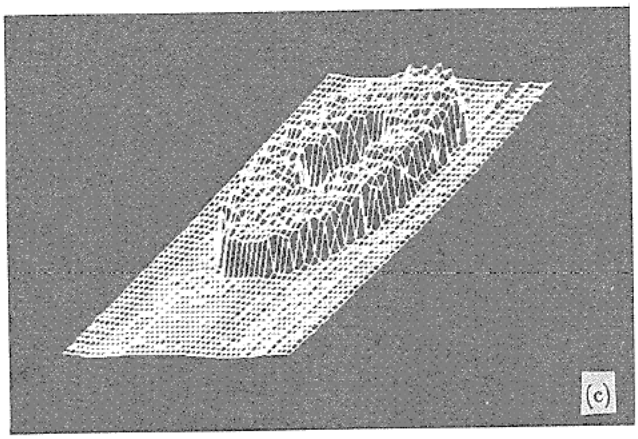


Figure 25: (c) The profile data in perspective.

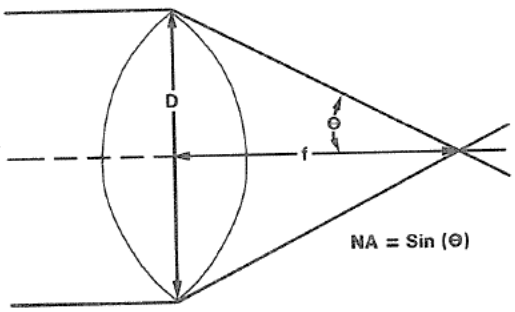


Figure 26: The definition of numerical aperture. f is the focal length of the lens, D is the lens diameter. θ is the angle subtended by the outermost ray intercepting the lens and the optical axis.

convergence angle of the rays towards the beam focus point. The beam spread, σ, at focus is given approximately by

$$\sigma = \frac{\lambda}{NA}. \tag{54}$$

The depth of field, δ, is also related to the numerical aperture of the objective lens. The focus spot diameter increases according a simple geometric interpretation of the divergence of the beam away from focus. That is,

$$\delta = \frac{\lambda}{NA^2}. \tag{55}$$

To get an idea of the magnitude of these quantities, consider a typical case where the objective lens has a .1 NA and the desired beam spread, σ, is .01 mm or .0004". The resulting depth of field is only .2 mm or about .010". This extremely limited range is difficult to maintain for manipulation over curved surfaces, so some form of profile feedback must be used to maintain the focus setting. On the other hand, with a beam spread of only .0004", the maximum profile error that can be caused by glint spikes is limited to about .001" for a parallax angle of 30^o.

5.2. Dynamic Focus

To maintain the beam focus for such a limited depth of field, one approach is to provide two levels of depth resolution. An example of this is shown in Figure 27. Here two wavelengths are used to provide dual channels. One channel is focused with a small NA and correspondingly large depth of field. This provides a crude profile reading to precisely locate the focus for the second beam. The second beam is focused, with a large NA and, consequently, fine profile resolution.

It is difficult to see how an approach along these lines can be extended to a parallel pattern of stripes as in the sensor described earlier. Each point must be individually focused, which implies that sensor readings are taken for each sample beam location and focus position. In the scheme just described, at least two sensor readings must be taken for each profile

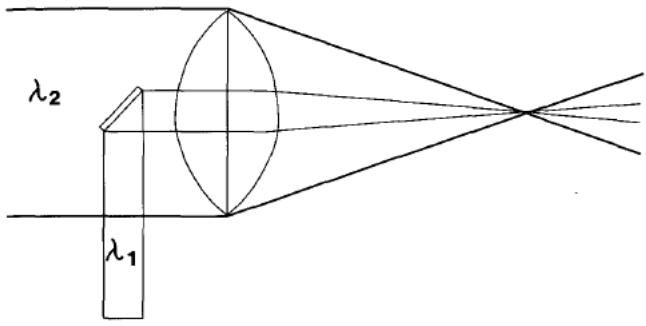


Figure 27: An arrangement for dual resolution beams. One beam is used to provide an approximate focus position for the other.

sample.

5.3. Structured Light and Stereo Matching

A more interesting prospect which avoids the need to maintain a finely focused system is the hybrid use of structured light and stereo feature matching. This idea is illustrated in Figure 28. Here, two receiver sensors are provided to observe the surface at the parallax angle with respect to the direction of the projected pattern.

If the reflectance of the surface is relatively uniform, the accuracy of profile measurements made by the differential normalization method will be good. In the presence of glint or other rapid variations in reflectance, the normalization procedure may degrade and lead to error in profile measurement.

However, this is exactly the condition in which stereo matching is effective. The rapid change in reflectance can be localized accurately in each sensor image using standard intensity edge detection methods [3]. The

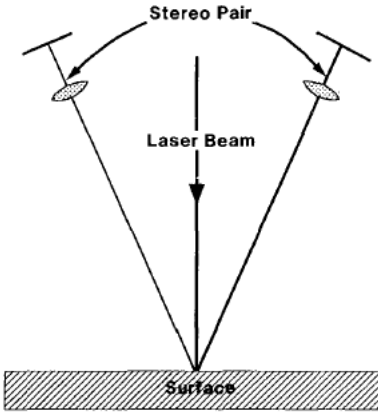


Figure 28: A hybrid arrangement using stereo matching and structured light.

location of these intensity features can be matched between the two views, which gives an accurate estimate of the surface position.

On the other hand, if the surface reflectance is uniform there are no intensity features to match in the stereo pair. The projected illumination pattern provides the needed triangulation features in this case. Thus the two methods nicely complement each other and are capable of high degrees of parallelism.

6. Conclusion

As we have described, the technology for direct profile sensing is complex and rapidly evolving. It is clear that the process of obtaining an accurate profile reading over rough metallic surfaces is computationally intensive. Fortunately, the present cost of computation is plunging rapidly through the application of VLSI technology. It is likely that in the near future many of the sensing and signal processing requirements described here will be available in highly integrated modules with correspondingly low cost.

In addition, other methodologies for profile sensing are becoming practical as a result of the improvement of semiconductor technology. For example, there is an increasing interest in time-of-flight methods. In these approaches, the round trip time or phase of a modulated laser beam is used to measure distance. It is not yet clear whether or not time can be measured with sufficient precision to compete with structured light triangulation methods for surface profile measurement.

References

[1] Beaudet, P.
Optimal linear operators.
In *Proc. 4th International Joint Conference on Pattern Recognition*. 1979.

[2] Canny, J. F.
Finding edges and lines in images.
Technical Report TR-720, MIT, Artificial Intelligence Lab, 1983.

[3] Grimson, W. E.
Computational experiments with a feature based stereo algorithm.
Technical Report 762, MIT, Artificial Intelligence Lab, 1984.

[4] Mundy, J. L.
Visual inspection of metal surfaces.
In *Proc. National Computer Conference*, pages 227. 1979.

3-D Vision System Analysis and Design

Steven K. Case, Jeffrey A. Jalkio, and Richard C. Kim

CyberOptics Corporation
2331 University Avenue S.E.
Minneapolis, MN 55414

and

University of Minnesota
Electrical Engineering Dept.
Minneapolis, MN 55455

Abstract

Structured light vision systems offer a practical solution to many 3-D inspection problems. This paper examines the tradeoffs involved in the design of 3-D inspection systems that use triangulation. General equations describing the behavior of such systems are derived. Problems common among structured light systems are explored and alternative solutions presented. The implementation of a multi-stripe system is discussed.

1. Overview

Vision systems are becoming an integral part of many automated manufacturing processes. Visual sensors for robot arm positioning, parts placement, and work cell security have greatly enhanced the utility of robots, while inspection systems employing similar sensors help ensure high quality products in many areas. However, standard vision systems provide only a two dimensional picture of the work area or finished product; unable to discern the distance to, or height of, the object viewed. Recently, various 3-D vision systems have been produced to alleviate this problem via several methods [10], [14]. The two major distance measurement techniques are triangulation and time of flight [12]. Triangulation systems can be further

subdivided into active [11] (structured light) and passive [13] (stereo disparity) systems. Most commercially available 3-D vision systems employ structured light triangulation because this method currently has the greatest potential for fast, accurate, and inexpensive high resolution systems.

As shown in Figures 1 and 2, structured light triangulation systems determine distance by projecting light from a source onto an object and imaging the resulting illumination pattern onto a detector. Knowing the position of the image on the detector, as well as the lateral separation between the detector lens and the light source, and the projection angle of the source, allows the determination of the distance, z, to the object. Multiple measurements at different (x,y) coordinates on the object lead to a full 3-D image of the object's surface.

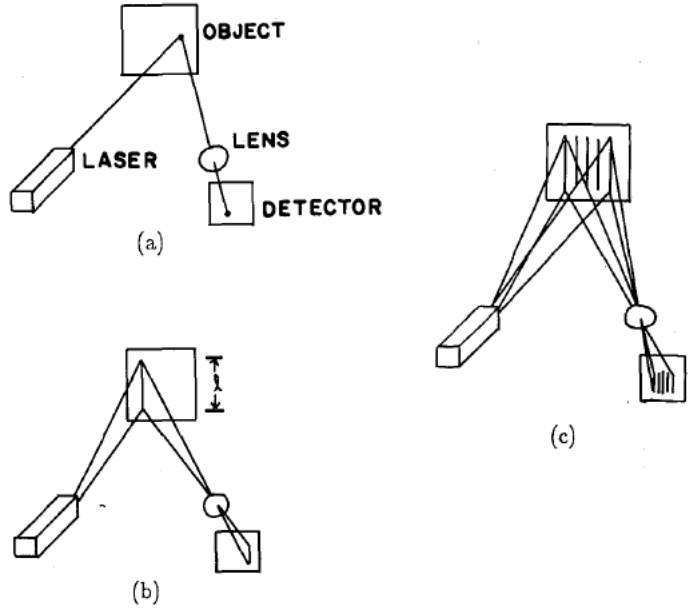


Figure 1: Structured light triangulation. (a) Single point projection; (b) Single line projection; (c) Multiple line projection

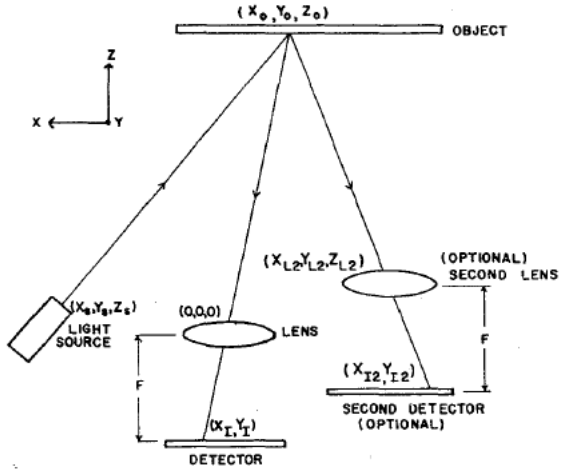


Figure 2: Geometric model for general triangulation system

In this paper we examine the design of structured light triangulation systems, starting with a discussion of illumination patterns, and an analysis of general triangulation geometry. We then consider the design of the illumination and detection subsystems (including light source and detector selection as well as the creation of illumination and receiving optics). Data processing techniques are addressed next, followed by a discussion of optical problems including a brief description of filtering techniques which minimize their effects. Finally, results from our prototype system are presented and our system's performance is discussed in the light of the original specifications.

2. Light Structures

The simplest structured light system projects a single point of light from the source onto an object for triangulation (Figure 1(a)). This method has the potential for delivering a great deal of light to that single point and can provide a high signal to noise ratio even when looking at dark or distant

objects. Only a single point is measured per sampling interval so that $n \times m$ sample times are required to acquire data for an n by m lateral resolution element image. Single point triangulation systems usually use lateral effect photodiodes or linear arrays as detectors. If a full 3-D image is to be obtained, a bi- directional scanning apparatus is required [9].

A second class of 3-D imaging instruments operates by projecting a light stripe onto the object and using a two dimensional detector array (Figure 1(b)) [1]. Fewer frames are needed to scan over an object (n frames for an n by m element picture) and scanning need only be done in the direction perpendicular to the stripe. This results in a system with fewer moving parts. In fact no moving parts are required if the object being inspected moves past the vision system (as on a conveyor belt). The image of the illumination stripe is free, in principle, to move across the entire width of the detector array and therefore the system can have depth resolution at least equal to the number of pixels in a horizontal line of the detector array.

Whether all of this depth resolution will in fact be used depends on both the object and the optics used in front of the detector. For proper optical design, the length, l, of the stripe in Figure 1(b) should be imaged so as to match the vertical dimension of the detector array. With standard spherical imaging lenses and a 45 degree angular separation between source and receiver optics axes, a height change of approximately l must be encountered in order for the stripe image to move all the way across the width of the detector array. Many objects in the world are quasi-flat, (e.g., circuit boards) so that in going from minimum to maximum height we use only a fraction of the detector array unless expensive anamorphic optics are used.

We can make use of this disparity between potential depth variation (determined by the resolution of the detector) and the useful depth variation (determined by object size and opto-mechanical geometry) by projecting multiple stripes onto the object (Figure 1(c)). Since with n stripes projected onto the object, each stripe can traverse only $1/n$ of a detector scan line, the

potential depth resolution has been decreased by a factor of n. However, as long as the useful depth variation for a quasi-flat object is smaller than the potential variation this is not a problem. A given lateral resolution can now be obtained with $1/n$ times the number of frames that a single stripe system would require.

3. Analysis of Triangulation Systems

Before designing a multi-stripe 3-D vision system we must understand the geometric constraints inherent in such a system.

3.1. Object Coordinate Calculation

In Figure 2 we show a general triangulation system, specifying the x,y, and z coordinates of the object, light source, and detector lens aperture centers. The coordinate system is centered on the primary detector lens. The (x,y) coordinates of an image point are given relative to the corresponding imaging lens center. A single illumination beam is represented at angles Θ_x and Θ_y to the z axis in the x and y directions respectively. It illuminates the object at the coordinates

$$x_o = x_s - (z_o - z_s)\tan\Theta_x\,, \tag{1}$$

$$y_o = y_s - (z_o - z_s)\tan\Theta_y\,. \tag{2}$$

An image of this point is formed on the primary detector at the coordinates

$$x_i = \frac{F}{z_o}x_o = \frac{F}{z_o}(x_s + z_s\tan\Theta_x) - F\tan\Theta_x, \tag{3}$$

$$y_i = \frac{F}{z_o}y_o = \frac{F}{z_o}(y_s + z_s\tan\Theta_y) - F\tan\Theta_y, \tag{4}$$

where F is the distance between the receiving lens and detector, which is nearly equal to the focal length of the receiving lens for long working distances (large z_o). If this image point is compared to that produced by an object at a calibration distance z_{ref} we find displacements of the image by

$$\Delta x_i = F(x_s + z_s \tan \Theta_x)\left(\frac{1}{z_o} - \frac{1}{z_{ref}}\right), \tag{5}$$

$$\Delta y_i = F(y_s + z_s \tan \Theta_y)\left(\frac{1}{z_o} - \frac{1}{z_{ref}}\right). \tag{6}$$

Using the x displacement to calculate the object distance we find

$$z_o = \frac{z_{ref}}{1 + \dfrac{\Delta x_i z_{ref}}{F(x_s + z_s \tan \Theta_x)}} \tag{7}$$

which implies that the relation between z_o and Δx_i is space variant (i.e., dependent on Θ_x) unless $z_s = 0$.

To simplify calculations and eliminate the need to buffer an entire frame, we would like image points to move only in the x direction (i.e., along a detector row or raster) for changes in object distance. For this we require $\Delta y_i = 0$ which implies $y_s = z_s = 0$, so that the effective source position must be on the x axis.

If we introduce a second camera, we can measure distance by comparing the relative separation of image points on two detectors instead of comparing the coordinates of a given image point to those corresponding to a previous reference point. As shown in Figure 2 the coordinates of the image on the second detector plane are

$$x_{i2} = \frac{F(x_o - x_{L2})}{(z_o - z_{L2})}, \tag{8}$$

$$y_{i2} = \frac{F(y_o - y_{L2})}{(z_o - z_{L2})}. \tag{9}$$

Using Equations (3) and (8) to eliminate x_o and solving for z_o we find

$$z_o = \frac{F x_{L2} - z_{L2} x_{i2}}{x_i - x_{i2}}, \tag{10}$$

which tells us that unless $z_{L2}=0$ we will have space variant behavior. The vertical displacement is given by:

$$\Delta y = y_i - y_{i2} = \frac{F y_{L2} - z_{L2} y_{i2}}{z_o}, \tag{11}$$

indicating that for the desired condition of $\Delta y=0$ we must have $y_{L2}=z_{L2}=0$, so lens 2 lies on the x axis.

Finally, we must consider the determination of x_o and y_o. Clearly we could use Equations (3) and (4) to find them from x_i and y_i, but it might be simpler to report coordinates in the z_o, Θ_x, Θ_y system; since Θ_x is fixed for a given illumination stripe, it need not be calculated. Furthermore, we note from Equation (2) that if $z_s=y_s=0$, Θ_y is uniquely determined by y_i and hence it need not be calculated either.

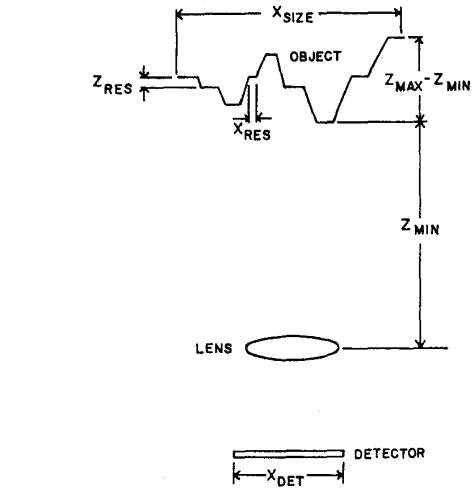
3.2. System Parameter Calculations

To demonstrate the procedure used to calculate the adjustable system parameters, we will assume that we wish to measure an object of overall dimensions x_{size}, y_{size} and varying in distance from z_{max} to z_{min}, as shown in Figure 3(a). We further assume that measurement resolutions of x_{res}, y_{res}, z_{res} are required. If we denote by $\Delta x_{i_{min}}$ the smallest measureable change in image stripe position on the detector (determined by both the pixel size and the stripe detection algorithm used), and by $\Delta x_{i_{max}}$ the change in image stripe position caused by the object distance changing from z_{max} to z_{min}, we find from Equation (5) that

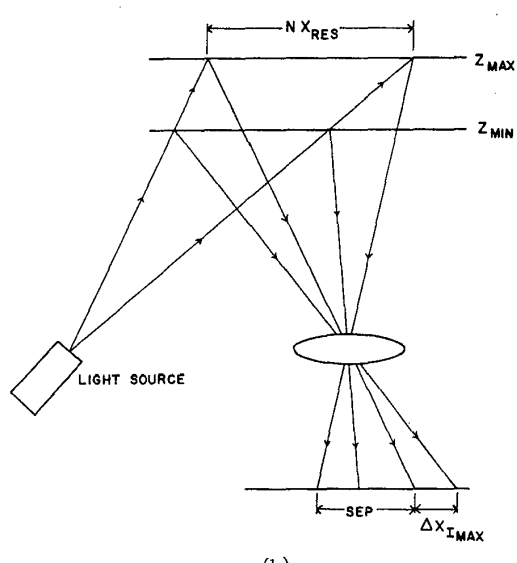
$$\Delta x_{i_{min}} = \frac{F x_s z_{res}}{z_{max}(z_{max} - z_{res})} \tag{12}$$

$$\Delta x_{i_{max}} = \frac{F x_s (z_{max} - z_{min})}{z_{max} z_{min}} \tag{13}$$

Assuming that $\Delta x_{i_{min}}$ is known, $\Delta x_{i_{max}}$ can be found in terms of $\Delta x_{i_{min}}$ by eliminating $F x_s$ from Equations (12) and (13):



(a)



(b)

Figure 3: Geometric model for the prevention of aliasing

$$\Delta x_{i_{max}} = \Delta x_{i_{min}} \frac{(z_{max}-z_{min})}{z_{res}} \frac{(z_{max}-z_{res})}{z_{min}}. \tag{14}$$

In Figure 3(b) we show two adjacent stripes projected from the source and imaged onto the detector. Shown are the image positions corresponding to maximum and minimum object distances. If we project M stripes onto the object and collect N frames of data wherein in each frame each stripe is shifted laterally on the object by an amount x_{res} from its position in the previous frame, we obtain a total lateral resolution of NM pixels, where to match the object resolution requirements

$$NM \geq \frac{x_{size}}{x_{res}}. \tag{15}$$

Adjacent, simultaneously projected stripes are separated by Nx_{res} and if we denote the separation between the images of these stripes on the detector by the distance sep, we have

$$sep = \frac{Nx_{res}F}{z_{max}}. \tag{16}$$

To image the entire object onto a detector of lateral extent x_{det}, y_{det} we impose the constraint

$$x_{det} > x_{size}F/z_{min}. \tag{17}$$

Eliminating F from Equations (16) and (17), we can find the number of frames required to obtain sufficient depth and lateral resolution:

$$N > \left(\frac{z_{max}}{z_{min}} \quad \frac{sep}{x_{det}} \quad \frac{x_{size}}{x_{res}} \right). \tag{18}$$

Once N is chosen, the number of stripes used in a single frame, M, can be found from Equation (15).

We define the ratio of the range of travel of a single stripe, $\Delta x_{i_{max}}$, to the separation between adjacent stripes, sep, and denote it by R. That is,

$$R = \frac{\Delta x_{i_{max}}}{sep} = \frac{x_s(z_{max} - z_{min})}{N x_{res} z_{min}} . \tag{19}$$

Clearly we require $|R| < 1$ to prevent aliasing. The exact value of R chosen for a system must take into account the stripe finding algorithm used, since some algorithms require more dead space between stripes than others. Once a value has been chosen for R, we can solve Equation (19) for the separation between the light source and detector lens pupil:

$$x_s = \frac{R N x_{res} z_{min}}{(z_{max} - z_{min})} . \tag{20}$$

Using this value in Equation (5) determines the focal length of the detector lens

$$F = \frac{z_{max} \Delta x_{i_{max}}}{R N x_{res}} . \tag{21}$$

If the dimensions of the object and detector are such that this focal length does not allow the entire y extent of the object to be imaged onto the detector, anamorphic optics can be used to provide a different focal length, and hence magnification, in the y direction, satisfying the relation

$$y_{det} = F_y y_{size} / z_{min} . \tag{22}$$

For systems employing two detectors, Equation (20) needs special interpretation. In particular, for the second detector x_s must be replaced by $x_s - x_{L2}$ and R_2 may differ from R_1. Combining the two versions of Equation (20), we can solve for the required lens separation x_{L2}:

$$x_{L2} = \frac{x_{res} N z_{min}}{z_{max} - z_{min}} (R_1 - R_2) . \tag{23}$$

It should be noted that for the case of two detectors located symmetrically about the light source, we have $R_2 = -R_1$ and $x_{L2} = 2x_s$.

From the above derivations we can see that if the dimensions and

resolution of the object and detector are known, all system parameters (i.e. number of stripes per frame, number of frames, lens focal length, and source and detector offsets) can be uniquely determined. As an example, our system was designed to view objects for which x_{size}=y_{size}= 60 mm, and $z_{max}-z_{min}$= 12.5 mm with x_{res}= 0.5 mm and z_{res}= 0.25 mm. We chose a working distance z_{max}= 650 mm. The active detector area has x_{det}=y_{det}= 5.5 mm, Δxi_{min}= 0.001 mm (using a super resolution algorithm) and we planned on R = 1/6 (see Section 6.2). Using these values and the above equations we find Δxi_{max}= 0.051 mm, $N > 6.8$ (we chose 8), x_s= 34 mm, and F= 50 mm.

3.3. Light Source Calculations

We have to this point presumed that stripes of light of suitable size could be accurately projected into the object space from the specified source position. Such does not happen by accident. In this section we will describe source design.

In Figure 4, a laser beam with Gaussian intensity profile impinges on a

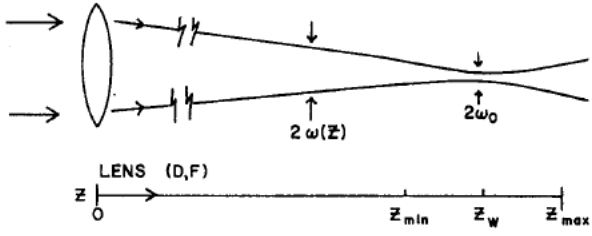


Figure 4: Focal properties for Gaussian beam

lens of diameter D and focal length F. The beam width as a function of position, z, is given by [15]

$$\omega(z) = \omega_o \left[1 + \left[\frac{\lambda(z-z_w)}{\pi\omega_o^2} \right] 2 \right] 1/2 \qquad (24)$$

where $\omega(z)$ is the distance from the center of the beam to the $1/e^2$ intensity point, and ω_o is the minimum value of $\omega(z)$ (which occurs at z_w).

We require that the beam diameter

$$2\omega(z) \leq x_{res} \tag{25}$$

in the interval

$$z_{min} \leq z \leq z_{max} \tag{26}$$

in order to obtain the desired lateral resolution. Selecting

$$\omega(z_{max}) = x_{res}/2 \quad \text{and} \quad z_w = (z_{max} + z_{min})/2 \tag{27}$$

allows us to solve Equation (24) for ω_o, the beam waist size, which in turn allows us to find $\omega(0)$, which is the minimum allowable lens radius. We find that

$$\omega_o = \frac{x_{res}}{2\sqrt{2}} \left[1 \pm \sqrt{1 - \left(\frac{4\lambda(z_{max} - z_{min})}{\pi x_{res}^2}\right)^2} \right]^{1/2} . \tag{28}$$

Hence there are two beams that satisfy our requirements. The beam with the smaller value of ω converges more rapidly than the other, hence it requires a larger lens diameter. For this reason we chose the larger value of ω, in hope of minimizing the size of our optical system. For this beam, the lens diameter must have size

$$D = 2\omega(z{=}0) = 2\omega_o \left[1 + \left(\frac{\lambda z_w}{\pi \omega_o^2}\right)^2 \right]^{1/2} . \tag{29}$$

As an example, with stand off distance $z_w = 650$ mm, depth of focus $z_{max} - z_{min} = 12.5$ mm, lateral resolution $x_{res} = 0.5$ mm, and $\lambda = 0.633 \times 10^{-3}$ mm, inserted into Equation (28) (with the positive radical) we find the beam waist diameter (full stripe width) to be

$$2\omega_o = 0.49990 \text{ mm} \approx 0.9998 x_{res} \tag{30}$$

so that the laser beam is only weakly convergent as it approaches the focus. Inserting Equation (30) into Equation (29) and applying our numerical values gives

$$D = 1.16 \text{ mm} \approx 2.32 x_{res} \quad .$$

4. Source Implementation

We saw in the previous section that it was advantageous for the fan beams producing our multiple illumination stripes to appear to diverge from a point which is only displaced in the x direction from the center of the detector lens pupil. Such an array of beams is easily produced via a multi-faceted hologram [3] which segments a single beam of light into M beams; these cross at a given distance to produce an effective source some distance from the hologram, and come into focus some distance later as shown in Figure 5. The cylinder lens diverges the beams to form stripes, although this function also could have been incorporated into the hologram.

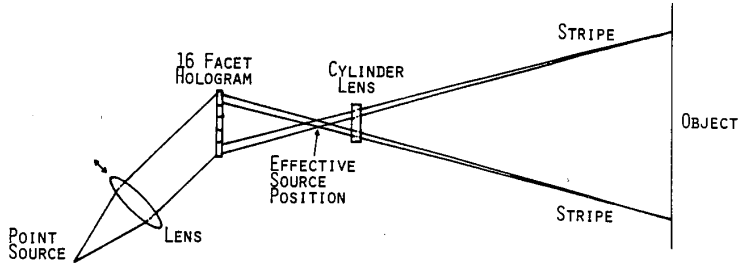


Figure 5: Schematic of illumination subsystem

In order to obtain a horizontal resolution of 128 pixels, our system produces M=16 stripes which are laterally shifted in each of 8 consecutive frames. This motion could be accomplished by moving a lens or mirror within the source optical system or by using 8 adjacent light sources that can be switched on sequentially. LEDs and laser diodes are small and easily switched. LEDs have lower brightness than lasers but eliminate speckle problems (explained later). For our experiments we used a HeNe laser whose

visible beam made optical alignment easier. Since it was not practical to use 8 HeNe lasers, our system moves a lens via a computer controlled translation stage to translate the stripes between frames. Our hologram facets were 1/16" = 1.6 mm in width and the illumination across each facet was of uniform irradiance so that depth of focus, etc. differed slightly from our above design analysis.

5. Detector Implementation

5.1. Detector Selection

Since our multiple stripe technique requires a two dimensional detector, we must choose between a solid state or vidicon type camera. Although solid state cameras are small, rugged, long life devices, our system uses a vidicon detector in order to obtain greater depth resolution. Not only does a vidicon camera offer higher resolution than currently available solid state cameras, but also the dead zones between the pixels of a solid state camera can cause distortion of narrow stripes, whereas data are obtained from a vidicon via smooth convolution with a scanning electron beam, which broadens the stripe but does not distort it.

Because of the scan instability inherent in vidicon devices, we chose to use a dual detector architecture to minimize the effects of jitter on our data. Comparing Equations (7) and (10) we see that in a single detector system, z data are obtained by comparing stripe positions in sequential frames, making the system quite sensitive to jitter (small changes in horizontal synchronization) and thermal drift, whereas with the dual detector architecture, z data are obtained by comparing two stripe images obtained at the same time. If the two detectors are produced via a double optical system in front of a single detector, the resulting system is immune to jitter effects. In fact, if the images from the dual optical system are interleaved so that the two image stripes corresponding to the same illumination stripe are adjacent, the resulting system is quite insensitive to fluctuations in the scan

rate during a single frame. In the next section we will discuss several techniques for producing the virtual camera pair and compare their strengths and weaknesses.

5.2. Dual Detector Implementations

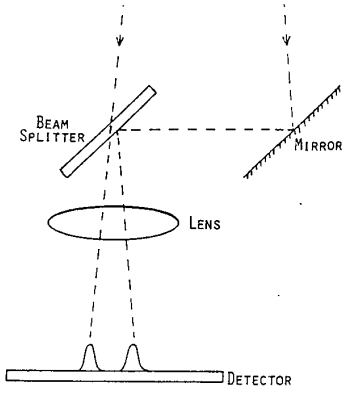


Figure 6: Mirror - Beamsplitter system produces asymmetric detector pair

One method for producing a pair of virtual detectors is to use a mirror and beamsplitter [4] as shown in Figure 6. With this arrangement the relative separation between the two stripe images increases with decreasing object distance in correspondence with Equation (5). This is inexpensive and simple to align, but it has three problems. First, the separation between the two detectors cannot be smaller than the aperture of the camera lens. Since there is an inverse relationship between the range of measurable distances, $z_{max}-z_{min}$, and the detector separation (Equation (20)), this gives us an undesirable trade-off between useful range and aperture size (light level). Second, the path length between the object and the camera is different for the two different paths. This means that the two images can never be simultaneously in focus. The third problem is also due to the path length

difference, namely that the two virtual detectors do not lie in a plane perpendicular to the optical axis, leading to a space variant relationship between distance and image separation as shown in Equation (7). Since we want to reduce the computational complexity of the system this is a very undesirable side effect.

One way to reduce the detector separation, as required for long working ranges (while maintaining depth resolution and lens aperture size), is to create an equivalent arrangement of mirrors by sputtering a pattern of gold on both sides of a glass plate. For glass with an index of refraction of n, the pattern shown in Figure 7 behaves similarly to the mirror beamsplitter arrangement; half of the incident light which passes the first surface is transmitted directly to the camera while the other half is reflected twice to enter the camera at a position laterally displaced by

$$x_{L2} = 2w \sin \psi \,, \tag{31}$$

where l and w are related by

$$l = \frac{w}{3\sqrt{n^2-1}} \,. \tag{32}$$

This technique has different path lengths, but the difference can be so small as to not affect focus appreciably. However, the space variant effect arising in Equation (14) is unchanged, since it is the ratio between the detector separation in the x and y directions that determines the relative importance of this effect. This system is half as light efficient as the system shown in Figure 6.

Another way of producing two images using the same camera is to equip the camera with a dual aperture and operate away from perfect focus [2] as shown in Figure 8. The light from the two apertures will focus in a plane before that in which the detector is located, and hence form out of focus images in two different places on the detector plane. Since we do not want to operate far from correct focus, this method can be modified by placing a

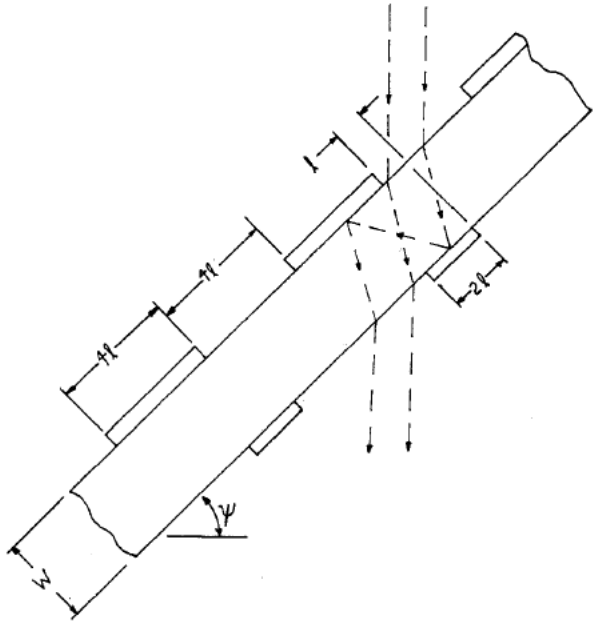


Figure 7: Multiple aperture asymmetric mirror system

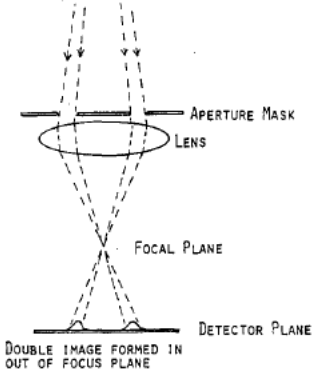


Figure 8: Dual aperture system producing defocused double images

chevron made of two glass plates in front of the apertures (Figure 9). In this manner a double, in-focus image is formed. Again the separation between

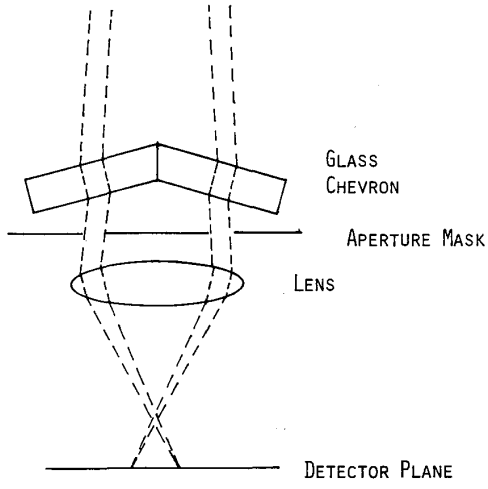


Figure 9: Dual aperture system employing a glass chevron to produce focused double images

the two cameras is forced to be greater than the lens aperture size. This can be overcome by using multiple chevron patterns adjoined to form the corrugated structure shown in Figure 10. However this severely limits the working range of the system since as the image moves far from proper focus the light from each segment of the glass 'wiggle' forms an 'image' at a different lateral position, leading to a great multiplicity of images.

All of the above techniques were investigated, and for our experiments we chose a symmetric modification of the first one mentioned above. Instead of a mirror and beamsplitter, four mirrors [7] are used as shown in Figure 11. This system provides symmetry for the two paths, simplifying focus and computation, but the separation between the two cameras is still constrained to be greater than the aperture size.

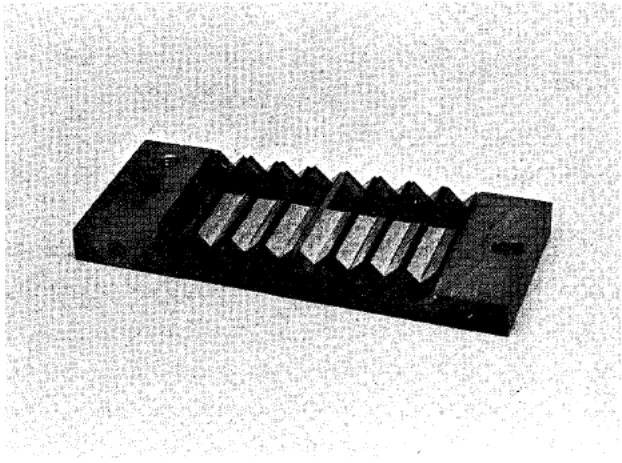


Figure 10: Multiple chevron

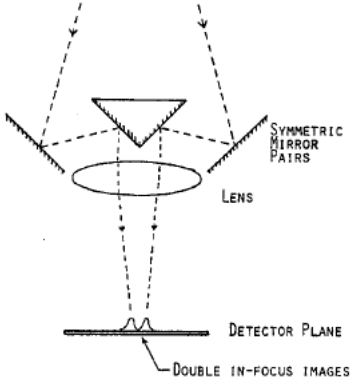


Figure 11: Symmetric mirror system

6. Data Interpretation

We must now face the question of how accurately the position of our stripes can be determined. The simplest approach to determining stripe position is to look for the highest intensity pixel in a region and assign its position to the center of the peak. For a noiseless system this gives an accuracy of $\pm 1/2$ pixel. This accuracy is rapidly degraded by noise; since the slope of the signal is zero at the peak, small amounts of noise can greatly affect the peak position.

If we assume that the peaks are symmetric or that the pair of peaks is symmetric about the center of the pair, we can determine the separation of the two peaks by noting where the intensity, $I(x)$, crosses a threshold, T. Assuming for a single peak that the threshold is crossed upward between pixel n and $n+1$ and downward between pixel m and $m+1$, the center of the peak, to a linear approximation, is at

$$C = \frac{1}{2}\left[n + m + \frac{T - I(n)}{I(n+1) - I(n)} + \frac{I(m) - T}{I(m) - I(m+1)} \right] \tag{33}$$

and the separation between two peaks is the difference between their centers. This technique is simple and effective for the peaks we have observed if the threshold is set at one half the maximum value of a Gaussian stripe image. It has lower noise sensitivity than peak detection because of the non-zero slope of the Gaussian at the threshold level.

A third technique uses the fact that the sampling theorem tells us that if a continuous bandlimited signal is sampled frequently enough, all of the details of the original signal can be reconstructed from the sampled data by convolution with a sinc function. Hence if the size of the peak on our detector is large enough to cover several pixels on the detector we should be able to determine the position of a noiseless signal exactly. This technique assumes that the sample values are not corrupted by noise, which is not a reasonable assumption for stripes reflected from real surfaces.

If the signal is not perfectly bandlimited, this reconstruction will not be ideal. Hence we expect that the error introduced by sampling will depend on the extent to which the high frequency components of the signal shift the peak position away from the position determined by the low frequency components. If this effect is slight (which we would expect for a Gaussian beam) we need not worry about it. Broadband noise will be a problem, however, since its effects cannot be eliminated by lowpass filtering. Some sources and potential solutions to this problem are discussed in the next sections.

A technique that reduces the effects of noise to obtain sub- pixel resolution is to perform a Gaussian least squares fit. Although this technique does provide a good estimate of the peak position, it is computationally intensive. Since one of our goals is to reduce the computational complexity of making height measurements, we did not choose to implement this algorithm as part of our system.

Yet another technique for achieving sub-pixel resolution is to view the intensity of the signal at a given pixel as the probability that the peak is centered at that pixel. Then averaging would give the mean position of the peak. The average is easily computed by the algorithm:

```
SUM = 0
SUM1 = 0
DO 10 x = N, 1, -1
SUM = SUM+I(x)
SUM1 = SUM1+SUM
10 CONTINUE
AVE = SUM1/SUM
```

producing a simple technique that works reasonably well. Our tests have shown that threshold crossing and averaging work equally well for the optical system described above.

7. Ubiquitous Optical Problems

Having discussed algorithms for improving the system's resolution, we must now turn to the optical problems that limit our ability to determine the object distance at a given point.

7.1. Surface Properties

The first two problems to be discussed are object surface roughness and varying object reflectivity. They have two deleterious effects on structured light triangulation systems. First they alter the shape of the stripe, so that the most intense point in the stripe image is not necessarily at the stripe center, and particularly dangerous, the images of the stripes in the two virtual detectors can have different intensity profiles. If the object has finite zones of locally planar surfaces as in Figure 12, then to overcome this

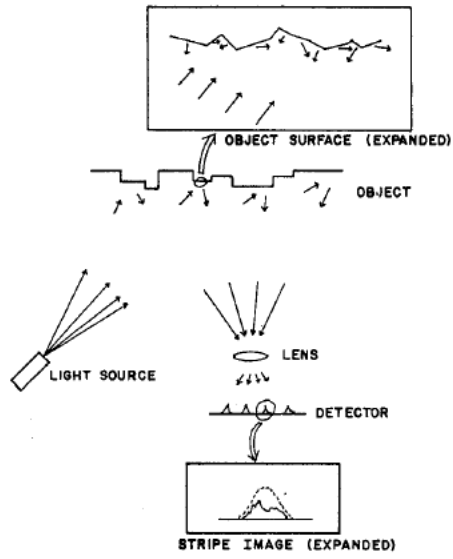


Figure 12: Effect of surface roughness

problem one is forced to make the illumination stripes on the object

narrower than the scale of surface texture and reflectivity changes. (Then the problems arise only at regional boundaries). To accomplish this and still achieve sub-pixel resolution we must magnify the image of the stripes so that each stripe occupies multiple pixels on the detector, reducing the field of view. This may not be possible if the object is rough on a microscopic scale or if we cannot tolerate a reduced field of view. The second problem arising from roughness and reflectivity variations is the large dynamic range of the detector required by the large variation in peak intensities of the stripe images. We have found that our peak finding algorithms work acceptably over a one decade range of peak intensities. Since object reflectivities commonly vary over a four decade range, this leads to errors if there is a large variation in reflectivity in the field of view. One way to solve this problem is to take multiple frames of the same object with different illumination levels. If a log-intensity detector were available this would be ideal.

7.2. System Properties

The next problems are distortions introduced by the optical system itself. The imaging lens introduces aberrations and the detector can have a non-linear and/or space varying response to intensity. Both of these problems lead to space varying errors in distance measurements which do not vary with time, and they can be reduced by proper choice of lens and camera. If the additional processing time is tolerable, the most practical way of eliminating their effects entirely is by calibrating the system to take their effects into account. For example, detector non-uniformities can be compensated by uniformly illuminating the detector and using the resulting intensity readings as normalization factors for future reference measurements. Pincushion or barrel distortion can be compensated in a similar manner by measuring stripe image separation in a reference plane and using this measurement for calibration purposes.

7.3. Speckle

Speckle may be a problem in any system which tries to achieve sub-pixel resolution while using a coherent light source. Since the stripe's image must occupy several pixels to achieve sub-pixel accuracy, the detector must sample the image (and hence the speckle pattern). Several suggestions have been made to reduce this problem which include rotating the polarization of the source, so that the observed speckle pattern is the superposition of two uncorrelated patterns, and making slight perturbations in the position of the effective source to achieve similar effects. Both of these techniques require cycle times shorter than a single frame time, so that the speckle pattern captured in a single frame is really the superposition of the produced patterns. None of these effects have yet been tested sufficiently to report any significant results.

The goal is to make the speckle small with respect to the pixel size of the detector (25 speckles within 1 detector pixel is sufficient for speckle free imaging [6]). Speckle size is similar to the spot size (impulse response) produced by the detector optics [8]. For a focused image, the speckle size is diffraction limited so that it varies as $1/D$, where D is the lens diameter. However, for an out of focus image, the speckle pattern defocuses to produce patches whose size in the geometrical limit is proportional to D. There is therefore a lens diameter D_0 which minimizes the size of speckle over a given range of defocusing. For practical systems this optimal lens diameter results in such a high f that the light level is unsatisfactory. Hence our approach thus far has been to reduce the lens aperture as much as possible while maintaining adequate light levels.

8. Filtering Techniques

In the previous two sections we have mentioned several noise sources, these and others can be reduced somewhat through the use of filters. High frequency noise such as that due to object roughness, detector non-

uniformities, and electronic noise in the detector system can be removed via an analog lowpass filter. The cutoff frequency for this filter can be readily determined by examining the image of a Gaussian stripe on the detector. For a Gaussian distribution with a variance of σ^2, lowpass filtering with a cutoff frequency of 2 σ will leave 95% of the spectrum of the stripe.

Low frequency noise due to variations in ambient lighting can be reduced by a narrow band filter in front of the camera lens. For our system this was entirely acceptable, for applications where it is not, we have found that local min - local max filtering nearly eliminates space variant biases due to ambient lighting.

Quantization noise can only be removed by using a high resolution A-D converter or avoiding the digital domain entirely. We have found that using an 8-bit A-D converter the noise introduced before quantization have always exceeded the quantization noise.

Once the distance data have been calculated, the noise in the resulting distance map can be reduced by using the assumption that distances vary smoothly and averaging distance data in small windows, or by assuming that distance features smaller than some minimal size are artifacts and eliminating them by median filtering [5]. However, it should be noted that severe noise in distance data usually corresponds to edges or holes where the stripes are not reflected back to the camera due to shadowing. Since edges are often important features of objects, it may be a good strategy to flag the absence of data caused by such features to provide data about possible object structure.

9. Experimental Results

A diagram of our prototype system is shown in Figure 13. We produce multiple illumination stripes via a multifaceted hologram, H, which spatially divides an expanded incident laser beam into 16 beams. The 16 beams collectively overlap at the position of lens L_3 (to form a small effective

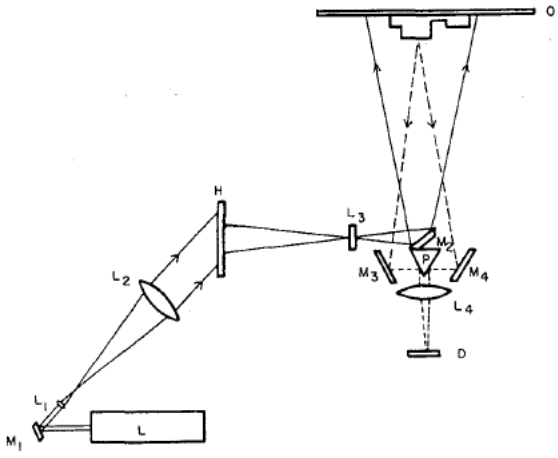


Figure 13: Schematic of multi-stripe 3-D vision system

source) but individually come to focus at the object plane, O. A cylindrical lens, L_3, spreads each beam into a vertical stripe so that a set of 16 focused vertical stripes illuminate the object. Transversely translating lens L_2 in our prototype system, scans the stripes across the surface of the object. In a final system the HeNe laser would be replaced by a laser diode and multiple laser diodes would be pulsed in sequence to scan the stripes, thus eliminating all moving parts. A television camera (lens L_4 and detector D) directed at this pattern can generate the data required for a 128 by 128 element 3-D picture in 8 frames. Again only 8 diodes would be required to produce a picture with the resolution mentioned above. The apparent position of the light source is at the lens pupil so that the center position of any pair of stripes does not change as z is varied, which further simplifies data interpretation and also allows one to treat the system as two single triangulation systems in the event of hidden part problems.

Figure 14 shows a sample image in the detector plane of the stripes corresponding to the staircase object seen in Figure 15. Adjacent steps differ by 2.5 mm in distance from the detector, while the average range is 65 cm.

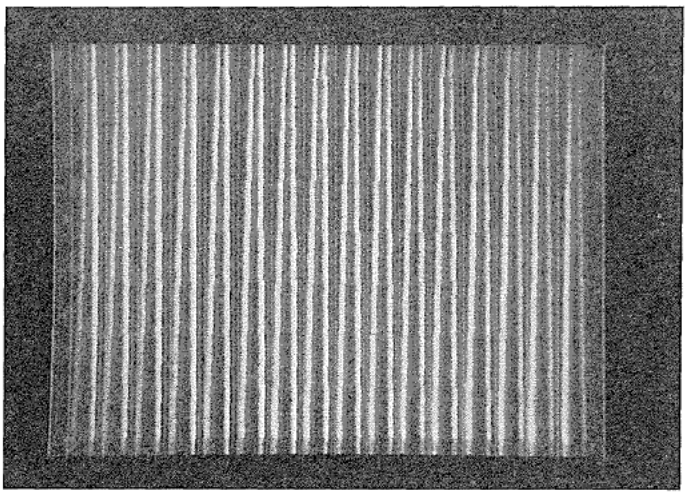


Figure 14: Image of stripes corresponding to step object in Figure 15

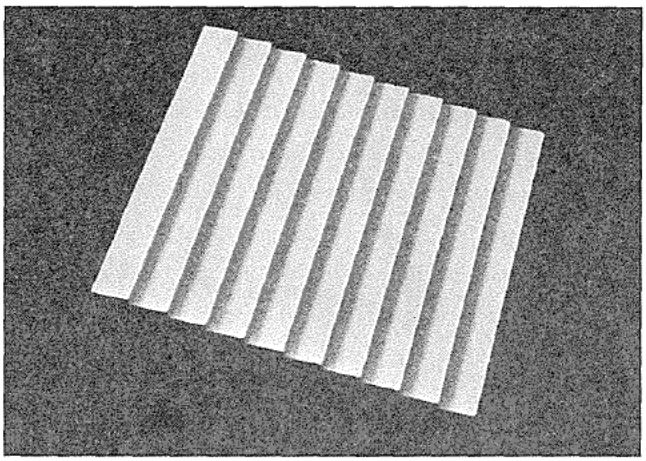


Figure 15: Step object corresponding to stripes in Figure 14

The green lines in Figure 14 are bin markers added by our computer to

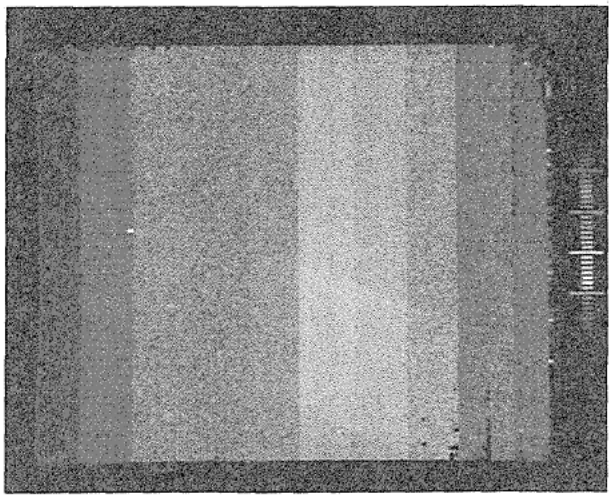


Figure 16: System output for step object in Figure 15

indicate the zone in which the stripes should be found. Figure 16 shows the output of our system when viewing the staircase object with height encoded in pseudocolor. A calibration scale has been added at the edge of the image where blue corresponds to 0" object height (located at z_{max}) and red corresponds to 1/2" height. Data acquisition for the image requires only 8 frame times. Doing all data processing in software, this picture was generated in 2 minutes, and is displayed from a digital frame buffer at TV frame rates. With a special purpose hardware processor the time required to produce a single 3-D picture could be brought down to well under a second. This step from software to hardware is facilitated by the low computational complexity of our algorithms.

The remaining figures present the system's pseudo-color output when viewing common objects in several application areas. Figure 17 shows a printed circuit board. Note that the orientation depressions on the integrated circuits are clearly visible. Figure 18 shows a pile of metal washers. The topmost washer is pseudo-colored green; clearly distinguishing

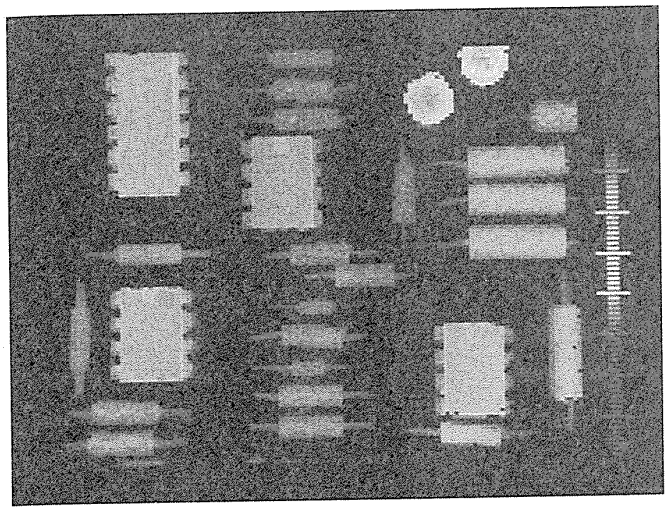


Figure 17: System output for a printed circuit board

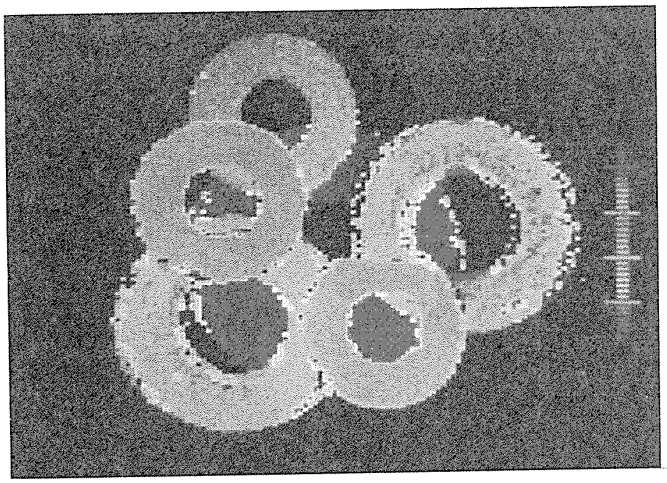


Figure 18: System output for a stack of washers

it from lower washers. Such data would be quite useful in robotics applications where sorting or grasping must be performed. This photograph also demonstrates the problems caused by varying surface reflectivity and multiple reflections at the edges. Figure 19 is the median filtered version of Figure 18 showing why filtering is often desirable for human monitoring of the system's output. Median filtering preserves edge acquity but removes impulse noise. Figure 20 shows a side view of a nose. This problem was posed by persons in the medical field interested in the use of 3-D inspection systems for the data gathering phase of reconstructive facial surgery. Finally, Figure 21 shows a mapping of distance vs. displacement along a single vertical line of the nose showing how various displays can be generated from the digital range data within the system.

The above results show that 3-D vision can find useful applications in a number of areas. Multiple stripe techniques and dual detector architectures may prove to be particularly useful due to the properties described in this paper.

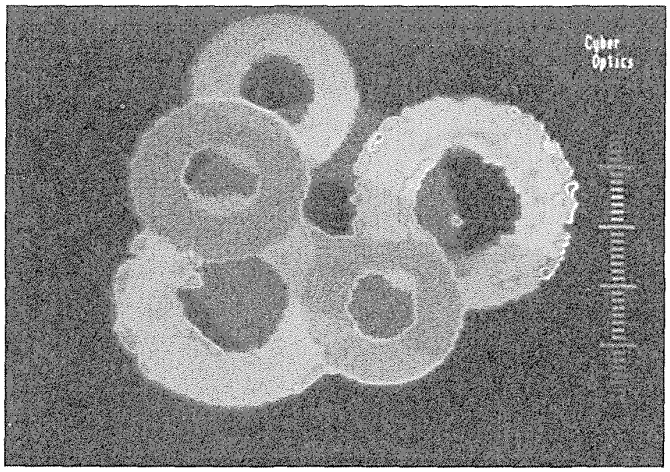


Figure 19: Median filtered version of Figure 18

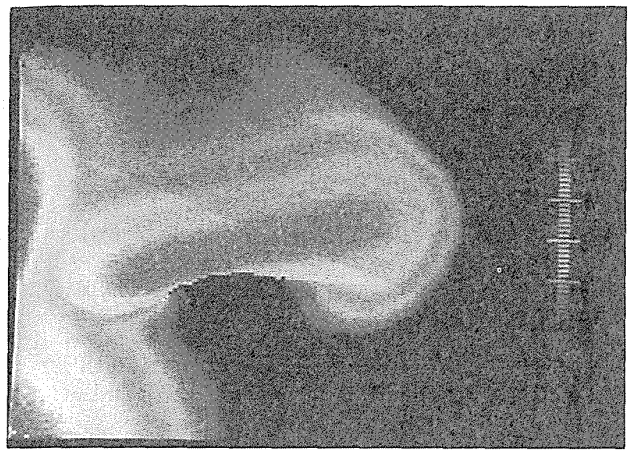


Figure 20: Side view of a nose

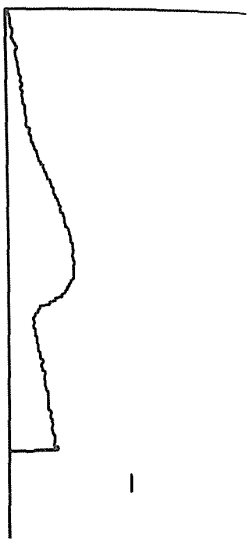


Figure 21: Distance data along a slice of the nose

Acknowledgments

This material is based upon work supported by a National Science Foundation Graduate Fellowship, a grant from the University of Minnesota Productivity Center, and by CyberOptics Corporation.

References

[1] Agin, G. J. and Binford, T. O.
Computer description of curved objects.
In *Proceedings of the Third International Joint Conference on Artificial Intelligence*, pages 629-640. August, 1973.
Also appeared in IEEE Transactions on Computers, 1976, Vol. C-25, pages 439-449.

[2] Blais, Francois, Rioux, Marc, and Pousart, Denis.
Very compact 3-D camera for robotic applications.
Machine Vision, Status and Directions :WB4, 1985.

[3] Case, S. K., Haugen, P. R., and Lokberg, O. J.
Multi-facet holographic optical elements for wavefront transformations.
Appl. Opt. 20:2670, 1981.

[4] Connah, D. M. and Fishbourne, C. A.
Using a laser for scene analysis.
In *Proc. 2nd Int. Conf. Robot Vision and Sensory Controls*, pages 233-240. 1982.

[5] Gallagher, Neil.
private communication, 1984.

[6] George, N.
private communication, 1984.

[7] Geschke, C.
Coordinated Science Lab Report.
Technical Report R-837, Univ. of Illinois at Urbana-Champaign, December, 1978.

[8] Goodman, J. W.
Some fundamental properties of speckle.
JOSA 66(11):1145, 1976.

[9] Haugen, P. R., Keil, R. and Bocci, C.
3-D active vision sensor.
In *Intell. Robots and Computer Vision.* SPIE, 52, 1984.

[10] Jarvis, R. A.
A perspective on range finding techniques for computer vision.
IEEE Trans. Pattern Anal. Machine Intell. PAMI-5:122-139, March, 1983.

[11] Kanade, T. and Asada, H.
Noncontact visual three-dimensional ranging devices.
In B. R. Altschuler (editor), *3-D Machine Perception*, pages 48-53. SPIE, 1981.

[12] Lewis, R. A. and Johnston, A. R.
A scanning laser rangefinder for a robotic vehicle.
Proc. 5th Int. Joint Conf. AI :762-768, 1977.

[13] Nishihara, H. K.
PRISM: A practical real time stereo matcher.
In *Intell. Robots: Third Int. Conf. on Robot Vision and Sensory Controls*, pages 134-142. SPIE, 449, 1985.

[14] Strand, T. C.
Optical three-dimensional sensing for machine vision.
Optical Engineering 24:33-40, 1985.

[15] Yariv, A.
Introduction to Optical Electronics.
Holt, Rinehart & Winston, New York, 1976.

Robot Vision by Encoded Light Beams

Martin D. Altschuler
Kyongtae Bae

Department of Radiation Therapy, School of Medicine
and
Department of Bioengineering, School of Engineering
University of Pennsylvania
Philadelphia, Pennsylvania, 19104

Bruce R. Altschuler
Jerome T. Dijak
Louis A. Tamburino

Systems Avionics Division
Air Force Wright Aeronautical Laboratories
Wright-Patterson AFB, Ohio, 45433

Barbara Woolford

Man-Systems Division
Lyndon B. Johnson Space Center
National Aeronautics and Space Administration
Houston, Texas, 77058

Abstract

A robot eye is more than a camera and less than a complete vision understanding system. It is a device which can scan or sense objects in the three-dimensional environment and extract useful numerical information about those objects. The information obtained by the robot eye should be of a form that can be interrogated by a vision understanding system. This article describes the development of a dependable, robust, and versatile robot eye which can rapidly mensurate a surface in three dimensions. Using an active/passive camera pair, surface mensuration is achieved with fast electro-optic implementation of well-known stereo-photogrammetric principles. We discuss the calibration of the robot eye and the application of the robot eye to exploring and mensurating 3-D objects.

1. Introduction

A computer vision system consists of both a robot eye (or preprocessor) and a vision-understanding (or analysis) component. The robot eye scans or images or otherwise records the environment (or 'scene'), and extracts information and patterns from the data. The vision-understanding component maintains a knowledge base of patterns (or mathematical functions) that may be present in the scene, interrogates the patterns obtained by the robot eye, and attempts to model the scene or classify the object using entities in its knowledge base. The robot eye can explore a scene according to pre-determined strategy or dynamically through feedback from the vision-understanding component.

A simple passive vision system consists of (1) a video camera (or image recorder) to view the scene, and software for image segmentation, and (2) an analyzer, essentially several 2-D patterns stored in memory, and software to recognize whether the observed image contains those patterns. The working environment of a simple vision system is usually strictly controlled with regard to lighting, the orientation and scaling of the camera image relative to the scene, the types of objects allowed in the scene, and the allowed orientations and spacings of those objects. Typical applications for such a vision system are to detect objects on an assembly line, to determine the width of a welding seam, and to inspect fabrics and LSI circuit masks [13].

More advanced passive vision systems attempt to extract 3-D information about a scene (in the form of models or object recognition) with the use of (a) camera images, (b) very sophisticated software for image processing (involving filtering, edge detection, segmentation, feature extraction, etc.), (c) the generation of data structures to organize patterns (such as polygons) extracted from the image, and (d) software to analyze or model the scene by comparing the information derived from the image with the information derived from known objects. When the scene contains many primitive objects that can occlude one another in an image, the

problem becomes very difficult. Model building software may require a hierarchy of inference rules to account for many different image situations that arise with varying scene illumination, shadows, surface texture, and the occlusion of objects along the line of sight. Determining when a complete set of inference rules is obtained or obtainable is always a difficulty [37], [26], [47], [7], [28].

So far we have assumed that the passive robot eye extracts and organizes features of a 2-D image, and that image understanding software infers the 3-D scene by analyzing 2-D image features rather than 3-D information. It is possible, however, to derive 3-D information directly by using two passive stereo cameras separated by sufficient parallax. The robot eye is then a pair of passive stereo cameras that view the same scene. To triangulate a point in the 3-D scene, the corresponding image points must first be identified in the two camera images. The matching of corresponding stereo image points is a difficult correlation problem for a computer, since the light intensity patterns in the image of a scene depend on the camera angle. Humans can match corresponding image points for triangulation with their ability to perceive three-dimensions when the images of a scene are viewed stereoscopically. See [14], [44], [35], [16], [17], [21], [23], [36].

To overcome the complexities of passive vision systems and to obtain directly numerical 3-D information about the scene, active/passive camera systems, sometimes called 'structured light systems' have been developed. In these systems, the robot eye consists of a passive camera and an active element separated by sufficient parallax for triangulation of a scene. The active element emits light patterns which illuminate the scene. Bright points in the passive camera image can then be correlated automatically with rays or planes of light emitted by the active element, and triangulation of points in the scene can be effected. A robot eye that directly acquires and stores 3-D information about a scene obviates much of the complex software needed to extract 3-D information from 2-D images [38], [50], [2], [32], [34],

[39], [43].

Several types of structured light systems have been constructed. The simplest conceptually has a single laser beam sweep across the scene [20]. At any instant, the passive image sees a single illuminated spot of the scene, and triangulation of the spot location is immediate. To obtain a fast sampling rate of the scene, the passive image detector can be a special electronic chip which returns only the location (x^*,y^*) of the centroid of illumination of the image. Then, as the laser beam sweeps the scene, four numbers for each sample point i are obtained, namely the measured direction of the laser beam (u_i,w_i) and the corresponding passive image location (x_i^*,y_i^*). If the passive camera is calibrated to the active element, then each set of four numbers will triangulate a single point of the scene and provide the three coordinates (x_i,y_i,z_i) for the point's location. The simple laser scanner is as good as the chip and the accuracy with which it determines the centroid of illumination. In a noisy environment, where several laser beams or their reflections may be present, or in cases of a laser beam grazing a surface, several surface points may be illuminated simultaneously. Since the chip returns only the illumination centroid of the passive image rather than multiple image points, erroneous mensuration of positions in the scene may sometimes occur.

In a second type of structured light system, the active element projects a plane of light which intersects the scene as a bright stripe [50], [38], [2], [32], [34]. Each bright point appearing in the passive image determines a line from the passive camera to the bright stripe in the scene. Thus three numbers are recorded for any illuminated sample point i in the scene, namely, its passive image location (x_i^*,y_i^*), and the light plane u_i projected by the active element. If the passive camera is calibrated with the active element, each set of three numbers will triangulate a single point of the bright stripe in the scene. After sufficient points of the illuminated stripe on the scene are triangulated, a new stripe can be generated by a parallel light

plane from the active element. In this way the stripe can be swept across the scene, and many sample points of the scene triangulated. Spatial resolution is limited by the thickness of the light stripe. Inaccuracies may result if the parallax is small or if random errors occur in measurement of u_i, x_i^*, or y_i^*.

The difference between the simple laser beam scanner and the stripe projection scanner lies in the precision of the triangulation. The laser beam scanner provides four equations (one each for u_i, w_i^*, x_i^*, y_i^*) to determine the three spatial coordinates of the illuminated point. The equations are overdetermined, hence inconsistent, since two lines in 3-D space need not intersect (and generally won't if there are random errors in image position or active element pointing). The overdetermined equations, however, permit a least-squares triangulation solution, which for the same parallax is less sensitive to random errors of measurement than is the consistent solution (3 equations for 3 unknowns) obtained with the light stripe generator. To achieve the same precision with the light stripe scanner, we can sweep the scene twice, once with vertical and once with horizontal stripes, thereby obtaining four numbers (u,w,x^*,y^*) for each scene point and overdetermining the triangulation problem as before.

Rather than sweep a light stripe across the scene so that N light stripes are necessary to mensurate the scene, we can illuminate (and observe) the entire scene with a sequence of only $\lceil \log_2 (N+1) \rceil$ binary-coded striped-light patterns. For image point (x_i^*, y_i^*), viewing sample point i of the scene, we can decode the brightness variation in the sequence of images and determine the coordinate u_i of the light plane (stripe) which illuminates point i. Thus eight binary coded light patterns and passive images provide the same information for triangulation (namely u_i, x_i^*, y_i^* for each sample point i) as do 128 individually-projected stripes and passive images. Although just $\lceil \log_2 N \rceil$ light patterns are needed to binary code N stripes, one stripe, zero, is always unilluminated in the binary code.

In this paper we describe the development over the past few years of an

advanced robot eye of the structured light type [3], [4], [33]. The eye contains an active element which can convert a beam from a single laser source into an interferometric array (currently 128 × 128) of laser 'beamlets' diverging from a common focus, then project and spatially modulate the distinct laser beamlets. For robot vision purposes, the beamlets are essentially rays. Thus the active element obeys exactly the same mathematics as a passive camera with an array (128 × 128) of pixels. We will call the active element an 'active camera', and the active/passive camera pair a 'numerical stereo camera' or 'NSC'. The computer controller can be programmed to turn on and off individual rays from the active camera, so that the NSC can simulate a single laser beam scanning the scene, a light stripe projector, or a projector of binary coded light patterns. Although we briefly describe the NSC hardware, we are concerned primarily with the algorithmic and software effort to make the NSC into as reliable and intelligent a robot eye or vision system preprocessor as possible. The goal is to make a robot eye that can adapt to different ambient light environments, can be calibrated simply and quickly, and is intelligent enough to decide on new viewing perspectives when concavities or holes are poorly observed from initially chosen perspectives. A robust robot eye which mensurates a 3-D scene can be an off-the-shelf item applicable to many different industrial and biomedical problems without being dependent on the vision understanding component of the system [19], [9], [5], [53]. Such an independent entity which directly obtains 3-D information could lead to vision understanding modules which are less complex and more standardized, require less development time for a particular application, require less time for computation in operation, and probably require less memory for pattern recognition of 3-D objects.

Some recent developments in direct acquisition of 3-D scene information also show promise:

1. Structured light systems which project multi-wavelength (i.e., multi-color) light patterns for stripe coding [52].

2. Radar-type rangefinders which emit picosecond pulses of laser light and record their echoes from the scene; this approach does not use triangulation [24], [43].
3. Passive fixed cameras which observe the same scene as it is illuminated from different directions; here some knowledge about the texture of the reflecting surface is useful, and the goal is to obtain the spatial orientation of surface elements (that is, partial 3-D information) [40], [18], [51].

We will not discuss these techniques further.

Over the past several years, the USAF has developed an advanced prototype NSC. This work began at the USAF School of Aerospace Medicine in the late 1970's and was transferred to the Wright Aeronautical Laboratories (AFWAL) in 1982. An advanced prototype system was constructed at the AFWAL Avionics Laboratory where it is still being developed and evaluated. Dr. J. Taboada (USAF School of Aerospace Medicine), a co-inventor of the NSC, has been and remains a key contributor to the development of the NSC optical subsystems. The integrated NSC hardware/software system developed at AFWAL is somewhat different from that described in this paper, and will be described in detail elsewhere.

2. System Description of the NSC

The NSC version of the robot eye consists of three components: an active camera, a passive camera, and software.

The active camera transmits an array of discrete laser beams outward from a common focus. Each beam can be identified and addressed by its row and column in the array of beams. The beams intersect an 'image plane' that controls which of them may pass through to illuminate the environment. We can use the same terminology for both the active and passive cameras by considering each discrete beam of the laser array to be a 'pixel' of the active camera image plane.

The active camera hardware consists of (a) a single laser, (b) a lens to expand the single beam into a larger solid angle, (c) an optical assembly to

generate (by reflection or refraction methods) an interference pattern of $M \times N$ discrete and orthogonally arranged laser beamlets, (d) optics to position the pattern so that the $M \times N$ laser beamlets are collimated and aligned to correspond precisely to optical window elements of a planar light modulator, (e) an electro-optic programmable spatial light modulator device (or PSLM) to modulate the projection of each of the discrete laser beamlets into the environment [10], [8], [22], [30].

All the laser beamlets have the same plane polarization orientation. A PSLM can control the transmission of beamlets because the plane of polarization of electro-optic material in the PSLM can be rotated as a function of an applied electric field, and because the applied electric field in the PSLM can be made to vary with position across the wavefront of beamlets. The presently implemented PSLM is a composite of two PSLMs, one to modulate the N columns and the other to modulate the M rows of the beamlet array, with $M=N=128$. A command by computer can cause a (preprogrammed) set of columns in a PSLM to rotate optically in polarization, thereby permitting either transmission or absorption of the corresponding incident beamlets (depending on the relative orientation of the polarized beamlets to that of the PSLM electro-optic material). A polarization rotator at the laser source is adjusted to maximize the transmission/absorption ratios of the PSLM in the optical system.

The generation of the initial interference pattern which creates the orthogonal array of beamlets is the product of diffraction effects. Individual beamlets are part of the interference pattern, so that range of projection is limited by the diffraction limits of the entire projecting lens aperture, and not by the aperture size of each beamlet.

The PSLM has window apertures greater than the beamlets, and light is so aligned and collimated that each beamlet passes through (or is blocked by) a corresponding window with minimum far field diffraction by the PSLM. Thus beamlets at infinite focus can project significantly into the far

field with little degradation.

Lenses can converge the laser beamlet array for the mensuration of microscopic surfaces or expand the array for the mapping of large surface areas. Each beamlet retains its relative spatial position and size in the array, and the entire pattern expands or contracts proportionately. Limitations to the system are set by the wavelength of the light and by the S/N ratio; the latter involves laser beam power, quality of the active camera hardware and optical alignment, monochromaticity, sensitivity and electronic characteristics of the passive camera, ambient light, and noise in the environment.

The passive camera is an optical image recorder which views the scene from a given perspective as different patterns of light are emitted by the active camera. The passive camera obtains a numerical image of the scene (that is, an 8-bit number for gray level for each pixel) for each pattern of scene illumination. Some types of passive camera can obtain an image directly in digital form; others must pass an analog image through an A/D converter. Buffer storage to hold a sequence of images, or else a device to transfer rapidly image data to and from a computer, is needed. The optical recorder must be synchronized with (the transmission of patterns by) the electro-optic light modulator of the active camera. To obtain a good signal to noise ratio, optical filters are used to make the passive camera specific to the wavelength of the laser light transmitted. The choice of wavelength is largely dependent on the application and the available lasers and sensors.

The region of the scene illuminated by a single beamlet should be observed by several passive camera pixels to ensure resolution. Ultimate mapping resolution is more dependent on a higher resolution video camera than on the laser array size.

Although available active cameras can create huge laser beamlet arrays, PSLM technology is presently limited to modulating 128 $\times$ 128 arrays. With video cameras and recorders of 2048 $\times$ 2048 on the verge of

practicality, mapping scenes illuminated with 512 $\times$ 512 beamlet arrays should be reasonable as soon as appropriate PSLMs become available.

When the first NSC was implemented several years ago, much laser light was lost in generating interference fringes by multiple reflection, thus lowering the signal to noise ratio. As a result, it was often difficult to determine whether or not a passive image pixel was viewing a laser illuminated part of the scene. Also, the active and passive cameras had relatively long focal lengths (that is, weak convergence toward the focal point of the camera), so that determining those calibration coefficients containing perspective information was an ill-conditioned (error sensitive) problem. An intense software effort was initiated to improve the reliability of the NSC and to simplify wraparound mapping of a 3-D object. Recently patented optical hardware to generate interference fringes by refraction, and other significant optical improvements in PSLM alignment, have led to very high efficiencies of laser light during array generation in the system. The array now has extremely discrete and sharp laser beamlets, and a much improved S/N ratio. With improved hardware and software, the NSC is becoming a reliable and versatile device. In this paper we discuss the improved algorithms developed for the NSC; the new hardware improvements will be described elsewhere.

Since we discuss only the robot vision applications rather than the engineering details of the system, we hereafter use the word 'beam' or 'ray' instead of 'beamlet'. Keep in mind, however, that the 'beams' we refer to are peaks of an interference pattern on a single laser beam, and cannot mutually overlap or interfere with increasing range of projection.

3. Mathematical Concepts

We consider a beam of the laser beam array to be simply an image point (or pixel) of the active camera. We can then use the same terminology for both the active and passive cameras, thus the terminology of

stereophotogrammetry. A correlation of a specific point in the passive image with a specific beam in the laser array is considered to be a correlation of image points (or pixels) in the active and passive cameras.

3.1. Triangulation

The point (x,y,z) on the surface of interest and its passive camera image (x^*,y^*) are related by the perspective transformation [14], [44], [35]:

$$(T_{11}-T_{14}\,x^*)x + (T_{21}-T_{24}\,x^*)y + (T_{31}-T_{34}\,x^*)z + (T_{41}-x^*) = 0 \quad , \qquad (1)$$

$$(T_{12}-T_{14}\,y^*)x + (T_{22}-T_{24}\,y^*)y + (T_{32}-T_{34}\,y^*)z + (T_{42}-y^*) = 0 \quad . \qquad (2)$$

If the scene-to-image transformation matrix T is known, we have for each illuminated point visible to the passive camera the known quantities T_{ij}, x^*, y^* and two equations for the three unknowns (x,y,z). We need one more equation for triangulation. The active camera point (or laser beam) (u,w) which illuminates the point (x,y,z) of the surface of interest is also given by a perspective transformation

$$(L_{11}-L_{14}u)x + (L_{21}-L_{24}u)y + (L_{31}-L_{34}u)z + (L_{41}-u) = 0 \quad , \qquad (3)$$

$$(L_{12}-L_{14}w)x + (L_{22}-L_{24}w)y + (L_{32}-L_{34}w)z + (L_{42}-w) = 0 \quad , \qquad (4)$$

where L is the scene-to-active-image transformation matrix. If we sequence the electronic shutter to project certain specific light patterns (that is, a spacecode) with the active camera columns, we can associate each bright image point (x^*,y^*) in the passive camera with a value of u corresponding to the column of a beam in the projected array of laser beams. Since only three equations are necessary for triangulation, we can solve for each of the laser-illuminated surface points with only column coding, making use of Equations (1)-(3). However, if we spacecode the optical shutter for rows and columns, we can improve the precision of the triangulation. The full set of four (usually inconsistent) Equations (1)-(4) can be solved for a 'best' value of (x,y,z) by least squares. In computation, the singular-value-

decomposition method (SVD) provides for least-squares answers with more significant figures for the same computer word size [15], and is preferred when the condition number of the matrix is large, as is the case when the triangulation parallax is small. A further justification for spacecoding both rows and columns lies in the detection of surface regions that are illuminated by the active camera but not seen by the passive camera.

Thus the problem of triangulation is: Given a (passive) image point (x^*,y^*), a corresponding active image point (u,w), and the constants $\{T\}$ and $\{L\}$, find the three-dimensional location (x,y,z) of the point being illuminated.

Although the triangulation problem is conceptually straightforward, the practical problem of programming the numerical stereo camera is not trivial. In a subsequent section we discuss the software needed to make the numerical stereo camera a practical device.

3.2. Spacecodes

The correlation of corresponding image points in cameras at two different perspectives can be made automatic with the use of active and passive cameras. The array of laser beams emanating from the laser-interference-electro-optical projector is equivalent to an array of pixels of an active camera. When the laser beam array is turned on and off in certain light patterns, and the passive camera, which is synchronized to the active camera, receives those light patterns in sequential images, the corresponding pixels of the active and passive cameras (that is, a stereo pair) can be determined for each observed surface point. For example, a simple binary spacecode can be projected to encode the address of each column of a 128 column laser array as follows:

Pattern	Columns transmitting (columns are numbered 0-127)
1	64-127
2	32-63,96-127
3	16-31,48-63,80-95,112-127

4	8-15,24-31,40-47,...,104-111,120-127
5	4-7,12-15,20-23,...,116-119,124-127
6	2-3,6-7,10-11,...,122-123,126-127
7	1,3,5,...,125,127 (i.e., every other column)
8	0,1,2,...,126,127 (i.e., every column)

Pattern 8 is not used for the binary code but to identify the regions of the scene illuminated by laser column 0. (See also Figure 1.)

From the table above, we see that if a pixel of the passive image is bright in the first image, dim in the second, bright in the third and fourth, dim in the fifth and sixth, and bright in the seventh, in accord with the pattern 1011001, then the pixel is viewing a surface position illuminated by a laser beam from column $(2^6+2^4+2^3+1=)$ 89 of the laser beam array.

Although binary coding of a scene by light patterns is conceptually simple, image noise can result from the hardware, from the nature of the surface of the objects in the scene, and from the chosen perspectives of the cameras. To compensate for noise, additional light patterns may be projected and imaged, and additional processing steps may be added.

3.3. Focal Point of a Camera

The focal point of a camera with calibration coefficients T_{ij} is

$$(x_f, y_f, z_f) = -(A/D, B/D, C/D) \tag{5}$$

where the values of A, B, C, D are given by the determinants

$$A = \begin{bmatrix} T_{41} & T_{21} & T_{31} \\ T_{42} & T_{22} & T_{32} \\ T_{44} & T_{24} & T_{34} \end{bmatrix} \quad C = \begin{bmatrix} T_{11} & T_{21} & T_{41} \\ T_{12} & T_{22} & T_{42} \\ T_{14} & T_{24} & T_{44} \end{bmatrix}$$

$$B = \begin{bmatrix} T_{11} & T_{41} & T_{31} \\ T_{12} & T_{42} & T_{32} \\ T_{14} & T_{44} & T_{34} \end{bmatrix} \quad D = \begin{bmatrix} T_{11} & T_{21} & T_{31} \\ T_{12} & T_{22} & T_{32} \\ T_{14} & T_{24} & T_{34} \end{bmatrix} \tag{6}$$

and $T_{44} = 1$.

Equations (5) and (6) derive from the perspective projection

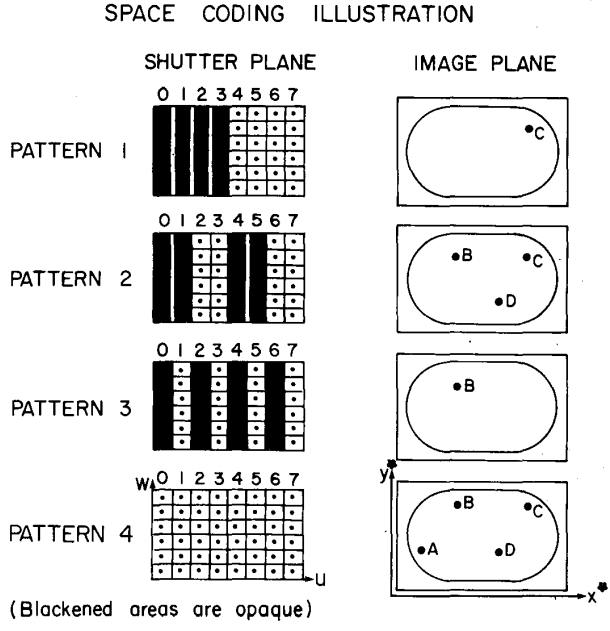


Figure 1: The array of laser beams is space coded by the sequence of binary patterns (shown at the left) obtained by controlling the opacity of the columns of the shutter plane. The intersections of the laser beams with the scene provide bright spots on the scene which can be imaged by a passive camera from a suitable perspective. The passive camera is synchronized with the active camera so that for each projected light pattern a passive image is obtained (shown at the right). In the figure, the passive image sees four bright spots of the scene. The column location u of a beam in the active image plane can be determined by the space coding received by the passive image plane. Point B is seen in passive images 2 and 3 but not in image 1. The binary code of the column of the corresponding active beam is therefore $u = 011 = 3$. The fourth (open) pattern illuminates column zero.

$$T_{11}x + T_{21}y + T_{31}z + T_{41} = Hx^* = X \tag{7}$$

$$T_{12}x + T_{22}y + T_{32}z + T_{42} = Hy^* = Y \tag{8}$$

$$T_{14}x + T_{24}y + T_{34}z + T_{44} = H \tag{9}$$

If $H \neq 0$, these three equations reduce to Equations (1) and (2). Thus the case $H \neq 0$ corresponds to a line or ray from the camera focal point to the point (x^*,y^*) of the image plane and then outward into the 3-D space of the scene. The focal point is on every ray of the camera, and cannot depend on the values of x^* or y^*. Only for the case $H = 0$ do Equations (7)-(9) give a solution for a point rather than a line; thus the focal point is given by the case H=0. Equations (5) and (6) correspond to the solution of Equations (7) through (9) with $H = 0$.

3.4. Camera Calibration

We now discuss camera calibration, the determination of the matrix coefficients connecting the coordinates of the scene and the image plane. In this section (only), the same notation T_{ij} and (x^*,y^*) is used for both the passive and the active cameras. To obtain the calibration mathematics in the notation used previously for the active camera, one need merely replace T_{ij} with L_{ij} and (x^*,y^*) with (u,w). By image plane we mean either the image plane of the passive camera or the electro-optic shutter plane of the active camera.

The values of the camera calibration coefficients T_{ij} depend on the orientation, translation, focal distance, and scaling of the camera relative to the 3-D scene. There are two independent coordinate systems: (x^*,y^*) for an image point and (x,y,z) for a point in the scene. The calibration procedure determines the T_{ij} values that satisfy Equations (1) and (2) when we are given a set of known points $\{(x_i,y_i,z_i)\}$ in the scene and their known corresponding image points $\{(x_i^*,y_i^*)\}$, where i=1, . . . ,n and $n \geq 6$.

We can set T_{44}=1 [44], [35] and obtain the $2n$ ($n \geq 6$) (inconsistent)

inhomogeneous equations for the 11 unknown values of T_{ij}:

$$\begin{bmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -x^*_1x_1 & -x^*_1y_1 & -x^*_1z_1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ x_n & y_n & z_n & 1 & 0 & 0 & 0 & 0 & -x^*_nx_n & -x^*_ny_n & -x^*_nz_n \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -y^*_1x_1 & -y^*_1y_1 & -y^*_1z_1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & x_n & y_n & z_n & 1 & -y^*_nx_n & -y^*_ny_n & -y^*_nz_n \end{bmatrix} \begin{bmatrix} T_{11} \\ T_{21} \\ T_{31} \\ T_{41} \\ T_{12} \\ T_{22} \\ T_{32} \\ T_{42} \\ T_{14} \\ T_{24} \\ T_{34} \end{bmatrix} = \begin{bmatrix} x^*_1 \\ \cdot \\ \cdot \\ \cdot \\ x^*_n \\ y^*_1 \\ \cdot \\ \cdot \\ \cdot \\ y^*_n \end{bmatrix} \tag{10}$$

or simply

$$A\,T = D \tag{11}$$

where A is the $2n \times 11$ matrix, T is the 11×1 column vector, and D is the $2n \times 1$ column vector.

The matrix Equation (11) can be solved for 'best' values of T_{ij}. The normal equation method would involve solving

$$A^tA\,T = A^tD \tag{12}$$

where A^t is the transpose of A, the matrix A^tA is an 11×11 symmetric matrix, and $A^t\,D$ is an 11×1 column vector. However, for the same word size, the normal equation method of Equation (12) loses more significant figures than other methods such as singular value decomposition [15].

Significant figures are important in this calibration procedure because the matrix A may not be very well conditioned if the known points in the scene (x_i,y_i,z_i) or the camera points (x_i^*,y_i^*), are not sufficiently separated. The calibration coefficients T_{14}, T_{24}, T_{34} derive from the converging of rays to the focal point; without significant separation of sample points and careful solution of Equation (11) they cannot be extracted accurately.

The problem of calibrating a camera is thus: Given a set of known noncoplanar points in 3-D and their corresponding image points in the camera, find the constants $\{T\}$. Thus given $\{(x_i,y_i,z_i,x_i^*,y_i^*); i=1,\ldots,n; n \geq 6\}$, find $\{T_{ij}; i=1,\ldots,4; j=1,2,4\}$.

4. Fundamental Software Problems

The fundamental software problems that must be solved to make the NSC robot eye a relatively independent preprocessor for a vision system are:

1. Signal Detection: To determine whether or not an observed bright point seen in a passive image corresponds to a laser-illuminated region of the scene, and if so, to determine the bit of the spacecode contributed by that light pattern at that point of the passive image.
2. Passive Camera Calibration: To calibrate the passive camera; that is, to determine the constants T_{ij} of Equations (1) and (2).
3. Active Camera Calibration: To calibrate the active camera; that is, to determine the constants L_{ij} of Equations (3) and (4).
4. Motion Recalibration: To recalibrate rapidly the passive and active cameras if the surface of interest or any of the cameras are displaced by a known amount.
5. Triangulation: To triangulate a point in the scene reliably with the minimal loss of significant figures.
6. Missing Region Detection: To detect regions of the scene that were not triangulated because of missing data (due to blocking of beams or occlusion of lines of sight by intervening hills and valleys).
7. Information Recovery: To pick new perspectives to attempt to mensurate regions of missing data in the scene.
8. Wraparound 3-D Mapping: To effect wraparound mapping of any 3-D object by merging in the same 3-D coordinate system the triangulation data of different stereo perspectives.

5. Signal Detection: the Problem of Determining the Spacecode

5.1. The Nature of the Problem

To stereo-correlate the active and passive camera images and triangulate points in the scene, the sequence of images acquired by the passive camera must be analyzed to determine the binary code received at every pixel of the passive image. To achieve this task, the passive camera must distinguish laser-illuminated areas of the surface from other areas. Unfortunately, there is no obvious single threshold value in the brightness distribution of the passive image to flag those image regions viewing laser-illuminated surface. The reason is that considerable variations of brightness may occur from one passive image to the next, because (1) regions of the surface of interest may reflect differently (anisotropically) for different angles of beam incidence, (2) different regions of the scene may have different textures and scatter properties, (3) the shutter configuration of each different projected light pattern may have different scatter and transmission properties, and (4) the environment may be noisy because of other lasers, grazing beams, or reflections. Detection (or thresholding) errors mean incorrect values for the spacecode, thus the creation of non- existent beams with concurrent loss of existing beams in the database created for 3-D stereo triangulation.

5.2. Algorithmic Considerations

A number of ways are available to reduce and correct errors. These methods are similar to the error detecting and correcting codes of information theory. All of them require additional projected light patterns and recorded passive images. Two methods have been tried for error detection and correction with the numerical stereo camera.

In the first method, the scene is illuminated by an all-laser-beams-on pattern and the passive image is scanned for brightness peaks and valleys. A single (global) threshold is determined to separate the bright regions (which

are presumably seeing laser illuminated surface) from the rest of the passive image. These 'valid' regions are remembered and followed in the subsequent sequence of images containing the light code. For each passive image in the sequence, the same brightness threshold determines the spacecode bit for each valid pixel; if a pixel is brighter than threshold, the spacecode bit contributed by that image is one. The method gives reliable spacecodes when there is a large S/N ratio and the image contrast does not change with the projected light pattern (shutter configuration). Experience has shown, however, that S/N ratio may vary over an image because the angles of incidence of the passive and active rays vary over the surface, and that the contrast at the same point in the passive image plane may vary significantly from one coded image to the next because the shutter transmission may vary with the configuration. Under such circumstances, a single global brightness value to threshold all the passive images in the coded sequence is not reliable.

The second method, developed by L. Tamburino, determines the spacecode received at each pixel of the passive image, and, in addition, a measure of statistical reliability for the spacecode [45]. Since 1983, several versions of the Tamburino detection algorithm have been implemented on the prototype NSC at the AFWAL Avionics Laboratory, and all demonstrate marked improvements in computational and spacecode accuracy [46], [12].

6. Calibrating the Passive Camera

In Section 3.4 we discussed the mathematics of camera calibration. For the case of the passive camera, the simplest calibration procedure is to (1) insert a known rectangular prism into the scene, (2) let the vertices of the prism define the scene coordinate system, (3) pick the 6 or 7 visible vertices of the prism and record their image positions as seen on the graphics screen, and (4) solve Equation (11) by singular value decomposition (SVD) for the

T_{ij} coefficients.

A sequence of two software programs is required. The first allows the user to define the prism coordinates and pick the vertices visible to the passive camera. The second solves for the values of T_{ij} by SVD.

This section and the next describe possible approaches to passive and active camera calibration. A somewhat different approach has also been implemented [11].

6.1. Picking Calibration Points

The first program records the spatial coordinates of six or seven points in the calibration prism and their image coordinates in the passive camera.

The algorithm steps are as follows:

S1: Enter the name of the data file for the sequence of coded images. The name acts as a tag to be associated with all the files associated with this problem.

S2: Enter the dimensions of the sides of the calibration prism. The first dimension is assumed to be along the x-axis; the second to be along the y-axis; and the third to be along the z-axis.

S3: Enter the 3-D locations of the calibration points on the prism. The points chosen are usually six or seven of the vertices of the prism since their images are easiest to identify in the passive camera image. The eight vertices of the prism can be conveniently chosen to be (0,0,0), (a,0,0), (a,b,0), (0,b,0), (0,0,c), (a,0,c), (a,b,c), (0,b,c), where a, b, c are the known dimensions of the prism for the x, y, z axes respectively. With an opaque calibration prism, at most seven of these eight vertices can be expected to be visible from any given perspective.

S4: Enter the (2-D) locations of the image points corresponding to the calibration points of the prism. In the program a corresponding image point is entered immediately after the chosen calibration point.

S5: Create a data file which contains the values

$\{(x_i,y_i,z_i,x_i^*,y_i^*);\ i=1,6\}$.

6.2. Determining the Passive Camera Calibration Coefficients

The second program finds the values of the calibration coefficients T_{ij} of the passive camera.

The algorithm steps are as follows:

S1: Read the data file generated by program 1.

S2: Find the coefficients T_{ij} . The input data are used to create a $2n\times11$ matrix and a $2n$ vector, where n is 6 or 7. The 11-vector containing the values of T_{ij} is then solved by SVD. If the parallax is so small that the converging of the rays to the camera focal point cannot be detected, the condition number of the solution will exceed a threshold determined by the number of significant figures for single precision real numbers. In that case, the program will reduce the $2n\times11$ matrix to a $2n\times8$ matrix and the 11-vector to an 8-vector, the rotation and translation components of T_{ij} will be computed, and the converging-ray components set to zero.

S3: Find the location of the focal point of the passive camera if the parallax permits (i.e., if the SVD provides sufficient significant figures for the available parallax). The focal point of the passive camera is used primarily as a check on the reasonableness of the calculation, and later in the program for active camera calibration to determine the visible faces of the calibration prism.

S4: Find the maximum and minimum values of $\{x_i^*\}$ and $\{y_i^*\}$. In the calibration of the active camera (next section), these values allow us to limit the search range in the passive image and to ignore background scatter (noise) from points outside the calibration prism.

S5: Write an output file which contains the values of T_{ij}, the focal point location, and the limits of the calibration prism in the passive camera.

7. The Problem of Calibrating the Active Camera

7.1. Algorithmic Considerations

The problem of calibrating the active camera requires that the constants L_{ij} of Equations (3) and (4) be determined. Just as for the passive camera, the values L_{ij} depend on the orientation, translation, focal distance, and scaling of the active camera relative to the 3-D scene. There are two independent coordinate systems: (u,w) for an active image point (i.e., for the address of a laser beam in the array of laser beams) and (x,y,z) for a point in the scene. The calibration procedure determines the L_{ij} values that satisfy Equations (3) and (4) for a set of known points (x_i,y_i,z_i) in the scene and their known corresponding active image points (u_i,w_i) where $i=1, \ldots, n$ and $n \geq 6$. We have tried four ways of calibrating the active camera.

Method 1: Turn on one beam in the laser array at a time and locate in 3-D by direct measurement where the beam falls on a surface. With six or more noncoplanar measured points (x_i,y_i,z_i) and their corresponding beams (u_i,w_i), we can solve for the L_{ij} using the same technique that we used for the passive camera calibration. This method has the advantage that the accuracy of the active camera is independent of the accuracy of the passive camera. The disadvantage is that the location of the intersection of the laser beam (u_i,w_i) with the 3-D surface at (x_i,y_i,z_i) must be physically measured in 3-D space.

Method 2: Place a rectangular prism of known dimensions in the scene. The six surfaces of the prism might, for example, satisfy the equations $x=0$, $x=a$, $y=0$, $y=b$, $z=0$, $z=c$ where the values for a, b, c are the lengths of the prism edges. Observe (sequentially) six or more laser-illuminated non- coplanar spots on the calibration prism (due to six or more separate laser beams of the array) and record each laser beam (u_i,w_i) and the corresponding position (x_i^*, y_i^*) in the passive image. Identify and record the passive image positions of the (observable) vertices of the prism (or retrieve

them from the files of the passive camera calibration). Then, use the image positions of the vertices to determine the face of the prism that each spot illuminates, and to locate in 3-D (by interpolation in the image) each of the illuminated spots on the prism. We then have $\{(x_i,y_i,z_i,u_i,w_i);\ i=1,\ldots,n;\ n \geq 6\}$ and can determine the L_{ij} calibration coefficients of the active camera. The advantage of the method is that no measurement need be made in the 3-D scene. The disadvantage is that correctness and accuracy depend on the accuracy of the pixel discretization of the passive camera, the absence of distortions in the passive image, and the foresight to ensure that the selected active camera beams do not all intersect a single plane surface in the 3-D scene.

Method 3: Turn on one beam (u_i,w_i) of the laser beam array and locate the illuminated prism point (x_i,y_i,z_i) by using (1) the known T_{ij} values previously calculated for the passive camera, (2) the known image point (x_i^*,y_i^*), and (3) the equations for the faces of the rectangular prism. To do this, trace the ray from the passive camera and find the intersections of this ray with the planes of each of the prism faces. Choose the correct intersection point automatically by considering the visibility of the prism faces relative to the passive camera. Repeat this procedure for $n \geq 6$ non-coplanar illuminated points on the prism (from n different laser beams), and determine a set of active camera and spatial points $\{(x_i,y_i,z_i,u_i,w_i);\ i=1,\ldots,n;\ n \geq 6\}$ to provide the calibration. The advantages and disadvantages of this method are similar to those of method 2. Now the accuracy also depends on the accuracy of the values of the T_{ij}. However, since this method determines the 3-D positions of the calibration points i from the ray and plane equations in space and not from interpolation in 2-D images, it is preferred if there are distortions in the passive image; this follows because even a distorted image plane corresponds point by point to rays radiating outward from the focal point, whereas interpolation in a distorted image must be done in non-Euclidean

coordinates.

Method 4: A variation of method 3, but now instead of imaging one active camera point (i.e., laser beam) at a time, we sequence through a spacecode pattern of illumination, correlate each (u_i,w_i) with the appropriate (x_i^*,y_i^*) which appears in the passive camera (provided (u_i,w_i) hits the surface at a point visible to the passive camera). Here as many as 128×128 points may be used to calculate the L_{ij} by least squares. Since a plane has the equation

$$A\,x + B\,y + C\,z = D \tag{13}$$

and the values of A, B, C, D are known for each face of the rectangular prism, we can solve Equations (1), (2), and (13) for (x,y,z) when we are given (x^*,y^*) and the T_{ij} .

The problem of calibrating the active camera is thus: Find a set of known points in 3-D that are illuminated by known laser beams (active image points) of the active camera. This may be obtained from knowledge of the prism faces, the passive image points, the passive camera calibration, and the spacecodes of the passive image points. Then calculate the L_{ij} .

7.2. Calculating the Coefficients of the Active Camera

We have implemented a program using method 4 above to find the values of the calibration coefficients L_{ij} of the active camera.

The algorithm steps are as follows:

S1: Retrieve the data files generated by previous programs for the prism parameters and for the passive camera calibration coefficients T_{ij} .

S2: From the passive camera images of a sequence of coded light patterns, derive the spacecode received at each pixel of the passive camera, and create both a column and row coded image. The column coded image is a 512×512 table of bytes, one byte for each pixel of the passive camera. The numerical value of each byte is either the (reliable) spacecode of an active camera column, or zero if the space code at the pixel of the passive

image is not reliable. (In a normal image, the numerical value of each byte corresponds to the intensity or gray level at a pixel; thus the spacecoded image should be distinguished from a normal picture image!) Similarly, the row coded image is a 512 × 512 table of bytes, each byte containing the spacecode for the active-camera row seen by a pixel of the passive image.

S3: Ask for an optional speed-up factor. A speed-up factor of 2 will process every other pixel of a coded image, allowing shorter times of computation at the cost of poorer statistics.

S4: For the column and row coded images separately, check each pixel. For each pixel with a non-zero (reliable) spacecode, trace back the ray from the pixel to the calibration prism. The ray of (x^*,y^*), determined by Equations (1) and (2), intersects the known planes (see Equation (13)) of the calibration prism. If an intersection point is found on a prism face visible to the passive camera, then the matrix and vector used to derive the active-camera calibration coefficients are updated. In effect, the matrix and vector contain both the 3-D locations of points on the calibration prism and the corresponding beams from the active camera (derived from the column or row spacecode values detected at the passive camera pixel). The number of intersection points found for each face of the calibration prism is recorded. At least two prism faces must be sampled for the active camera to be calibrated.

S5: Use the matrix and vector generated by the data to determine the values of the calibration coefficients L_{ij} of the active camera. Singular value decomposition is used to solve the large matrix equation, with one matrix row for each reliable spacecode byte.

S6: Calculate and report the focal point of the active camera. The location of the focal point checks the reasonableness of the calculation and helps pick the face of the prism intersecting a line of sight from the passive camera.

S7: Store the values of L_{ij} .

8. Rapid Recalibration for Changing Perspectives

8.1. Algorithmic Considerations

To map the surface of a truly 3-D object, we must view the object from several different perspectives. With the calibration technique so far described, the wrap-around mapping procedure would be as follows: (1) Set up the active and passive cameras; (2) Insert a calibration prism into the scene and calibrate the cameras; (3) Remove the calibration prism from the scene and insert the object of interest; (4) Map the object from this perspective; (5) Move the object or cameras to a new perspective; (6) Repeat steps 2 through 5 until the entire 3-D surface of the object is mapped. This calibration procedure is slow and error prone because the object of interest must be removed prior to each new perspective to allow camera calibration and then replaced in exactly the same position it was originally.

If a precision positioning device can be attached to the camera mounts or object platform, the calibration procedure can be somewhat simplified to: (1) Set up the active and passive camera; (2) Insert a calibration prism into the scene and calibrate the cameras; (3) Move the prism or camera to a new perspective, note the position readings, and calibrate the cameras; (4) Repeat step 3 for a sufficient number of perspectives; (5) Remove the calibration prism from the scene and insert the object of interest; (6) Map the object from each of the previously calibrated perspectives. The advantage of this method is that the calibration is done initially for all the perspectives so that the object of interest can be mapped in 3-D without disturbing its position on the platform. The disadvantage is that we must decide on the viewing perspectives in advance.

If a precision positioning device keeps track of the translations and rotations of the object and cameras, we need only calibrate the cameras and object initially. Thereafter we can recalibrate the cameras with a mathematical transformation without actually inserting a physical prism

into the scene. The advantage is that we can choose new perspectives as the topology of the surface dictates; we need not restrict ourselves to previously-chosen perspectives of uncertain efficacy. Thus the simplest calibration procedure with a precision positioning device is: (1) Set up the active and passive camera; (2) Insert a calibration prism into the scene and calibrate the cameras; (3) Remove the calibration prism from the scene and insert the object of interest; (4) Map the object from this perspective; (5) Move the object or a camera to a new position; (6) Recalibrate the cameras mathematically from the motion information, in accord with the discussion below; (7) Repeat steps 4 to 6 for a sufficient number of perspectives. (See also [42] and [41] for more on calibration techniques.)

Let T be the perspective operation $(x,y,z,1) \rightarrow (X,Y,Z,H)$ or

$$[x\ y\ z\ 1]T = [x^*\ y^*\ z^*\ 1]H$$

and let P be the projection operation onto the z=0 plane

$$[x\ y\ z\ 1]P = [x\ y\ 0\ 1]\ .$$

Then the operation $T\,P$ (i.e., first T then P) is

$$[x\ y\ z\ 1]T\,P = [x^*\ y^*\ 0\ 1]H\ . \tag{14}$$

Equation (14) is shorthand for Equations (7) to (9), or (1) and (2). Let R be the (known) displacement operator which rotates or translates points (x,y,z) to (x',y',z') so that

$$[x\ y\ z\ 1]R = [x'\ y'\ z'\ 1]\ .$$

Then the 3-D points would project onto new (primed) camera points by

$$[x\ y\ z\ 1]R\,T\,P = [x^{*\prime}\ y^{*\prime}\ 0\ 1]H' \tag{15}$$

Because we want to map a 3-D object from multiple perspectives, we want coordinates (x,y,z) fixed to the object regardless of any motions of the cameras or the object. We thus interpret Equation (15) by saying that there exists new calibration coefficients T' where

$$T' = R\,T/k \tag{16}$$

$$H'' = H'/k \tag{17}$$

and k is a normalization constant, so that Equation (15) becomes

$$[x\ y\ z\ 1]T'P = [x^{*\prime}\ y^{*\prime}\ 0\ 1]H'' \tag{18}$$

We can now triangulate using the primed (new perspective) image points and the known calibration coefficients $T'P$ and continue to locate the surface points of the object in the same object coordinate system as before. Thus

$$T'P = \frac{R\,T\,P}{k}$$

$$= \frac{1}{k}\begin{bmatrix} R_{11} & R_{12} & R_{13} & 0 \\ R_{21} & R_{22} & R_{23} & 0 \\ R_{31} & R_{32} & R_{33} & 0 \\ D_1 & D_2 & D_3 & 1 \end{bmatrix}\begin{bmatrix} T_{11} & T_{12} & 0 & T_{14} \\ T_{21} & T_{22} & 0 & T_{24} \\ T_{31} & T_{32} & 0 & T_{34} \\ T_{41} & T_{42} & 0 & T_{44} \end{bmatrix} \tag{19}$$

where the R_{ij} are rotation terms, the D_i are displacement terms, and the T_{ij} are the original calibration coefficients. Since we assigned $T_{44} = 1$ (normalization) to reduce the number of coefficients T_{ij} from 12 to 11, we similarly require $T_{44}'=1$ for the new calibration coefficients T'; thus

$$k = D_1\,T_{14}+D_2\,T_{24}+D_3\,T_{34}+1\,. \tag{20}$$

Note that the normalization factor k is only non-unity when translation occurs (that is, D_1, D_2, D_3 are not all zero).

The same type of transformation applies to the active camera, namely

$$L' = R\,L/k\,. \tag{21}$$

On the other hand, if the object remains stationary but the camera is displaced, we use the inverse of R for the camera that moves and obtain the same form of Equation (18). But now only the camera that moves requires new calibration coefficients. If both cameras move, then two different (inverse) operators of the form of R are used. A sequence of motions for the camera or object leads to a sequence of new calibration coefficients given by

Equation (16) and (21) but with

$$R = R_1 R_2 R_3 \ldots \tag{22}$$

8.2. Rapid Recalibration After Relative Motion

A program has been developed to find the values of the new calibration coefficients T_{ij} and L_{ij} after relative motion between the object and the passive and/or active cameras. If the relative motion is known precisely, there is no need to insert the calibration prism back into the scene or to disturb the cameras or object of interest. The recalibration can be done mathematically rather than physically.

The algorithm steps are as follows:

S1: Read the data files for the present calibration points and camera calibration coefficients T_{ij} and L_{ij}. These files contain information about the calibration prism, the original calibration coefficients for the passive and active cameras, and the calibrations done at each step in a chain of previous motions chosen by the user or system. That is, every time the object or a camera is moved and a recalibration is done, the new calibration information is stored. If a string of motions is carried out, calibration is continually updated. Error buildup is less if the motions are known more precisely.

S2: For interactive mode, display a menu of options for the user:

1. rotate passive camera
2. rotate active camera
3. rotate object
4. translate passive camera
5. translate active camera
6. translate object

After a motion option is completed, the program returns to display this menu, and the user can continue to move the object and/or a camera until he chooses an exit option.

S3: To rotate the object, specify the axis and angle of rotation. The axis specification requires any point on the axis and a vector parallel to the axis.

S4: Calculate new calibration coefficients for the passive and/or active

cameras directly from the old coefficients and the given rotations and translations (see Equations (19) and (20)). The case of translation has the complication of requiring a renormalization in the calculation for the new calibration coefficients (cf. Section 8.1). The accuracy of the new calibration coefficients does not depend on the parallax; although there is little point in doing so, one can move the object or camera miles apart and still recalibrate accurately.

S5: Determine the new focal points of the active and passive cameras directly from the calibration coefficients.

S6: Although the calibration prism has been physically removed from the scene, we can imagine the vertices to be still attached to the object and can follow them mathematically. If the object is rotated or translated, determine the new positions for the vertices. From the 6 or 7 vertices of the calibration prism, calculate the corresponding image points in the passive and active cameras. If only the passive (or active) camera is rotated, calculate new image points only for the passive (or active) camera. From these image points, calculate the new limits of the image of the (imaginary) calibration prism on the passive camera. In this way we can keep track of the field of view.

S7: Store the new calibration coefficients, and the locations of the calibration prism vertices and their images, in a user-designated file.

9. Triangulation

When the passive and active cameras have been calibrated by the methods described in Sections 6 to 8, the NSC can be used to triangulate multiple points in the scene. With a 128 $\times$ 128 array of laser beams, up to 16,384 points of the scene or of a surface can be mensurated in 3-D space. The basic mathematics of the triangulation problem was discussed in Section 3.1.

The major problem that arises in actual practice is that the parallax

between the passive and active cameras is often small, so that the solution of Equations (1) through (4) becomes ill-conditioned. When this happens, the number of significant figures in the solution decreases and one must be careful not to end up with (meaningless) answers that have no significant figures. One approach is to use double precision real numbers in the calculation. The other is to use SVD [15] to solve the four overdetermined equations. Conceptually, the loss of significant figures corresponds to the error-sensitivity of the triangulated position in the direction perpendicular to the baseline between the active and passive cameras.

Since the T_{ij} and L_{ij} coefficients of the cameras are known from the calibration procedures, the position (x^*,y^*) of a passive image point (or pixel) is known, and the corresponding laser beam (u,w) is known from the spacecode received at the pixel, Equations (1) through (4) can be written as the matrix equation

$$T\,x = b\,. \tag{23}$$

Here T is a 4×3 matrix containing the expressions in parentheses in the first three terms of each of the Equations (1) through (4), b is the 4×1 vector containing the negative of the expressions in parentheses in the last term of each of the equations, and x is the 3×1 column vector containing the unknown coordinates (x,y,z) of the scene point to be triangulated.

If the parallax is small, then Equation (23) represents a set of four ill-conditioned inhomogeneous linear equations. These four equations are usually inconsistent because of measurement errors, and correspond to planes in space which intersect in a tetrahedron rather than at a single point. If we were to solve the equations by creating the 3×3 square matrix $T^t\,T$ and solving for x, we would lose many more significant figures than if we simply solved Equation (23) directly by SVD. The least squares solution obtained by SVD corresponds to a point in the tetrahedron satisfying a condition of minimum error.

Constraint equations can be added to Equation (23) to weight the

solution and move the solution point around within the tetrahedron. If the tetrahedron is elongated away from the cameras because of ill-conditioning, then the additional constraints may significantly affect the location of the solution. One should introduce constraints into the least squares solution only if one has some prior knowledge as to how the solution should behave. With large parallax and well-conditioned equations, the tetrahedron is generally small, and any point in the tetrahedron should provide an adequate solution for triangulation.

In the algorithm and software that we have developed, the SVD method is used to flag cases with a very high condition number (that is, too few significant figures for the accuracy desired).

10. Missing Region Detection

Stereo cameras, separated by parallax, view a scene from different perspectives. As the parallax increases, the problem of triangulating points in the scene becomes more well-posed (less error sensitive) but the number of points in the scene that are imaged simultaneously by both cameras generally decreases. In stereophotogrammetry, an observer views a stereo pair of photographs and perceives which points of the scene are visible to both cameras and thus can be triangulated. If some scene regions are not viewed by either camera because mountains or valleys obstruct the line-of-sight from a camera perspective, the observer may use or request images taken from yet other perspectives.

The same problem of missing regions can occur with the NSC, except that now missing regions can be detected by an algorithm rather than by human perception. For the NSC, there are two separate cases, depending on which camera (active or passive) is obstructed.

Case 1: Light projected by the active camera does not illuminate a part of the scene because of a surface convolution (obstructing mountain, cliff, or hole). The passive camera viewing the unilluminated part of the scene sees a

region of unreliable spacecodes in the image plane because there is little change in light intensity received when either a light pattern or its complement is on. Thus, if the passive image has a region which (1) contains pixels with mostly unreliable spacecodes, and (2) is surrounded by regions which contain many pixels with reliable spacecodes; then we conclude that lines of sight from the active camera to the corresponding region of the scene are occluded because of intervening mountains or holes.

Case 2: Light reflected from an illuminated part of the scene does not reach the passive camera because of an intervening surface convolution. The passive camera may obtain reliable spacecodes in the image plane, but may see spacecode values which change discontinuously from one observed bright spot to the next across some unseen boundary in the image plane, indicating some spacecode values are missing. Thus if the passive image sees two adjacent regions, each containing pixels with mostly reliable spacecodes, and a sudden discontinuity in the value of the spacecode from one region to the next, then we conclude that some lines of sight from the scene to the passive camera are occluded because of intervening mountains or holes.

In different regions of the same passive image both cases 1 and 2 may arise (Figure 2).

We have two tasks in this section: (1) to detect which regions of the scene are not illuminated by the active camera, or viewed by the passive camera, and (2) to estimate a minimum depth of any hole or minimum height of any hill in the unmensurated regions of the scene.

We assume that the active camera can sample the scene sufficiently; that is, every significant wiggle or convolution in the scene can be sampled at least twice. Equivalently, the separation of laser bright spots on the scene nowhere exceeds half the smallest wavelength of any 3-D spatial harmonic contained in the scene. Without sufficient sampling by the active camera, the mensuration of the scene could be aliased. (See for example [27].)

In practice the grid of pixels in the passive image has much finer spatial

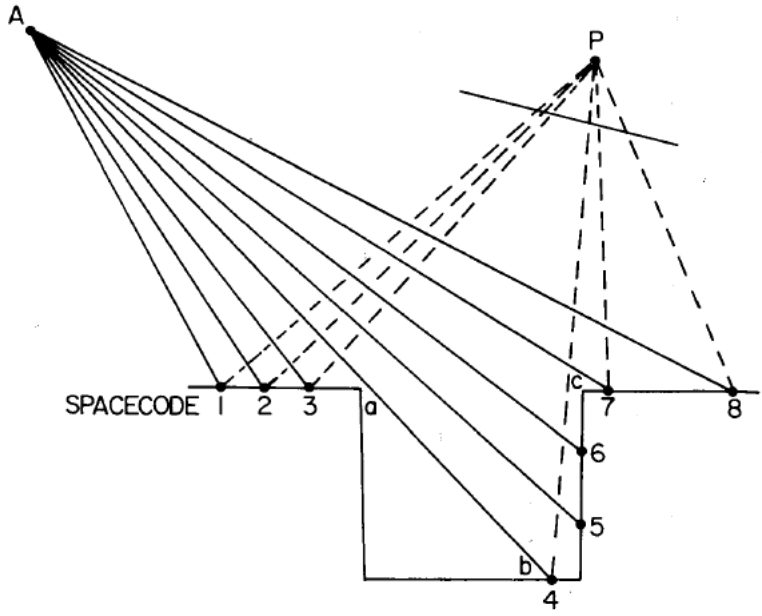


Figure 2: Region *ab* is unilluminated by the active camera so that the passive image gets an unreliable spacecode there (thus Case 1). The bright dots within region *bc* are not seen by the passive camera; the camera sees bright dots at *b* and *c* and a jump in the spacecode values from 4 to 7 (Case 2).

resolution than the grid of pixels in the active image. Thus, the passive image may have (1) pixels which view the same bright spot of the scene and thus detect the same spacecode, and (2) pixels which view regions between the bright spots of the scene and thus have unreliable spacecode. Pixels in the second group do not correspond to obstructed rays from the active camera, and must be distinguished from them.

It is therefore convenient to define a neighbor of a passive image pixel as follows. Consider a passive image pixel which has a reliable spacecode corresponding to active camera beam (p,q). Then a neighbor of that pixel is a passive image pixel which has a reliable spacecode corresponding to beam $(p',q')=(p+i,q+j)$ in the allowed domain of the active camera beam array such that p, q, p', q', i, j are integers and (i,j) is in one of the classes

$(i > 0,0)$, $(0,j > 0)$, $(i < 0,0)$, $(0,j < 0)$ called the directions of the neighbor. A neighbor of order n is a neighbor such that the magnitude of either i or j is n. Note that adjacent passive pixels which view the same bright spot are not neighbors by the definition above, but share the same neighbors. See Figure 3.

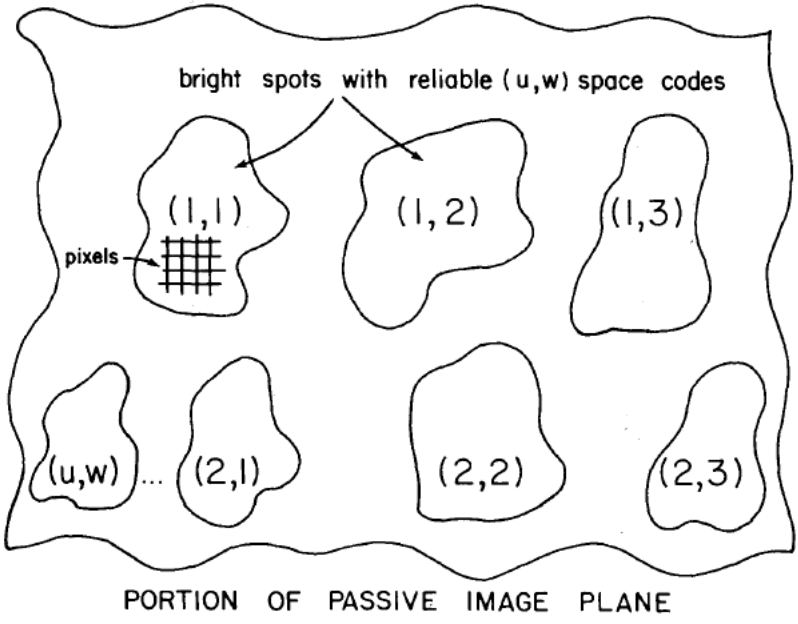


Figure 3: All pixels within the labelled regions (or spot images) have reliable spacecodes and are seeing the reflections of laser beams from the surface of interest. Every pixel in region (1,1) is a neighbor of order 1 to every pixel in regions (1,2) and (2,1), a neighbor of order 2 to every pixel in region (1,3), and not a neighbor to pixels of regions (2,2) and (2,3). Pixels in regions (1,2) and (2,1) are not neighbors.

Task 1a: To detect regions of the scene that are not illuminated by the beams of the active camera (case 1):

S1: In the passive image, find the mean and standard deviation of the separation distance (in pixel units) of pixels from their neighbors of order 1.

S2: Flag all passive pixels which are separated by two or more standard deviations from (at least) one neighbor of order 1. Remove duplication by keeping only the median pixel of any group of pixels which view the same illuminated spot on the scene (and have the same spacecode).

S3: Determine which of the flagged pixels are neighbors of order 1 and are separated by more than two standard deviations. These should form the boundary of the region of unreliable spacecodes in the passive image.

Task 1b: To detect regions of the scene that are illuminated by the active camera but are not seen by the passive camera (case 2):

S1: Find all beams of the laser array whose spacecodes are missing from the passive image.

S2: In the passive image, record the locations of all pixels which have (1) a reliable spacecode and (2) a neighbor of order $n \geq 2$ in a certain direction but no neighbor of order 1 in that direction.

S3: Find the contours in the active image plane that correspond to the loci of pixels found in step 2.

S4: Find which of the missing beams of step 1 lie within these loci in the active image. This subset of the missing beams reaches the scene, but the rays from the scene to the passive camera are obstructed.

Task 2: We have now delineated the regions of missing (blocked or obstructed) rays in the active and passive images. However, the boundaries of these missing ray regions can be used in triangulation of the scene. The boundary of an unmensurated region appears in the passive image plane for case 1 and in the active image plane for case 2. Since the mathematics is the same for both cases, we simply say the image boundary of the unmensurated region is in the image plane.

The steps to determine the minimum depth of an unmensurated region of the scene are as follows (see Figure 4):

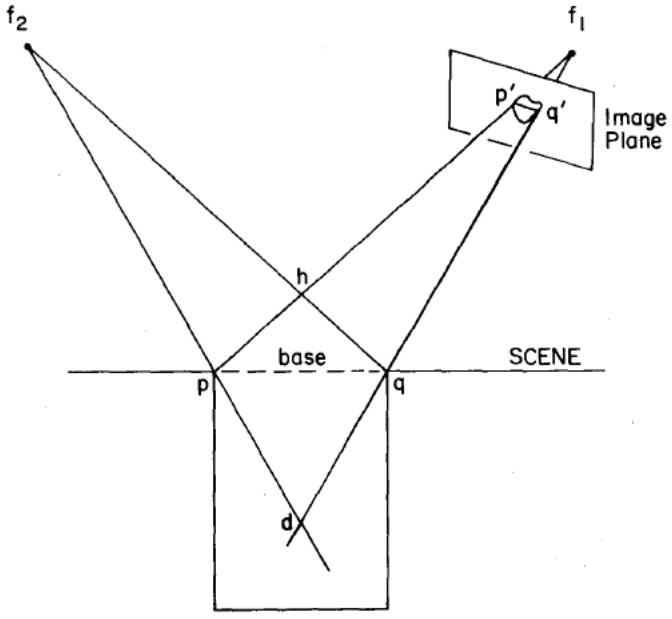


Figure 4: Cameras are at f_1 and f_2 and an unmensurated region is outlined in the image plane of camera 1. Image points p' and q' at the image boundary of the unmensurated region are correlated by the NSC with points of the other camera so that the boundary points p and q in the scene can be triangulated. The quadrilateral $phqd$ in the plane $f_1\ f_2 p$ gives the height of the possible hill or the minimum depth of the possible hole that could cause obstruction of rays.

S1: Pick an image point p' on the image boundary of the unmensurated region; this image point is associated with a scene point p which can be triangulated.

S2: Determine the plane f_1,f_2,p containing that scene point and the focal points of both the active and passive cameras.

S3: Determine the line of intersection $p'q'$ of that plane with the image plane. The image point chosen in step 1 is on this line.

S4: If the point q' is on the image boundary of the unmensurated region, we have a line segment in the image plane which cuts through the image of the unmensurated region. Find the scene points p,q corresponding to the

endpoints of the line segment in the image plane; these scene points will be considered the base of a triangle.

S5: Draw rays from the focal points of the active and passive cameras to these measured endpoints in the scene. These four rays intersect in a quadrilateral. Calculate the altitudes of the two triangles containing the measured scene points as a base. These altitudes determine the minimum possible height of a hill or the minimum possible depth of a hole (in the plane of the rays) to explain the unmensurated region.

S6: Choose a neighboring image point of p' on the image boundary of the unmensurated region and repeat steps S1-S5. Repeat for consecutive points on the image boundary until every point has been considered. Choose the greatest minimum height and greatest minimum depth derived from all the line segments through the image of the unmensurated region in the image plane. These values correspond to the minimum possible height and minimum possible depth consistent with the entire unmensurated region, as determined from the perspectives of the active and passive cameras.

11. Information Recovery

Once we have detected a missing region of the scene and have found the minimum values of height and depth for that NSC perspective, we can choose new perspectives to allow triangulation of points in that region.

We begin on the assumption that the unmensurated region is a hole. Since we know nothing about the hole, we assume it extends as a right cylinder in a direction perpendicular to the least squares plane of the hole boundary. We find the principal axis of the hole boundary (projected) on the least squares plane, and measure depth along the central axis which bisects the principal axis and parallels the cylinder of the hole. Since we do not know the depth of the hole, we must consider the tradeoffs in positioning the cameras:

1. If we position the cameras so that both are within the cylinder delineated by the hole boundary, we can look directly into the

hole. Unfortunately, triangulation becomes more ill-conditioned as the parallax between the cameras decreases. Moreover, the error is greatest in the direction of depth into the hole. Thus for every possible depth of the hole bottom, the parallax angle must exceed some minimum if triangulation is to provide a given number of significant figures.

2. To map the floor of the hole with several sample points, the passive image must resolve the bright dots illuminated by adjacent beams of the active camera. For laboratory situations, there will be little difference in resolution from object surface to hole bottom.
3. For a hole of a given depth, maximum parallax is obtained as follows: Consider a triangle whose base is the principal axis of the hole and whose third point lies on the central axis at the depth of the hole (Figure 5). Extend the sides of the triangle from the third point and let the cameras be within the angle of these lines and in the plane of the triangle. Position the cameras as near as possible to the sides of the extended triangle. Note that the triangle becomes more acute as the hole depth increases.

The strategy we choose is as follows:

S1: For each unmensurated region, begin by assuming it is a hole.

S2: Decide on the minimum depth for which repositioning cameras is worthwhile. In particular, if the hole opening is so narrow that only a small depth into the hole can be measured without losing significant figures in the triangulation calculation, moving the cameras may not be efficacious.

S3: To position both cameras within but at the edge of the hole cylinder, calculate the parallax of measurement to the hole bottom for each possible depth. Determine the maximum depth for which some desired number of significant figures can be obtained with the available parallax.

S4: To position both cameras to maximize parallax (Figure 5), calculate the maximum parallax for different possible depths (from the extended triangles of tradeoff (3) above). Again calculate the maximum depth for the hole consistent with significant figures in triangulation.

S5: If neither S3 nor S4 allow probing of the hole to any significant depth, the hole is too small; do not reposition the cameras. If S4 allows probing to a significant depth but S3 does not, then reposition the cameras

in accord with S4. If S3 allows probing to a significant depth, reposition the cameras in accord with S3.

S6: If it should turn out that the mensuration at the new camera positions is poor and indicative of a mountain rather than a hole, then the camera positions should be shifted (in the plane of the central and principal axes of the region boundary) so that both cameras are first on one side of the unmensurated region and then on the other.

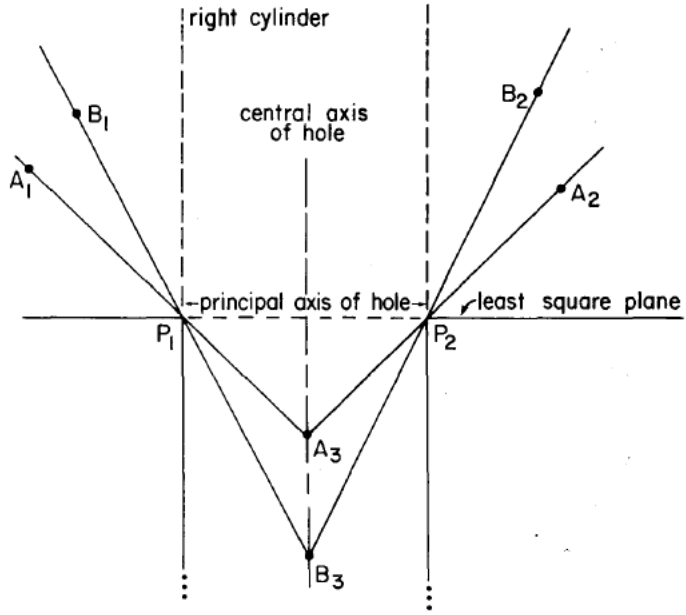


Figure 5: The cross-section of an unmensurated hole is used to illustrate how the maximum parallax decreases when we want to see deeper into the hole. Let P_1P_2 be the principal axis (thus the widest opening of the hole). To probe the hole to depth A_3 we can separate the cameras by a maximum parallax angle of $A_1A_3A_2$. To probe to depth B_3, the maximum parallax angle $B_1B_3B_2$ is less.

12. Fully Three-Dimensional Wrap-Around Surface Mapping

To reconstruct the surface of a three-dimensional object, we need images from multiple perspectives around the object.

12.1. Data Acquisition

The first step in reconstructing the complete surface of a 3-D object is to triangulate numerous surface sample points from each of several perspectives. The numerical stereo camera is capable of illuminating and mensurating up to 128 × 128 different regions (spots) of a surface at each perspective. For real-time 3-D data acquisition, several active and/or passive cameras may be used simultaneously at various perspectives around an object; the reflections from the different active cameras can be distinguished if each laser operates in a different spectral region. When real time is not required, a single setup can be used sequentially at different perspectives (for example, by placing the object on a turntable). Since object and camera repositioning need not require recalibration, cameras can be relocated to any perspective to achieve better measurements.

The problem then is to combine the mensurations made at each different perspective to achieve the overall surface map.

12.2. Representation of the Surface Data Points

Suppose, for an object or scene, that the surface points which have been triangulated from different perspectives or camera positions are given by

$$\{(x_{ip},y_{ip},z_{ip}) : i=1,2,\ldots,I(p);\ p=1,2,\ldots,P\}$$

where P is the number of camera perspectives used in the mensuration, and $I(p)$ is the number of surface points measured from perspective p. A vision preprocessor or robot eye should be able to specify the complete surface of the object or scene from these sample points. This involves two tasks: (A) the surface locations determined from the different perspectives must be

consistent, and (B) the surface measurements must permit an accurate representation of the surface on the global scale. It is left for the vision-understanding component of the system to segment the 3-D information and identify the object or scene components.

As long as the camera positions and their changes from perspective to perspective are accurately known, and the parallax between the active and passive cameras of the NSC is sufficient to provide well-conditioned solutions, triangulation from the different perspectives will be consistent with one another. In fact, any inconsistency for a surface location measured with overlapping camera perspectives is itself evidence of either a systematic error in the determination of camera positions or insufficient parallax. Experimental work with the NSC indicates that task (A) has been achieved. We are therefore concerned primarily with task (B) of the robot eye.

To obtain a useful global representation of a surface, it is necessary to sample every 'significant structure' of the surface at least twice by bright dots (laser beam/surface intersections) from some active camera perspective. The passive camera is always assumed to sample a surface of interest more finely than the active camera. If some surface features are smaller than twice the spacing between sample points on the surface, then the reconstructed surface derived from the triangulated bright dots will be undersampled (aliased). An aliased representation of a 3-D surface not only loses the fine structures (that is, the high harmonics), but provides an incorrect global (low harmonic) map as well. Great care must be taken to ensure that surface mensuration is not done with aliased data.

A complication is that 'significant structure' is determined not only by the spatial harmonic spectrum of the object but also by the minimum information required by the vision-understanding component to recognize the object. If one needs to know only that an object of a certain size is present, then the 'significant structure' of the object can be obtained with only a few surface sample points. On the other hand, if one needs to

distinguish one human face from another or to detect small shape abnormalities for anthropometry, then very high harmonic information, thus fine sampling, is needed to obtain the 'significant structure' of the surface. This complication requires feedback loops between the robot eye and vision-understanding modules of the system. In the case of the NSC, collecting high density sampling data is only logarithmically more expensive than collecting low density data. (In fact, the data collection time, even for high density data, is only about 1 second at TV camera scan rates. Since the solid state shutter switches in microseconds, data collection times of 1 ms are conceivable with faster and more sensitive cameras.) The processing of the data, however, is linear with the number of sample points when a sequential computer is used. Hence to apply the NSC in a 3-D vision system, the number of sample points to be processed for any application would be read from the vision- understanding component. Thus a single number for the sampling rate would feed back from the higher understanding level to the level of surface data acquisition. The higher level may also decide where the robot eye should look, that is, the boundaries of the scene, but once this is done, the 3-D data acquisition for the delineated scene is done at the preprocessor level. The modular integrity of the eye and the understanding components is thereby maintained.

Some surfaces of interest may have undersampled fine structure only over relatively small patches. For example, a small grating in a street may be undersampled, when the rest of the street is adequately sampled. A good global representation of a surface should not be unduly affected by small undersampled patches.

We are studying several approaches to the representation of a 3-D object from local position measurements. The first method uses voxels [6], the second uses the convex hull [29], [49], and the third uses surface patches [35], [31]. All of the methods are reasonably insensitive to small areas of undersampling. (See also [1], [48].)

12.2.1. Simple Voxel Approach

S1: For each surface region, choose the smallest dot spacing from among those active camera perspectives which sample the region.

S2: Choose the voxel size to be twice the largest of these smallest dot spacings of the different surface regions.

S3: Fill the three-dimensional space (which contains the object or scene of interest) with voxels of the chosen size.

S4: For each sample point (bright dot) on the surface, find the address of the voxel which contains it. The voxels which contain bright dots constitute the surface of the object or scene. The addresses of these voxels are the only numbers that need be stored to describe the surface.

S5: Because of the way we chose the voxel size there should be few gaps (empty voxels) in the object surface. Filled (surface) voxels are connected if they share a surface, edge, or point. Close any gaps in the mensurated surface which are at most a voxel thick.

12.2.2. Convex Hull Approach

S1: Find the convex hull of all the triangulated surface points. The convex hull is chosen as the initial approximation of the object's surface.

S2: For each point within the convex hull, effect a local connection to the hull to achieve a better approximation of the object's surface. Continue this procedure until all mensurated points are attached to the surface. Several methods have been proposed to do this based on various optimization goals such as minimum area [29], or minimizing a cost function of various local geometric properties [49].

12.2.3. Bicubic Spline Approach

S1: Construct a separate surface patch for each perspective of the active and passive cameras.

S2: If the edges of every patch overlap with edges of other patches so that an extended path from a patch always lies within some patch, we can

construct a global representation of the surface of the 3-D object.

This method is used in aerial photogrammetry, and in mapping moons from interplanetary fly-bys [25].

13. Appendix: Singular Value Decomposition

To solve least squares problems with a minimum loss of significant figures, the method of singular value decomposition has been used throughout this work. Here we review briefly the salient ideas of the method.

The problem is to solve the linear system of equations

$$A\,x = b \tag{A1}$$

where A is an $m \times n$ real matrix with $m \geq n$, the vector x is an unknown n-vector, and b is a given m-vector. The classical method of solution is to form and solve the normal equations

$$A^t\,A\,x = A^t\,b \tag{A2}$$

where $A^t\,A$ is an $n \times n$ symmetric matrix.

To solve this equation, one can apply the well-known diagonalization algorithms such as the Gaussian elimination method and the QR decomposition. There is, however, a serious fundamental disadvantage in using the normal equations. The computation of $A^t\,A$ involves excessive and unnecessary loss of significant figures, thus numerical inaccuracy, because of its ill-conditioned nature.

The most reliable method for computing the solutions of least squares problems is based on a matrix factorization known as the singular value decomposition (SVD). Although there are other methods which require less computer time and storage, they are less effective in dealing with errors and linear dependence [15].

The SVD is based on the following theorem. Any $m \times n$ matrix A with $m \geq n$ can be factored into the form

$$A = U D V^t \tag{A3}$$

where

$$U^t U = V^t V = V V^t = I \text{ and } D = diag(d_1, \ldots, d_n). \tag{A4}$$

The diagonal elements of D are the non-negative square roots of the eigenvalues of $A^t A$; they are called 'singular values'.

The SVD is a little more complicated factorization than that given by Gaussian elimination or by the QR decomposition but it has a number of advantages. The idea of the SVD is that by proper choice of orthogonal matrices U and V, it is possible to make D diagonal with non-negative diagonal entries. The orthogonal matrices have highly desirable computational properties; they do not magnify errors because their norm is unity.

In the least squares problem for the solution of $Ax=b$, we require the vector x for which $norm(b-Ax)$ is a minimum. Then

$$norm(b-A\,x) = norm(U^t(b-A\,x)) = norm(U^t\,b - D\,V^t\,x)$$

$$= norm(c-D\,y) \tag{A5}$$

where

$$c = U^t\,b \quad \text{and} \quad y = V^t\,x\,. \tag{A6}$$

Since D is a diagonal matrix with elements d_i,

$$norm(c-D\,y) = \sum_i (c_i - d_i y_i)^2 + \sum_i c_i^2\,. \tag{A7}$$

Therefore the minimal solution is

$$y_i = c_i/d_i \text{ for } i = 1, \ldots, n; \quad y_i = 0 \text{ for } i = n+1, \ldots, m\,. \tag{A8}$$

The most fundamental question in the least squares problem is the reliability of the computed solution x. In other words, how can we measure

the sensitivity of x to changes in A and b? The answer can be obtained by the idea of matrix singularity. If matrix A is nearly singular, we can expect small changes in A and b to cause very large changes in x. Since the singular values d_i of A are sensitive to perturbations in the values of A, we can determine the measure of nearness to singularity by defining the 'condition number' of A to be

$$cond(A) = d_{max}/d_{min} \tag{A9}$$

where

$$d_{max} = max\{d_i\} \text{ and } d_{min} = min\{d_i\} . \tag{A10}$$

That is, the condition number of a matrix is defined to be the ratio of its largest and smallest singular values.

If $cond(A) = 1$, then the columns of A are perfectly independent. If $cond(A)$ is very large, then the columns of A are nearly dependent and the matrix A is close to singular. The condition number can also be interpreted as a relative error magnification factor in computation.

References

[1] Aggarwal, J. K., Davis, L. S., Martin, W. N., and Roach, J. W.
Survey: Representation methods for three-dimensional objects.
In L. N. Kanal and A. Rosenfeld (editor), *Progress in Pattern Recognition*, pages 377-391. North-Holland Pub., Amsterdam, 1981.

[2] Agin, G. J. and Binford, T. O.
Computer description of curved objects.
In *Proceedings of the Third International Joint Conference on Artificial Intelligence*, pages 629-640. August, 1973.
Also appeared in IEEE Transactions on Computers, 1976, Vol. C-25, pages 439-449.

[3] Altschuler, M. D., Altschuler, B. R., and Taboada, J.
Laser electro-optic system for rapid three-dimensional topographic mapping of surfaces.
Optical Engineering 20:953-961, 1981.

[4] Altschuler, M. D., Posdamer, J. L., Frieder, G., Altschuler, B. R., and Taboada, J.
The numerical stereo camera.
In B. R. Altschuler (editor), *3-D Machine Perception*, pages 15-24. SPIE, 1981.

[5] Altschuler, M. D., Posdamer, J. L., Frieder, G., Manthey, M. J., Altschuler, B. R., and Taboada, J.
A medium-range vision aid for the blind.
In *Proc. of the Int. Conf. on Cybernetics and Society, IEEE*, pages 1000-1002. 1980.

[6] Artzy, E., Frieder, G., and Herman, G. T.
The theory, design, implementation, and evaluation of a three-dimensional surface detection algorithm.
Computer Graphics and Image Processing 15:1-24, 1981.

[7] Ballard, D. H. and Brown, C. M.
Computer Vision.
Prentice-Hall, NJ, 1982.

[8] Bleha, W. P., Lipton, L. T., Wiener-Avnear, E., Grinberg, J., Reif, P. G., Casasent, D., Brown, H. B., and Markevitch, B. V.
Application of the liquid crystal light valve to real-time optical data processing.
Optical Engineering 17:371-384, 1978.

[9] Cornelius, J. Gheluwe, B. V., Nyssen, M., and DenBerghe, F. V.
A photographic method for the 3-D reconstruction of the human thorax.
In *Applications of Human Biostereometrics*, pages 294-300. SPIE, 1978.

[10] Cutchen, J. T., Harris, Jr., J. O., and Laguna, G. R.
PLZT electro-optic shutters: Applications.
Applied Optics 14:1866-1873, 1975.

[11] Dijak, J. T.
Precise three-dimensional calibration of numerical stereo camera systems for fixed and rotatable scenes.
Technical Report AFWAL-TR-84-1105, Air Force Wright Aeronautical Labs. (AAAT-3), September, 1984.

[12] Dijak, J. T.
System software for the numerical stereo camera.
Technical Report AFWAL-TR-85-1078, Air Force Wright Aeronautical Labs. (AAAT-3), 1985.

[13] Dodd, G. G. and Rossol, L. (eds.).
Computer Vision and Sensor-Based Robots.
Plenum Press, New York, 1979.

[14] Duda, R. O. and Hart, P. E.
Pattern Classification and Scene Analysis.
Wiley, New York, 1973.

[15] Forsythe, G. E., Malcolm, M., and Moler, G. B.
Computer Methods for Mathematical Computations.
Prentice-Hall, 1977.

[16] Gennery, D. B.
A stereo vision system for an autonomous vehicle.
In *5th Int. Joint Conf. on AI*, pages 576-582. 1977.

[17] Helmering, R. J.
A general sequential algorithm for photogrammetric on-line processing.
Photogrammetric Engineering and Remote Sensing 43:469-474, 1977.

[18] Horn, B.K.P.
Obtaining shape from shading information.
In P.H. Winston (editor), *The Psychology of Computer Vision*, pages 115-156. Mc Graw-Hill, New York, 1975.

[19] Hugg, J. E.
A portable dual camera system for biostereometrics.
In *Biostereometrics '74*, pages 120-127. American Society of Photogrammetry, Falls Church, VA, 1974.

[20] Kanade, T. and Asada, H.
Noncontact visual three-dimensional ranging devices.
In B. R. Altschuler (editor), *3-D Machine Perception*, pages 48-53. SPIE, 1981.

[21] Karara, H. M., Carbonnell, M., Faig, W., Ghosh, S. K., Herron, R. E., Kratky, V., Mikhail, E. M., Moffitt, F. H., Takasaki, H., Veress, S. A.
Non-topographic photogrammetry.
Manual of Photogrammetry.
Amer. Soc. of Photogrammetry, Falls Church, VA, 1980, pages 785-882.

[22] Land, C. E.
Optical information storage and spatial light modulation in PLZT ceramics.
Optical Engineering 17:317-326, 1978.

[23] LaPrade, G. L., Briggs, S. J., Farrel, R. J., Leonardo, E. S.
Stereoscopy.
Manual of Photogrammetry.
Amer. Soc. of Photogrammetry, Falls Church, VA, 1980, pages 519-544.

[24] Lewis, R. A. and Johnston, A. R.
A scanning laser rangefinder for a robotic vehicle.
Proc. 5th Int. Joint Conf. AI :762-768, 1977.

[25] Light, D. L., Brown, D., Colvocoresses, A. P., Doyle, F. J., Davies, M., Ellasal, A., Junkins, J. L., Manent, J. R., McKenney, A., Undrejka, R., and Wood, G.
Satellite photogrammetry.
Manual of Photogrammetry.
Amer. Soc. of Photogrammetry, Falls Church, VA, 1980, pages 883-977.

[26] Mero, L. and Vamos, T.
Medium level vision.
Progress in Pattern Recognition.
North-Holland Publ., Amsterdam, 1981, pages 93-122.

[27] Montgomery, W. D.
Sampling in imaging systems.
Journal of the Optical Society 65:700-706, 1975.

[28] Nevatia, R.
Machine Perception.
Prentice-Hall, NJ, 1982.

[29] O'Rourke, J.
Polyhedra of minimal area as 3-D object models.
Proc. Int. Joint Conf. on AI :664-666, 1981.

[30] Oliver, D. S.
Real-time spatial modulators for optical/digital processing systems.
Optical Engineering 17:288-294, 1978.

[31] Pavlidis, T.
Algorithms for Graphics and Image Processing.
Computer Science Press, Rockville, MD, 1982.

[32] Popplestone, R. J., Brown, C. M., Ambler, A. P., and Crawford, G. F.
Forming models of plane-and-cylinder faceted bodies from light stripes.
In *Proc. 4th Int. Joint Conf. AI*, pages 664-668. Sept., 1975.

[33] Posdamer, J. L. and Altschuler, M. D.
Surface measurement by space-encoded projected beam systems.
Computer Graphics and Image Processing 18:1-17, 1982.

[34] Rocker, F. and Kiessling, A.
Methods for analyzing three dimensional scenes.
Proc. 4th IJCAI:669-673, September, 1975.

[35] Rogers, D. F. and Adams, J. A.
Mathematical Elements for Computer Graphics.
McGraw-Hill, 1976.

[36] Ryan, T. W. and Hunt, B. R.
Recognition of stereo-image cross-correlation errors.
Progress in Pattern Recognition.
North-Holland Publ., Amsterdam, 1981, pages 265-322.

[37] Shapira, R. and Freeman, H.
Reconstruction of curved-surface bodies from a set of imperfect projections.
5th Int. Joint Conf. on AI:628-634, 1977.

[38] Shirai, Y. and Suwa, M.
Recognition of polyhedrons with a range finder.
In *Proceedings of the Second International Joint Conference on Artificial Intelligence*, pages 80-87. 1971.

[39] Shirai, Y.
Three-dimensional computer vision.
Computer Vision and Sensor-Based Robots.
Plenum Press, NY, 1979, pages 187-205.

[40] Shirai, Y. and Tsuji, S.
Extraction of the line drawings of 3-dimensional objects by sequential illumination from several directions.
2nd Int. Joint Conf. on AI :71-79, 1971.

[41] Slama, C. C. (ed.).
Manual of Photogrammetry
4th edition, American Soc. of Photogrammetry, 1980.

[42] Sobel, I.
On calibrating computer controlled cameras for perceiving 3-D scenes.
3rd Int. Joint Conf. on AI :648-657, 1973.

[43] Strand, T. C.
Optical three-dimensional sensing for machine vision.
Optical Engineering 24:33-40, 1985.

[44] Sutherland, I. E.
Three-dimensional input by tablet.
Proc. IEEE 62:453-461, 1974.

[45] Tamburino, L. A.
Complementary pair detection algorithm and improved noise immunity for numerical stereo camera systems.
Technical Report AFWAL-TR-83-1117, Air Force Wright Aeronautical Labs., Wright-Patterson AFB (AAAT-3), August, 1983.

[46] Tamburino, L. A. and Dijak, J. T.
Error correction in the numerical stereo camera system.
Technical Report AFWAL-TR-85-1077, Air Force Wright Aeronautical Labs., Wright-Patterson AFB (AAAT-3), 1985.

[47] Thomason, M. G. and Gonzalez, R. C.
Database representations in hierarchical scene analysis.
Progress in Pattern Recognition.
North-Holland Publ., Amsterdam, 1981, pages 57-91.

[48] Tomita, F. and Kanade, T.
A 3-D vision system: Generating and matching shape description.
First Conf. on AI Applications, IEEE :186-191, 1984.

[49] Uselton, S. P.
Surface reconstruction from limited information.
PhD thesis, Univ. of Texas at Dallas, 1981.

[50] Will, P. M. and Pennington, K. S.
Grid coding: A preprocessing technique for robot and machine vision.
2nd Int. Joint Conf. on AI 2:319-329, 1971.

[51] Woodham, R. J.
A cooperative algorithm for determining surface orientation from a single view.
5th Int. Joint Conf. on AI :635-641, 1977.

[52] Yang, H. S., Boyer, K. L., and Kak, A. C.
Range data extraction and interpretation by structured light.
First Conf. on AI Applications, IEEE :199-205, 1984.

[53] Young, J. M. and Altschuler, B. R.
Topographic mapping of oral structures -- problems and applications in prosthodontics.
In B. R. Altschuler (editor), *3-D Machine Perception*, pages 70-77. SPIE, 1981.

A Noncontact Optical Proximity Sensor for Measuring Surface Shape

Takeo Kanade and Michael Fuhrman

Computer Science Dept. and Robotics Institute
Carnegie-Mellon University
Pittsburgh, Pa. 15213

and

Equipment Development Division
Alcoa Laboratories
Alcoa Center, PA 15069

Abstract

We have developed a noncontact multi-light source optical proximity sensor that can measure the distance, orientation, and curvature of a surface. Beams of light are sequentially focused from light emitting diodes onto a target surface. An analog light sensor - a planar PIN diode - localizes the position of the resultant light spot in the field of view of the sensor. The 3-D locations of the light spots are then computed by triangulation. The distance, orientation, and curvature of the target surface is computed by fitting a surface to a set of data points on the surface.

The proximity sensor that has been built uses 18 light sources arranged in 5 conical rings. The sensor has a range of 10 cm where it can measure the distance of approximately 200 discrete points per second with a precision of 0.1mm, and then compute surface orientation with a precision of 1.0°. This sensor may be used by a robotic manipulator to home in on an object and to trace the object's surface.

1. Introduction

Noncontact proximity sensors that can measure the range and shape of a surface have useful robotic applications such as surface inspection, seam or edge tracking, obstacle avoidance, and homing a manipulator to an object. The simplest devices are optical switches that detect the presence of a

nearby surface [3], [11]. In these devices, a beam of light originating from a light source on the sensor illuminates a surface. The closer the surface is to the sensor the greater the intensity of the light reaching the sensor. By assembling the light emitter and the detector on small moving platforms the range of this device can be continuously altered [14]. However, this technique for measuring distance is sensitive to the orientation and reflectivity of a target surface as well as the distance of a surface. One method that was developed to overcome this limitation uses a single detector and several light sources. Distance is measured by comparing the relative intensities of the light reflected from a surface [13].

When light detectors sensitive to light spot position are used in a proximity sensor, distance can be measured by triangulation. The measurement then does not depend on the amount of reflected light reaching the sensor. Linear or planar CCD arrays [5], and linear or planar PIN diodes [1], [2], [6], [9], [8], [12] have been incorporated into proximity sensors. For example, in one application, a mirror deflects a beam of light into the field of view of a sensor. The position of the light spot is detected by a planar PIN diode and then correlated with the mirror position to compute distance and eventually the shape of a weld seam [2].

The proximity sensor that we have developed is configured around a planar PIN diode. Multiple conical arrays of light sources focus beams of light into the center of the field of view of the sensor where the distortion and nonlinearity of the sensor chip are smallest. The proximity sensor uses 18 light sources configured into 5 conical arrays. The sensor has a range of 10 cm where it can measure the distance of approximately 200 discrete points per second with a precision of 0.1 mm, and then compute surface orientation with a precision of 1°.

This paper focuses on the design and performance of the new proximity sensor. How well this type of sensor can measure the orientation of a target surface depends on the number of projected light spots, and how they are

distributed. Hence, the new sensor was designed with the aid of a statistical argument that takes into account the geometry of the sensor to minimize measurement uncertainty. The performance of the sensor was tested by measuring the distance and shape of various target surfaces.

2. Measurement of Distance and Shape

2.1. Overview

The proximity sensor is based on the principle of active illumination and triangulation. Figure 1 shows the basic configuration of the proximity The sensor head in this drawing consists of a ring of light sources, a lens, and a light spot position sensor chip.

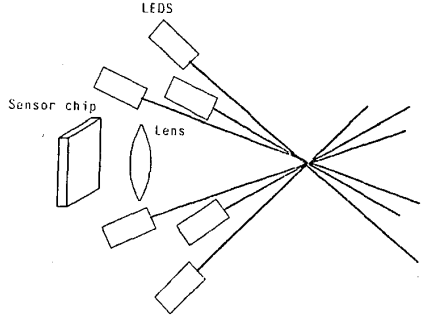


Figure 1: Configuration of the Multi-Light Source Proximity Sensor

The principle of operation is as follows. A beam of light emitted from the sensor head is interrupted by a target surface. The resultant light spot on the target surface is focused by the lens onto the analog light sensor that is sensitive to the intensity and position of a light spot in its field of view. Since the direction of the incident beam of light is known, and the light sensor measures the location of the light spot in the field of view, the three dimensional (3-D) coordinates of the light spot on the target surface can be

computed by triangulation. The proximity sensor has multiple light sources, and as a result the coordinates of several discrete points on the surface can be measured. The average distance, orientation, and curvature of the target surface are computed by fitting a plane and then a quadric surface to this set of points.

Three features of the analog sensor chip make it an attractive device for use in the proximity sensor: its position linearity, its intrinsic ability to measure the centroid of a light spot on its surface, and its speed of response. The chip, a planar PIN diode, measures both the intensity and the position of the light spot on its surface. If the spot of light on a target surface is distorted because of the curvature or orientation of the surface, the sensor chip responds to the stimulus by measuring the coordinates of the centroid of the image of the light spot. To minimize error when computing the coordinates of the light spot, the spot size must be kept small. Therefore, light emitting diodes (LEDs) are used that have a light source diameter of about 150 μm.

2.2. Measuring the Distance of a Surface

The coordinates of a light spot on a target surface are computed by triangulation. The trajectory of each light beam that illuminates a surface is fixed and is measured in advance. The line of sight to the light spot on the target surface is obtained from the sensor chip. The calculation of the coordinates of the light spot is based on a simple optical model: first, the light source beam is a line, and thus the projected light spot is a point; second, the camera lens forms a flat, undistorted image; and finally the sensor chip is linear in finding the light spot position and is insensitive to defocus. Since the sensor is subject to small errors because of deviations from the simple model, they are treated as perturbations from the model and error corrections are calculated beforehand.

The position and orientation of a line in 3-D space can be defined by the

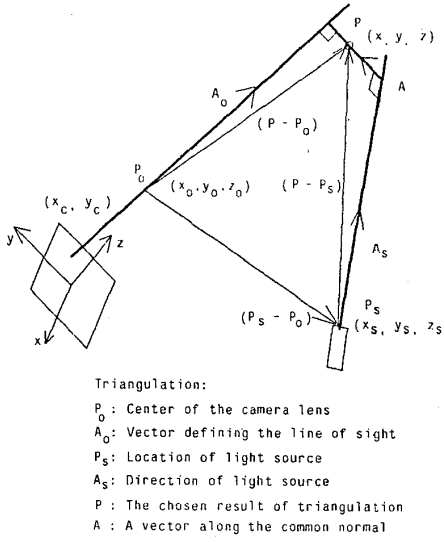


Figure 2: Calculation of Point Coordinates by Triangulation

coordinates of a point on the line and the direction cosines of the line. In Figure 2 light beam trajectory is defined by the light source origin $\mathbf{P}_s = (x_s, y_s, z_s)$ and the light beam direction vector $\mathbf{A}_s = (a_s, b_s, c_s)$. The line of sight has an orientation $\mathbf{A}_o = (a_o, b_o, c_o)$ that is defined by the camera lens center $\mathbf{P}_o = (x_o, y_o, z_o)$ and the centroid of the spot of light $\mathbf{P}_c = (x_c, y_c)$ on the sensor chip. Let us denote the coordinate of the spot position we want to locate as $\mathbf{P} = (x,y,z)$. In principle, the two three-dimensional lines (the light source beam and the line of sight) should intersect at the location of the spot on the surface. In practice, however, the two spatial lines thus obtained only come very close due to noise and the simplified optical model being used. We therefore determine the spot location $\mathbf{P}$ to be located along the line of closest approach (i.e., along the common normal) between the two skew lines. A set of equations from which the coordinates of the light spot

can be calculated is found by observing that the four vectors consisting of the baseline from the center of the camera lens to the light source $(\mathbf{P}_s - \mathbf{P}_o)$, the light beam from the light source to the common normal $(A_s\mathbf{A}_s)$, the common normal $A\mathbf{A}$, and the line of sight from the common normal to the center of the camera lens $(-A_o\mathbf{A}_o)$ sum to zero, that is,

$$(\mathbf{P}_s - \mathbf{P}_o)+(A_s\mathbf{A}_s)+A\mathbf{A}+(-A_o\mathbf{A}_o) = 0. \tag{1}$$

Let the baseline have a length B and a direction specified by the unit vector $\mathbf{B}=(\mathbf{P}_s - \mathbf{P}_o)/|\mathbf{P}_s - \mathbf{P}_o|$. Then Equation (1) can be rewritten as

$$B\mathbf{B}+A_s\mathbf{A}_s+A\mathbf{A}-A_o\mathbf{A}_o = 0 \tag{2}$$

Taking the scalar product of Equation (2) first with the direction vector of the line of sight $\mathbf{A}_o$ and then with the direction vector of the light beam $\mathbf{A}_s$ leads to a set of two simultaneous equations. Noting that $\mathbf{A}_s \cdot \mathbf{A} = 0$ and $\mathbf{A}_o \cdot \mathbf{A} = 0$ yields

$$A_s\mathbf{A}_o \cdot \mathbf{A}_s - A_o = -B\mathbf{A}_o \cdot \mathbf{B} \tag{3}$$

$$A_s - A_o\mathbf{A}_s \cdot \mathbf{A}_o = -B\mathbf{A}_s \cdot \mathbf{B} \tag{4}$$

After solving for A_s and A_o, the coordinates of the light spot can be chosen to be on the light beam or on the line of sight, or at any point along the common normal. Since the output of the sensor chip is subject to noise, and the position and orientation of each beam of light are stable quantities, we choose the computed coordinates of a light spot to lie on the light beam.

2.3. Surface Fitting

To estimate the position and orientation of a surface, a plane is fit to the set of three dimensional data points obtained by the proximity sensor. To estimate the curvature of a surface, a second order equation is fit. The accuracy of the calculation of position, orientation and curvature depends, therefore, on the accuracy of the individual data points, and the number and distribution of the data points on the surface. For example, if the points are too closely spaced, the orientation of the plane that is fit to the points will be very uncertain. These relationships were analyzed, and the results were used to design a sensor whose expected uncertainties are less than the design specifications.

Suppose there are n measured points on a surface, where each point (x_i,y_i,z_i) has an estimated total uncertainty in position of σ_i. A surface can be fit to this set of points using the method of least squares to determine the coefficients a_j in the equation

$$z(x,y) = \Sigma a_j X_j(x,y) \tag{5}$$

where $X_j(x,y)$ is a polynomial of the form $\Sigma c_{qr} x^q y^r$, as for example, when fitting a plane, $q + r \leq 1$, and when fitting a quadric surface, $q + r \leq 2$. We are concerned with how the uncertainty in the measured points and their distribution on the surface affects the uncertainty of the computed coefficients.

The values of the coefficients a_j are chosen so that they minimize the aggregate error E:

$$E = \sum_{i=1}^{n} \{\frac{1}{\sigma_i^2}(z_i - z(x_i,y_i))^2\} = \sum_{i=1}^{n} \{\frac{1}{\sigma_i^2}(z_i - \Sigma a_j X_j(x_i,y_i))^2\} \tag{6}$$

By minimizing E, each a_j can be computed by:

$$a_j = \sum_k \epsilon_{jk} \beta_k \tag{7}$$

where

$$[\epsilon_{jk}] = [\alpha_{jk}]^{-1} \tag{8}$$

$$\alpha_{jk} = \sum_{i=1}^{n} \{\frac{1}{\sigma_i^2} X_j(x_i, y_i) X_k(x_i, y_i)\} = \frac{1}{2} \frac{\partial^2 E}{\partial a_j \partial a_k} \tag{9}$$

$$\beta_k = \sum_{i=1}^{n} \{\frac{1}{\sigma_i^2} z_i X_k(x_i, y_i)\} \tag{10}$$

The symmetric matrix $[\alpha_{jk}]$ is known as the curvature matrix because of its relationship to the curvature of E in the coefficient space [4].

We can estimate the uncertainties of the coefficients, and determine their dependence on the geometry of the proximity sensor if we assume that fluctuations in the coordinates of the individual data points are uncorrelated, and assign the total uncertainty in the measurement of position to the variable z. The uncertainty of the coefficient a_j is then

$$\sigma_{a_j}^2 = \sum_{i=1}^{n} \sigma_i^2 \left(\frac{\partial a_j}{\partial z_i} \right)^2 \tag{11}$$

By calculating the partial derivatives of Equation (7), it follows that the uncertainties of the coefficients are proportional to the diagonal elements of the inverse of the curvature matrix [4], that is

$$\sigma_{a_j}^2 = \epsilon_{jj} \ . \tag{12}$$

Let us first examine the case where the plane

$$z = Gx+Hy+I \tag{13}$$

is fit to a set of data points. Here we have set $a_1 = G$, $a_2 = H$, $a_3 = I$, $X_1 = x$, $X_2 = y$, and $X_3 = 1$ in Equation (5). The local surface normal is $\mathrm{N} = (G, H, -1)$. Since all of the measured points on a surface are at about the same distance from the sensor in any one measurement cycle, the total uncertainty σ_i can be taken as a constant σ for all of the data points. More accurately, since the uncertainty in the location of a light spot depends on where it is in the field of view and the intensity of the measured light, σ is considered an upper limit on the uncertainty in the location of a light spot. The curvature matrix for this case is

$$\alpha = \frac{1}{\sigma^2}\begin{bmatrix} \Sigma x^2{}_i & \Sigma x_i y_i & \Sigma x_i \\ \Sigma x_i y_i & \Sigma y^2{}_i & \Sigma y_i \\ \Sigma x_i & \Sigma y_i & n \end{bmatrix} \tag{14}$$

In many applications the proximity sensor will be used to maintain an orientation normal to a surface. The light sources can be arranged symmetrically so that the resulting spots of light on a flat surface facing the sensor perpendicularly satisfy the condition

$$\Sigma x_i = \Sigma y_i = \Sigma x_i y_i = 0 \tag{15}$$

that is, the α matrix is diagonal. The uncertainties in the coefficients of Equation (13) are then simply

$$\sigma_G = \frac{\sigma}{\sqrt{\Sigma x_i^2}} \tag{16}$$

$$\sigma_H = \frac{\sigma}{\sqrt{\Sigma y_i^2}} \tag{17}$$

$$\sigma_I = \frac{\sigma}{\sqrt{n}} \tag{18}$$

The angle between the optical axis of the proximity sensor and the surface normal is $\phi = \tan^{-1}\sqrt{G^2+H^2}$. The uncertainty in the angular orientation of a surface perpendicular to the sensor can be estimated by

$$\sigma_\phi^2 \leq \sigma_G^2+\sigma_H^2 . \tag{19}$$

Equations (16) through (19) relate the uncertainty in the measured position and orientation of a surface with the number of data points, the uncertainty in their measurement, and their distribution on the surface.

2.4. Normal Curvature

To compute the curvature of a surface and to compute a more accurate value for the distance of a curved target surface from the proximity sensor, the second order equation

$$z = Ax^2+Bxy+Cy^2+Dx+Ey+F \tag{20}$$

of a surface is fit to the set of data points. The curvature of the target surface can be computed at any point from the coefficients of this equation. In particular, the normal curvature, κ_N, and the principal normal curvatures, κ_1 and κ_2, can be computed where the surface intersects the axis of the sensor.

Figure 3 shows the proximity sensor over a target surface. The projected

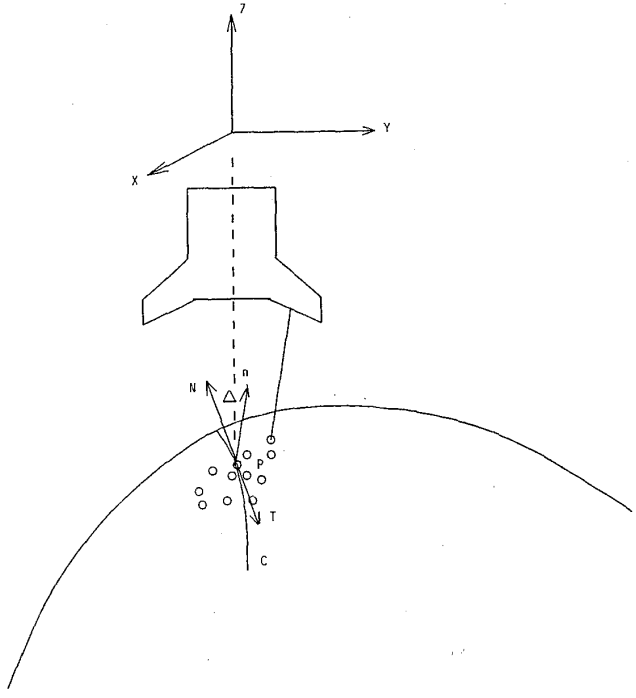


Figure 3: Measuring the Curvature of a Surface

light spots are shown as small circles on the surface. The intersection of the axis of the sensor with the target surface is located at P. The curve C on the surface is the intersection of the target surface with a plane (not shown) containing the axis of the sensor. The vector **n** is the principal normal of the curve C at P; it lies in the intersecting plane. The vector **T** is tangent to the surface, and to the curve C at P. The vector N is normal to the surface at P. The angle between the vectors **n** and N is Δ.

The equation of the fitted surface can be expressed as

$$\mathbf{R} = x\mathbf{i} + y\mathbf{j} + (Ax^2+Bxy+Cy^2+Dx+Ey+F)\mathbf{k}. \tag{21}$$

The surface normal, N, is given by

$$\mathrm{N} = \frac{\mathbf{R}_x \times \mathbf{R}_y}{|\mathbf{R}_x \times \mathbf{R}_y|} = \frac{((2Ax+By+D), -(Bx+2Cy+E), 1)}{\sqrt{(2Ax+By+D)^2 + (Bx+2Cy+E)^2+1}}, \tag{22}$$

where the subscripts x, y denote partial derivatives. The vector, T, tangent to the surface, and to the curve C at P, is given by

$$\mathbf{T} = \frac{d\mathbf{R}}{ds}, \tag{23}$$

where $ds = |d\mathbf{R}|$. The curvature vector, K, whose magnitude is the curvature of C at P is then given by

$$\mathbf{K} = \frac{d\mathbf{T}}{ds}. \tag{24}$$

The normal curvature vector, $\mathbf{K}_N$, is the projection of K onto the surface normal N. The component of $\mathbf{K}_N$ in the direction of N, κ_N, is called the normal curvature of C at P:

$$\mathbf{K}_N = (\mathbf{K}\cdot\mathrm{N})\mathrm{N} = \kappa_N \mathrm{N}. \tag{25}$$

The magnitude of $\mathbf{K}_N$ is the curvature of a curve on the surface, in the plane of N, with tangent vector T at P.

To compute the curvature of a surface, one first computes the first and second fundamental coefficients of the surface [10]. The first fundamental coefficients of the surface are

$$\mathcal{E} = \mathbf{R}_x^{\,2} = 1 + (2Ax+By+D)^2 , \tag{26}$$

$$\mathcal{F} = \mathbf{R}_x \cdot \mathbf{R}_y = (2Ax+By+D)(Bx+2Cy+E) , \tag{27}$$

$$\mathcal{G} = \mathbf{R}_y^{\,2} = 1 + (Bx+2Cy+E)^2 . \tag{28}$$

The second fundamental coefficients of the surface are

$$\mathcal{L} = \mathbf{R}_{xx} \cdot \mathbf{N} = 2A , \tag{29}$$

$$\mathcal{M} = \mathbf{R}_{xy} \cdot \mathbf{N} = B , \tag{30}$$

$$\mathcal{N} = \mathbf{R}_{yy} \cdot \mathbf{N} = 2C . \tag{31}$$

In terms of these coefficients, the normal curvature of the surface is given by

$$\kappa_N = \frac{\mathcal{L}\,dx^2+2\mathcal{M}\,dx\,dy+\mathcal{N}dy^2}{\mathcal{E}dx^2+2\mathcal{F}dx\,dy+\mathcal{G}dy^2} , \tag{32}$$

where the ratio $dx{:}dy$ specifies the direction of a line tangent to the surface at point P. In terms of the coefficients of the second order surface defined by Equation (20), the normal curvature at $x = y = 0$ is given by

$$\kappa_N = \frac{1}{\sqrt{1+E^2+D^2}} \frac{2[A\cos^2\theta+B\sin\theta\cos\theta+C\sin^2\theta]}{[(1+D^2)\cos^2\theta+2DE\sin\theta\cos\theta+(1+E^2)\sin^2\theta]} . \tag{33}$$

Here, θ defines the orientation of the intersecting plane with respect to the x–y axis.

The principal normal curvatures of the target surface where the axis of the sensor intersects the surface, κ_1 and κ_2, are the extreme values of Equation (32). In terms of the fundamental coefficients of the surface, κ_1 and κ_2 are the roots of the equation

$$(\mathcal{EG}-\mathcal{F}^2)\kappa^2-(\mathcal{EN}-2\mathcal{FM}+\mathcal{GL})\kappa+\mathcal{LN}-\mathcal{M}^2=0\,, \tag{34}$$

The directions of the principal axes are given by the roots of the equation

$$(\mathcal{EM}-\mathcal{FL})\tan^2\theta+(\mathcal{EN}-\mathcal{GL})\tan\theta+\mathcal{FN}-\mathcal{GM}=0\,. \tag{35}$$

The principal curvature, κ_n, of the curve C at P is given by

$$\kappa_n=\frac{\kappa_N}{\cos\Delta}=\kappa_N\,\frac{\sqrt{1+D^2+E^2}}{\sqrt{1+D^2}} \tag{36}$$

where κ_N is computed in a direction tangent to C at P, and Δ is the angle between N and n.

3. Measurement Error

The computed coordinates of a light spot in the field of view of the sensor is subject to error. The affects of various sources of error were taken into account in designing the sensor. Features of the sensor include: the beams of light from the sensor are focused into the center of the field of view where non-linearity and distortion are smallest; the light sources are modulated and the resultant signals from the sensor chip are synchronously detected; the resultant light spots on a target surface are small when compared to the area of the field of view of the sensor; and a means for adjusting the light source intensity has been provided so that the measured light intensity is within a small range.

One of the features of the sensor chip is it intrinsically measures the center of gravity of a light spot on its surface. If the shape of a target surface doesn't vary rapidly, the location of the light spot in the field of view of the sensor can be measured accurately. However, if the surface is

concave, light can be reflected about the surface. The center of gravity of the reflected light in the field of view will not coincide with the center of gravity of the projected light spot.

3.1. Distortion

To measure and then compensate for the distortion of the lens and the nonlinearity of the sensor chip, a square array of light spots was generated in the field of view of the sensor. The coordinates output by the sensor chip were then compared with the results computed according to the geometrical model of the sensor. A two dimensional transformation, $\{x_c, y_c\} \rightarrow \{x_c', y_c'\}$, maps the measured sensor chip coordinates into their expected values [7]. This transformation is applied to measured sensor chip coordinates each time a light source is turned on when the proximity sensor is in use.

3.2. Light Source Trajectories

The sensor chip was used to measure the trajectories of the beams of light that are emitted from the sensor head. To do this, a flat surface was mounted perpendicular to the optical axis of the sensor head. The z coordinate of a spot of light on the surface is simply the z position of the surface. As each light source was turned on, the sensor chip coordinates were measured and the x and y coordinates of each light spot were computed. By moving the plane along the z axis of the sensor, a set of points $\{x, y, z\}$ for each light source was acquired.

For each light source, a line was fit through a subset of the acquired data points, $\{x, y, z\}$, near the center of the field of view where distortion and nonlinearity are smallest. These lines were chosen as best estimates of the light beam trajectories. The intersection of a line with the target plane yields the coordinates of a set of points $\{\bar{x}, \bar{y}, \bar{z}\}$. These points are considered to be the best estimate of points along the trajectory of a light beam.

3.3. Error Correction

Just as the set of points $\{\bar{x}, \bar{y}, \bar{z}\}$ were computed, now a set of points $\{x, y, z\}$ are computed for each LED by triangulation. The error is taken to be the difference between the coordinates $\{x, y, z\}$ computed by triangulation and the coordinates $\{\bar{x}, \bar{y}, \bar{z}\}$ computed according to the geometrical model.

Assuming that $(\bar{x}, \bar{y}, \bar{z})$ is the correct location of a light spot, let us set

$$\bar{x}=x+\epsilon_x\,, \tag{37}$$

$$\bar{y}=y+\epsilon_y\,, \tag{38}$$

$$\bar{z}=z+\epsilon_z\,. \tag{39}$$

It would be difficult to model the effect of the departures from the simplified model precisely to determine the terms ϵ_x, ϵ_y, and ϵ_z. However, since the errors are small and usually repeatable with respect to the measured spot position (x_c,y_c) on the sensor chip, we model them as a whole by means of polynomials, for example, by third order correction polynomials of the form

$$\begin{aligned}\epsilon_t = f_t(x_c,y_c) = {} & a_{t1}x_c^{\,3}+a_{t2}x_c^{\,2}y_c+a_{t3}x_cy_c^{\,2}+a_{t4}y_c^{\,3} \\ & +a_{t5}x_c^{\,2}+a_{t6}x_cy_c+a_{t7}y_c^{\,2} \\ & +a_{t8}x_c+a_{t9}y_c+a_{t10}\end{aligned} \tag{40}$$

where

$$t = x,y,z\,.$$

When the proximity sensor is calibrated, these polynomials are designed for each LED using a calibrated data set. During operation, the values

ϵ_x, ϵ_y, and ϵ_z computed by Equations (40) can be added to the spatial coordinates (x,y,z) computed by triangulation to obtain the corrected location of the spot.

4. Multiple Conical Arrays of Light Sources

Uncertainty in the measurement of surface orientation ultimately depends upon the uncertainty with which the sensor chip measures light spot position. Although the resolution of the sensor chip is potentially high (about 1/5000 across its surface), the precision of a single measurement is limited by several factors. The intensity of the light that reaches the sensor chip depends upon the distance, orientation and reflectivity of the target surface. The shape of a surface also affects the intensity distribution of the light spot on the sensor chip, causing a small error in measurement. However, because the sensor chip is very fast we can use redundant multiple light sources and choose the orientations and positions of the light sources so as to increase the accuracy of the sensor.

When measuring the distance of a surface, the region of greatest interest is generally the center of the field of view of the sensor. The effects of the various distortion and non-linearity are smallest near the optical axis. In addition, even if a spot of light is out of focus or grossly distorted by a surface, errors that are the result of a portion of the light spot falling beyond the edge of the sensor chip will be minimized. For these reasons the proximity sensor is configured using multiple conical arrays of light sources. In this way, several light sources are always focused near the optical axis over the range of the sensor.

Since the sensor chip measures the centroid of the intensity distribution on its surface, the distortion of the spot of light on a flat surface does not lead to an appreciable error. However, a large light spot size does decrease the linear range of the proximity sensor, and certainly will affect the measurement of curved surfaces. To minimize the size of the light spots, the

sensor uses light emitting diodes that were intended for fiber optic communication. These LEDs have a light source diameter of 150 microns. As a result the projected light spots on a target surface are on the order of 350 microns in diameter.

4.1. Placement and Orientation of Conical Arrays

Having decided that all of the light sources are to be arranged in multiple conical arrays, the angles of orientation for these light sources, and then the number of light sources comprising each cone, the number of cones, and the placement of the cones are determined by means of the statistical analysis of errors.

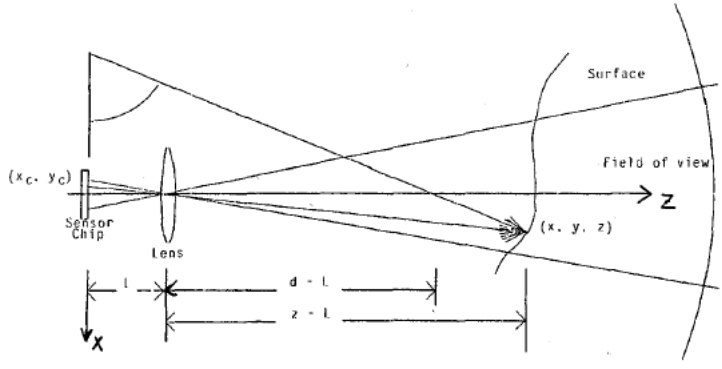


Figure 4: Camera Model of the Proximity Sensor

Figure 4 shows the locations of the sensor chip, the lens, and one light source focused on a point along the optical axis. The angle of the light beam relative to the x axis is θ. A line is drawn from the point on a surface interrupting the light beam, through the effective center of the lens, to the sensor chip. The equation of the beam of light in the x-z plane is

$$z = x \tan \theta + d \,. \tag{41}$$

The x coordinate of the spot of light on the sensor chip is

$$x_c = -\frac{L}{z-L}x\,. \tag{42}$$

A small change in the z coordinate of the target surface by Δz corresponds to a shift in the spot of light on the sensor chip by Δx_c:

$$S = \frac{\Delta x_c}{\Delta z} = -\frac{L}{(z-L)\tan\theta} + \frac{L(z-d)}{(z-L)^2\tan\theta}\,. \tag{43}$$

The larger the magnitude of S, the more sensitive the measurement. The orientation of the light beam θ can be chosen so any measurement of distance using a single light source will have a nominal sensitivity S_N in the center of the field of view, *i.e.*, at $z = d$. This condition places an upper bound on the value of θ:

$$\theta < \tan^{-1}\frac{L}{(d-L)}\frac{1}{S_N}\,. \tag{44}$$

Suppose one particular cone of light is generated using n light sources equally spaced in a circle. When a flat surface intersects the cone of light beams normal to the axis of the cone, the spots of light fall on a circle. The radius R of the circle is given by

$$R = |z-d|\cot\theta\,. \tag{45}$$

To determine the uncertainty in the orientation of a plane that is fit to these points, the mean-square values of the coordinates are first calculated. For $n \geq 3$:

$$\sum_{i=1}^{n} x_i^2 = R^2 \sum_{i=1}^{n} \cos^2 \frac{2\pi i}{n} = \frac{nR^2}{2}, \tag{46}$$

$$\sum_{i=1}^{n} y_i^2 = R^2 \sum_{i=1}^{n} \sin^2 \frac{2\pi i}{n} = \frac{nR^2}{2}. \tag{47}$$

Using Equations (16), (17), (18), and (19), the uncertainty in measured orientation when using a single cone of light can be calculated in terms of the uncertainty σ in the measurement of spot position, the radius of the circle of points, and the number of light sources:

$$\sigma_\phi \leq \frac{\sigma}{R} \frac{2}{\sqrt{n}}. \tag{48}$$

When using multiple cones of light, where the jth cone consists of n_j light sources focused towards a point d_j, the radius of the jth cone at z is

$$R_j = |z - d_j| \cot \theta_j. \tag{49}$$

The mean square value of the x coordinates of the data points is then

$$\sum_i x_i^2 = \frac{1}{2} \sum_j n_j \cot^2 \theta_j (z - d_j)^2. \tag{50}$$

The uncertainty in measured orientation becomes

$$\sigma_\phi \leq \frac{2\sigma}{\sqrt{\Sigma n_j \cot^2 \theta_j (z - d_j)^2}}. \tag{51}$$

This equation guides the determination of the number of light sources n_j in each cone, their orientation θ_j, and their placement d_j.

4.2. Design of the Proximity Sensor

Several goals guided the design of the proximity sensor: the sensor should be compact, with dimensions on the order of 10 cm, in order that it can be used by a robotic manipulator; the sensor should have a useful range of about 10 cm; the incident angle of the light beams on a target surface should be small to minimize light spot distortion; there should be at least three separate conical rings of light sources; and each individual light source should measure position with a sensitivity of at least 0.25 mm.

The primary constraint on the design of the proximity sensor was the light source intensity. The intensity of the light that is incident on a target surface depends on the intensity of the light emitter, the distance of the emitter from its focusing lens, and the diameter and focal length of the focusing lens. These last two quantities determine the size of the sensor and the number of light sources that can be mounted on the sensor head. The intensity of the reflected light that reaches the sensor chip depends on the distance of the camera lens from the target surface and the diameter of the lens.

In a tradeoff between sensitivity, range, and the size of the field of view, the distance from the camera lens to the plane that is in best focus was chosen to be 3.42 times the distance from the sensor chip to the lens. As a result, the area of the field of view that is in best focus is approximately 4.0 cm square. In the version of the sensor that has been built we use a 16mm, f/1.6 camera lens. If a camera lens with a shorter focal length were used, a smaller sensor could have been built. This sensor would focus over a closer range and have a wider field of view. However, the sensitivity of the sensor would then be decreased at longer distances as the field of view of the sensor rapidly expanded. If we used a longer focal length camera lens the angular field of view would be smaller, and the sensitivity greater, but it is difficult to focus small spots of light with sufficient intensity at the greater distance that would then be in focus.

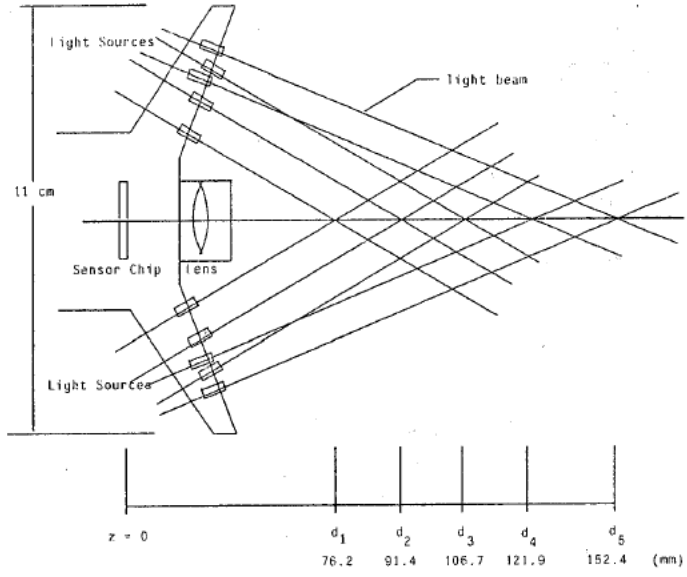


Figure 5: Geometry of the New Proximity Sensor

Figure 5 shows the configuration of the proximity sensor that has been built. To begin the analysis, we choose three cones of light each consisting of a three light sources. Equation (44) determined the choice of the angle of orientation of the light sources. With this angle chosen, we use Equation (51) to calculate the minimum spacing between the vertices d_1, d_2 and d_3 of the cones of light, where $z = d_2$ is in best focus.

First we have to estimate the uncertainty σ in the measurement of spot position at $z = d_2$. Referring to Equation (43) and Equation (44), we first set $\theta = 60^\circ$, and $(d_2 - L) \,/\, L = 3.42$ and find $S_N = 6.67$. A smaller angle would increase the sensitivity of the sensor, but the light spot on a surface would then be further elongated. We assume that when a flat surface is measured, a signal to noise ratio can be achieved so that the effective resolution of the sensor chip is at least 1/500 its linear dimensions. This

corresponds to $\Delta x_c = 0.026$ mm , and a shift in the position of the target surface by $\Delta z = 0.15$ mm . This value of Δz is used to estimate σ_z. The total uncertainty σ is estimated by observing that when the position of a point is calculated by triangulation, the solution is constrained to lie on the line determined by the light beam. Since $\sigma^2 \approx \sigma_x^2 + \sigma_y^2 + \sigma_z^2$, and $\tan 60^o = \sqrt{3}$,

$$\sigma_z^2 \approx 3(\sigma_x^2 + \sigma_y^2) \tag{52}$$

and therefore,

$$\sigma^2 \approx 1.33\sigma_z^2. \tag{53}$$

The maximum value of the uncertainty in the measurement of surface orientation σ_ϕ occurs near the vertex of the middle cone. We use Equation (51) to calculate the minimum spacing between the vertices d_1, d_2 and d_3 of the cones of light by insisting the uncertainty in any single measurement of orientation at $z = d_2$ be less than 1.0^o (17.4 milliradians). Assuming these three cones are equally spaced, and consist of three light sources, we find

$$14.0 \text{ mm} \leq d_2 - d_1 = d_3 - d_2 \tag{54}$$

Figure 6 shows how the uncertainty in orientation σ_ϕ calculated by Equation (51) decreases as cones of light are added to the proximity sensor head. $\sigma_\phi(1)$ is graphed assuming a single cone of light consisting of three light sources is focused toward d_2. Then $\sigma_\phi(2)$ and $\sigma_\phi(3)$ are graphed as successive cones of light are added: first the cone with a vertex at d_3, then the cone with a vertex at d_1.

In order to further increase the accuracy of the sensor a second cone of light is focused at d_1 to augment the original light sources. In order to extend the range of the sensor, two more cones of light with an orientation of

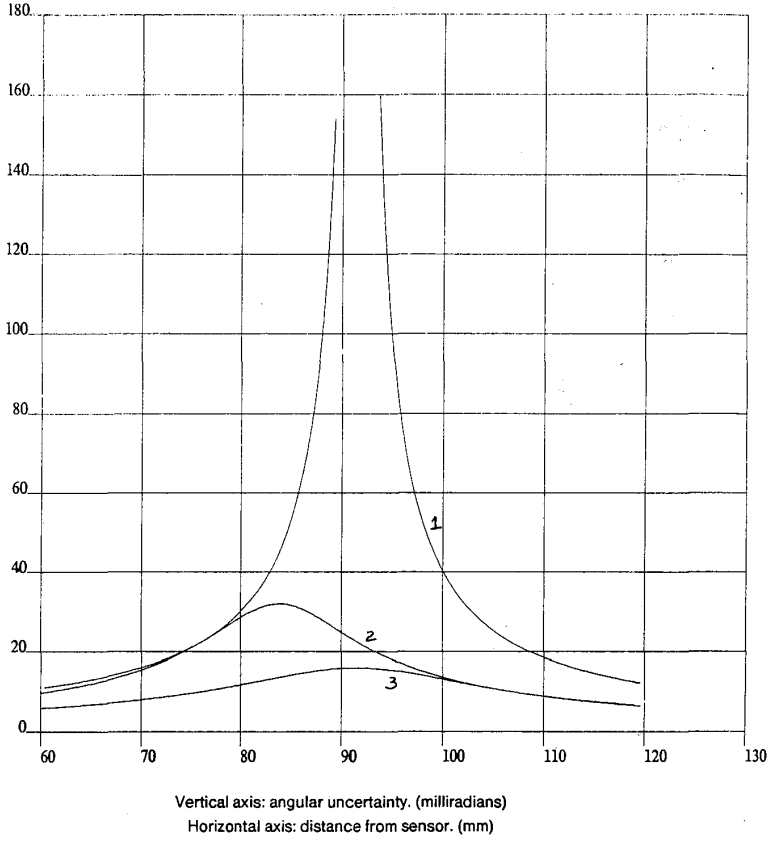


Figure 6: These graphs display the uncertainty when measuring the orientation of a flat target surface. Curves 1, 2, and 3 show the uncertainty when using a single, then two, and then three cones of light, each having three light sources

70 ° are focused towards points further along the optical axis. As a result, the range of the proximity sensor consists of two overlapping regions: first an inner region with a small field of view, a depth of 6 cm, and many redundant light sources where the sensor accurately measures the distance and shape of a surface, and then an outer region that extends the range of

the sensor by another 4 cm . The proximity sensor actually has an extended range of 20 cm. At this distance several of the light beams leave the field of view. However, the center of the field of view is devoid of light spots.

The computed distance and orientation of a target surface are estimated to be the values at the intersection of the surface with the axis of the sensor. If there are data points in this region, the estimate is more reliable than otherwise. The spacing of the vertices of the cones of light, and the orientations of the light sources determine how localized the data points are near the axis of the sensor over the range of the sensor. In the sensor that has been built, a light spot on the target surface is always within 5mm of the axis of the sensor.

The distances where the beams of light enter and leave the field of view of the sensor were computed for each light source. The angular field of view of the sensor depends on the active area of the sensor chip. Table 1 lists the entrance and exit distances for each light source assuming the active area of the sensor chip has a radius of 4 mm. In addition, the table lists the distances where each cone of light intersects the optical axis. From the table, one can see over what distances the cones of light overlap.

	Entrance (mm)	Axis Intercept (mm)	Exit (mm)
Cone 1	62.26	76.2	104.18
Cone 2	73.67	91.4	127.11
Cone 3	85.09	106.7	150.03
Cone 4	86.78	121.9	236.79
Cone 5	106.68	152.4	301.85

Table 1: Entrance and Exit Distances for Each Light Source.

This proximity sensor uses a total of 18 light sources arranged in 5 conical arrays. The diameter of the sensor is 11 cm. Assuming a sufficiently

high signal to noise ratio, the overall accuracy in the measurement of position when using the innermost 12 light sources is expected to be on the order of .1 mm. The accuracy in the measurement of surface orientation is expected to be better than 1°.

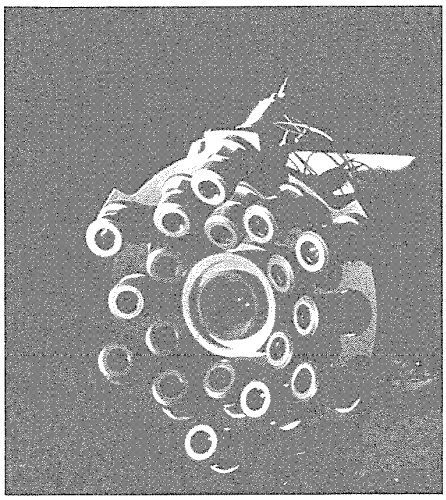


Figure 7: Photograph of the new proximity sensor

The proximity sensor head shown in Figure 7 is composed of a barrel that houses the analog sensor chip, and into which the camera lens and eighteen light sources are fixed. The barrel was machined from a single piece of aluminum to insure the concentricity of the cones of light sources with the optical axis of the camera lens and with the center of the sensor chip. Any

suitable C-mount lens can be mounted in front of the sensor chip. The light sources screw into the flange of the barrel.

To summarize, the final parameters of the proximity sensor are:

Camera lens focal length: $f = 16$ mm
Sensor chip to lens distance: L = 20.67 mm
Distance from sensor chip to plane in best focus: $D = 91.4$ mm
Orientation of inner rings of light sources: $\theta_{1,2,3} = 60^{\circ}$
Orientation of outer rings light sources: $\theta_{4,5} = 70^{\circ}$
Number of light sources in ring 1: $n_1 = 6$
Number of light sources in outer four rings: $n_{2,3,4,5} = 3$
Distance between inner light source target points along the optical axis: 15.24 mm

This particular sensor head weighs approximately one pound. Future versions of the sensor can be constructed of composite materials to decrease the sensor's weight.

5. Performance

5.1. Measuring Distance and Orientation

The proximity sensor was used to measure the distance and orientation of flat target surfaces to determine the accuracy of the sensor. The target surfaces were moved by high resolution linear and rotary platforms in the field of view of the sensor. Each surface was rotated in 2° increments from -16° to $+16^{\circ}$, and moved in 2 mm steps in front of the sensor. The distance and orientation of the surfaces were measured and compared with their expected values.

The results for a uniform white target surface are summarized in Figures 8 and 9. The graphs show the error in the measured distance and orientation of the surface as a function of the actual distance and orientation of the surface. The error in orientation is typically less than 2°, and the error in the measurement of distance is less than 0.2 mm.

The amplifier gains of the proximity sensor were set so that the sensor could measure the distance of a matte surface over a range of 10 cm. The

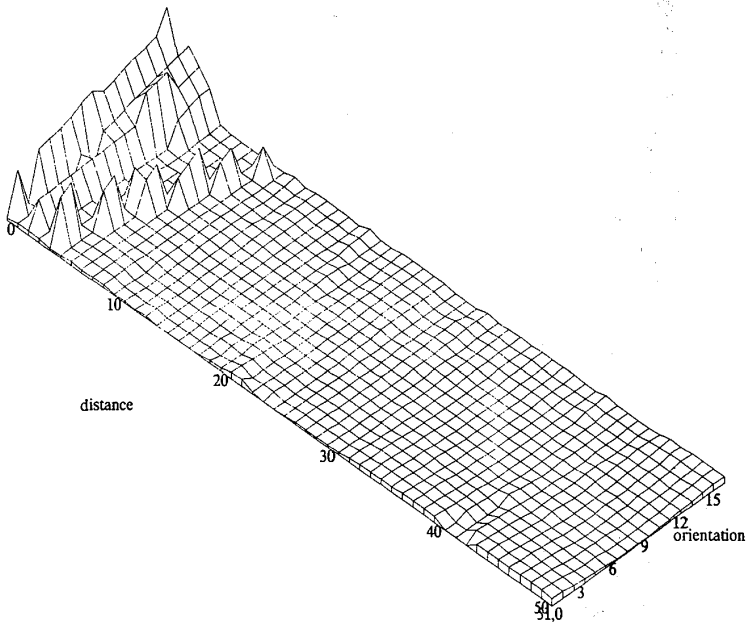


Figure 8: White surface: error in measured distance versus orientation and distance. The error in measured distance is typically less than 0.2mm

light intensity measured by the sensor agreed with the expected result when the white surface was used as a target. However, the reflectivity of an unpainted aluminum surface, for example, is less than that of a white surface, and in addition the reflected light has a large specular component. As a result, the range of the sensor was limited to 5 cm when measuring the distance of this type of surface. Either an insufficient amount of light is scattered by the surface towards the sensor, or alternatively, light is specularly reflected directly towards the sensor and overflows the amplifiers.

The measured coordinates of a light spot on a target surface also has a

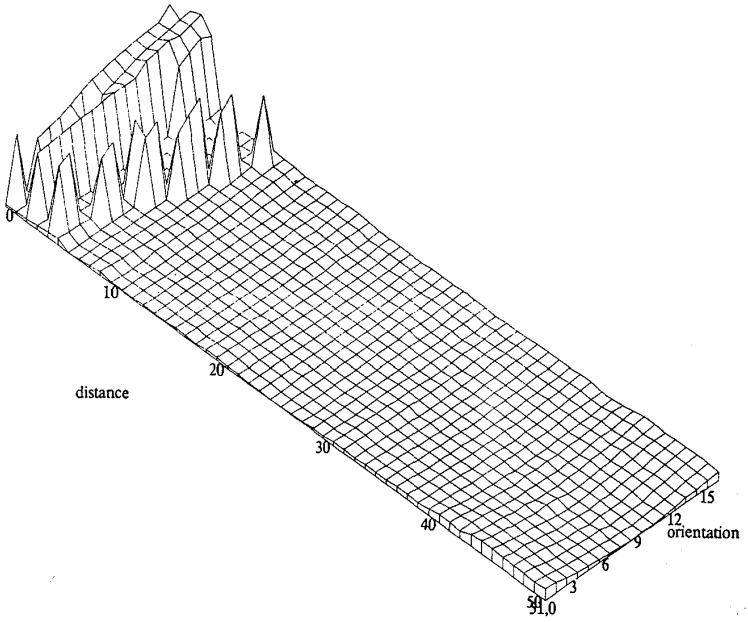
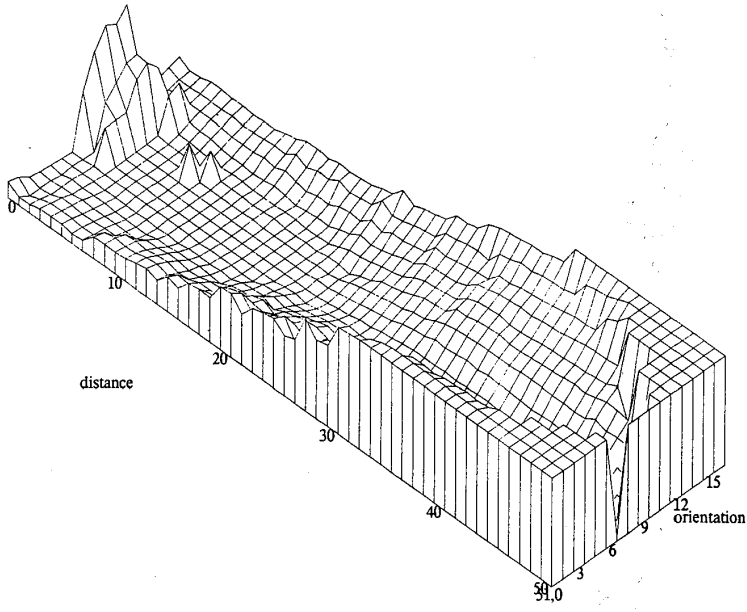


Figure 9: White surface: error in measured orientation versus orientation and distance. The error in measured orientation is typically less than 2^o

small dependence on the intensity of the light reaching the sensor chip. As a result, when the sensor was calibrated for the white target surface and then used to measure the distance and orientation of the other target surfaces the results were in error. However, by recalibrating the proximity sensor for each of these surfaces, (that is, recomputing error correction polynomials), the performance of the sensor was improved so that the measurement error at normal incidence was less than 0.5 mm. Figures 10 through 13 show the errors computed for two target surfaces, a brushed and a tarnished aluminum surface, after recalibrating the sensor for each of these surfaces.



Orientation axis: -16° to 16° in 2° steps.
Distance axis: 66.4 mm (0) to 166.4 mm (50) in 2 mm steps.

Figure 10: Aluminum surface: error in measured distance versus orientation and distance after recalibrating the sensor.

5.2. Curvature

To measure the contour of a surface, a surface was moved through the field of view of the proximity sensor. At each position of the surface, the distance of the surface from the proximity sensor was measured, and the orientation and curvature of the surface were computed. The results were compared with the actual shape of each surface, and with shape of each surface that was computed assuming the proximity sensor behaved ideally.

The measurement of surface shape was simulated to evaluate the expected results. The coordinates of the resultant light spots on a simulated target surface were computed for each position of the surface. The distance,

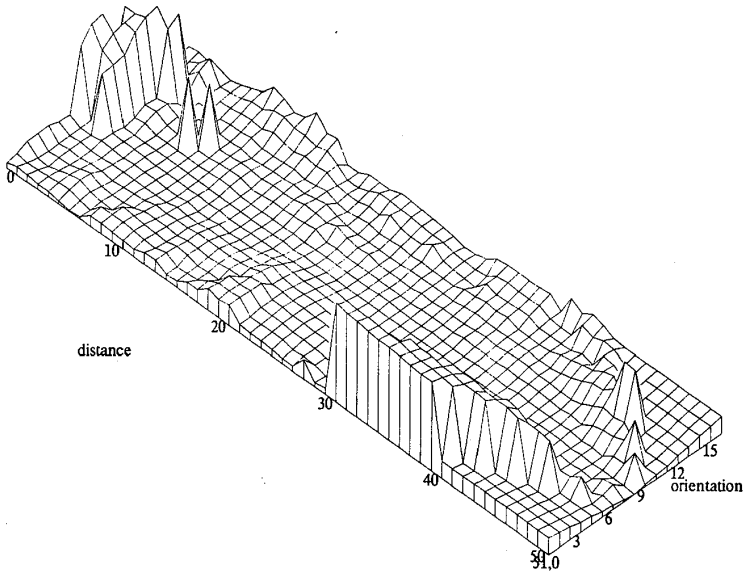
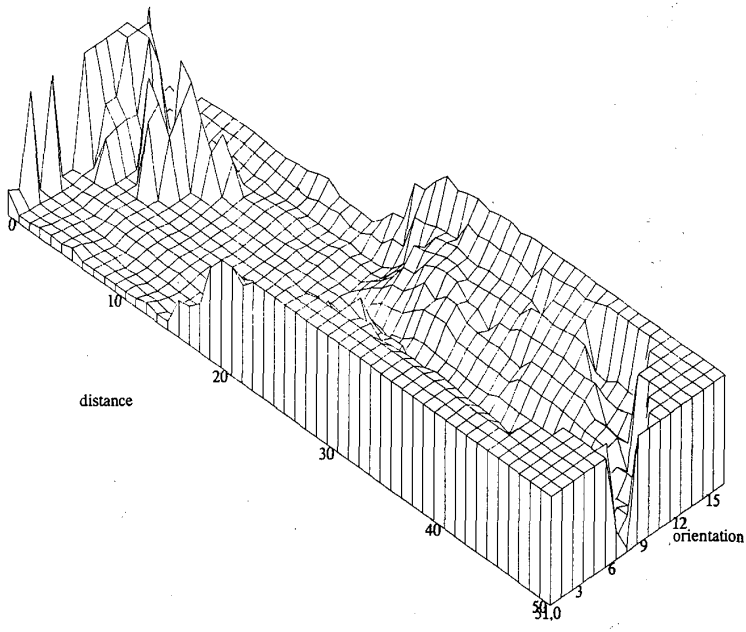


Figure 11: Aluminum surface: error in measured orientation versus orientation and distance after recalibrating the sensor.

orientation and curvature of the simulated target surface were then computed, and the results were compared with the actual measurements.

Since measuring the contour of a surface uses only a small interval of the range of the sensor, relative measurement errors are generally small. For example, when a flat surface with uniform reflectivity was moved across the field of view of the sensor, the error in measured distance was much less than the errors that were reported when the distance and orientation of the target surface were varied over a range of 10 cm. In many applications of the proximity sensor, a target surface will be nearly flat. In addition, the sensor can be moved to maintain a fixed distance and orientation above a target



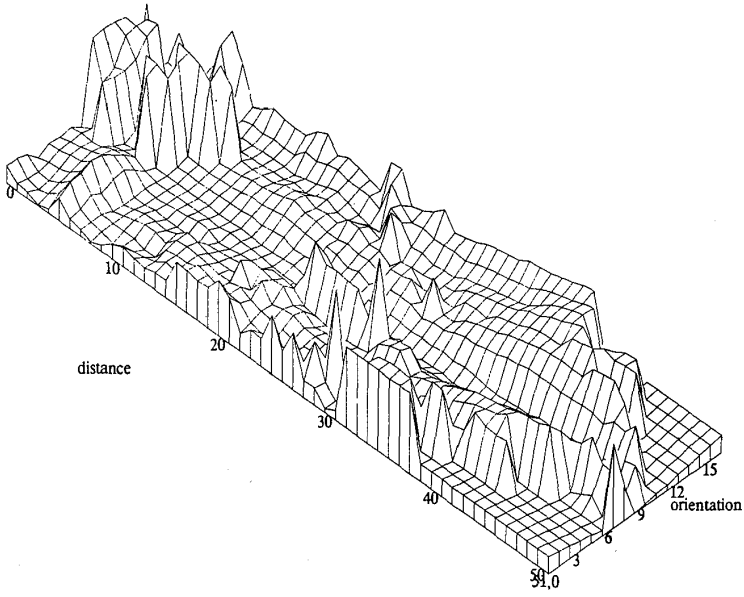
Orientation axis: -16° to 16° in 2° steps.
Distance axis: 66.4 mm (0) to 166.4 mm (50) in 2 mm steps.

Figure 12: Tarnished surface: error in measured distance versus orientation and distance after recalibrating the sensor.

surface to achieve a better measurement.

The shapes of a cylinder and a cone were measured by moving each surface linearly across the field of view of the sensor a distance of 5 cm in 2 mm steps. The cylinder was in focus when closest to the sensor; the cone was positioned 6 mm further away. Both the cylinder and the cone were covered with a sheet of matte white paper. The cylinder had a 57.15 mm (2.25 inch) radius. In Figure 14 the contour of the cylinder is graphed as the sensor was moved across its surface.

The cone that was used as a target had a base angle of 81°. At the height at which the axis of the sensor intersected the surface of the cone, the cone



Orientation axis: -16° to 16° in 2° steps.
Distance axis: 66.4 mm (0) to 166.4 mm (50) in 2 mm steps.

Figure 13: Tarnished surface: error in measured orientation versus orientation and distance after recalibrating the sensor.

had a radius of 45 mm. The distance between the sensor and the cone was 97.4 mm at their closest position. This is 6 mm further away from the sensor than the distance of best focus. In Figure 15 the contour of the cone is graphed as the sensor was moved across its surface. Curve '1' shows the measurement results. Curve '2' shows the expected results that were computed according to the geometrical model of the sensor which assumes, for example, there were no extraneous reflections. Curve '3' shows the actual contour of the cone.

Figures 14 through 17 show that the proximity sensor can accurately measure the distance and orientation of a curved surface when the surface is

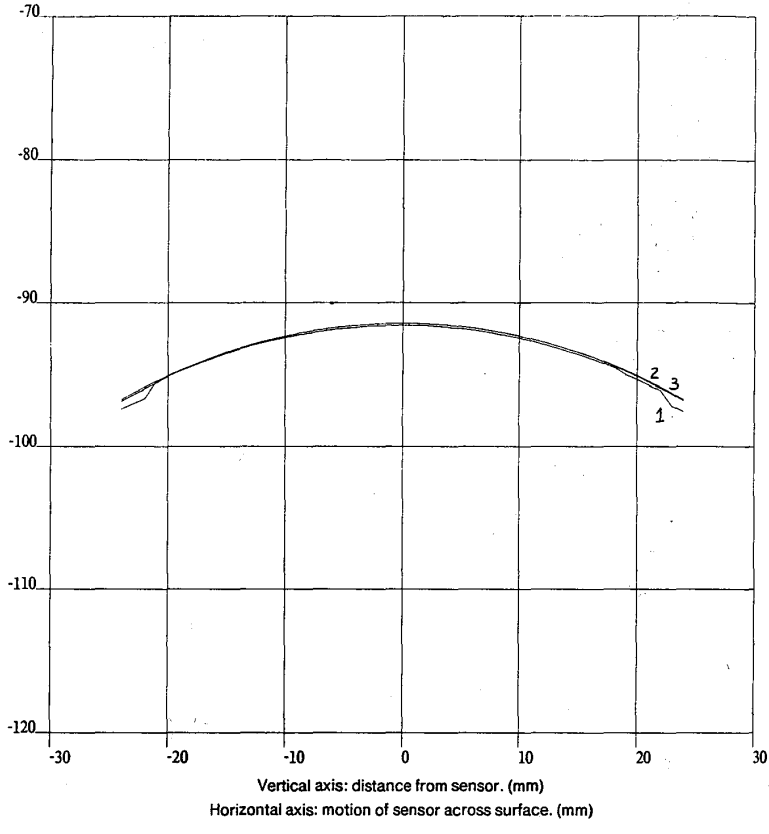


Figure 14: Contour of the surface of a cylinder: the measured 1 and the simulated 2 results, along with the actual 3 values

near normal to the axis of the sensor. When the orientation of each of these surfaces was computed from the coefficients of the second order equation, the results were not accurate. Fitting a quadric surface to the data points on a curved surface provides a better estimate of the range of a surface, but the gradient and the curvature of the surface are not stable. The local orientation of the target surfaces was better represented by fitting a plane to the data points. Figures 16 and 17 show the measurement results and the

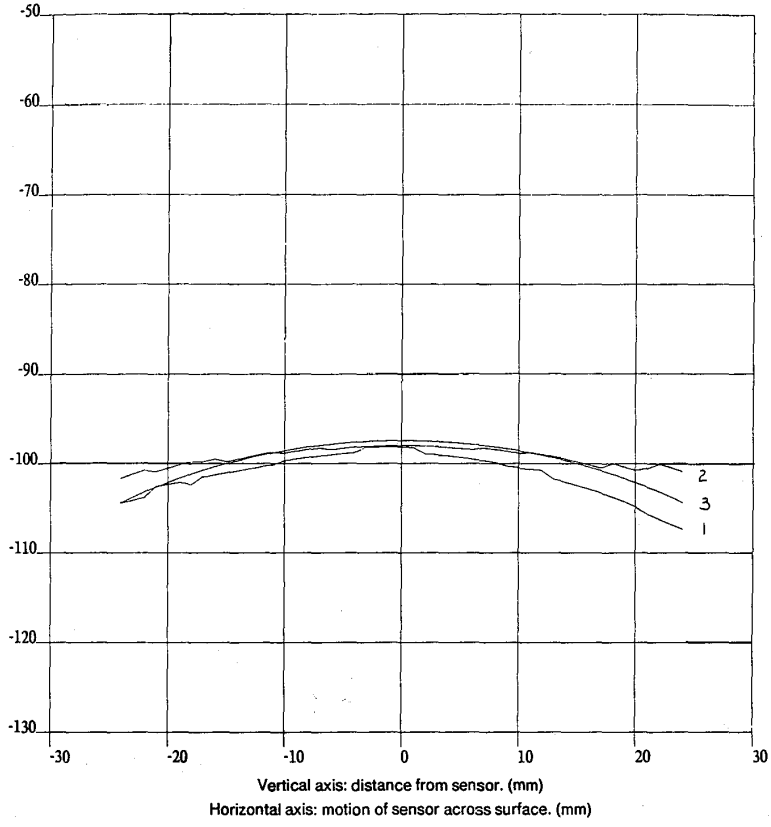


Figure 15: Contour of the surface of a cone: the measured 1 and the simulated 2 results, along with the actual 3 values

simulation results.

The principal normal curvatures, κ_1 and κ_2, of the cylinder and the cone were also computed. Figures 18 and 19 respectively show the smaller value for the computed principal radius of curvature of the cylinder and of the cone. The curves labeled '1' are the measured results and the curves labeled '2' show the simulation results. The solution for the curvature of the cylinder and the cone can be chosen where the other principal radius of

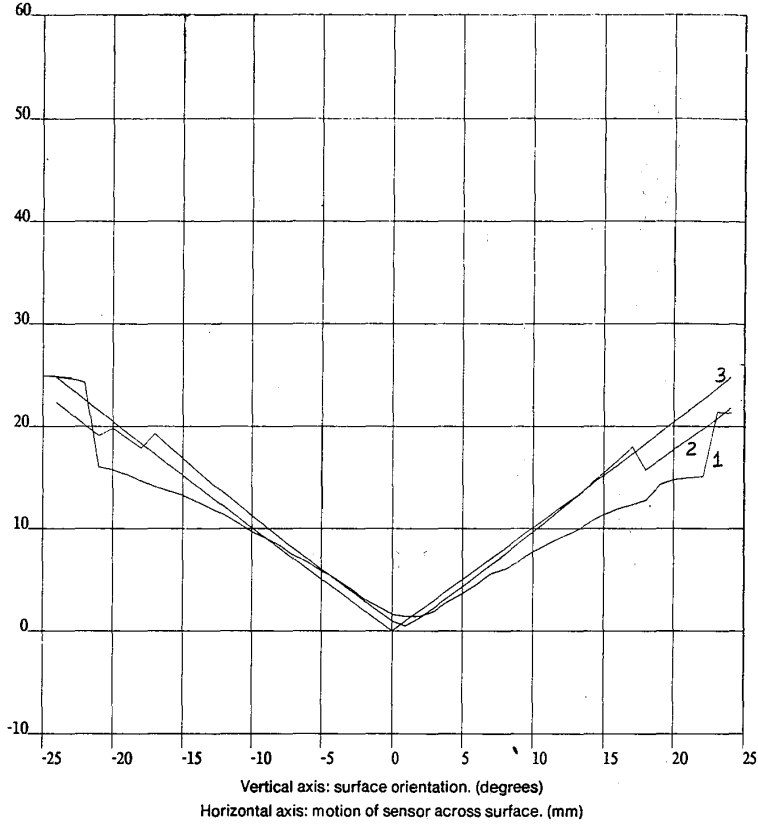


Figure 16: Surface orientation of the cylinder: measured 1 and simulated 2 results, and the actual value 3. The orientation is measured with respect to the axis of the proximity sensor

curvature has a maximum. For the cylinder this occurs where $x = -6$. For the cone this occurs where $x = 2$.

The proximity sensor can accurately measure the distance and orientation of a smooth surface when maintained at a near normal position relative to the surface. According to the simulation results, the proximity sensor has the potential to accurately measure the curvature of a surface.

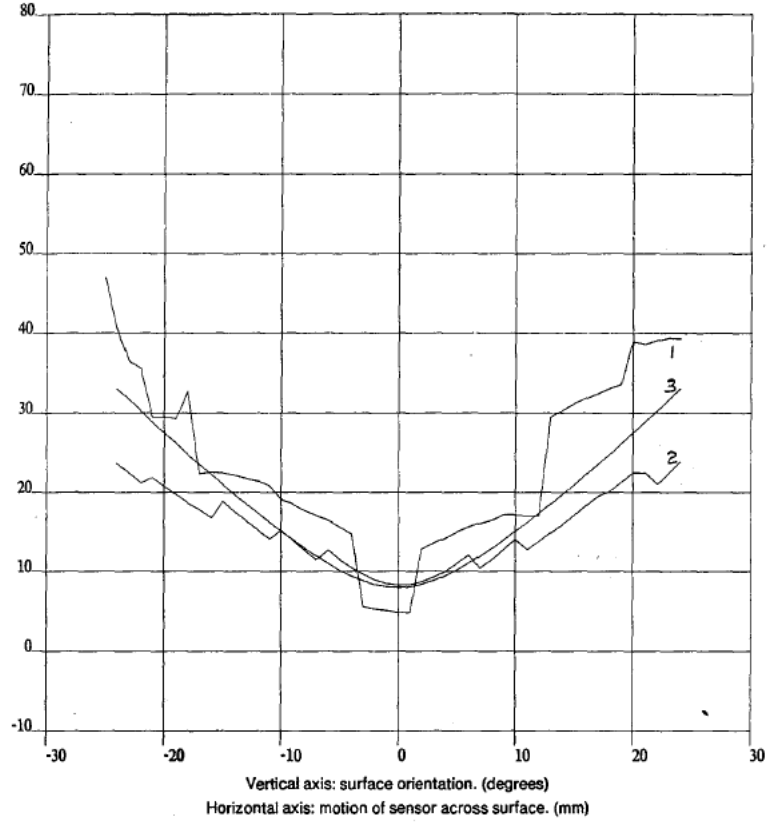


Figure 17: Surface orientation of the cone: measured 1 and simulated 2 results, and the actual value 3. The orientation is measured with respect to the axis of the proximity sensor

The simulated measurement of curvature is stable over a range of orientation and distance relative to a surface. However, the simulation did not take into account specular reflection of light toward the sensor head, finite light spot size and other perturbations. Either increasing the number of light sources, focusing smaller light spots onto the target surface, or using data points from earlier measurements in the computation, should increase

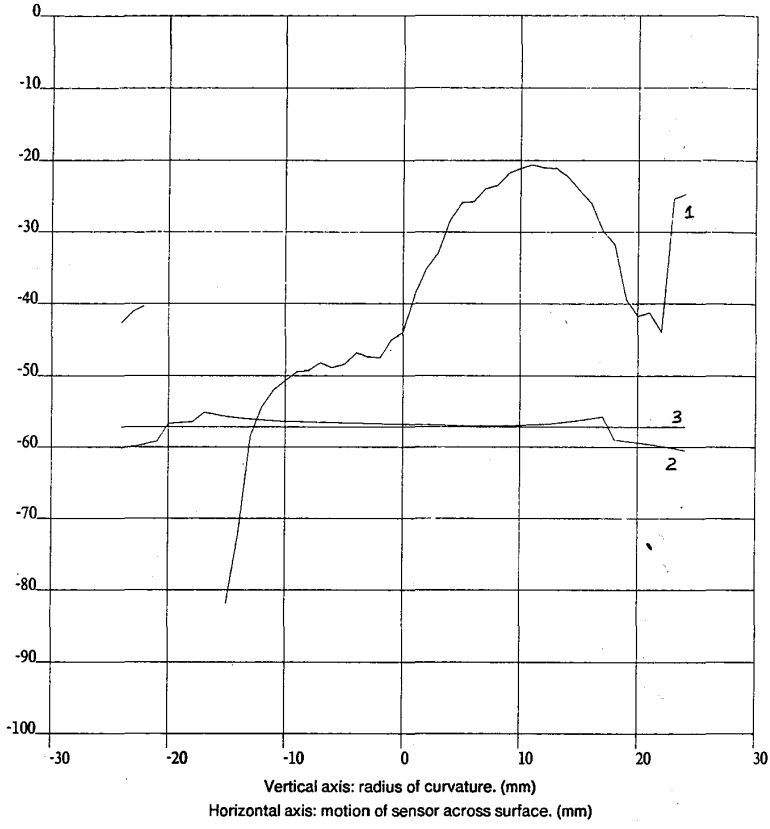


Figure 18: The smaller principal radius of curvature of a cylinder: measured 1 and simulated 2 results, and the actual value 3.

the accuracy of the computed curvature.

6. Conclusion

A new compact multi-light source proximity sensor has been developed. The sensor is based on the principle of active illumination and triangulation: each time a light source is pulsed, the coordinates of the resulting spot of the light on a surface are calculated. The proximity sensor that has been built

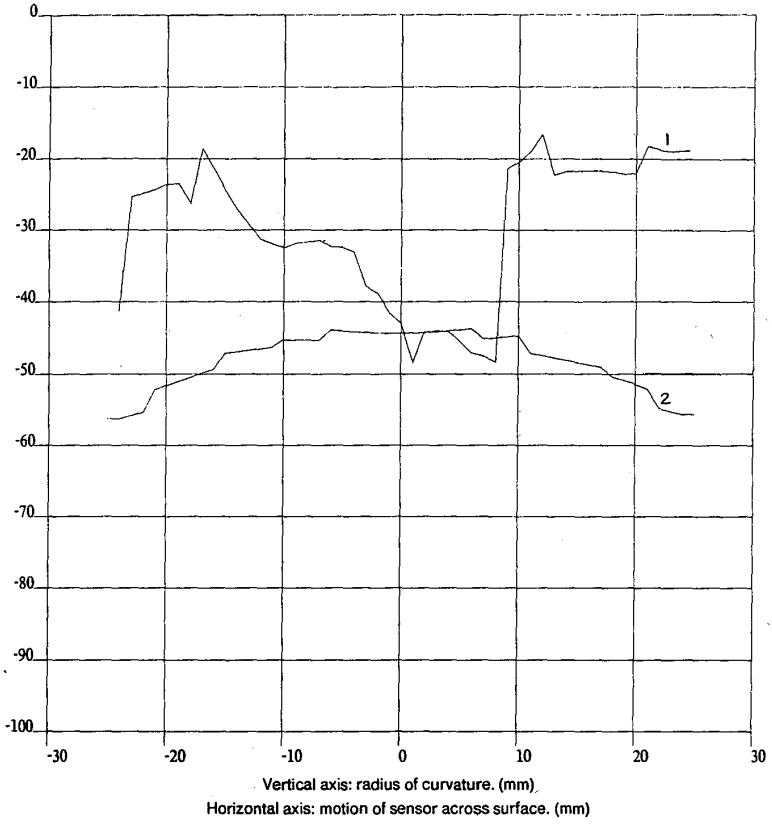


Figure 19: The smaller principal radius of curvature of a cone: measured 1 and simulated 2 results, and the actual value 3.

has a range of 10 cm where it can measure distance with a precision of 0.1 mm and surface orientation with a precision of 1.0°. The sensor can acquire data at a rate exceeding 200 points per second.

The design of the proximity sensor was guided by a statistical analysis of the uncertainty in the measurement of position using a single light source and the measurement of surface orientation using several light sources. The analysis related the number of light sources needed to measure orientation

within a specified accuracy to the distribution of light spots on a target surface. The new design is also intended to be a compact sensor for use on a robotic manipulator.

The proximity sensor is subject to error because of various types of distortion, nonlinearity, and noise. To compensate for the distortion of the sensor chip and imaging optics, a square array of light spots was generated in the field of view. A two dimensional transform was computed that maps the measured image of this array on the sensor chip into its ideal undistorted image. This transform was applied to the sensor chip coordinates during the operation of the sensor.

To further compensate for errors in the computation of distance, error correcting polynomials were computed that map the distance of a surface computed by triangulation into its actual distance. These polynomials were used during the operation of the sensor to improve measurement results.

The features of this proximity sensor include its simple principle of operation and its fast speed. The sensor chip which detects the spot of light on the target surface is an analog device which outputs the position of the centroid of a spot of light on its surface. Although the accuracy of any individual measurement of light spot position is limited by the distortion of a spot of light by a surface, and the noise and sensitivity of the sensor, we take advantage of the speed of the sensor and use multiple light sources to increase the overall accuracy of measurements. The geometrical arrangement of the light sources was guided by the statistical analysis to achieve the design specification for accuracy. Measurement of distance, surface orientation and surface curvature can exploit the geometrical redundancy of the device.

Acknowledgments

We thank Regis Hoffman and Donald Schmitz for useful discussions and help in software and hardware development. This research was partially supported by the Office of Naval Research (ONR) Grant No. N00014-81-K-0503, and by the National Science Foundation Grant No. ECS-8320364.

References

[1] Bamba, T.,Maruyama, H., Kodaira, N. and Tsuda, E.
A Visual Seam Tracking System for Arc Welding Robots.
In *Fourteenth International Symposium on Industrial Robots*. 1984.

[2] Bamba, T.,Maruyama, H., Ohno, E. and Shiga, Y.
A Visual Sensor for Arc Welding Robots.
In *Eleventh International Symposium on Industrial Robots*. 1982.

[3] Bejczy, A.
Smart Sensors for Smart Hands.
Nasa Report Number 78-1714, 1978.

[4] Bevington, Philip R.
Data Reduction and Analysis for the Physical Sciences.
McGraw-Hill Book Company, 1969.

[5] Diffracto Limited.
Diffracto Robotics Vision Sensors.

[6] Fuhrman, M. and Kanade, T.
Design of an Optical Proximity Sensor using Multiple Cones of Light for Measuring Surface Shape.
Technical Report TR-84-17, Carnegie Mellon University Robotics Institute, 1984.

[7] Hayes, J.G.
Numerical Aproximation to Functions and Data.
Athlone Press, University of London, 1970.

[8] Kanade, T. and Sommer, T.
An Optical Proximity Sensor for Measuring Surface Position and Orientation for Robot Manipulation.
Technical Report TR-83-15, Carnegie Mellon University Robotics Institute, 1983.

[9] Kanade, T. and Asada, H.
Noncontact visual three-dimensional ranging devices.
In B. R. Altschuler (editor), *3-D Machine Perception*, pages 48-53. SPIE, 1981.

[10] Lipschutz, M. M.
Schaum's Outline of Theory and Problems of Differential Geometry.
McGraw Hill Book Co., 1969.

[11] Morander, E.
The Optocator. A High Precision, NonContacting System for Dimension and Surface Measurement and Control.
In *Proc. Fifth Intern. Conf. on Automated Inspection and Product Control*, pages 393-396. , 1980.

[12] Nakamura, Y., Hanafusa, H.
A New Optical Proximity Sensor for Three Dimensional Autonomous Trajectory Control of Robot Manipulators.
In *Proceedings: '83 International Conference on Advanced Robotics*. 1983.

[13] Okada, T.
A Short Rangefinding Sensor for Manipulators.
Bulletin of Electrotechnical Laboratory 42, 1978.

[14] Thring, M.W.
Robots and Telechirs.
Ellis Horwood Limited, 1983.

PART II: 3-D FEATURE EXTRACTIONS

Toward a Surface Primal Sketch

Jean Ponce
Michael Brady

Massachusetts Institute of Technology
Artificial Intelligence Laboratory
545 Technology Square
Cambridge, MA 02139

Abstract

This paper reports progress toward the development of a representation of significant surface changes in dense depth maps. We call the representation the Surface Primal Sketch by analogy with representations of intensity changes, image structure, and changes in curvature of planar curves. We describe an implemented program that detects, localizes, and symbolically describes: steps, where the surface height function is discontinuous; roofs, where the surface is continuous but the surface normal is discontinuous; smooth joins, where the surface normal is continuous but a principal curvature is discontinuous and changes sign; and shoulders, which consist of two roofs and correspond to a step viewed obliquely. We illustrate the performance of the program on range maps of objects of varying complexity.

1. Introduction

This paper describes an implemented program that detects, localizes, and symbolically describes surface changes in dense depth maps:

- *steps*, where the surface height function is discontinuous;
- *roofs*, where the surface is continuous but the surface normal is discontinuous;
- *smooth joins*, where the surface normal is continuous but a principal curvature is discontinuous and changes sign; and
- *shoulders*, which consist of two roofs and correspond to a *step* viewed obliquely.

Figures 4, 7, 9, and 10 show the idealized instances of these surface changes that are the basis of the mathematical models used by the program.

The work reported here continues our investigation [3] of surface descriptions based on the concepts of differential geometry. Section 2 summarizes our ideas and shows the kind of geometric (CAD) description we are aiming at. An important component of our work is the identification and isolation of a set of *critical surface curves*, including significant surface changes. To this end, we report progress on the development of a representation we call the *Surface Primal Sketch* by analogy with:

- Marr's *Primal Sketch* [22] representation of significant intensity changes;
- Asada and Brady's *Curvature Primal Sketch* [1] representation of significant curvature changes along planar contours; and
- Haralick, Watson, and Laffey's *Topographic Primal Sketch* [14] representation of image structure.

In each case, there are three distinct problems: to *detect* significant changes; to *localize* those changes as accurately as possible; and to *symbolically describe* those changes. We follow the approach of Asada and Brady [1], as sketched in Section 3. A key component of that approach is scale space filtering, pioneered by Witkin [28]. Yuille and Poggio [30], [31] have proved that, in principle, scale space filtering enables a discontinuity to be accurately localized. Canny [7] uses the smallest scale at which a given intensity change can be detected to most accurately localize it.

Brady, Ponce, Yuille, and Asada [3] report initial experiments that adapt Asada and Brady's algorithm [1] to find surface changes. We describe a number of problems, both mathematical and implementational, with that approach.

Section 4 describes a robust algorithm to find *roofs*, *steps*, *smooth joins*, and *shoulders*. *Roofs* are found from extrema of curvature (positive maxima and negative minima), whereas *steps*, *shoulders*, and *smooth joins* are found from parabolic points: zero crossings of the Gaussian curvature. We use scale space behavior to discriminate *steps*, *shoulders*, and *smooth joins*. Section 6 shows the algorithm at work.

2. Background

In this section, we recall some of the main features of our work on representing visible surfaces. We work with dense depth maps that are the output of "shape-from" processes such as stereo or, more usually, direct ranging systems. There are three principal problems to be addressed:

1. Finding surface intersections. These enable the *description* of the depth map to be partitioned into a set of smooth surface patch descriptions. This is the problem addressed in the present paper. Surface intersections do not, in general, partition the depth map. Consider, for example, a bulbous end of an American telephone handset (Figure 1). The surface intersection marked on the figure peters out by the time the cylindrical portion is reached. Each surface intersection has an associated description that includes its type (*step*, *roof*, *smooth join*, etc.). In general, the type of surface intersection may vary along its length [16], [27]. If a surface intersection has a special property, such as being planar, that property is included in the description.
2. Generating descriptions for the smooth surface patches that result from the partitioning in (1). This is the problem addressed by Brady, Ponce, Yuille, and Asada [3], who introduce a representation called *Intrinsic Patches*. This is discussed further below.
3. Matching surface descriptions to a database of object models that integrate multiple viewpoints of a surface. We have not yet addressed this problem. Grimson and Lozano-Pérez [13], Faugeras, Hebert, Pauchon, and Ponce [12], and Faugeras and Hebert [10] have made a solid start on the problem, though they restrict attention to the case of polyhedral approximations to surfaces. However, Faugeras and Hebert [10], [11], illustrate the advantages of representations based on sculptured surfaces. Brou [4], Little [21], and Ikeuchi and Horn [17] have developed the Extended Gaussian Image (EGI) representation for recognition and attitude determination. The EGI is an information-preserving representation only for complete maps of convex objects, a rare situation in practice. Not much has been done to extend the representation to handle non-convex objects.

The *Intrinsic Patch* representation that we are developing is based on concepts of differential geometry, principally because it provides a hierarchy of increasingly stringent surface descriptions. A surface may simply be (doubly) curved, but, in some cases, it may be ruled, even developable, even

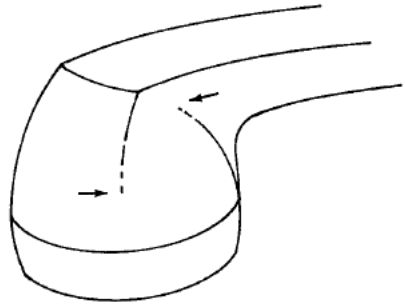


Figure 1: A telephone handset illustrates that surface intersections on curved surfaces do not, in general, partition the surface into a patchwork of smooth components.

conical. Our aim is to find the most appropriate and most stringent descriptors for portions of a surface. If, for example, there is a connected region of umbilic points, indicating that part of the surface is spherical, then it is made explicit, as is the center of the corresponding sphere (Figure 2). If there is a portion of the surface that is determined to be part of a surface of revolution, it is described as such, and the axis is determined (see Figures 2 and 16).

Similarly, if there is a line of curvature or an asymptote that is planar or whose associated curvature (principal curvature or geodesic curvature respectively) is constant, then it is made explicit. For example, the asymptote (which in this particular case is also a parabolic line) that marks the smooth join of the bulb and the stem of the lightbulb in Figure 2, as well as the surface intersections marked on the oil bottle in Figure 16, are noted in the representation. The program described in Section 3 cannot compute the asymptote on the lightbulb; but that described in Section 4 can. We may associate a description with a curve that is a surface intersection; but only if it has an important property such as being planar. For example, a slice of a cylinder taken oblique to the axis of the cylinder produces a planar

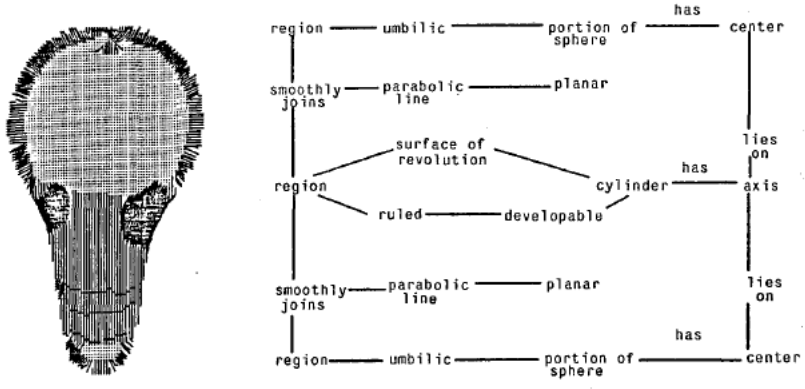


Figure 2: The representation of a light bulb. (a) The dotted region consists of umbilic points, indicating that the bulb is spherical. The parallel lines are the meridians of the cylindrical stem. The parallels, which are also rulings, are not shown. (b) The representation that we are working towards for the lightbulb. All save the rightmost column can be automatically computed by existing programs.

curve of intersection. Machining operations such as filleting tend to produce planar curves. Similarly, the intersection of a finger of a dextrous robot hand [25], [18], [19] and an object surface is planar. On the other hand, the intersection of two cylinders is not a planar curve.

Figure 2(b) illustrates the representation we are aiming at. The stem of the lightbulb is determined to be cylindrical, because it is ruled and because it is a surface of revolution. We can compute the axis of the stem. The bulb is determined to be a portion of a sphere, because it is a connected region of umbilic points. The center of the sphere can be computed. Similarly, the center of the spherical portion that forms the threaded end can be determined. The stem is smoothly joined to the bulb. Moreover, the axis of the cylindrical stem passes through the centers of the spheres defined by the bulb and threaded end. This distinguishes the diameters of each sphere that are collinear with the stem axis, showing that the lightbulb is a surface of

revolution. All of Figure 2(b) can be computed by the algorithms described in this paper and in Brady, Ponce, Yuille, and Asada [3], except for the rightmost column, which relates to the inferences that derive from attaching the spherical portions to the cylindrical stem. Currently, we are working on the inference engine (see also Kapur, Mundy, Musser, and Narendran [20]).

3. Surface intersections from lines of curvature

Asada and Brady [1] introduce a representation, called the *Curvature Primal Sketch*, of the significant changes of curvature along a planar curve. We review that work here because our extension to surfaces follows an analogous development. Asada and Brady describe an algorithm that not only detects and localizes significant changes, but describes those changes symbolically. The simplest descriptor is *corner*, where two arcs meet continuously but where the tangent is discontinuous. Other descriptors are composed of two or more instances of the corner model. The curve that is input to the algorithm is represented by its tangent $\theta(s)$ where s is the intrinsic arclength coordinate. The algorithm is based on a mathematical analysis of a set of models that are idealized instances of the descriptors. For example, the corner model is formed by the intersection of two circles. Note that this is intended as a *local* approximation to a corner to facilitate analysis. It does not prejudice the subsequent approximation of the contour to be piecewise circular. Rather, it suggests a set of knot points for any appropriate spline approximation.

Asada and Brady derive a number of salient features of the curvature of the models as they vary with the scale of the smoothing (Gaussian) filter. For example, a corner generates a curvature maximum, equivalently a positive maximum flanking a negative minimum in the first derivative of curvature. The height and separation of these peaks varies in a characteristic fashion over scale. The salient features are the basis of the tree matching algorithm that locates a curvature change and assigns it a

descriptor. Note that the distance between peaks varies approximately linearly (in arclength) with scale.

In the next section we develop an analogous mathematical framework for significant surface changes. Our surface analysis is local and based upon smoothing (locally) cylindrical functions with a Gaussian distribution. This is because of the following theorem, proved in Brady, Ponce, Yuille, and Asada [3].

The Line of Curvature Theorem: The convolution of a cylindrical surface with a Gaussian distribution is cylindrical. In more detail, let $f(x,y,z)$ be a surface that is the cross product of a planar curve and a straight line. The lines of curvature of the convolution of f with a Gaussian distribution are in the plane of the curve and parallel to the generating line.

In vector notation, a cylindrical surface has the form $\mathbf{r}(x,y)=x\mathbf{i}+y\mathbf{j}+f(x)\mathbf{k}$, and consists of parallel instances of a curve $f(x)$ in the $x-z$ plane. Our models for *roof*, *step*, *smooth join*, and *shoulder* correspond to different choices for the function $z=f(x)$.

The curvature of the smoothed curve is given by the non-linear expression

$$\kappa_{smooth}(x) = \frac{z''_{smooth}}{(1+ z'^{2}_{smooth})^{3/2}}. \qquad (1)$$

Since Asada and Brady [1] could work with tangent directions $\theta(s)$ along a planar curve, the curvature was the linear expression $d\theta(s)/ds$, so that the curvature of a smoothed contour is simply equal to the smoothed curvature of the original contour. This is *not* the case for surfaces represented as height functions $z(x,y)$. For example, the (constant) curvature of the parallels of a surface of revolution are modified (see Figure 15(a)). The non-linearity of curvature considerably complicates the analysis of surface change models presented in Section 4 relative to those used by Asada and Brady. Non-linearity affects smoothing too, as we discuss in Section 5.

Brady, Ponce, Yuille, and Asada [3] used the *Line of Curvature*

Theorem directly in a *two-step process* to detect, localize, and symbolically describe surface intersections, as follows:

1. Compute the lines of curvature on the surface;
2. Compute significant changes of curvature along the lines of curvature found in the first step.

The lines of curvature are computed using a best-first region growing algorithm [3]. A *good continuation* function is defined between neighboring points of the surface. The function involves the Cartesian distance between the points and the inner product of the tangent vectors corresponding to the curvature principal directions at the two points. The region growing algorithm joins the point pair whose good continuation function is globally maximum, and incorporates the new link into the developing set of lines of curvature. Brady, Ponce, Yuille, and Asada [3] show several illustrations of the algorithm's performance. In the second step of finding surface intersections, Asada and Brady's algorithm for computing the Curvature Primal Sketch, described in the previous section, is applied to the lines of curvature in turn.

The two step process has been tested on the objects shown in Brady, Ponce, Yuille, and Asada [3]: a lightbulb, a styrofoam cup, and a telephone receiver. It is robust and gives good results, suggesting that the method has competence. Nevertheless, there are several problems with the method:

- **The method is inefficient.** Typical running times on a lisp machine for a smoothed depth map that is 128 points square are of the order of one hour. Much of the time is spent on further smoothing each of the (typically hundreds of) lines of curvature at multiple scales, as required by the Curvature Primal Sketch algorithm.
- **Multiple multiple-smoothing is mathematically confused.** The raw surface data are smoothed at multiple scales σ_i, giving a set of surfaces z_i. The Curvature Primal Sketch algorithm further smooths the lines of curvature of z_i at multiple scales σ_j yielding a set of smoothed lines of curvature $\mathbf{r}_{ij}(s_{ij})$. There is no obvious relation between the scales σ_i and σ_j.
- **Discretization makes implementation difficult.** The lines of curvature of an analytic surface form a dense orthogonal

web. The (smoothed) depth maps we work with are discrete approximations to analytic surfaces. In practice, the lines of curvature found by the two step process are sometimes broken. The lines of curvature near the perceptual join of the stem and bulb of the lightbulb shown in Figure 2 illustrates this problem. This is due in part to quantization effects, but is also because the principal directions change rapidly near surface discontinuities. This is why the *smooth join* between the bulb and the stem of the lightbulb is not found by the two step process.

- **The *Line of Curvature Theorem* only applies locally.** In practice, few surfaces are cylindrical in the sense of the *Line of Curvature Theorem*. The Theorem is only approximately true in general, and then only locally. The application of the Curvature Primal Sketch algorithm in the second step does not respect this.
- **Lines of curvature on smoothed surfaces are not planar curves.** The models that are embodied in the Curvature Primal Sketch algorithm are not a complete set for surface intersections.

The success of the two step process suggests that the method is on the right track. The problems just enumerated suggest that reducing the problem to apply an existing algorithm developed for planar curves, though expedient, is wrong. Together, these observations suggest that a real two-dimensional extension of the Curvature Primal Sketch should be developed along analogous lines. The next section reports our progress toward such an extension.

4. Toward a surface primal sketch

4.1. A three-step process

In this section we develop a method for finding certain types of changes in the height of a surface that overcomes the difficulties of the two-step process described in the previous section. The types of changes we have analyzed and implemented are as follows: *step*s, where the surface height function is discontinuous; *roof*s, where the surface is continuous but the surface normal is discontinuous; *smooth join*s, where the surface normal is

continuous but a principal curvature is discontinuous and changes sign; and *shoulders*, which consist of two roofs and correspond to a *step* viewed obliquely. It turns out that *roofs* consist of extrema of the dominant curvature; that is, maxima of the positive maximum curvature or minima of the negative minimum curvature. On the other hand, *steps*, *smooth joins*, and *shoulders* consist of parabolic points, that is zero crossings of the Gaussian curvature. They are distinguished by their scale space behavior.

We have implemented the following *three-step process* (Figure 3) that is illustrated in the examples presented in Section 6:

1. Smooth the surface with Gaussian distributions at a set of scales σ_i, yielding surfaces z_i. Compute the principal directions and curvatures everywhere;
2. In each smoothed surface z_i, mark the zero-crossings of the Gaussian curvature and the (directional) extrema of the dominant curvatures.
3. Match the descriptions of the surfaces z_i to find points that lie on *roof*, *step*, *smooth join*, and *shoulder* surface discontinuities.

The computation of principal curvatures is described by Brady, Ponce, Yuille, and Asada [3]. They investigate parabolic lines and lines of curvature as *global* descriptors of surfaces, and suggest that such lines need additional global properties, such as planarity, to be perceptually important. In this paper, we are interested in parabolic points and curvature extrema as *local cues* for significant surface intersections.

We discuss smoothing in more detail in the next section. The next four subsections analyze *steps*, *roofs*, *smooth joins*, and *shoulders*. Finally, we discuss the matching algorithm and further work that is needed to elaborate the model set.

Is it necessary to use multiple scales to find surface intersections? Arguments supporting multiple scales for edge finding in images have been advanced elsewhere (see [23]). However, it might be supposed that it would be sufficient to smooth depth maps with a single coarse filter. Figures 12(b) and 15(a) show that this is not so. Even after thresholding, there is still a large curvature extremum in the neck of the bottle running parallel to the

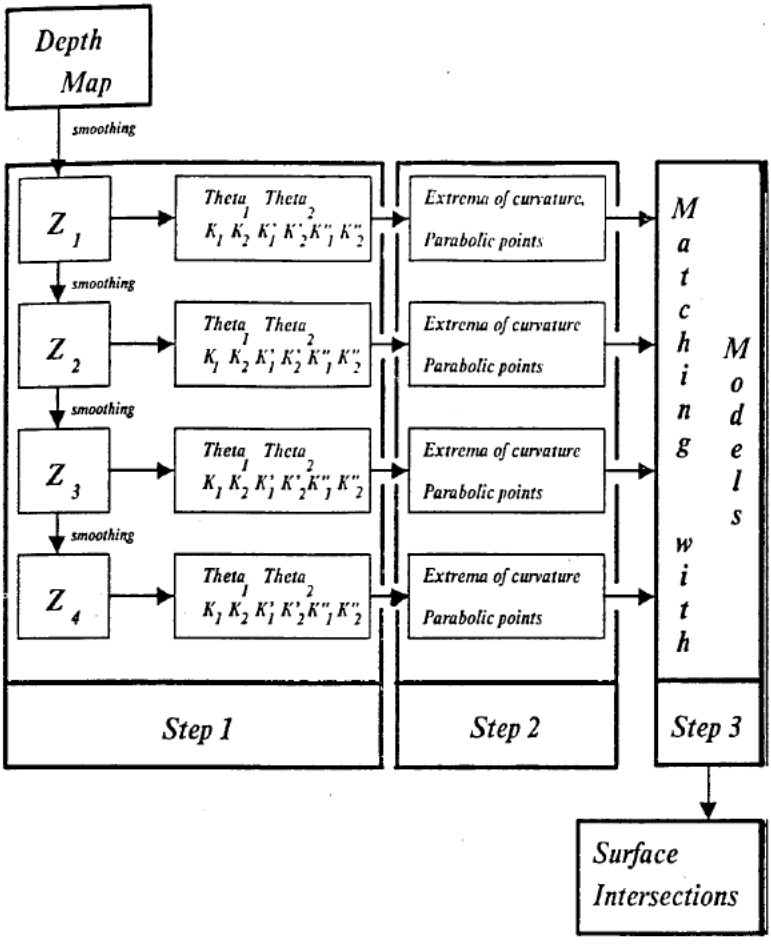


Figure 3: Schematic of the three-step process described in this paper and in Brady, Ponce, Yuille, and Asada [3]. The analysis of the set of models and the matching algorithm are the topic of the present paper. Results of running the process are shown in Section 6.

axis. This extremum is an artifact of non-linear smoothing, and it cannot be eliminated at a single scale. Instead, we reject it because it does not change over scale in the characteristic manner of a *roof*.

4.2. Step discontinuities

A *step* occurs when the surface itself is discontinuous. The model we use consists of two slanted half planes whose normals lie in the $x-z$ plane. They are separated by a height h at the origin (Figure 4).

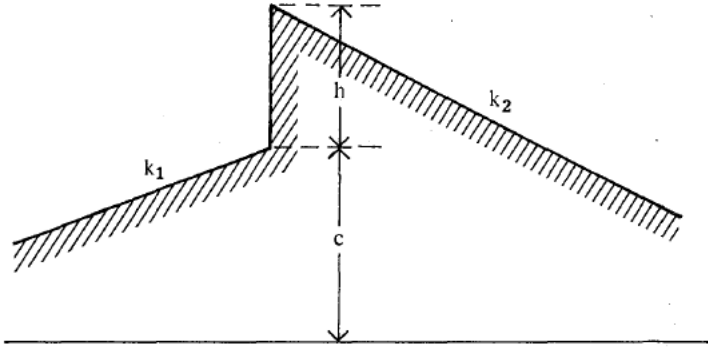


Figure 4: The *step* model, consisting of two slanted planes separated by a height h at the origin. The *roof* model corresponds to the case $h=0$ and $k_1 \neq k_2$.

Using the line of curvature theorem, we study the one dimensional formulation of this model.

Let the curve $z = f(x)$ be defined by

$$z = \begin{cases} k_1 x + c, & x < 0; \\ k_2 x + c + h, & x > 0; \end{cases} \tag{2}$$

where h and c are constants. In this expression, h is the height of the *step*. We now derive the result of smoothing this function with a Gaussian distribution at a given scale σ. To obtain a symmetric form for this smoothed version, we introduce the two following parameters:

$$k = (k_1 + k_2)/2;$$

$$\delta = k_2 - k_1 .$$

If we denote the smoothed curve $G_{\sigma} * z_{\sigma}$ by z_{σ}, we then obtain

$$z_\sigma = c + \frac{h}{\sigma\sqrt{2\pi}}\int_{-\infty}^{x} exp(-\frac{t^2}{2\sigma^2})dt \tag{3}$$

$$+ kx + \frac{\delta x}{\sigma\sqrt{2\pi}} \int_0^x exp(-\frac{t^2}{2\sigma^2})dt$$

$$+ \frac{\delta\sigma}{\sqrt{2\pi}} exp(-\frac{x^2}{2\sigma^2}).$$

The first and second derivatives of z_σ are given by

$$z'_\sigma = k + \frac{\delta}{\sigma\sqrt{2\pi}}\int_0^x exp(-\frac{t^2}{2\sigma^2})dt + \frac{h}{\sigma\sqrt{2\pi}} exp(-\frac{x^2}{2\sigma^2}); \tag{4}$$

$$z''_\sigma = \frac{1}{\sigma\sqrt{2\pi}}(\delta - \frac{hx}{\sigma^2}) exp(-\frac{x^2}{2\sigma^2}) . \tag{5}$$

In particular, the curvature κ, given by Equation (1), has a zero crossing at the point $x_\sigma = \sigma^2\delta/h$. This is at the origin if and only if $k_1 = k_2$; otherwise, the distance from x_σ to the origin is proportional to σ^2. This is illustrated in Figure 5 for the *step* between the cylindrical body and the cylindrical base of the oil bottle shown in Figure 16. From Figure 5, we calculate δ/h to be 0.105. The actual height of the step is about 1.5 millimeters. By the way, the position of the zero crossing shown in Figure 5 moves by about 3 pixels over one octave.

Using the fact that the second derivative of z_σ is zero at x_σ, it is easy to show that

$$\frac{\kappa''}{\kappa'}(x_\sigma) = \frac{z''''_\sigma}{z'''_\sigma}(x_\sigma) = -\frac{2\delta}{h} \tag{6}$$

So the ratio of the second and first derivatives of the curvature at the zero crossing is constant over the scales. Calculating δ/h this way gives 0.11,

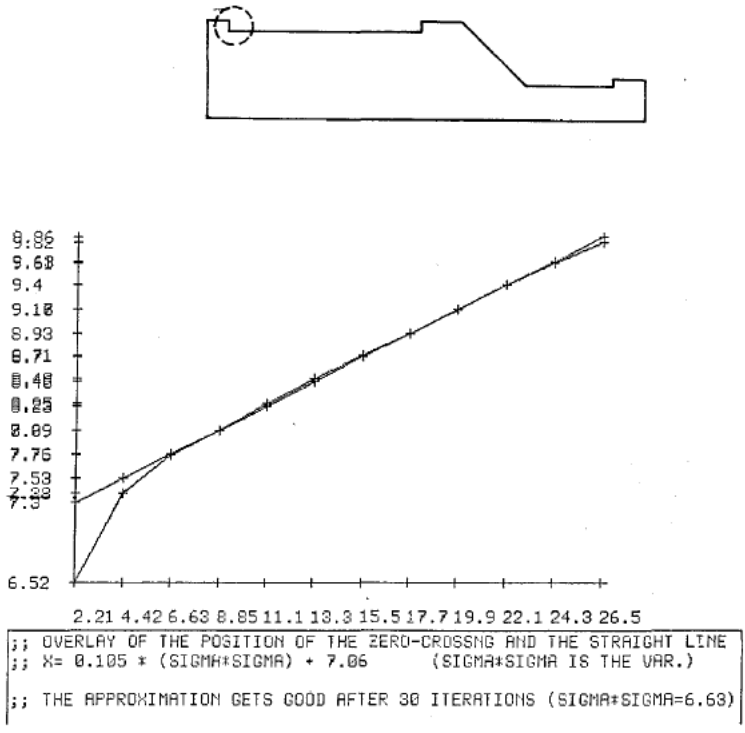


Figure 5: Variation over scale of the position of the zero crossing of the curvature of the smoothed *step* between the cylindrical body and the cylindrical base of the oil bottle shown in Figure 16. The abscissa is σ^2, and the ordinate is the position of the zero crossing. The height of the step is about 1.5 millimeters. The slope is $\delta/h = 0.105$.

which is close to the value given by the slope in Figure 5. This suggests that one ought not be overly coy about computing first and second derivatives of curvature of appropriately smoothed versions of a surface, even though they correspond to third and fourth derivatives.

4.3. Roof discontinuities

A *roof* occurs when the surface is continuous, but the surface normal is discontinuous. Specializing Equations (2) to (5) to the case h=0, we obtain

$$\kappa = \frac{1}{\sigma\sqrt{2\pi}} \frac{\delta e^{-\frac{x^2}{2\sigma^2}}}{\left[1 + (k + \frac{\delta}{\sigma\sqrt{2\pi}} \int_0^x exp(-\frac{t^2}{2\sigma^2}) dt)^2\right]^{3/2}} \tag{7}$$

$$= \frac{1}{\sigma\sqrt{2\pi}} \frac{\delta e^{-\frac{x^2}{2\sigma^2}}}{\left[1 + (k + \frac{\delta}{\sqrt{2\pi}} \int_0^{x/\sigma} exp(-\frac{u^2}{2}) du)^2\right]^{3/2}}$$

From Equation (7), we deduce that for a roof, we have $\kappa(x,\sigma)=(1/\sigma)\kappa(x/\sigma,1)$. In particular, this implies that the extremum value of κ is proportional to $1/\sigma$, and that its distance from the origin is proportional to σ. This is illustrated in Figure 6 for the *roof* discontinuity between the cylindrical neck and the conical shoulder of the oil bottle shown in Figure 16. Figure 6(a) shows the variation in the position of the negative minimum of curvature as a function of scale. Figure 6(b) shows that curvature is directly proportional to $1/\sigma$. It is also easy to show that the second derivative of the curvature, κ'', is proportional to $1/\sigma^3$. However, we do not use this property in the current implementation of the program, relying instead on the the variation of the extremum height over scale.

We first look for points that are local maxima (minima) of the maximum (minimum) curvature in the corresponding direction. The curvature directions can be estimated accurately. Then we use non-maximum suppression [7] to reject local extrema. The location of the peak, its height, its type (maximum or minimum), and its orientation, are the features we use

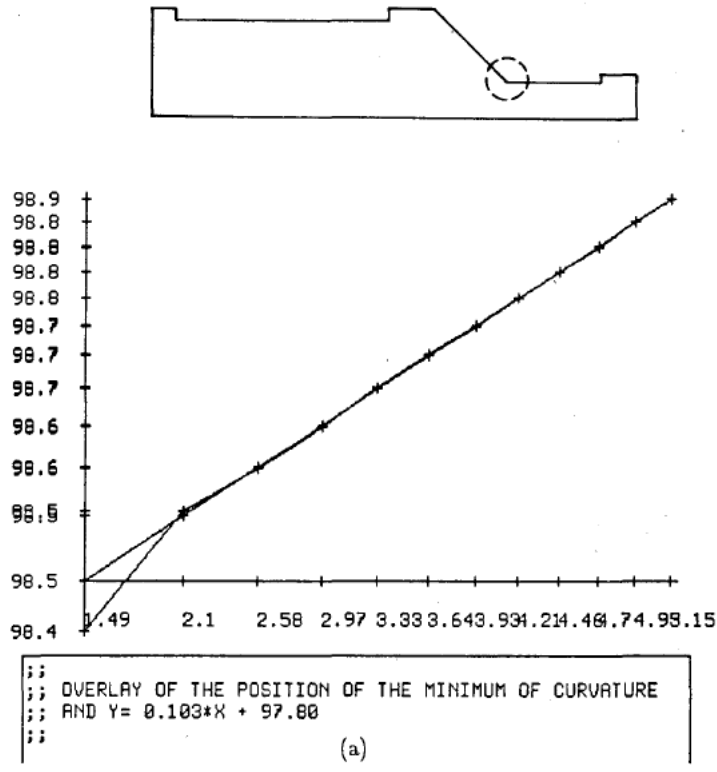
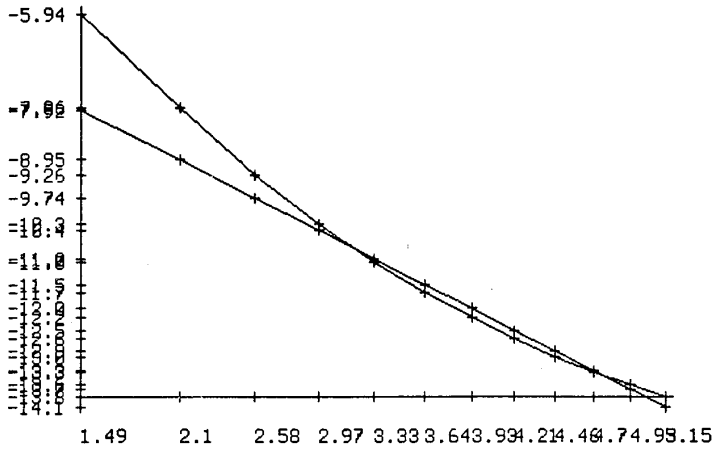


Figure 6: Scale space behavior of the *roof* discontinuity between the cylindrical neck and the conical shoulder of the oil bottle shown in Figure 16. (a) The position of the negative minimum of curvature varies linearly as a function of scale as predicted by the analysis.

for the subsequent matching over scales.

4.4. Smooth join discontinuities

In certain circumstances, one can perceive surface changes where both the surface and its normal are continuous, but where the curvature is discontinuous. We call such a surface change a *smooth join* discontinuity. If the curvature changes sign at a *smooth join*, the surface has a parabolic point. As we shall see, such changes can be found from zero crossings of a principal curvature. It is well-known (see [1] for discussion and references)



(b)

Figure 6: (b) The curvature is directly proportional to $1/\sigma$, as predicted.

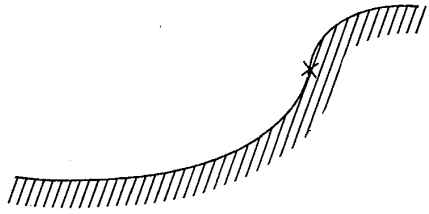
that a *smooth join* where the curvature does not change sign is perceptible only when the (discontinuous) jump in curvature is "sufficiently large". In such a case, there is not a zero crossing of curvature; rather there is a level crossing, and the curvature typically inflects. We do not yet have a complete analysis of that case.

Our model of a *smooth join* consists of two parabolas that meet smoothly at the origin (the curve is differentiable). Figure 7 shows the two distinct cases of the model. Though the two cases appear to be perceptually distinct and lead to different matching criteria, they are governed by the same equation, so it is convenient to analyze them together at first.

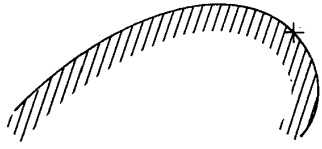
Consider the curve $z(x)$ defined by

$$z = \begin{cases} \frac{1}{2}c_l x^2 + b_l x + a_l, & x < 0; \\ \frac{1}{2}c_r x^2 + b_r x + a_r, & x > 0. \end{cases} \tag{8}$$

The continuity and differentiability of the curve at $x=0$ imply that $b_l=b_r=b$, say, and $a_l=a_r=a$, say. As in the case of the *step*, we introduce



(a)



(b)

Figure 7: The model for a *smooth join* consists of two parabolas meeting smoothly at the origin. (a) The curvature changes sign generating a parabolic point on the surface; (b) The curvature does not change sign. Such *smooth join*s are typically perceivable only when there is a large, discontinuous jump in curvature.

the parameters

$$c = (c_l + c_r)/2,$$

$$\delta = c_r - c_l.$$

We can express the surface, smoothed at the scale σ, as

$$z_\sigma = \frac{1}{2}\left(c + \frac{\delta}{\sigma\sqrt{2\pi}} \int_0^x exp(-\frac{t^2}{2\sigma^2})dt\right) x^2 \qquad (9)$$
$$+ \left(b + \frac{\delta\sigma}{2\sqrt{2\pi}} exp(-\frac{x^2}{2\sigma^2})\right) x$$
$$+ \left(a + \frac{c\sigma^2}{2} + \frac{\delta\sigma}{2\sqrt{2\pi}} \int_0^x exp(-\frac{t^2}{2\sigma^2})dt\right)$$

The first and second derivatives of z_σ are now given by:

$$z'_\sigma = \left(c + \frac{\delta}{\sigma\sqrt{2\pi}} \int_0^x exp(-\frac{t^2}{2\sigma^2})dt\right) x \qquad (10)$$
$$+ \left(b + \frac{\delta\sigma}{\sqrt{2\pi}} exp(-\frac{x^2}{2\sigma^2})\right);$$

$$z''_\sigma = c + \frac{\delta}{\sigma\sqrt{2\pi}} \int_0^x exp(-\frac{t^2}{2\sigma^2})dt\,. \qquad (11)$$

In particular, we deduce from Equation (11) that the curvature has a zero crossing if and only if

$$\frac{1}{\sqrt{2\pi}} \int_0^{x/\sigma} exp(-\frac{u^2}{2})du = -\frac{c}{\delta}\,. \qquad (12)$$

This equation has a solution if and only if the absolute value of c/δ is less than $1/2$, which simply corresponds to c_l and c_r having opposite signs. It follows that *smooth joins* of the sort shown in Figure 7(a) generate a zero crossing in curvature, hence a parabolic point on the surface. Those shown in Figure 7(b) do not.

For example, the parabolic lines found on the light bulb shown in Figure 8 are *smooth joins*. The two-step process utilizing the Curvature Primal

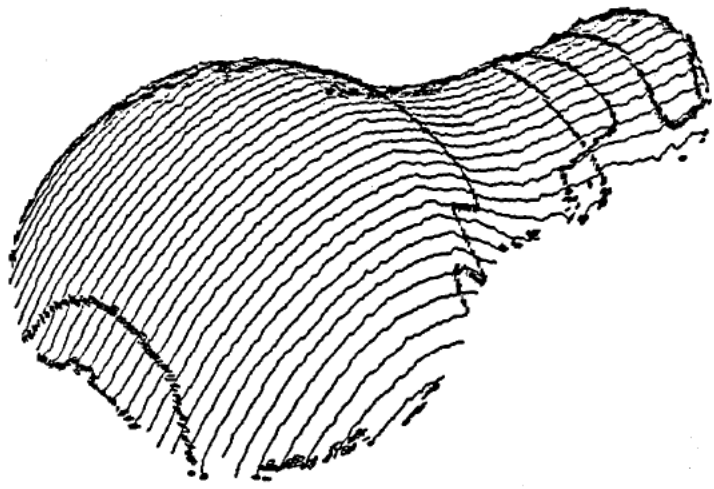


Figure 8: Parabolic lines found by the program described in Section 4 for the lightbulb shown in Figure 2. The smooth join between the stem and bulb were not found by the program described in Section 3.

Sketch algorithm, discussed in the previous section, failed to find the *smooth joins* (see [3] Figure 1).

Equation (12) implies that the distance from the zero crossing location x_σ to the origin is proportional to σ. Using this property and the fact that z'' is zero at x_σ, it is then easy to use the Implicit Function Theorem to show that

$$\frac{\kappa''}{\kappa'}(x_\sigma) = \frac{z''''_\sigma}{z'''_\sigma}(x_\sigma) = \frac{\gamma}{\sigma} \tag{13}$$

for some constant γ. It follows that the ratio of the second and first derivatives of the curvature in x_σ is inversely proportional to σ. This scale space behavior allows us to discriminate zero crossings due to steps from those due to smooth joins.

4.5. Shoulder discontinuities

A *step* discontinuity confounds information both about the geometry of the surface and the viewpoint. Shifting the viewpoint to the half space defined by the outward normal of the "riser" of the step typically changes the depth discontinuity to a pair of *roofs* of opposite sign whose separation again confounds geometry and viewpoint. We introduce the *shoulder* discontinuity to cater for this situation (Figure 9).

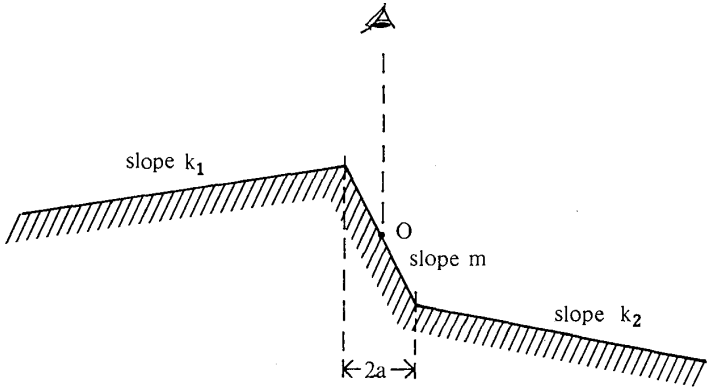


Figure 9: The *shoulder* discontinuity consists of two *roofs* of opposite sign. The shoulder appears as a *step* when viewed from the half-space defined by the inward normal of the "riser".

We may expect the scale space behavior of the *shoulder* to closely resemble that of the *step* when the projected separation $2a$ of the *roofs* is small compared to the filter size σ, perhaps becoming more like a pair of *roofs* as the viewpoint shifts. This is what we find.

We model the *shoulder* by the function

$$z = \begin{cases} k_1 x + (k_1 - m)a, & x < -a; \\ mx & x \in [-a, +a]; \\ k_2 x + (m - k_2)a, & x > +a\,. \end{cases} \tag{14}$$

If we denote $k_1 - m$ by δ_1 and $k_2 - m$ by δ_2, then $\delta_i \neq 0$, and δ_2/δ_1 is positive

(otherwise the curve is always convex, or always concave). It is easy to show that the second derivative of the shoulder, smoothed at scale σ is

$$z''_\sigma = \frac{\delta_2}{\sigma\sqrt{2\pi}} exp(-\frac{(x-a)^2}{2\sigma^2}) - \frac{\delta_1}{\sigma\sqrt{2\pi}} exp\,(-\frac{(x+a)^2}{2\sigma^2}). \tag{15}$$

Since δ_2/δ_1 is assumed positive, we deduce from Equation (15) that the curvature has a zero crossing. The location of the zero crossing is given by

$$x_\sigma = \frac{\sigma^2}{2a} log\,(\frac{\delta_1}{\delta_2}). \tag{16}$$

Using the Implicit Function Theorem as in the case of the *roof*, it is then straightforward to show that

$$\frac{\kappa''}{\kappa'}(x_\sigma) = \frac{z''''_\sigma}{z'''_\sigma}\,(x_\sigma) = -\frac{1}{a} log(\frac{\delta_1}{\delta_2}), \tag{17}$$

so the ratio of the first and second derivatives at the zero crossing is constant over scales.

4.6. Thin bar and other compound discontinuities

The models considered so far involve *isolated* surface changes. Even though a *shoulder* may appear as a pair of *roofs* if it is viewed close to the normal to the riser and if the riser subtends a sufficient visual angle, its more typical behavior is like that of a *step*. As two (or more) surface changes are brought more closely together, so that the filter width σ approaches half the separation of the changes, the filter responses due to the individual changes interfere with each other. Since certain kinds of compound surface discontinuity are important for recognition and use of objects, they must be modeled and matched by the program.

This observation raises two questions: (i) which compound surface changes should be modeled and matched; and (ii) how shall instances be found by the program? Ultimately, the answer to (i) is application-

dependent, though the *thin bar*, consisting of a *step* up closely followed by a *step* down, presses for inclusion (Figure 10). *Thin bars* occur as ribs on many

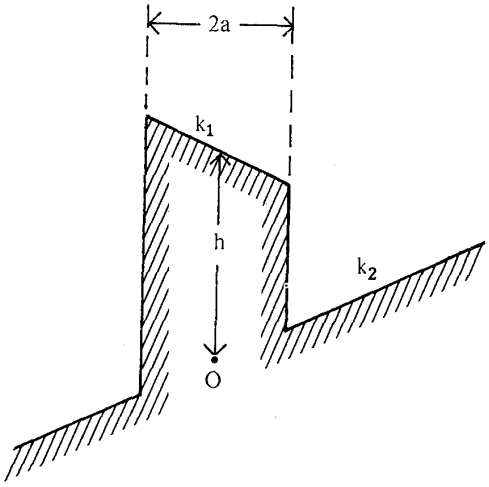


Figure 10: Model of a *thin bar* compound surface discontinuity. It consists of a plateau of height h, width $2a$, and slope k_2, resting on flat ground of slope k_1.

surfaces, for example along the sides of the neck of a connecting rod. Also, it seems [22], [24] that the mammalian visual system is sensitive to thin intensity stripes. In the case of curvature changes along a planar curve, the *crank* [1] is analogous to a thin bar since it consists of a corner followed closely by one of opposite sign. Other compound surface changes that might be important are a rounded corner and a moulding that is like a thin bar but with one of its risers smooth and concave. We have not studied such configurations.

Restricting attention to *thin bars* raises question (ii): how shall instances be recognized? First, let us conjecture what the curvature response to a *thin bar* might look like. We may base our conjecture on Asada and Brady's analysis [1] of a *crank*, though we need to be cautious because their

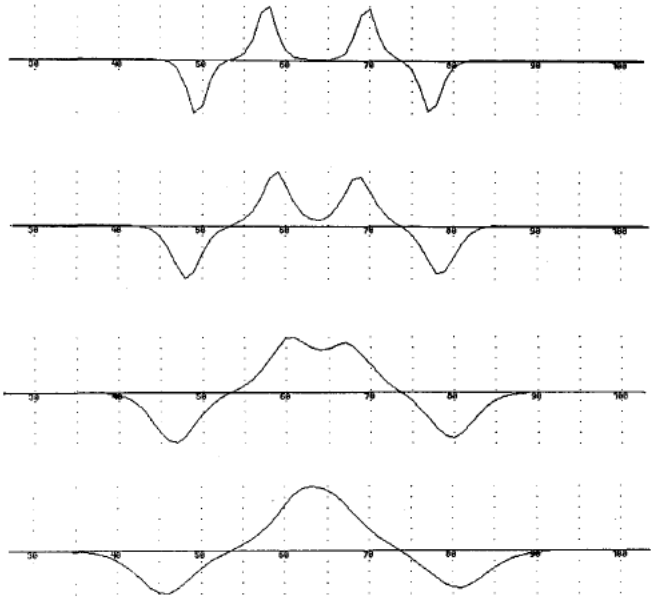


Figure 11: Expected curvature response of a *thin bar* to Gaussian filters. At fine scales, the *thin bar* signals two separate *steps*; at coarser scales it resembles a difference-of-Gaussians. The *step* responses begin to interfere when σ equals half the separation of the risers.

operators were linear. Figure 11 shows the response that might be expected, indeed the response that is generated in one special case (see below). Unfortunately, the response becomes substantially more complex in the general case.

Note that the *thin bar* in Figure 11 generates as many as five curvature peaks at fine scales, reducing to three at coarser scales. Note also that there appears to be a curvature peak at the origin. Asada and Brady extracted peaks at all scales and developed a matcher that linked peaks across scales. The *crank* model explicitly checked for three peaks splitting to five in the way shown in Figure 11. Matching such compound (planar curve curvature) changes was the source of the complexity of their program. In view of the

non-linearity of surface curvature, and the two-dimensionality of surfaces, we are reluctant to implement an analogous peak matching program. In practice, the peaks from a *thin bar* may cover as many as fifteen pixels, suggesting error-prone and inefficient search. In this paper, we have sought *local statements* that apply to a single zero crossing or curvature extremum and studied its scale space behavior in isolation. As we shall see, however, the analysis is quite difficult for a thin bar, even in simple cases.

We analyze a model of a *thin bar* consisting of a plateau of height h, width $2a$, and slope k_2, resting on flat ground of slope k_1 (Figure 10). We model the *thin bar* by the function

$$z = \begin{cases} k_1 x, & x < -a; \\ h + k_2 x, & x \in [-a,+a]; \\ k_1 x, & x > +a. \end{cases} \tag{18}$$

We denote $k_2 - k_1$ by δ. The first and second derivatives of the smoothed *thin bar* are given by

$$z'_\sigma = k_1 + \frac{\delta}{\sigma\sqrt{2\pi}} \int_{x-a}^{x+a} exp(-\frac{t^2}{2\sigma^2})dt \tag{19}$$

$$+ \frac{(h-\delta a)}{\sigma\sqrt{2\pi}} exp(-\frac{(x+a)^2}{2\sigma^2}) - \frac{(h+\delta a)}{\sigma\sqrt{2\pi}} exp(-\frac{(x-a)^2}{2\sigma^2}) ;$$

$$z''_\sigma = \frac{1}{\sigma\sqrt{2\pi}}(\delta - \frac{(h-\delta a)(x+a)}{\sigma^2}) exp(-\frac{(x+a)^2}{2\sigma^2}) \tag{20}$$

$$- \frac{1}{\sigma\sqrt{2\pi}}(\delta - \frac{(h+\delta a)(x-a)}{\sigma^2}) exp(-\frac{(x-a)^2}{2\sigma^2}) .$$

These expressions simplify considerably in the case that the plateau surface is parallel to the ground, that is $\delta = 0$. In particular, in that case $z'''_\sigma = 0$. However, the curvature attains an extremum at the origin (equivalently, $\kappa' = 0$) only when $k_1 = 0$. This is the case depicted in Figure 11, but it is

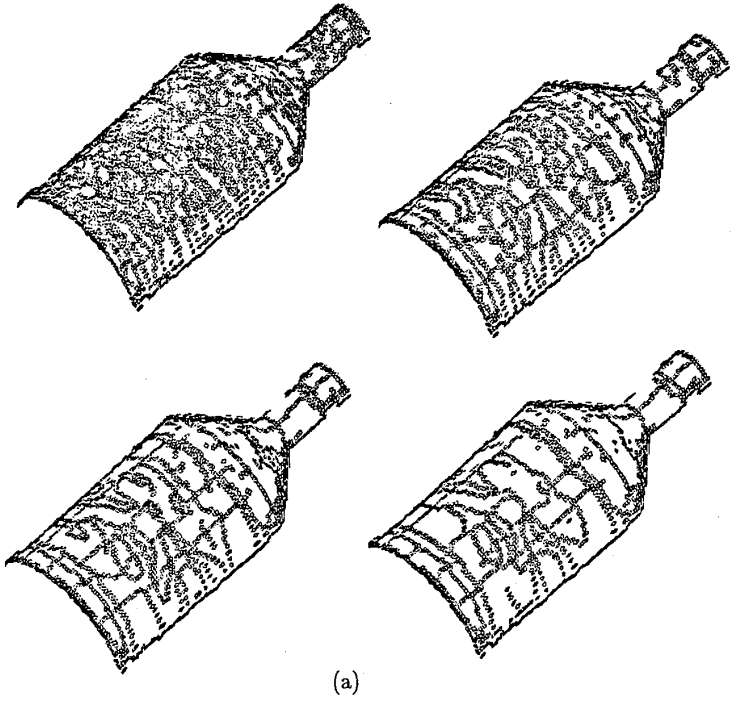
too restrictive since it is too sensitive to changes in viewpoint. Further work is needed here. The special case $k_1 = k_2 = 0$ is that typically studied in psychophysical studies of intensity *thin bars* (e.g., [24]). It would be interesting to know what is the response to *thin bars* of intensity superimposed on a linear intensity ramp.

4.7. The matching algorithm

We now use the models introduced in the previous sections to *detect* and *localize* the surface intersections. We use a coarse to fine tracking of the extrema of curvature and parabolic points found at each scale, and direct it by using the particular features associated to each model.

We first smooth the original depth map with a Gaussian distribution at a variety of scales σ (see Figure 3). We then compute, for each smoothed version of the surface, the principal curvatures and their directions (using the method described in [3]). We also compute the first and second derivatives of the principal curvatures in their associated directions. All parabolic points, as well as directional maxima of the maximum curvature and minima of the minimum curvature, are then marked (Figure 12(a)). The marked points are thresholded, according to their type: all extrema whose curvature value is less than a given value are removed; all zero crossings whose slope is less than another value are also removed (Figure 12(b)).

The values of the thresholds vary according to the scale. The extremum threshold varies proportionally to $1/\sigma$, as suggested by the *roof* model. There is unfortunately no such clue for the zero crossing threshold. This is not a major problem however, as the thresholding step is used for selecting a set of candidates for the subsequent matching process, rather than finding the surface intersections themselves. For example, the curvature extrema parallel to the axis of the oil bottle, that are due to the non-linear smoothing (Figure 15), cannot be eliminated by thresholding, but are rejected by the



(a)

Figure 12: (a) The extrema and zero crossings of curvature marked on the oil bottle at four different increasing scales, from left to right.

matching algorithm (Figure 16).

The basic matching algorithm is a simple 2D extension of the Asada and Brady method. We track the thresholded extrema and zero crossings across the scales, from coarse to fine. We obtain a forest of points, equivalent to a "fingerprint" [29]. We then use the models developed in the previous sections to parse this forest. Paths in the forest are assigned a type, *roof*, *step*, *smooth join*, or *shoulder*, according to the behavior of the associated curvature and its first and second derivatives across the scales. This provides us with a *symbolic description* of the models instances that are found. Finally, we use a stability criterion (see Witkin [28]) to find the significant surface intersections (*detection phase*); they correspond to paths in the

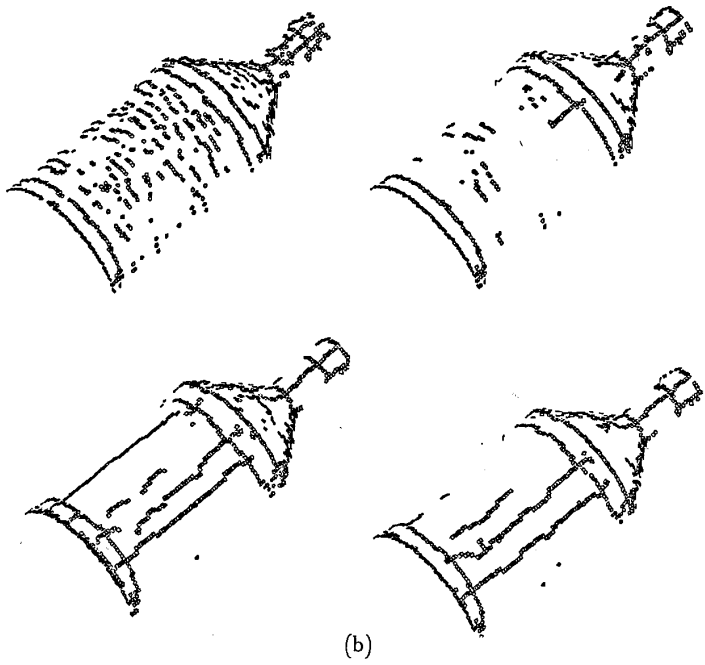


Figure 12: (b) The thresholded points. Note the curvature extrema parallel to the bottle axis are due to the non-linear smoothing, and the numerous parabolic points not thresholded at the finest scale. These non-significant points will be eliminated by the matching algorithm (Figures 13 and 16).

forest that go from the coarsest scale to the finest one. Points at the finest scale that have no ancestor at the coarsest scale are eliminated. The remaining points at the finest scale ensure the best *localization* of the surface intersections.

There are some problems with a straightforward implementation of this method. In order to stress the first one, we momentarily restrict ourselves to the case of the *roof*, which is the only one of our models to be characterized by a curvature extremum. In the *Curvature Primal Sketch*, the extrema of curvature are isolated points on a one-dimensional curve, and this makes the

construction of the trees relatively simple. However, in the two-dimensional case, the surface intersections themselves form continuous curves. This makes it difficult, for each scale, to associate to each point a single ancestor, and to build an explicit representation of the forest. This, in turn, makes difficult an *a posteriori* interpretation of this forest. Instead, we associate to each couple of marked points belonging to two successive images a compatibility function. This function involves the Cartesian distance between the points and the angle between their associated principal directions. It also takes into account the *roof* model by comparing the ratio of the curvatures to the inverse of the ratio of the associated scales. At each scale, and for each thresholded extremum, we look for an ancestor inside a square window of the previous scale image. If an ancestor with a good enough score is found, then the point is kept as a potential ancestor for the next scale. Otherwise it is removed. This way, the forest is never explicitly built, and the interpretation is done during the tracking itself, as the only extrema tracked are those which correspond to potential *roofs*. In particular, this is how the artifacts due to non-linear smoothing are removed.

The case of the parabolic points is slightly more complicated, as they may correspond to different types of discontinuities. In fact, a point may be at the same time a zero crossing of the Gaussian curvature and an extremum of a principal curvature. We again define compatibility functions for *steps*, *shoulders*, and *smooth joins* that take into account their mathematical models. The behavior of the ratio of the second and first derivatives of the curvatures is the basis for these compatibility functions. At each scale, a point may be a candidate for several different types of intersections, and be associated to an ancestor of each of these types. The use of the succession of scales is usually sufficient to disambiguate these cases. However, if several interpretations subsist until the finest level, the interpretation with the best cumulated compatibility score is chosen.

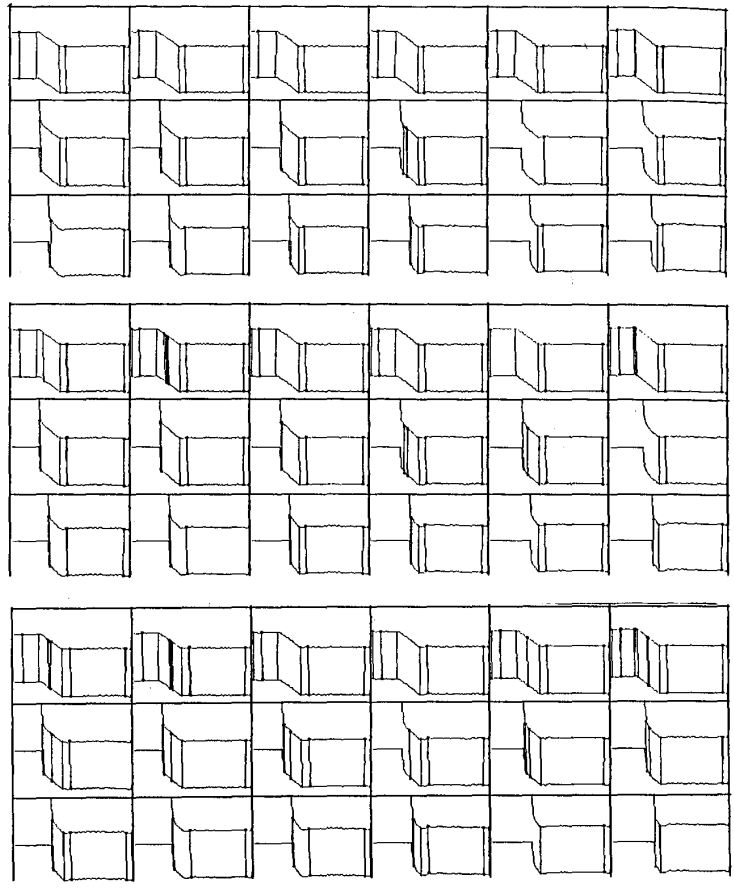


Figure 13: Matching between the different scales. The three parts of the figure show, for each couple of scales, points that have been matched on different slices of the oil bottle. These points are linked by a vertical line. Note that the matching is in fact made between points that are in a square window, not necessarily in the same slice. The display is done slice by slice for the sake of readability.

Figure 13 shows the matching algorithm at work, and Figure 16 shows the final result on the oil bottle. Note that, although we have analyzed the behavior of the position of the characteristic point of each of our models, and

have found it simple and reliable, we do not use it in the compatibility functions. The reason is that, for each intersection found, the real origin on the surface is unknown. This implies that at least three matched points are necessary to estimate the parameters of the motion of the extremum or zero crossing across the scales, and makes the measure of this movement unsuitable for a scale-to-scale tracking. Note however that the estimation of the movement parameters could be used for an *a posteriori* verification of the surface intersections found.

5. Smoothing a surface with a Gaussian distribution

Brady, Ponce, Yuille, and Asada [3] discuss techniques for smoothing a depth map with a Gaussian distribution. The main difficulty stems from bounding contours, where the surface normal turns smoothly away from the viewer, and where there is typically a substantial depth change between points on the surface of the object and the background. In general, the bounding contour is easy to find with a simple edge operator or by thresholding depth values. The problem is how to take the boundary into account when smoothing the surface.

Brady, Ponce, Yuille, and Asada observed that if the smoothing filter is applied everywhere, the surface "melts" into the background and changes substantially. Figure 14 is reproduced from [3] and shows this. They suggested instead using repeated averaging [5], [6] as well as adapting Terzopoulos' technique [26] of computational molecules to prevent leakage across depth boundaries. This smooths the surface without substantially altering it (see Figure 14(c)).

Here we point out a slight difficulty in smoothing surfaces using the technique illustrated in Figure 14(c), and suggest a refinement. Although the smoothed surface appears to be close to the original, small orientation-dependent errors are introduced. These errors are magnified in computing the curvature (Figure 15(a)), to produce "false" curvature extrema near the

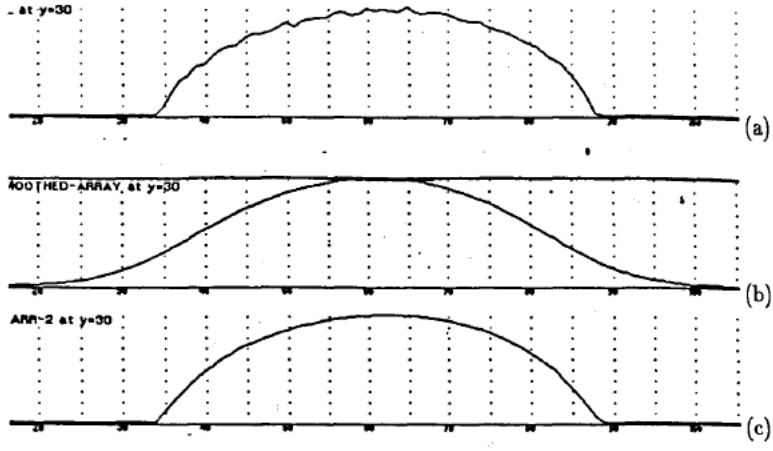
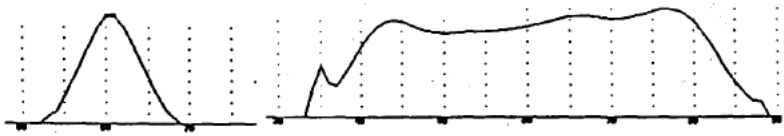


Figure 14: (a) Raw data from a cross section of an oil bottle after scanning using the INRIA system; (b) Smoothing across surface boundaries with a Gaussian mask that is applied everywhere; (c) Gaussian smoothing using repeated averaging and computational molecules. (Reproduced from [3], Figure 12)

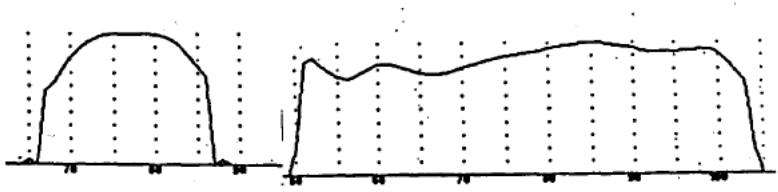
boundary (compare the overshoot phenomenon in Terzopoulos' work [26] on detecting surface discontinuities). The overshoots do not exemplify Gibbs' ringing as we originally thought. Instead, the phenomenon has two causes:

- **The coordinate frame is not intrinsic.** The smoothing filter is applied in the $x-y$ plane, and since this is not intrinsic to the surface, the result is orientation-dependent. For example, the difference between a cylinder and its smoothed version monotonically increases toward the boundary from a value of zero where the normal faces the viewer.
- **Points near the boundary are not smoothed as much.** Such points are relatively unsmoothed as several of their computational molecules are continually inhibited. The result is that the difference between the smoothed and original surfaces decreases at a certain distance from the boundary. This creates an inflection point, which in turn creates an extremum of curvature.

It is possible to substantially reduce the effect of this problem by smoothing in intrinsic coordinates. At each point of the surface, the normal



(a)



(b)

Figure 15: (a) The curvature computed from the smoothed surface shown in Figure 14(c). Small orientation-dependent errors in smoothing are magnified. The first figure is the neck, the second part of the body; (b) The curvature on the slices shown in (a) computed using the intrinsic coordinate method described in the text

is estimated. Instead of smoothing z, the surface point is moved along its normal a distance that depends upon the projected distances of the point's neighbors from the tangent plane. In the case that the normal faces the viewer, this is equivalent to the previous technique. However, the result is no longer orientation-dependent. Figure 15(b) illustrates the computation of curvature after smoothing the oil bottle by this method. The drawback with the technique is the computation time; it requires one to compute the tangent plane at every point.

The technique described in Brady, Ponce, Yuille, and Asada [3] has been used in all the examples presented here, as it represents a good tradeoff between computational efficiency and faithful rendering of the smoothed surface.

6. Examples

In this section, we present a number of examples of the surface discontinuities found on simple objects by our algorithm. In all the examples, we use four different scales corresponding to 20, 40, 60, and 80 iterations of the smoothing filter described in Brady, Ponce, Yuille, and Asada [3]. Viewing the resulting centrally-limiting Gaussian distributions as approximately bandpass filters, they span one octave. Figure 16 shows the final output of the algorithm for the oil bottle. The points detected during the matching step are linked together using a connected components exploration algorithm. The smallest components (less than 3 or 4 pixels) are then removed. Conversely, points may have been missed during the previous phases, creating gaps in the lines that are found. These gaps are filled by adding points that have characteristics compatible with the detected points. The bottle is finally segmented into six parts, separated by three step edges and two roofs.

Brady, Ponce, Yuille, and Asada [3] showed that the coffee cup shown in Figure 17 is best represented as the join of a cylindrical body and a tube surface that corresponds to the handle. Here we show that the handle can be separated from the body using the algorithms described in this paper. Note that the surface intersections are of type *roof*.

The third example shows the surface intersections found on a telephone handset, Figure 18. All the major intersections have been found. The representation is not symmetric because the handset was not quite perpendicular to the scanner, causing part of the surface to be occluded. Note that the surface intersections are more reliably detected at the coarsest scale, but are more accurately localized at the finest scale.

The surface intersections found on a few simple tools, namely a hammer, a drill, and the head of a screwdriver are shown in Figure 19. Figure 20 shows the surface intersections found on an automobile part that has featured in several papers by the group at INRIA. On this complicated

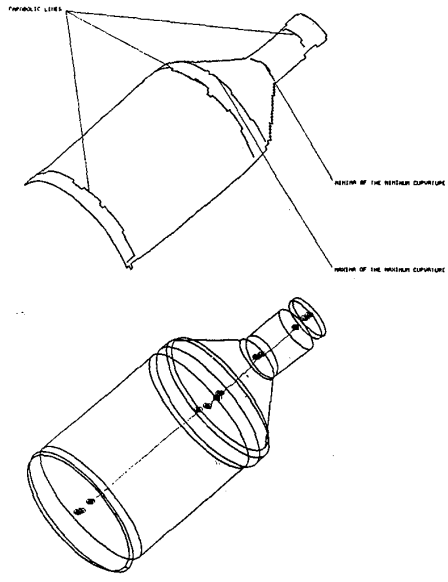


Figure 16: The oil bottle is segmented into six parts. Three step and two roof intersections are found by the algorithm described in this paper. The algorithms described in Brady, Ponce, Yuille, and Asada [3] determine the lines of curvature of the parts of the oil bottle, fit circles to the parallels, and fit axes to the centers of those circles.

object, global lines of curvature have no significance, so the Curvature Primal Sketch would not perform well. Notice in particular the circular step edge found on the left "head" of the part: it corresponds to a shallow depression whose depth is about one millimeter. This is approximately at the resolution limit of the laser scanner, and underlines the practical significance of the algorithms described here.

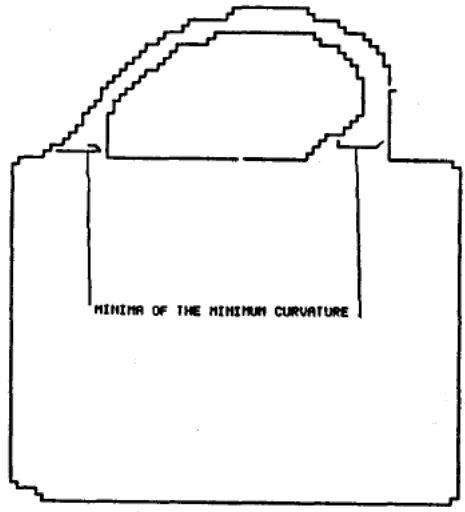
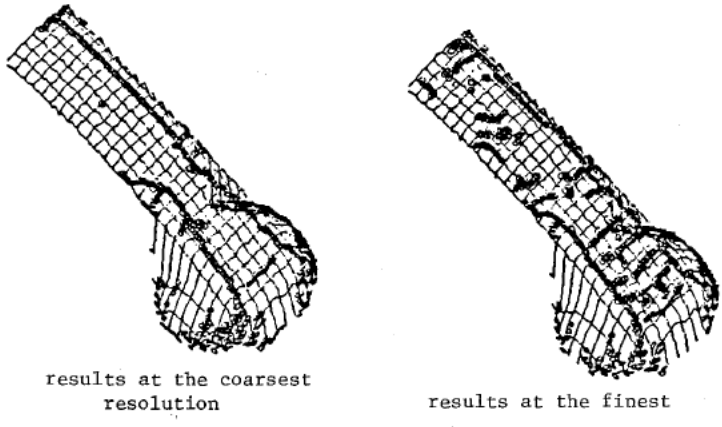


Figure 17: The joins of the handle to the body of the coffee mug are computed by the algorithms described in this paper. They are determined to be of type *roof*.

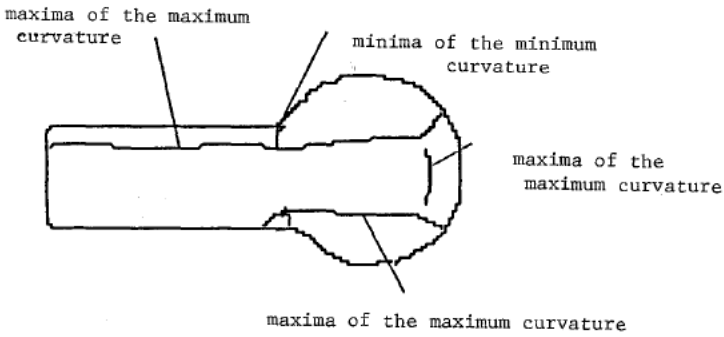
Figure 21 shows the surface intersections found on the head of a connecting rod. The current state of our algorithm cannot deal with the *thin bars* located on the sides of the neck of the connecting rod, but performs well for the other intersections.

The last example is the mask of a human face (Figure 22). The program finds face features as the nose, the eyes, and the mouth. This shows its ability to deal with arbitrary curved surfaces, usually not found in man-made objects.

Although our primary concern in this paper has been with the intrinsic geometry of a surface as found by a three-dimensional vision system, one might suppose that the methods described in this paper could be applied straightforwardly to *extract* and *interpret* significant intensity changes in images, considered as surfaces. To interpret intensity changes, it is necessary to take irradiate effects into account, since intensity changes do not always correspond to surface changes. Rather, they may signify



(a)



(b)

Figure 18: (a) The surface intersections found on a telephone handset at the coarsest and finest scales, approximately an octave apart. The intersections are more reliably detected at the coarsest scale; they are more accurately localized at the finest scale. (b) The results of matching the changes across scales.

reflectance or illumination changes. Extraction and interpretation was in fact the intent of Marr's original work [22] on the Primal Sketch, though considerably more attention was paid to extraction than interpretation. More recently, Haralick, Watson, and Laffey [14] have advocated a representation of image surface geometry that involves concave and convex

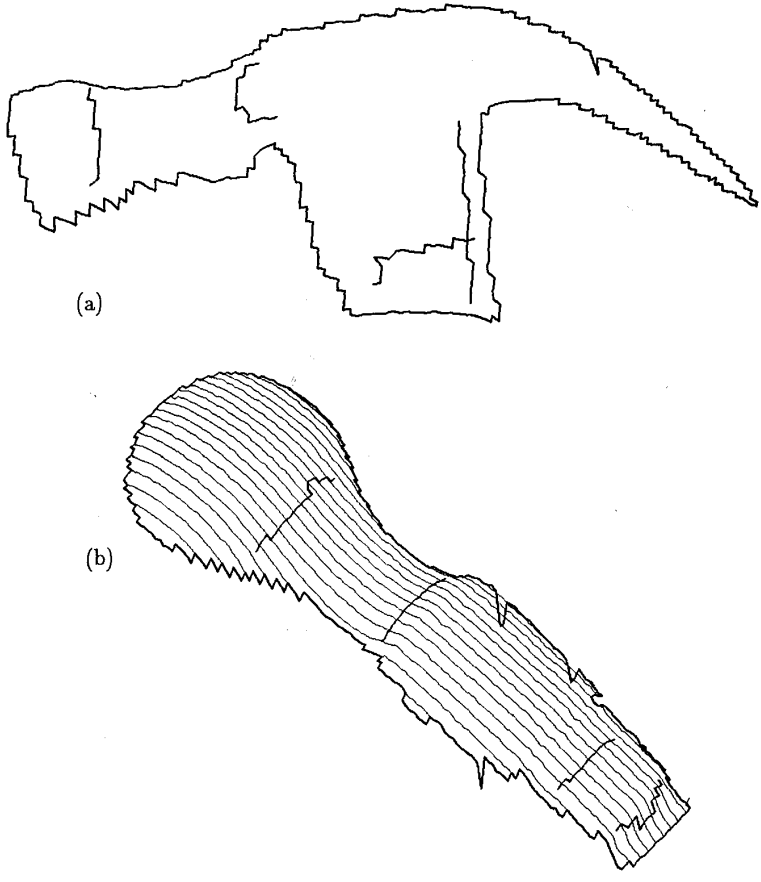


Figure 19: The intersections found by the program on simple tools. (a) a hammer; (b) a drill

hills, planar regions, and saddles. These geometrical aspects of the image surface are not interpreted in terms of the intrinsic geometry of the surface, or of illumination or reflectance changes.

Some preliminary work exists on interpretation of intensity changes. An early edge finder developed by Binford and Horn [2] included filters for step, roof, and "edge effect" changes. Horn's study of intensity changes [15] included a suggestion that occluding boundaries and reflectance changes correspond to *step* intensity changes, while concave surface

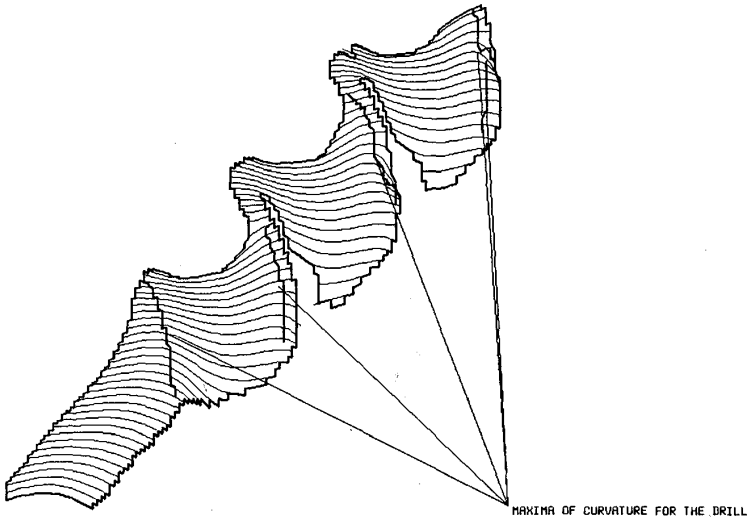


Figure 19: (c) a screwdriver

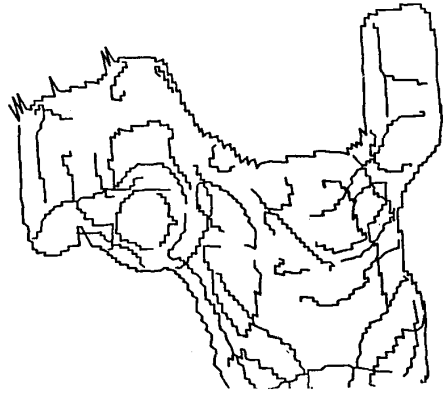


Figure 20: Surface intersections found on an automobile part (see, for example, [8], [9]). The circular step edge found on the left "head" of the part corresponds to a shallow depression whose depth is about one millimeter. This is approximately at the resolution limit of the laser scanner.

intersections generate *roof* intensity changes (because of mutual illumination). Finally, Yuille [29] suggests that certain points along lines of

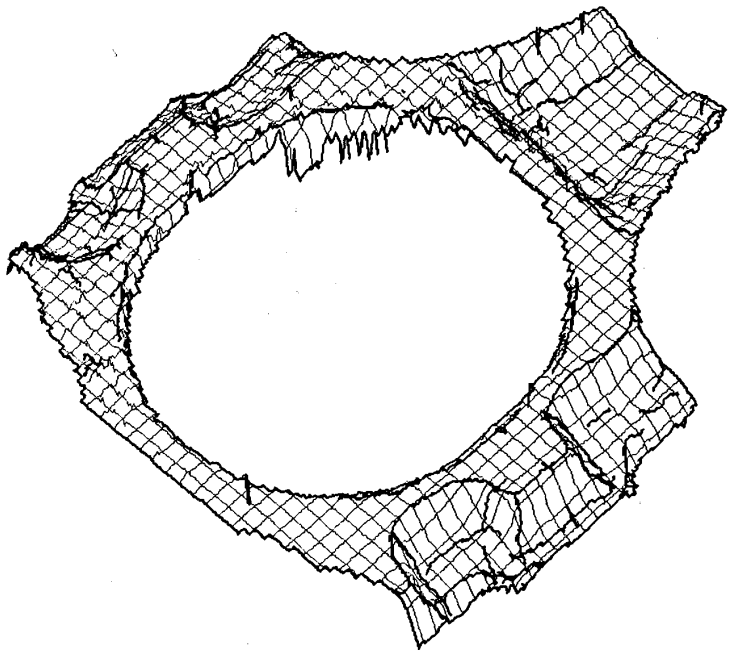


Figure 21: Surface intersections found on a connecting rod. Only the head of the part is shown. The inclusion of the *thin bar* models in the algorithm would allow a fine description of the neck of this object.

curvature of a surface can be extracted directly from an image. There is much scope for additional work along these lines.

Figure 23 shows an initial experiment we have carried out on applying the methods developed in this paper to image surfaces. The join of the wing to the fuselage of the airplane is determined to be *roof* changes, consistent with Horn's suggestion.

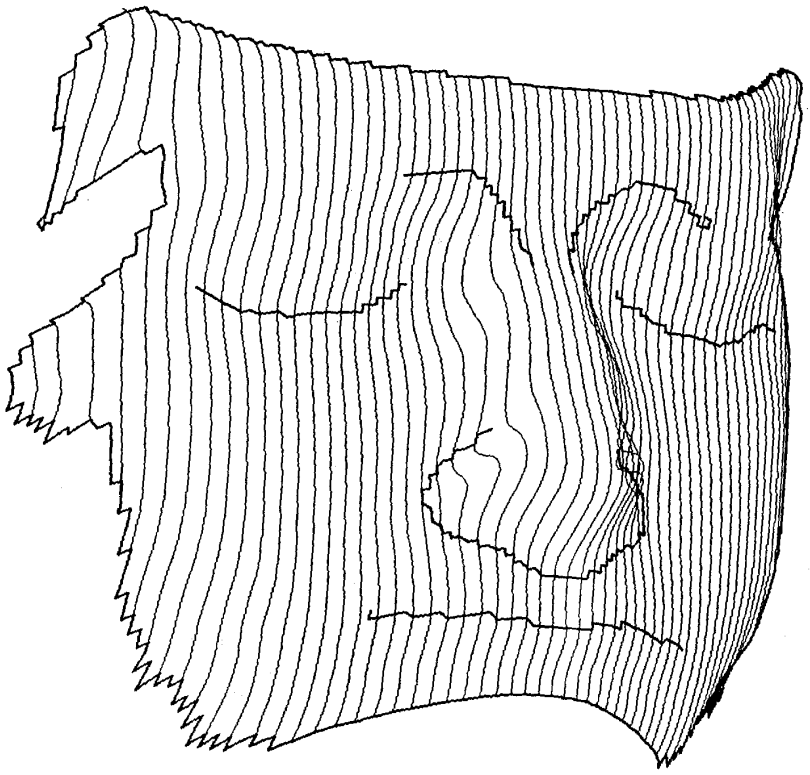


Figure 22: Surface intersections found on a mask. The mouth, the nose and the eyes are found as corners by the program.

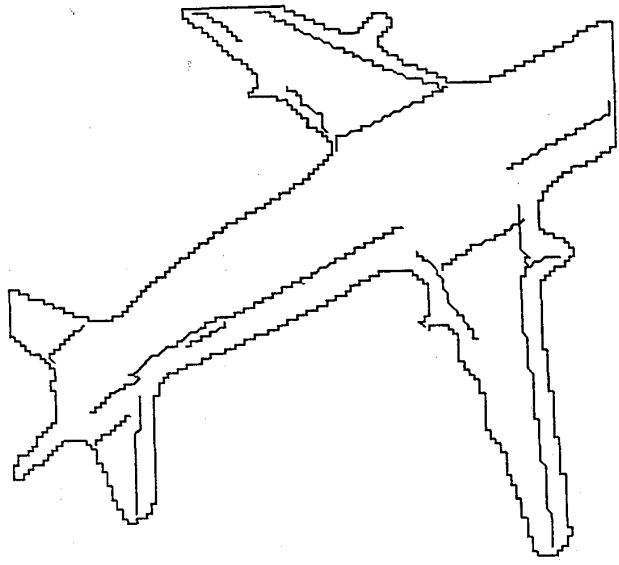


Figure 23: Application of the methods of the paper to an intensity surface. The interest is in the type of the internal intensity changes. The join between the wings and fuselage is a roof, suggesting a concave surface intersection.

Acknowledgments

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the laboratory's Artificial Intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-80-C-0505, the Office of Naval Research under contract number N00014-77-C-0389, and the System Development Foundation. This work was done while Jean Ponce was a visiting scientist on leave from INRIA, Paris, France. We thank several people who have commented on the ideas presented in this paper, particularly Ruzena Bajcsy, Bob Bolles, Jean-Daniel Boissonat, Bernard and Hilary Buxton, Olivier Faugeras, Eric Grimson, Berthold Horn, Ramesh Jain, Takeo Kanade, Christopher Longuet-Higgins, David Mumford, Joe Mundy, Tommy Poggio, Robin Stanton, Demetri Terzopoulos, and Alan Yuille. We are indebted to Olivier Faugeras and the INRIA group for giving us access to their ranging system, and to Philippe Brou for bringing his ranging system back to life.

References

[1] Asada, Haruo and Brady, Michael.
The curvature primal sketch.
Technical Report AIM-758, MIT, Artificial Intelligence Lab, 1984.

[2] Binford, Thomas O.
Inferring surfaces from images.
Artificial Intelligence 17:205-244, 1981.

[3] Brady, M., Ponce, J., Yuille, A. and Asada, H.
Describing surfaces.
In *Proc. Seond Int. Symp. on Robotics Research*, pages 434-445. Kyoto, Japan, 1984.

[4] Brou, P.
Finding the orientation of objects in vector maps.
Int. J. Rob. Res. 3(4):89-175, 1984.

[5] Burt, Peter J.
Fast filter transforms for image processing.
Comp. Graph. and Im. Proc. 16:20-51, 1981.

[6] Burt, Peter J.
Fast algorithms for estimating local image properties.
Comp. Graph. and Im. Proc. 21:368-382, 1983.

[7] Canny, J. F.
Finding edges and lines in images.
Technical Report TR-720, MIT, Artificial Intelligence Lab, 1983.

[8] Faugeras, O.D., et. al.
Towards a flexible vision system.
In A. Pugh (editor), *Robot Vision.* IPS, UK, 1982.

[9] Faugeras, O. D.
New steps toward a flexible 3-D vision system for robotics.
Preprints of the Second International Symposium of Robotics Research. Note: also see In Proc. 7th Int. Conf. Pattern Recogn., pages 796-805; Montreal, Canada, July :222-230, 1984.

[10] Faugeras O.D., Hebert, M.
A 3-D recognition and positioning algorithm using geometrical matching between primitive surfaces.
In *Proc. Eighth Int. Joint Conf. On Artificial Intelligence*, pages 996-1002. Los Altos: William Kaufmann, August, 1983.

[11] Faugeras, O. D. and Hebert, M.
The representation, recognitin, and positioning of 3-D shapes from range data.
to appear in the Int. Journal of Robotics Research , 1985.

[12] Faugeras, O. D., Hebert, M., Pauchon, E., and Ponce, J.
Object representation, identification, and positioning from range data.
First International Symposium on Robotics Research.
MIT Press, 1984, pages 425-446.

[13] Grimson, W. E.
Computational experiments with a feature based stereo algorithm.
Technical Report 762, MIT, Artificial Intelligence Lab, 1984.

[14] Haralick, Robert M., Watson, Layne T., Laffey, Thomas J.
The topographic primal sketch.
Int. J. Rob. Res. 2(1):50-72, 1983.

[15] Horn, B. K. P.
Understanding image intensities.
Artif. Intell. 8, 1977.

[16] Huffman, D. A.
Impossible objects as nonsense sentences.
Machine Intelligence 6.
Edinburgh University Press, Edinburgh, 1971, pages 295-323.

[17] Ikeuchi, K., Horn, B. K. P., et al.
Picking up an object from a pile of objects.
First International Symposium on Robotics Research.
MIT Press, 1984, pages 139-162.

[18] Jacobsen, S. C., Wood, J. E., Knutti, D. F., and Biggers, K. B.
The Utah/MIT dextrous hand: work in progress.
Int. J. Rob. Res. 3(4), 1984.

[19] Jacobsen, S. C., Wood, J. E., Knutti, D. F., Biggers, K. B., and Iversen, E. K.
The verstion I UTAH/MIT dextrous hand.
Proc. of the 2nd International Symposium on Robotics Research.
MIT Press, Cambridge, 1985.

[20] Kapur, D., Mundy, J. L., Musser, D., and Narendran, P.
Reasoning about three-dimensional space.
In *IEEE Int. Conf. on Robotics*, pages 405-410. St. Louis, MO, March, 1985.

[21] Little, J. J.
Recovering shape and determining attitude from extended Gaussian images.
Technical Report, Univ. of British Columbia, Dept. of CS, 1985.

[22] Marr, D.
Early processing of visual information.
Phil. Trans. Roy. Soc. B275:843-524, 1976.

[23] Marr, D. and Hildreth, E. C.
Theory of edge detection.
Proc. R. Soc. London B 207:187-217, 1980.

[24] Richter, J. and Ullman, S.
A model for the spatio-temporal organisation of X and Y type ganglion cells in the primate retina.
Biol. Cyb. 43:127-145, 1982.

[25] Salisbury, J. Kenneth and Craig, John J.
Articulated hands: force control and kinematic issues.
Int. J. Rob. Res. 1(1):4-17, 1982.

[26] Terzopoulos, D.
The role of constraints and discontinuities in visible-surface reconstruction.
Proc. 7th Int. Jt. Conf. Artif. Intell., Karlsruehe :1073-1077, 1983.

[27] Turner, K.
Computer perception of curved objects using a television camera.
Technical Report, Edinburgh University, 1974.

[28] Witkin, A.
Scale-space filtering.
In *7th Int. Jt. Conf. Artificial Intelligence*, pages 1019-1021. Karlsruhe, 1983.

[29] Yuille, A. L.
Zero-crossings on lines of curvature.
In *Proc. Conf. Amer. Assoc. Artif. Intell.*. Washington, August, 1983.

[30] Yuille, A. L. and Poggio, T.
Fingerprints theorems for zero-crossings.
Technical Report AIM-730, MIT, AI Lab, 1983.

[31] Yuille, A. L. and Poggio, T.
Scaling theorems for zero crossings.
Technical Report AIM-722, MIT, AI Lab, 1983.

3-D Object Representation from Range Data Using Intrinsic Surface Properties

B. C. Vemuri, A. Mitiche, and J.K. Aggarwal

Laboratory for Image and Signal Analysis
College of Engineering
The University of Texas at Austin
Austin, TX 78712

Abstract

A representation of three-dimensional object surfaces by regions homogeneous in intrinsic surface properties is presented. This representation has applications in both recognition and display of objects, and is not restricted to a particular type of approximating surface.

Surface patches are fitted to windows of continuous range data with tension splines as basis functions. Principal curvatures are computed from these local surface approximations, and maximal regions formed by coalescing patches with similar curvature properties.

1. Introduction

Range (depth) data provide an important source of 3-D information about the shape of object surfaces. Range data may be derived from either intensity images or through direct measurement sensors.

In recent years, considerable attention has been devoted to the problem of analyzing intensity images of 3-D scenes to obtain depth information [18], [2], [14]. Much of the work has been focussed on overcoming the difficult problems caused by projection, occlusion, and illumination effects such as shading, shadows, and reflections. Many monocular depth cues can be exploited to produce three-dimensional interpretations of two-dimensional images. These cues include texture gradient [15], size perspective (diminution of size with distance), motion parallax [25], occlusion effects [17], depth from optic flow, surface orientation from surface contours [17],

and surface shading variations [12].

According to many researchers the problem of deriving the 3-D structure of an object from its intensity image is equivalent to inferring the orientations of the surfaces of the object. The word inference is used because there is insufficient information in the intensity image to allow the visual system to merely recover or extract the surface orientation. Rather, the visual system must make explicit assumptions about the nature of the surfaces it sees. Although depth cues in a single intensity image are weak, there are generally strong cues to surface orientation; these were exploited by Stevens [27]. To determine 3-D structure from intensity information obtained from a single view of an object would require several restrictive constraints in the form of additional information about the objects that populate the environment of interest. Fundamentally this is because of the many-to-one nature of perspective projections; an infinite number of 3-D objects can correspond to any single view. Stereo-scopic vision [19] and structure from motion [30] may be used to effect three-dimensional reconstruction of objects. Both these methods may be considered as correspondence techniques, since they rely on establishing a correspondence between identical items in different images; the difference in projection of these items is used to determine the surface shape. If this correspondence is to be determined from the image data there must be sufficient visual information at the matching points to establish a unique pairing. Two basic problems arise in relation to this requirement. The first arises at parts of the image where uniformity of intensity or color makes matching impossible, the second when the image of some part of the scene appears in only one view of a stereo pair because of occlusion (the missing part problem) or a limited field of view. The further apart the two camera positions, the potentially more accurate the disparity depth calculation, but the more prevalent the missing part problem and the smaller the field of view. Another limitation of correspondence-based methods stems from the large number of

computations required to establish the correspondence.

The use of direct range measurements can eliminate some of the problems associated in deriving 3-D structure from intensity images. Capturing the third dimension through direct range measurement is of great utility in 3-D scene analysis, since many of the ambiguities of interpretation arising from occasional lack of correspondence between object boundaries and from inhomogeneities of intensity, texture, and color can be trivially resolved [13]. One of the disadvantages of direct range measurement is that it is somewhat time consuming. Several range finding techniques have been proposed in literature; a good survey of those in use is presented in Jarvis [13]. Two widely used ranging methods are based on the principle of triangulation [26], [1], [24] and time of flight [22].

There have been several studies on scene description using range data. Will and Pennington [32] describe a method by which the locations and orientations of planar areas of polyhedral solids are extracted through linear frequency domain filtering of images of scenes illuminated by a high contrast rectangular grid of lines. Shirai and Suwa [26] use a simple lighting scheme with a rotating slit projector and TV camera to recognize polyhedral objects. The facts that projected lines are equally spaced, parallel and straight on planar surfaces are exploited in reducing the computational complexity; only the end points of each line segment are found, and line grouping procedures are used to identify distinct planar surfaces. Agin and Binford [1] describe a laser ranging system capable of moving a sheet of light of controllable orientation across a scene; their aim was to derive descriptions of curved objects based on a generalized cylinder model. Their method is interactive and is useful in describing objects with cylinder like or elongated parts. Mitiche and Aggarwal [20] discuss an edge detection technique for noisy range images. Their motivation for detecting edges was to delineate regions on the surface of objects, and the technique is limited to finding edges in prespecified directions. Duda, Nitzan and Barret [8] use

intensity as a guide for finding planes, with plane equations being determined from the registered range images. Their technique assumes underlying surfaces to be planar. Faugeras, et al. [9] are quite successful in segmenting range data from approximately planar and quadratic surfaces. Their segmentation process partitions data points into subsets in which approximation by a polynomial of order at most two yields an error less than a given threshold. However, their edge detection scheme is prone to large errors due to the fact that planes are used to detect surface creases. Also, region merging requires explicit updating of region equations and could be quite time consuming. Grimson [10] presents a theory of visual surface interpolation, where the source of range data is a pair of stereo images. The theory deals with determining the best-fit surface to a sparse set of depth values obtained from the Marr and Poggio [19] stereo algorithm. Grimson's algorithm assumes that the entire scene is a single surface although this is not a valid assumption along occluding contours. Also, the iterative algorithm suggested converges slowly. Oshima and Shirai [23] describe scenes by surfaces and relations between them. Their intent was to develop a program which described a scene containing both planar and curved objects in a consistent manner. The primitives used in their algorithm are planar surface elements which are not adaptable to the shape of the surface. Following a region growing process, which requires explicit updating of region equations, a region refinement procedure is described which involves fitting quadrics to curved regions. This process is computationally expensive. Terzopolous [29] presents a technique for visible surface computation from multiresolution images. His surface reconstruction process treats surfaces of objects as thin plate patches bounded by depth discontinuities and joined by membrane strips along loci of orientation discontinuities. He suggests the use of intrinsic properties of surfaces, computed post reconstruction, to describe shape. His motivation for computing these intrinsic properties was to demonstrate the robustness of

his surface representation scheme.

In this paper a representation of visible 3-D object surfaces by regions that are homogeneous in certain intrinsic surface properties is presented. First smooth patches are fitted to the object surfaces; principal curvatures are then computed and surface points classified accordingly. The surface fitting technique is general, efficient and uses existing public domain software for implementation. An algorithm is presented for computing object descriptions. The algorithm divides the range data array into windows and fits approximating surfaces to those windows that do not contain discontinuities in range. The algorithm is not restricted to polyhedral objects nor is it committed to a particular type of approximating surface. It uses tension splines, which make the fitting patches locally adaptable to the shape of object surfaces. Computations are local and can be implemented on a parallel architecture . Our ultimate goal is to recognize objects independent of the viewpoint; therefore, this necessitates a representation which is invariant to rigid motion.

In Section 2 we describe the data acquisition mechanism, some relevant concepts in differential geometry are recalled, and the surface fitting technique used in building object descriptions is introduced. Section 3 describes in detail the algorithm for building object descriptions. In section 4 some experimental results using real range images are presented. Section 5 contains a summary.

2. Surface Fitting and Intrinsic Features Computation

The range information used in this study is obtained from a single view of the scene by the White scanner [28], a laser ranging system shown in Figure 1. This system works on the principle of light sheet triangulation. A sheet of light produced with a laser and a cylindrical lens is cast on a scene, resulting in a bright curve when viewed through a video camera. A rotating mirror causes this sheet of light to move across the scene. The intersection of

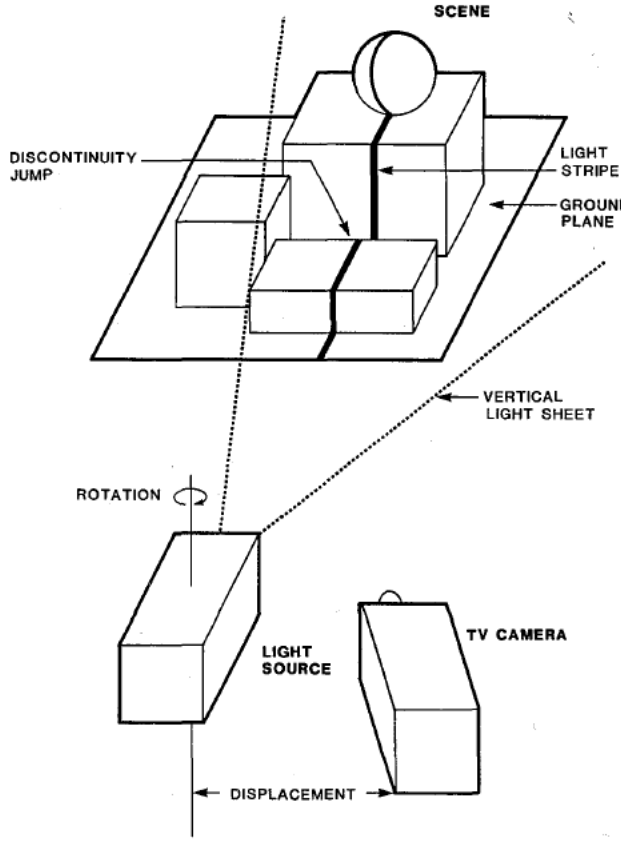


Figure 1: Laser ranging system [13]

this plane of laser light with projecting rays through the video image (bright curve) determines the position of points in space. The actual output of the device is a set of (x,y,z) coordinates of points in the scene. These coordinates are stored in three two-dimensional arrays of size $m \times n$.

2.1. Surface Fitting

In this section we discuss the details of the surface fitting technique used in computing object descriptions.

The surface fitting problem can be stated as follows: given $m \times n$ arrays of $x, y,$ and z coordinates, we want to determine a surface that approximates the data in the least-squares sense and is smooth. Smoothness is interpreted as the twice differentiability of the surface at all points. Fitting a surface to an $L \times L$ window of data is equivalent to computing a surface fit for a roughly rectangular grid of data values. Intuitively, a roughly rectangular grid is one that has been obtained from a rectangular mesh that has been deformed to fit the surface. The surfaces need not be single valued; i.e., one coordinate need not be a function of the other two, as in $z=f(x,y)$. In fact, the surfaces are represented in the parametric form

$$\begin{aligned} x &= f(s,t), \\ y &= g(s,t), \\ z &= h(s,t), \end{aligned} \tag{1}$$

where s and t are the parameters and the functions f, g, and h are tensor products of splines in tension. This fitting is a mapping from a two-dimensional parameter space to a three-dimensional Cartesian space.

Note that points (x,y,z) and (x,y,z') could both occur within the grid with $z \neq z'$. It is assumed however that no two adjacent rows and no two adjacent columns are identical.

More precisely, let $p_{ij} = (x_{i,j}, y_{i,j}, z_{i,j})$. We require

$$p_{i,j} \neq p_{i+1,j}, \tag{2}$$

$$p_{i,j} \neq p_{i,j+1} \tag{3}$$

for all $i=1, \ldots, m$ and $j=1, \ldots, n$. If these conditions are not satisfied then the parameterization fails.

The first step towards a solution to the surface fitting problem is to obtain the parametric grids $\{s_i\}_{i=1}^{m}$ and $\{t_j\}_{j=1}^{n}$ so that for each $i = 1, \ldots, m$ and each $j = 1, \ldots, n$ there exist s_i and t_j such that

$$(x(s_i, t_j), y(s_i, t_j), z(s_i, t_j)) = (x_{i,j}, y_{i,j}, z_{i,j}) \tag{4}$$

where x, y and z are twice differentiable functions of s and t. It is required to have $s_1=t_1=0$ and $s_m=t_n=1$. The quantity $s_{i+1} - s_i$ is made equal to the average (over $j = 1, \ldots, n$) normalized distance between $p_{i+1,j}$ and $p_{i,j}$. Similarly, $t_{j+1}-t_j$ is made equal to the average (*over* $i = 1, \ldots, m$) normalized distance between $p_{i,j+1}$ and $p_{i,j}$. Precisely,

$$s_{i+1} = s_i + \frac{1}{n} \sum_{j=1}^{n} \left(\frac{\|p_{i+1,j} - p_{i,j}\|}{\sum_{l=1}^{m-1} \|p_{l+1,j} - p_{l,j}\|} \right), \tag{5}$$

$$t_{j+1} = t_j + \frac{1}{m} \sum_{i=1}^{m} \left(\frac{\|p_{i,j+1} - p_{i,j}\|}{\sum_{l=1}^{n-1} \|p_{i,l+1} - p_{i,l}\|} \right), \tag{6}$$

where $\|p_{i+1,j} - p_{i,j}\|$ is the Euclidean distance. It is easy to show, given these definitions and $s_1=t_1=0$, that indeed $s_m=t_n=1$.

Having determined the parametric grids, the surface approximation problem is now reduced to three standard surface fitting problems over rectangular grids, namely

- $x(s_i, t_j) = x_{i,j}$ *for* $i=1, \ldots, m$ *and* $j=1, \ldots, n$
- $y(s_i, t_j) = y_{i,j}$ *for* $i=1, \ldots, m$ *and* $j=1, \ldots, n$
- $z(s_i, t_j) = z_{i,j}$ *for* $i=1, \ldots, m$ *and* $j=1, \ldots, n$

These problems can be solved using the following formulation by Cline [3], [4], [5]. We are given a grid defined by two strictly increasing abscissa sets, $\{s_i\}_{i=1}^m$ and $\{t_j\}_{j=1}^n$, an ordinate set $\{\{z_{ij}\}_{i=1}^m\}_{j=1}^n$, positive weights $\{\delta s_i\}_{i=1}^m$ and $\{\delta t_j\}_{j=1}^n$, a non-negative tolerance S, and a non-negative tension factor σ. We wish to determine a function g which minimizes

$$\int_{s_1}^{s_m}\int_{t_1}^{t_n}\{g_{sstt}(s,t)\}^2 ds dt + \sigma^2 \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\int_{s_i}^{s_{i+1}}\int_{t_j}^{t_{j+1}}\{Q_{ij}(s,t)\}^2 ds dt \qquad (7)$$

over all functions $g \in C^2[s_1,s_m]\times[t_1,t_n]$ such that

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\left\{\frac{(g(s_i,t_j)-z_{ij})}{\delta s_i\,\delta t_j}\right\}^2 \leq S. \qquad (8)$$

where

$$Q_{ij}(s,t) = g_{st}(s,t) - \left\{\frac{(g(s_{i+1},t_{j+1}) - g(s_i,t_{j+1}) - g(s_{i+1},t_j) + g(s_i,t_j))}{(s_{i+1}-s_i)(t_{j+1}-t_j)}\right\}$$

It can be shown that the solution to this problem is a tensor product of splines under tension (i.e., g is a one dimensional spline under tension when restricted to any value s or to any value of t). Also, g can reproduce local monotonicity, local positivity and local convexity as the one dimensional smoothing spline under tension can [5]. Because all three subproblems have several components in common it is possible to solve them concurrently.

The surface fitted to the roughly rectangular grid can be represented by three tensor products of splines under tension:

$$x(s,t) = \sum_i \sum_j \alpha_{ij}\, \phi_i(s)\psi_j(t)\,,$$

$$y(s,t) = \sum_i \sum_j \beta_{ij}\, \theta_i(s)\nu_j(t)\,, \tag{9}$$

$$z(s,t) = \sum_i \sum_j \gamma_{ij}\, \rho_i(s)\kappa_j(t)\,,$$

where ϕ, ψ, θ, ν, ρ and κ are tension splines; and α, β, and γ are coefficients of the tensor products.

2.2. Curvature Computation

In this section we discuss the computation of principal curvatures at a point on the parameterized, fitted, surface. Principal curvatures at a point on a surface indicate how fast the surface is pulling away from its tangent plane at that point. In the presence of an edge the principal curvatures will achieve a local maximum. Therefore it is appropriate to declare as edges those points at which the curvature is above a given threshold. The partial derivative information required to compute the principal curvatures are provided by the surface fitting software [5].

In general, when a plane passing through the normal to the surface at a point P is rotated about this normal, the radius of curvature of the section changes; it will be at a maximum r_1 for a normal section s_1 and a minimum r_2 for another normal section s_2 [11]. The reciprocals $k_1=1/r_1$ and $k_2=1/r_2$ are called the principal curvatures; the directions of tangents to s_1 and s_2 at P are called the principal directions of the surface at P (see Figure 2). The Gaussian curvature at point P is defined as the product of the two principal curvatures. The Gaussian curvature is an intrinsic property of the surface,

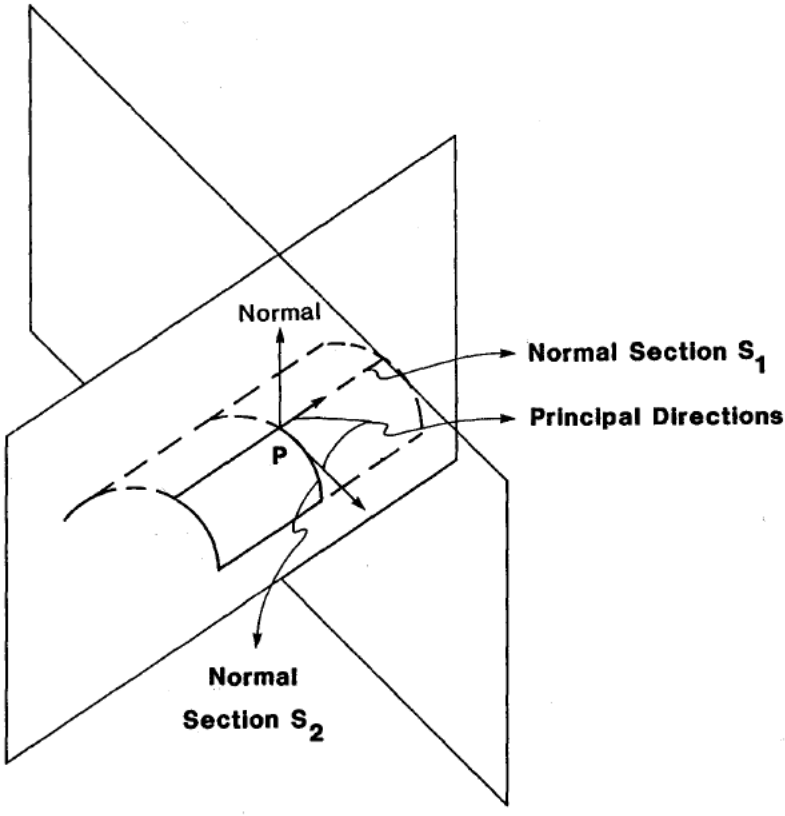


Figure 2: A surface with normal sections along the principal directions

since it depends only on the coefficients of the first fundamental form and their derivatives, whereas the two principal curvatures are extrinsic properties. The coefficients of the first and second fundamental forms (E, F, G and e, f, g as given below) determine the surface uniquely up to a rigid body transformation [7]. The principal curvatures at a point on a surface

are computed in terms of the parameters s and t as follows. Let $\vec{X}(s,t)$ represent the surface.

$$\vec{X}(s,t) = (x(s,t), y(s,t), z(s,t)), \tag{10}$$

$$\vec{X}_s = dX/ds = [x_s\ y_s\ z_s], \tag{11}$$

$$\vec{X}_t = dX/dt = [x_t\ y_t\ z_t], \tag{12}$$

$$\vec{X}_{st} = d^2X/dsdt = [x_{st}\ y_{st}\ z_{st}], \tag{13}$$

$$\vec{X}_{ss} = d^2X/ds^2 = [x_{ss}\ y_{ss}\ z_{ss}], \tag{14}$$

$$\vec{X}_{tt} = d^2X/dt^2 = [x_{tt}\ y_{tt}\ z_{tt}]. \tag{15}$$

Let

$$E = |\vec{X}_s|^2 = (x_s^2 + y_s^2 + z_s^2), \tag{16}$$

$$F = \vec{X}_s \cdot \vec{X}_t = (x_s x_t + y_s y_t + z_s z_t). \tag{17}$$

$$G = |\vec{X}_t|^2 = (x_t^2 + y_t^2 + z_t^2). \tag{18}$$

The unit normal is then given by

$$\vec{X} = \frac{\vec{X}_s \times \vec{X}_t}{\sqrt{EG - F^2}}, \tag{19}$$

The Gaussian curvature K at any point on a surface is defined as the

product of two principal curvatures k_1 and k_2, and is given by

$$K = k_1 k_2 = \frac{eg - f^2}{EG - F^2}, \tag{20}$$

where e, f and g are

$$\begin{aligned} e &= \vec{N} \cdot \vec{X}_{ss}, \\ f &= \vec{N} \cdot \vec{X}_{st}, \\ g &= \vec{N} \cdot \vec{X}_{tt}. \end{aligned} \tag{21}$$

The mean curvature H is given by

$$H = \frac{(k_1 + k_2)}{2} = \frac{eG - 2fF + gE}{2(EG - F^2)}, \tag{22}$$

and the principal curvatures are

$$k_1 = H - \sqrt{H^2 - K}, \tag{23}$$

$$k_2 = H + \sqrt{H^2 - K}. \tag{24}$$

The above formulas are used to compute Gaussian curvature, and points on a surface are classified according to its sign. Points for which $K > 0$ are called elliptic, $K < 0$ are called hyperbolic, $K = 0$ are called parabolic, $K =$ constant (i.e., $k_1 = k_2$) are called umbilic, and $k_1 = k_2 = 0$ are called planar umbilic [7]. This classification will be used subsequently to group object points into regions.

3. Computation of Object Description

In this section we present an algorithm to compute the object description in terms of jump boundaries, internal edges, and regions homogeneous in sign of Gaussian curvature. Our algorithm is:

1. Divide the range image into overlapping windows.
2. Detect jump boundaries and fit patches to windows not containing jump boundaries.
3. Compute the principal curvatures and extract edge points.
4. Classify each non-edge point in a patch into one of parabolic, elliptic, hyperbolic, umbilic or planar umbilic, and group all points of the same type in a patch and its neighboring patches into a region.

The details of each step in the algorithm follow. In step 1 the division of the input arrays of x, y, and z coordinates into $L \times L$ windows should be overlapped in order to ensure that an internal edge is always contained in a patch.

In step 2 the standard deviation of the Euclidean distance between adjacent data points in the $L \times L$ window gives an indication of the scatter in the data. In the vicinity of the jump boundaries of an object the Euclidean distance between adjacent points will be large. This will cause the standard deviation of the windows of data containing such jump boundaries to be very high. By setting a threshold on the standard deviation, jump boundaries can be detected. After the jump boundaries have been detected, smooth patches are fitted to the data in the windows not containing a jump discontinuity. In the presence of a jump boundary, the window size is adapted so that the window no longer contains the jump boundary. The sizes of the objects in the scene are assumed to be much larger than the $L \times L$ window, allowing us to detect fine detail such as internal edges in an object. Fitting a surface to an $L \times L$ window of data is equivalent to computing a surface fit to a roughly rectangular grid of data values, as mentioned in the

previous section. The surfaces are represented parametrically, and this fitting is a mapping from the two-dimensional parameter space to three-dimensional space.

In step 3 of the algorithm we detect all those points that belong to or fall on internal edges of objects in the scene. Intuitively it is reasonable to expect high curvature to occur in the vicinity of an edge, and we define a point P in a patch as an edge point if the absolute value of one of the principal curvatures becomes greater than a threshold [31].

Due to the presence of noise in the data and inappropriate choice of thresholds for detecting edges, clusters of edge points may appear in the vicinity of a true edge position. In order to eliminate these clusters of computed edge points, non-maxima suppression is applied at every edge point in a direction perpendicular to the edge, which is the direction associated with the maximum absolute principal curvature. Non-maxima suppression will declare no edge present at points where the curvature is not a local maximum.

The next step is to group object points into homogeneous regions. All points in a patch are classified as either elliptic, hyperbolic, parabolic, planar umbilic, or umbilic. All points of the same type are grouped together into a large region. The merging of points does not require explicit fitting of a new surface to this homogeneous region. This is because we can make the second derivatives across patch boundaries match one another, thereby resulting in a C^2 surface [6]. Each homogeneous region is assigned a label depicting its type (elliptic, hyperbolic, parabolic, umbilic, or planar umbilic). The grouping of points into regions immediately follows the surface fitting process.

The above definition of a region allows for internal edges to occur within regions. The extent of regions can be delineated by (a) jump boundaries, (b) internal edges, and (c) curvature edges, which we define as places where there is a change in curvature-based classification. Figure 4(d) shows an

example of a wedge, where the regions are bounded by jump boundaries and internal edges. For instance, Figure 8(c) depicts an example of a coffee jar, where the regions are bounded by jump boundaries and curvature edges.

The object representation in terms of regions and curvature based properties of regions, has the advantage of being invariant under the transformation of independent parameters of the surface or rigid body transformations and is therefore independent of viewpoint. In addition, our algorithm for computing object descriptions has the advantage of being able to operate directly on the laser range data without applying any coordinate transformations, and uses existing, tested public domain software.

4. Experimental Results

This section presents examples of the object description algorithm. Real and synthetic data are used to demonstrate the performance of the algorithm.

We consider three elementary surfaces and a composite surface in our examples. Elementary surfaces are those that contain regions made up of one of parabolic, umbilic, planar umbilic, hyperbolic or elliptic points. A composite surface is one that is made up of a combination of elementary surfaces. The three elementary surfaces are chosen because they constitute regions that occur most often in practice. The composite surface is chosen to demonstrate the performance of the algorithm on complex objects composed of different types of regions.

In order to extract maximal regions of an object from a scene in the presence of noise, preprocessing of the range data is necessary. A simple filtering scheme, neighborhood averaging, is applied to the range image to smooth out noise in the data.

Synthetic data are obtained using the approach discussed in [16]. After completing the surface fitting process to each window of data, the (x,y,z) coordinate values on the surface are converted into intensity values using

Lambert's law of direct illumination [21]; this is done to facilitate easy viewing of the reconstructed surface. Figure 3(a) shows the range image of an automobile. The image has been pseudo-colored; red indicates points far from the viewpoint and blue indicates points nearer to the viewpoint. Figure 3(b) illustrates the reconstructed object. Figure 3(c) depicts the

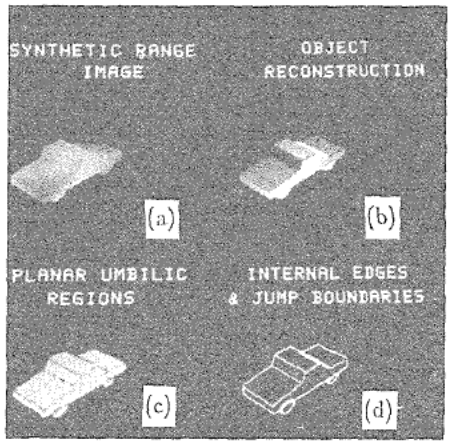


Figure 3: (a) Synthetic range image; (b) Object reconstruction; (c) Planar umbilic regions on the automobile; (d) Jump boundaries and internal edges

various regions extracted. In this case all the regions obtained are planar-umbilic regions. Regions in this and all the examples to follow are colored with the following code:

Yellow	Planar umbilic
Red	Parabolic
Green	Elliptic
Blue	Hyperbolic
Cyan Blue	Umbilic

The results are consistent with the theoretical predictions. Figure 3(d) depicts the various jump boundaries (in dark) and internal edges (in light) of

the automobile.

The real data are obtained using the laser scanner described earlier. Figure 4(a) shows the pseudo-colored range image of a wedge obtained from

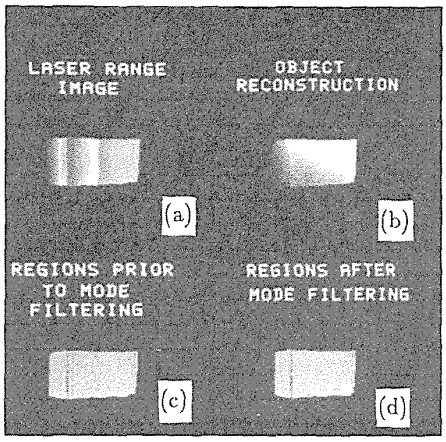


Figure 4: (a) Laser range image; (b) Reconstructed object; (c) Region classification prior to mode filtering; (d) Planar umbilic regions obtained after mode filtering

a laser range finder. A number of points in the scene have no (x,y,z) coordinate values; such points are called holes, (Figure 5). Holes are caused, when parts of the scene illuminated by the laser are not visible to the camera or when parts of the scene visible to the camera are not illuminated by the laser. Figure 4(b) depicts the reconstructed object, Figure 4(c) shows the regions extracted from the range image. Note that the regions obtained are not maximal. In order to extract maximal regions, mode filtering is applied to the initial region classification shown in Figure 4(c). Mode filtering replaces each label of a point in a region by the most dominant label in its neighborhood. Figure 4(d) shows the regions obtained after applying mode

filtering to the Figure 4(c). Every region obtained is a planar umbilic region. The results obtained here are in agreement with the theoretical predictions.

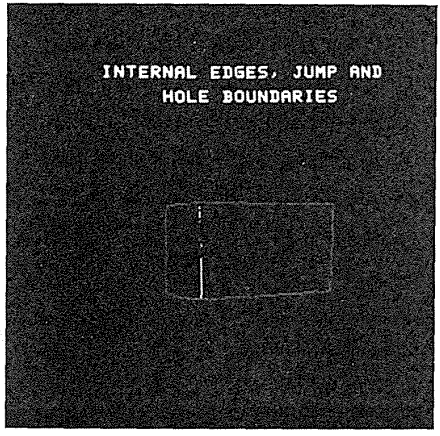


Figure 5: Jump, hole boundaries and internal edges on the wedge

Figure 5 shows the hole boundaries and internal edges. Figure 6(a) shows the pseudo-colored range image of a cylinder placed on a block, obtained from the laser scanner. Figure 6(b) shows the reconstructed objects. Figure 6(c) depicts the regions extracted (after mode filtering) from the range image. In this case there are two regions, one parabolic (on the cylinder) and the other planar umbilic (on the block). Figure 6(d) shows the hole and jump boundaries detected in the range image. Figure 7(a) shows another example of objects consisting of elementary surfaces, a balloon placed on a block. Figure 7(b) is the reconstructed objects and Figure 7(c) depicts the regions extracted after mode filtering. In this case there are two regions, one elliptic (on the balloon) and the other planar umbilic (on the block). Figure 7(d) shows the jump boundaries and hole boundaries on the balloon and the block.

Finally, Figure 8(a) shows the pseudo-colored range image of a composite

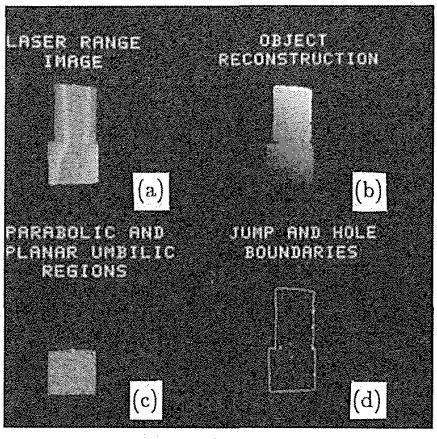


Figure 6: (a) Laser range image of a cylinder placed on a block; (b) Reconstructed objects; (c) Parabolic (cylinder) and planar umbilic (block) regions; (d) Jump and hole boundaries on the objects

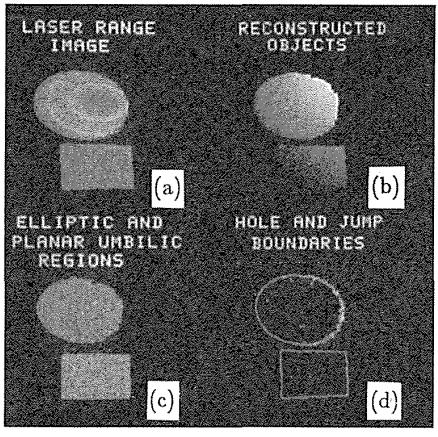


Figure 7: (a) Laser range image of a balloon placed on a block; (b) Reconstructed objects; (c) Elliptic (balloon) and planar umbilic (block) regions; (d) Jump and hole boundaries on the objects

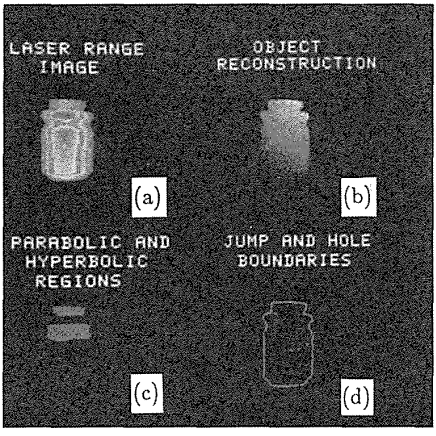


Figure 8: (a) Laser range image of a coffee jar; (b) Reconstructed object; (c) Parabolic and hyperbolic regions on a coffee jar; (d) Jump boundaries on a coffee jar

surface (a coffee jar) obtained from the laser scanner. Figure 8(b) shows the reconstructed object. Figure 8(c) depicts the regions extracted after mode filtering. In this case and as expected, there are four regions, two parabolic and two hyperbolic as shown in Figure 8(c). Figure 8(d) depicts the jump boundaries and hole boundaries on the coffee jar.

It should be noted that surface fitting is an important operation in the overall process. Indeed, refer to Figures 9(a), 9(b), 9(c), and 9(d) which show the classification results obtained when the principal curvatures are computed directly. These results confirm the need for smoothing (surface fitting) the raw data prior to computation of principal curvatures. The results of direct computation are very sensitive to noise in the data, and misclassification is high as can be seen from the examples.

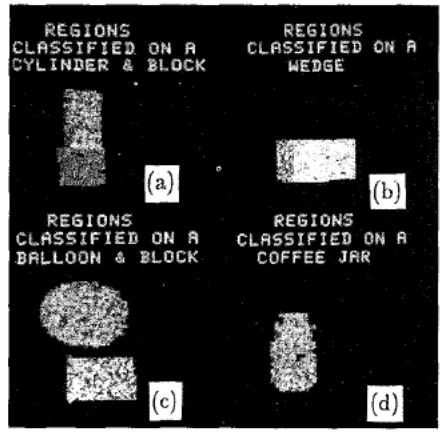


Figure 9: (a) Region classification of a cylinder placed on a block; (b) Region classification of a wedge; (c) Region classification of a balloon placed on a block; (d) Region classification of a coffee jar

The examples chosen constitute both planar and curved surfaces of fair complexity. The results obtained are most often in agreement with the theoretical predictions.

5. Summary

This paper has presented an approach where 3-D objects are represented by regions which are defined as a collection of patches homogeneous in curvature-based surface properties.

The input to the object description algorithm consists of a collection of 3-D points which need not be represented with one coordinate as a function of the other two. First a local process fits a smooth, tension spline based surface to the data. The surface fitting algorithm is general, efficient (linear time [4]) and uses existing public domain numerical software. The principal

curvatures are then computed and the surface points are classified accordingly. Regions on the surface of an object are grown on the basis of this classification. The object description so obtained is viewpoint independent and hence will prove to be valuable in the context of object recognition.

Future research will be directed towards organizing the region descriptions in a meaningful fashion (segmentation) and developing efficient algorithms for matching object descriptions with model descriptions to achieve recognition.

Acknowledgments

The work was supported in part by the Air Force Office of Scientific Research under Contract F49620-85-K-0007 and in part by National Science Foundation under contract DCR8517583.

References

[1] Agin, G. J. and Binford, T. O.
Computer description of curved objects.
In *Proceedings of the Third International Joint Conference on Artificial Intelligence*, pages 629-640. August, 1973.
Also appeared in IEEE Transactions on Computers, 1976, Vol. C-25, pages 439-449.

[2] Barnard, S. T. and Thompson, W. B.
Disparity analysis of images.
IEEE Trans. Pattern Anal. Mach. Intell. PAMI-2(4):333-340, 1980.

[3] Cline, A. K.
Curve fitting using splines under tension.
Atmos. Technology 3:60-65, September, 1973.

[4] Cline, A. K.
Smoothing by splines under tension.
Technical Report CNA-168, Univ. of Texas at Austin, Dept. of CS, 1981.

[5] Cline, A. K.
Surface smoothing by splines under tension.
Technical Report CNA-170, Univ. of Texas at Austin, Dept. of CS, 1981.

[6] Coons, S. A.
Surfaces for computer aided design of space forms.
MIT Project MAC TR-41, MIT, June, 1967.

[7] Do Carmo, M. P.
Differential Geometry of Curves and Surfaces.
Prentice-Hall, Inc., Englewood Cliffs, NJ, 1976.
pages 134-156.

[8] Duda, R. O., Nitzan, D., and Barrett, P.
Use of range and reflectance data to find planar surface regions.
IEEE Trans. Pattern Anal. Machine Intell. PAMI-1:259-271, July, 1979.

[9] Faugeras, O. D., Hebert, M., and Pauchon. E.
Segmentation of range data into planar and quadratic patches.
In *CVPR*, pages 8-13. 1983.

[10] Grimson, W. E. L.
An implementation of computational theory for visual surface interpolation.
CVGIP 22:39-69, 1983.

[11] Hilbert, D. and Cohn-Vossen, S.
Geometry and the Imagination.
Chelsea Publishing Co., New York, 1952.
pages 183-198.

[12] Horn, B.K.P.
Obtaining shape from shading information.
In P.H. Winston (editor), *The Psychology of Computer Vision*, pages 115-156. Mc Graw-Hill, New York, 1975.

[13] Jarvis, R. A.
A perspective on range finding techniques for computer vision.
IEEE Trans. Pattern Anal. Machine Intell. PAMI-5:122-139, March, 1983.

[14] Levine, M. D., O'Handley, D. A., and Yagi, G. M.
Computer determination of depth maps.
Computer Graphics and Image Processing 2:131-150, 1973.

[15] Lieberman, L. I.
Computer recognition and description of natural scenes.
PhD thesis, Univ. of Pennsylvania, 1974.

[16] Magee, M. J. and Aggarwal, J. K.
Intensity guided range sensing recognition of three dimensional objects.
Proc. CVPR :550-553, June, 1983.

[17] Marr, D.
Analysis of occluding contours.
Technical Report AI Memo 372, MIT, 1976.

[18] Marr, D. and Poggio, T.
Cooperative computation of stereo disparity.
Science 194:283-288, 1976.

[19] Marr, D. and Poggio, T.
A computational theory of human stereo vision.
Proc. R. Soc. Lond. (B204):301-328, 1979.

[20] Mitiche, A. and Aggarwal, J. K.
Detection of edges using range information.
In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pages 1906-1911. Paris, France, May, 1982.
Also appeared in IEEE Trans. on PAMI, Vol. 5, No. 2, March, 1983, pp. 174-178.

[21] Newman, W. M. and Sproull, R. F.
2nd edition: Principles of Interactive Computer Graphics.
McGraw-Hill, 1979.
pages 498-501.

[22] Nitzan, D., Brain, A. E., and Duda, R. O.
The measurement and use of registered reflectance and range data in scene analysis.
Proc. IEEE 65:206-220, Feb., 1977.

[23] Oshima, M. and Shirai, Y.
A scene description method using three-dimensional information.
Pattern Recognition 11:9-17, 1978.

[24] Rocker, F. and Kiessling, A.
Methods for analyzing three dimensional scenes.
Proc. 4th IJCAI :669-673, September, 1975.

[25] Rogers, B. and Graham, M.
Motion parallax as an independent cue for depth perception.
Perception 8:125-134, 1979.

[26] Shirai, Y. and Suwa, M.
Recognition of polyhedrons with a range finder.
In *Proceedings of the Second International Joint Conference on Artificial Intelligence*, pages 80-87. 1971.

[27] Stevens, K. A.
Representing and analyzing surface orientation.
Artificial Intelligence: An MIT Perspective.
P. H. Winston and R. H. Brown, 1980, pages 103-125.

[28] Technical Arts Corp.
White Scanner 100a User's Manual
Seattle, WA, 1984.

[29] Terzopolous, D.
Multiresolution computation of visible surface representation.
PhD thesis, MIT, Dept. of EE and CS, January, 1984.

[30] Ullman, S.
The interpretation of visual motion.
MIT Press, Cambridge, MA, 1979.

[31] Vemuri, B. C. and Aggarwal, J. K.
3-D reconstruction of objects from range data.
In *7th International Conference on Pattern Recognition*, pages 752-754. Montral, July, 1984.

[32] Will, P. M. and Pennington, K. S.
Grid coding: A preprocessing technique for robot and machine vision.
2nd Int. Joint Conf. on AI 2:319-329, 1971.

Use of Vertex-Type Knowledge for Range Data Analysis

Kokichi Sugihara[1]

Department of Mathematical Engineering
and Instrumentation Physics
University of Tokyo
Hongo, Bunkyo-ku
Tokyo 113, Japan

Abstract

This paper describes a knowledge-guided system for range data analysis. Inputs to the system are raw range data obtained by the slit-light projection method and outputs are surface patch descriptions (i.e., organized structures of vertices, edges, and faces) of the visible part of the objects. A junction dictionary is constructed to represent the knowledge about the physical nature of the object world, and is used by the system for the extraction and organization of edges and vertices. At each step of the analysis, the system consults the dictionary to predict positions, orientations and physical types of missing edges. Those predictions guide the system to decide where to search and what kinds of edges to search for as well as how to organize the extracted features into a description.

1. Introduction

Three-dimensional range data contain rich information about object shapes. However, raw range data are simply collections of distance values from the viewer to surface points, and consequently are not directly suitable for use as object identification and object manipulation; they should be condensed into more concise descriptions.

Generalized cylinders [1], [11] and surface patches [12], [13], [5], [3] are the prevailing tools in terms of which the descriptions are constructed from

[1]Previously in the Department of Electrical Engineering, Nagoya University

range data. The generalized cylinders are based on natural principal axes of objects, and so enable us to construct object-centered descriptions relatively easily [9]. Descriptions based on generalized cylinders, however, are unstable or non-unique if the objects are not composed of elongated parts. The surface patches, on the other hand, can be used for non-elongated objects, but it is usually difficult to segment smooth, curved surfaces into patches in a stable manner. If the objects are limited to polyhedrons, however, the surfaces can be decomposed into natural patches, that is, planar faces. Thus, polyhedral scenes can be described naturally in terms of surface patches.

This paper presents a method for extracting surface patch descriptions of polyhedral objects from range data [18], [20]. In doing this, knowledge about vertex types is represented in terms of a junction dictionary, and used for constructing scene descriptions. For this purpose we borrow some results in the study of interpretation of line drawings.

One of the greatest milestones in interpretation of line drawings is a labelling scheme proposed by Huffman [6] and Clowes [2]. They introduced labels to represent physical properties of edges (such as convex and concave edges), and found that the number of junction types that can appear in labeled line drawings is very small. On the basis of this observation they proposed the labeling scheme in which, once the list of all possible junctions is constructed and registered, line drawings can be interpreted by finding consistent assignments of labels in the sense that all the resulting junctions belong to the list. This scheme has been extended to pictures with cracks and shadows [22], pictures with hidden lines [19], and pictures of the Origami world [8]. Thus, junction knowledge has been used for categorizing lines in line drawings.

In the case of range images, on the other hand, we can classify convex, concave, and some other types of lines directly from the data without using any additional information; the junction knowledge is not necessary for

categorizing lines. Therefore, this knowledge can be used for another purpose, that is, for predicting missing edges. Comparing a line drawing extracted from the range data with the list of the possible junctions, we can tell whether the present drawing is consistent with physical reality. If it is not, we can predict the locations and physical categories of possible missing edges. The extraction of edges can be efficiently guided by those predictions. This approach is practical, since noisy raw range data can be dealt with, while the junction knowledge has been used for the analysis of "perfect" line drawings in the previous studies.

In Section 2 we introduce a local operator that can extract edges from raw range data, and in Section 3 we construct a junction dictionary, which is a condensed representation of vertex-type knowledge. Using these two tools, we construct a system for extracting surface patch descriptions of the scenes, which is described in Section 4.

2. Edge Detection from Range Data

This section describes the method for detecting edges from range data. First, the representation of range data is explained, then we define a local operator to detect edge points, and finally we present the strategy to track edge points, which can extract selectively a particular type of a line running in a particular direction.

2.1. Range Data

One way to get range information is using a rangefinder by active illumination. As shown in Figure 1, a beam of light is projected from the source, say O, onto an object, and the illuminated spot is observed from another angle by a camera. Let $P(X,Y,Z)$ be the illuminated spot on the surface, and P' be its image. Furthermore, let $D(P)$ denote the distance from the left margin of the image to the image point P'. This $D(P)$ is what we can observe directly by the method in Figure 1. Indeed, if a TV camera is used, $D(P)$ can be measured as the time interval from the time when the

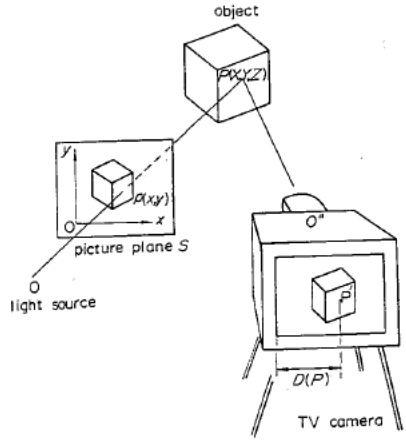


Figure 1: Range Finding Method

horizontal scan begins to the time when the video signal shows the maximum intensity.

Let us place a virtual plane, say S, with a Cartesian coordinate system (x,y) between the light source O and the object, and let $p\,(x,y)$ be the point of intersection of the light beam OP and the plane S. Since the direction of the light beam OP is specified uniquely by the point $p\,(x,y)$ on S, the value $D(P)$ can be considered as a function of the point $p\,(x,y)$. Hence, we hereafter denote $D(x,y)$ instead of $D(P)$.

When the scene is swept by the light beam, the point $p\,(x,y)$ sweeps some region on S and we get raw data $D(x,y)$ at many points (x,y), say

$$\{D(i,j) \mid i=1,\ldots,m,\ \ j=1,\ldots,n\}\,,$$

where we use (i,j) instead of (x,y) in order to indicate that the data are obtained only at discrete points. This collection of the data is similar to a usual digital image in that the pixel values are defined at grid points (i,j) on the image plane, i.e., the plane S. The only difference is that the values $D(i,j)$'s defined above are not light intensity values at the grid points. We

shall call the collection of $D(i,j)$'s a raw range picture.

If we use a spot light, $D(i,j)$ is measured only at one point for each light projection, which requires a long time to measure ranges for the whole scene [7]. For practical purposes, therefore, we project light through a narrow vertical slit so that we can obtain many data $\{D(i,j) \mid j=1, \ldots, n\}$ along a vertical line simultaneously [16] (see also [10] and [15]).

2.2. Edge Detecting Operators

To construct an operator for detecting edges from a range picture, we need to consider the relations between geometrical configurations of objects and the values of $D(i,j)$'s. Let ξ be a line on a picture plane. Let $p_\xi(s)$ denote a point which moves along the line ξ with a line parameter s (that is, s is the length along the line from a certain fixed point on ξ to $p_\xi(s)$). Further, let $P_\xi(s)$ denote the corresponding point on the three-dimensional surface of the object. Referring to Figure 2, we will examine how $D(P_\xi(s))$ varies as $P_\xi(s)$ moves across geometrical features of objects. The following observations are intuitively obvious.

When $P_\xi(s)$ moves on a flat surface, $D(P_\xi(s))$ is approximately a linear function of s as shown in Figure 2(a) (indeed, it is strictly linear if the light is projected in parallel and its image is obtained orthographically). When $P_\xi(s)$ moves across a convex edge, $D(P_\xi(s))$ changes continuously and bends upward in the s–D plane as illustrated in Figure 2(b). Similarly, when $P_\xi(s)$ moves across a concave edge, $D(P_\xi(s))$ bends downward (see Figure 2(c)). Note that these upward and downward bends corresponding to convex and concave edges, respectively, are true only when the camera is to the right of the light source. If the camera and the light source exchange their locations to each other, the changes of $D(P_\xi(s))$ at a convex edge and at a concave edge are reversed. Finally, when $P_\xi(s)$ moves across an obscuring edge, $D(P_\xi(s))$ changes discontinuously as shown in Figure 2(d).

According to our observation we can devise the following operator to

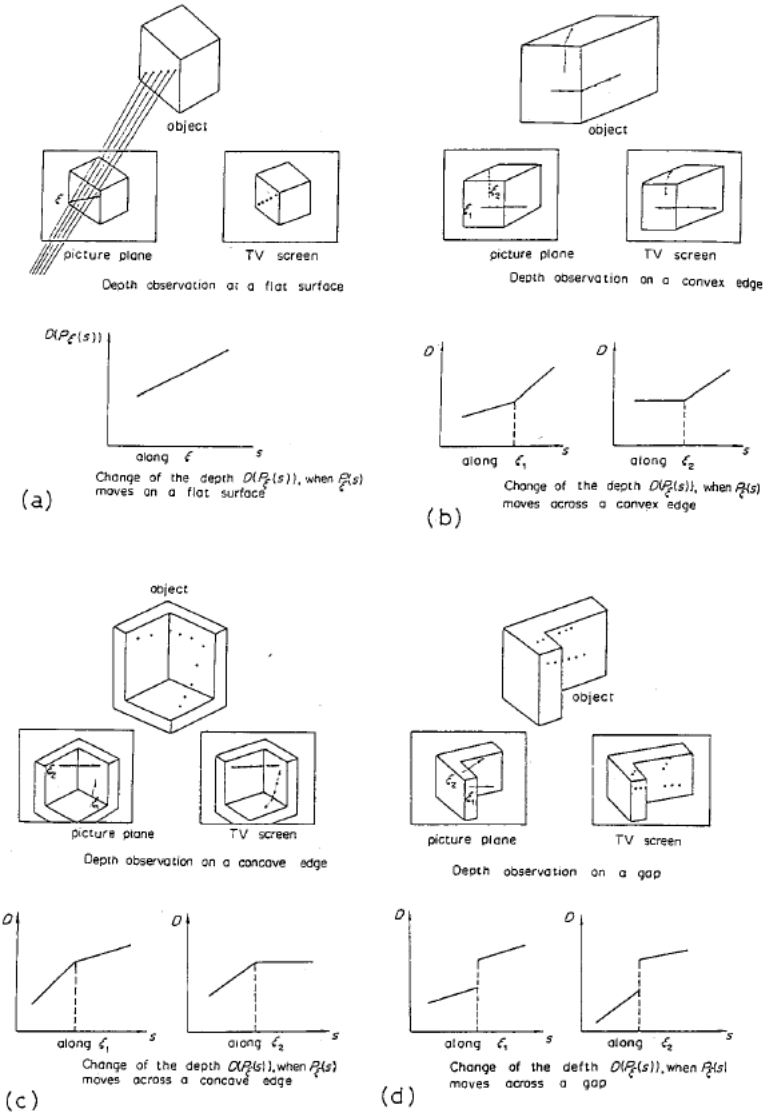


Figure 2: Changes of $D(P_\xi(s))$ as $P_\xi(s)$ Moves Across Various Features on the Object

detect edges. Consider three points $p_\xi(s)$, $p_\xi(s+\Delta s)$, and $p_\xi(s-\Delta s)$ along the line ξ on the image plane S, where Δs is a short length (see Figure 3).

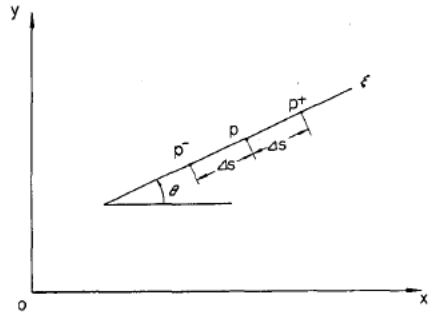


Figure 3: Three Points Used by the Edge Detecting Operator

Let θ represent the orientation of the line ξ in the image and measured counterclockwise from the x axis. For simplicity in notation, let

$$p^+ = p_\xi(s+\Delta s) \text{ and } p^- = p_\xi(s-\Delta s)$$

as illustrated in Figure 3. We will define the operator $O_\theta(p)$ by

$$O_\theta(p) = \frac{1}{2\Delta s}[D(p^+) + D(p^-) - 2D(p)]. \tag{1}$$

Figure 4 shows profiles of $O_\theta(p)$ when P moves across various types of features of the object corresponding to the cases in Figure 2. We can see that $O_\theta(p) = 0$ when P moves on a planar surface, $O_\theta(p) > 0$ when P moves across a convex edge, and $O_\theta(p) < 0$ when P moves across a concave edge.

For practical purposes we can redefine the operator on a digital range picture by restricting $\theta = 0, \pi/4, \pi/2$, and $3\pi/4$:

$$O_0(i,j) = [D(i+k,j) + D(i-k,j) - 2D(i,j)]/(2k), \tag{2}$$

$$O_{\pi/2}(i,j) = [D(i,j+k) + D(i,j-k) - 2d(i,j)]/(2k), \tag{3}$$

$$O_{\pi/4}(i,j) = [D(i+k,j+k) + D(i-k,j-k) - 2D(i,j)]/(2\sqrt{2}k), \tag{4}$$

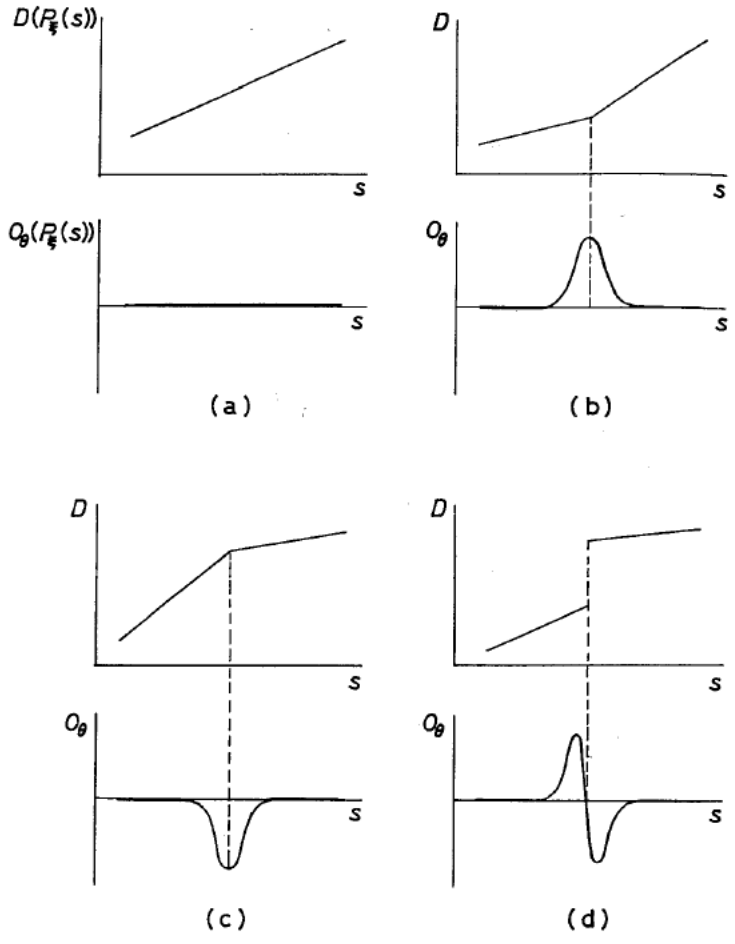


Figure 4: Behaviors of the Edge Detecting Operator

$$O_{3\pi/4}(i,j)=[D(i+k,j-k)+D(i-k,j+k)-2D(i,j)]/(2\sqrt{2}k). \quad (5)$$

As we have just seen, $O_\theta(i,j)$ takes the greatest absolute value when an edge

lies in the direction $\theta + \pi/2$.

In addition to the directional operators, we can also consider nondirectional operators:

$$O_{M4}(i,j) = \frac{1}{4}\sum_{\theta=0,\pi/4,\pi/2,3\pi/4} O_{\theta}(i,j), \tag{6}$$

$$O_{M2}(i,j) = \frac{1}{2}[O_0(i,j) + O_{\pi/2}(i,j)], \tag{7}$$

The operator in (6) or (7) will usually have a non-zero value when P is on an edge, but it does not give information about the direction of the edge.

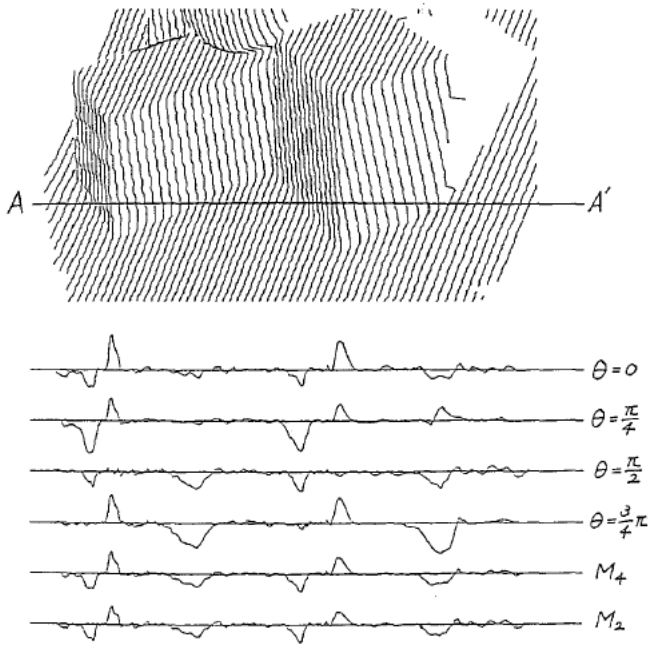


Figure 5: Output of the Edge Detecting Operators Along the Line AA' in the Range Picture

Figure 5 shows the output of the operators from (2) through (5), (6), and

(7) when the point $p(i,j)$ moves along the horizontal line AA' in the range picture, where the range picture is displayed as the image of vertical light stripes. We can see that the operators in (2) through (5) are sensitive to particular edge directions. In contrast, the operators in (6) and (7) have non-zero values at all edges. Moreover, the values of all the operators are positive at convex edges and negative at concave edges. There is not a great deal of difference between the value of O_{M4} and that of O_{M2}, and thus we mainly use only O_{M2}, which is more economical in computational time than O_{M4}.

2.3. Edge Tracking

The previous subsection presented operators to detect edge points locally. In order to extract edge lines, the following edge-tracking strategy is used. We first scan the range picture with the nondirectional operator O_{M2} or O_{M4}, and when we come across a point where the absolute value of the operator output is large enough, that is, a promising point for an edge, we then determine the direction of the line using the four directional operators from (2) through (5). We next follow the line in the given direction from that promising point. That is, we search the neighboring area for the next point along the line step by step and construct a string of points forming the edge line.

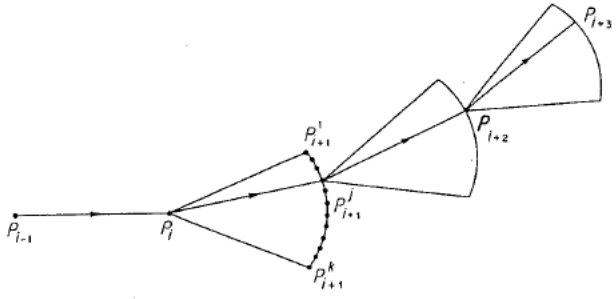


Figure 6: Edge Following

Figure 6 illustrates these steps for edge following. Suppose that we have

just found a new point p_i by searching the neighboring area of a point p_{i-1}. Then, in the next step we search the fan-shaped area spanning from the pivotal point p_i as shown in Figure 6. For practical purposes, the discrete points $p^1_{i+1}, p^2_{i+1}, \cdots, p^k_{i+1}$ on the arc of the fan-shaped area are chosen as the candidates for the next point, and the edge detecting operator is applied to those candidate points. Since the expected direction of the line is given, we need only use one of the four operations for each candidate point. Figure 7 gives the associated direction θ for each candidate point used in our system. We consider 32 points around the pivotal point and divide them into four groups, one for each directional operator. Among the candidates, we choose the point with the maximum value of the operator if the lines are convex, or the one with the minimum value (i.e., the maximum absolute value with a negative sign) if the line is concave.

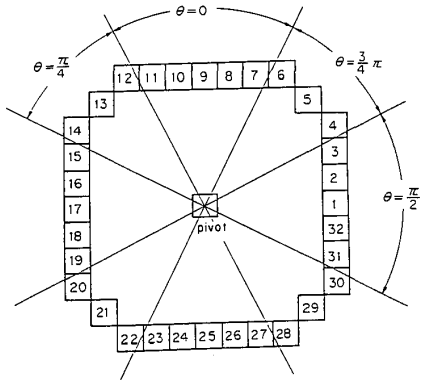


Figure 7: Thirty-two Candidate Points for the Next Point in the Edge Following

In our system we usually need not scan the whole picture to find the starting points for edge following, because the vertex-type knowledge suggests the starting point, the direction, and the type of an expected line as will be described later.

3. Junction Dictionary

Here we present our second tool for range data analysis, that is, a junction dictionary. The junction dictionary contains possible configurations at a junction, and can be considered as a kind of representation of knowledge about three-dimensional objects. It is used as a predictor of missing edges that should be extracted from range pictures.

3.1. Line Types

We consider a world consisting of solid polyhedrons, which are bounded only by planar *faces*. An *edge* is a line where two faces meet, and a *vertex* is a point where three or more faces meet. We call a collection of polyhedrons in a three-dimensional space a *scene*, and its projection on the plane a *picture*. The visible faces, edges, and vertices of the scene are associated with the polygonal *regions*, the straight *lines* (actually, line segments), and *junctions*, respectively, of the picture.

In the three-dimensional space there are only two types of edges for our planar faced objects: convex edges and concave edges. When a concave edge is visible, both of its side faces are also visible from the eye. Hence, we always call a line representing a concave edge a *concave line*, and represent it by a line segment with a minus (−) label. On the other hand, when a convex edge is visible, one of the two side faces is sometimes hidden by the other. Therefore, we distinguish between two types of lines representing convex edges: a *convex line* to represent a convex edge both of whose side faces are visible, and an *obscuring line* to represent a convex edge where one of the sides faces hides the other. A plus (+) label is used for a convex line and an arrow (→) is used for an obscuring line with the convention that, when one stands on the line facing the direction indicated by the arrow, both of the associated side faces are to the right.

The three types of labels above were introduced by Huffman [6] and have been used in various works on interpretation of line drawings. However,

those three types are not enough to constitute line drawings representing range pictures. In order to introduce another line type, we have to consider the method for obtaining range information.

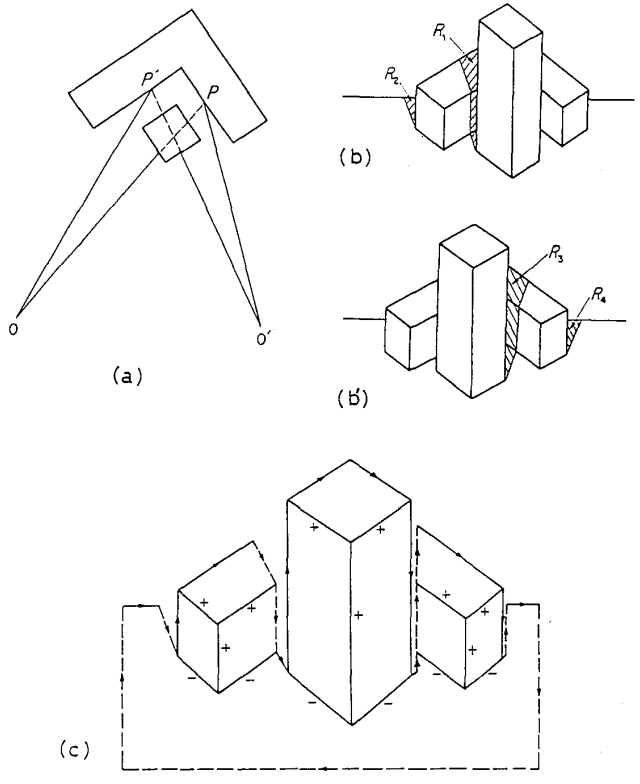


Figure 8: Line Labels for Range Pictures

A light-stripe rangefinder measures the range of a point on an object by seeing the point from two angles. As shown in Figure 8(a), we project a beam of light from a light source O and observe the illuminated spot with a camera at the point O'. This method does not measure the range of a point either when it is not illuminated (for example, see the point P in Figure 8(a))

or when it is not seen from the camera (point P'). Thus, after sweeping the scene with the beam of light, there remains some regions of points whose ranges are not determined. Examples of such regions are illustrated by hatched areas in Figure 8(b) and (b'). Figure 8(b) is the picture of the scene viewed from the light source, and hence the shaded regions represent surfaces that cannot be seen from the camera. On the other hand, Figure 8(b') is the picture seen from the camera, and the shaded regions represent surfaces that cannot be seen from the light source. We will call boundary lines of those obscured regions *obscured lines*; we will represent them by broken lines with arrows with the convention that, when we stand on the line facing the direction indicated by the arrow, the obscured region is to the left. Figure 8(c) shows the labeled line drawing of the picture in Figure 8(b) according to our conventions.

Since we have defined a range picture as that obtained by seeing the scene from the light source, unilluminated regions (for example, the regions R_3 and R_4 in Figure 8(b')) are always out of sight. Nevertheless we will represent the existence of those regions by a pair of contour lines: an obscuring line and an obscured line, parallel to each other and opposite in direction (see Figure 8(c)). Moreover, for convenience we will represent the margin of the picture plane also by obscured lines.

3.2. Possible Junctions

Now that we have introduced the four types of lines, we next enumerate possible junctions, that is, junctions that can appear in labeled pictures. For this purpose we restrict our objects to trihedral ones, that is, objects in which exactly three faces meet at each vertex.

Junctions in a picture usually correspond to vertices in a scene. However, there are junctions which are not images of actual vertices. One typical class of such junctions is found on an obscured line. An obscured line is caused when one body is in front of another. Therefore, a shape of an

obscured line is sometimes due to the shape of the obscured body, sometimes due to the shape of the obscuring body, and sometimes due to both of them. Although the configurations of obscured lines may contain much information about shapes of the objects, it seems difficult to extract helpful information for our purpose. Hence, we will ignore junctions on obscured lines except in the following case.

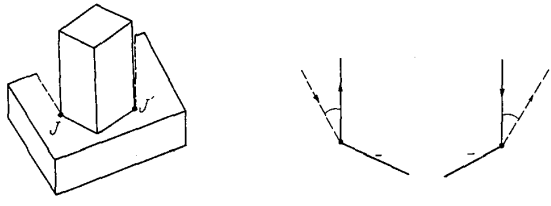


Figure 9: Shadow Causing Junctions and Their Representation

Figure 9(a) shows a line drawing of a scene with two blocks, one supporting the other. Junctions J and J' in the figure have the property that each of the junctions includes one obscuring line and one obscured line, and that the former causes the latter; in other words, the latter is a shadow of the former. We call a junction with this property a *shadow causing junction*. We represent a shadow causing relationship by an arc linking the obscuring line to the obscured one as illustrated in Figure 9(b). Any vertex that can generate a shadow causing junction in the picture has at least one convex edge and at least one concave edge, for only a convex edge can cast its shadow on a face associated with a concave edge.

The enumeration of possible junctions can be executed in a systematic manner. Let us consider an isolated vertex. Since the vertex is trihedral, there are exactly three faces meeting at the vertex. If the three faces are extended into unbounded planes, then they divide the surrounding space into eight parts, which we call *octants*. Some of the octants are occupied with material and the others are empty. The eye and the light source can only be in empty octants. Considering all the combinations of placing the

light source and the camera in empty octants, we can exhaust possible views of the vertex.

Actually, there are only four types of trihedral vertices: a vertex composed of one nonempty octant, that of three nonempty octants, that of five nonempty octants, and that of seven nonempty octants (see [6] for the details). Hence, by the enumeration method above we eventually find that there are only fourteen possible views of trihedral vertices provided that the light source and the camera are in general positions so that neither of them is coplanar with any faces. These fourteen junctions are listed in Figure 10,

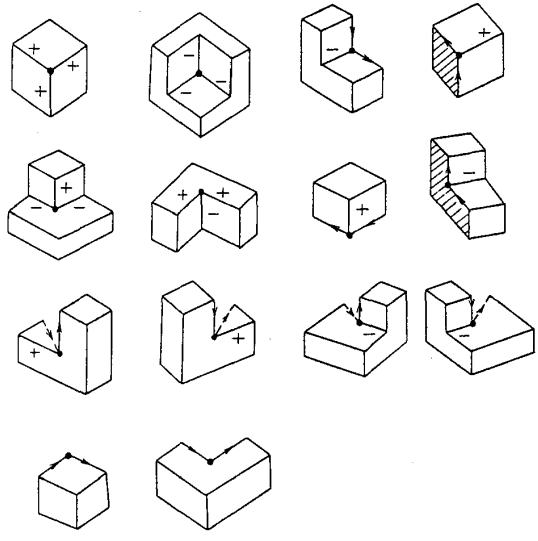


Figure 10: All Possible Junctions Resulting from Isolated Trihedral Vertices

where the shaded regions represent the regions that can be seen from the light source but cannot be seen from the camera.

If the scene contains two or more objects, other types of junctions can

also appear because contact of objects often gives rise to non-trihedral vertices. Let us assume that objects in the scene do not align accidentally, that is, neither two distinct edges nor two distinct vertices share a common position and no vertex lies on an edge of another object. Then, by the exhaustive search we can find that there are only six additional types of junctions as shown in Figure 11. After all, we have a total of twenty possible junctions, which are listed in Figure 12.

3.3. Representing a Junction Dictionary

A junction dictionary is represented in a form of a directed graph whose nodes are all subconfigurations of the possible junctions and whose arcs represent the relationships that one node can be changed into another node by an addition of one line.

Part of the junction dictionary is shown in Figure 13, where seven junctions and relationships between them are illustrated. The three junctions in rectangles (t,v, and y) are impossible ones and the four in circles (u,w,x, and z) are possible ones. The drawings beside the possible junctions show the examples of situations in which the junctions occur. An arrowed arc connecting two junctions denotes an addition of a line. If, for example, we add a convex line in the shaded area of the junction t in Figure 13, we obtain junction u. Adding a concave line to u results in w, and thus we get the path (t,u,w) from t to w. This kind of a path suggests where new lines exist and thus guides the analysis of the range data. For example, if a junction of the type u is found during the analysis, the dictionary tells us that a concave line "may" exist at that junction. If, on the other hand, a junction of the type v is found, the dictionary says that a convex line "must" exist to form w because v is impossible and w is the only possible junction that includes v as a subconfiguration. In this way, the junction dictionary can suggest what type of edges exist and in what directions they must be searched for near the vertex.

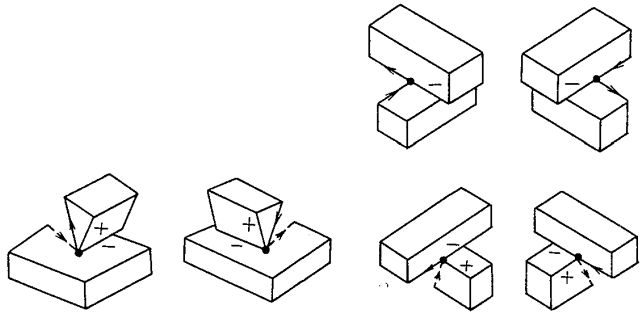


Figure 11: All Possible Junctions Resulting From Contact of Two Objects Without Accidental Alignments

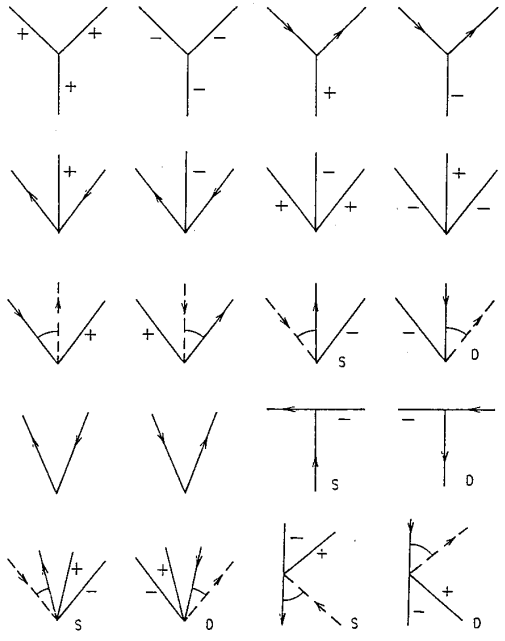


Figure 12: Final List of Possible Junctions

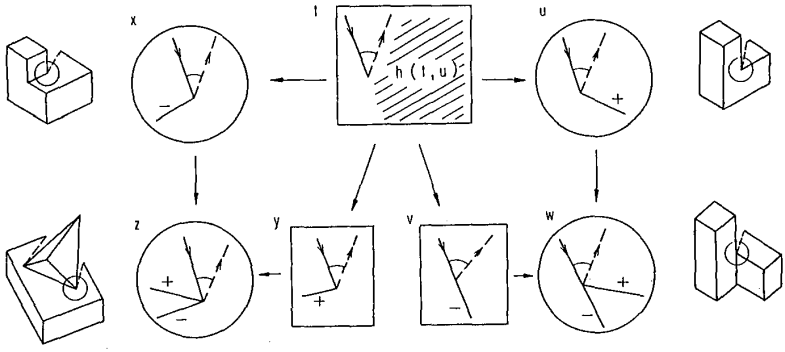


Figure 13: Part of the Junction Dictionary

3.4. Body Partitioning Rules

For object recognition it is useful to decompose a complicated scene into simple elements and to describe the spatial relationships among them. We show that the junction dictionary can also be used for the decomposition of a scene. Note that this decomposition is inherently ambiguous (see [4]). We get no information from a picture (no matter what kind of a picture it is, a light intensity picture or a range picture) to judge, for example, whether a block is merely placed on a desk or it is fixed to the desk with glue. Consequently, the rules we will propose here are necessarily empirical ones.

Our junction dictionary contains only such junctions that are found when polyhedrons in a scene are mutually in general positions; that is, without accidental alignments. However, when there are accidental alignments of objects in a scene, many new junction types can appear. If we enumerated all such accidental junctions, the junction dictionary would have become unmanageably large. Instead, the partitioning rules to be described will sometimes decompose an illegal junction (that is, a junction that is not included in the dictionary) into two legal ones. As a result, the partitioning rules help in handling cases when accidental alignments of bodies are involved.

The first partitioning rule is the following:

Partitioning Rule 1.

If there are two pairs of lines with arrows and both pairs have shadow causing relationships as shown in Figure 14(a), then partition the junction into two as shown in Figure 14(b).

The condition of this rule is usually found when a body is supported by another at a point. Figure 14(c) gives an example scene to which the rule can be applied.

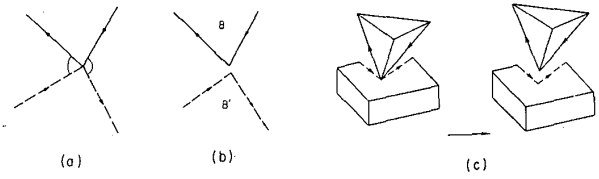


Figure 14: Partitioning Rule 1

The second partitioning rule is based on the particular types of junctions labeled S and D in the list of possible junctions of Figure 12. These junctions occur at the point where two objects contact each other. Hence, they can be used as cues for partitioning.

Partitioning Rule 2.

If a chain of concave lines connects two S and D type junctions as shown in Figure 15(a), or if a chain of concave lines connects an S or D type junction with a junction on an obscured line as shown in Figure 15(b), then partition the body into two by changing the chain of concave lines into a pair of contour lines as shown in Figure 15(c).

Figure 15(d) shows an example scene to which the rule is applicable. The scene is divided into two along the concave lines between junctions S and D.

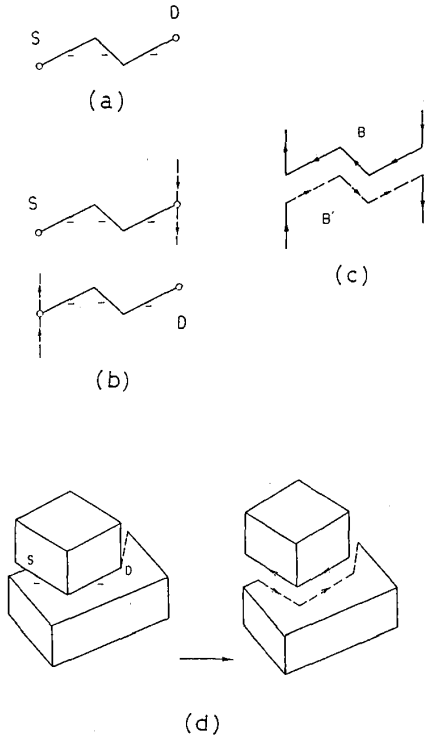


Figure 15: Partitioning Rule 2

4. Range Data Analysis Guided by a Junction Dictionary

In the preceding two sections we have developed two tools, the edge detecting operators and the junction dictionary. Here we combine them into a whole system which generates a surface patch description from a range picture and, at the same time, which partitions the scene into simple bodies. We first describe the outline of the system, and then present experimental results.

4.1. Experts in the System

The input to the system is a raw range picture of a real scene and the output is a collection of labeled line drawings of all the objects in the scene. The system is composed of the following experts: a line predictor, a prediction tester, a line tracker, a straight-line adjuster, a body partitioner, and a vertex position corrector.

When a junction is given, a *line predictor* consults the junction dictionary and predicts missing lines around the junction. New lines are predicted if and only if addition of the new lines results in possible junctions. If the present junction is an impossible one, the prediction is strong in the sense that at least one of the predicted lines must be found. If the present junction is a already a possible one, on the other hand, the prediction is weak in that the predicted lines may not exist.

A *prediction tester* judges whether the predicted lines really exist or not. The judgment is based upon two kinds of tests. One is to use the edge detecting operators to see if there are new lines to be tracked in the range picture, and the other is to examine old lines in the present line drawing if their properties meet the prediction. The latter search is necessary, because when a new line is found and is added to a junction, the end of the new line may be left unconnected though it should be joined to one of its neighboring junctions. When old lines whose properties meet the prediction are found, the prediction tester concludes that the prediction is true. Otherwise, it examines all the predictions in the range picture and concludes differently depending on the strength of the prediction. That is, if the present junction is impossible, the prediction tester returns the direction in which the edge detecting operator gives the maximum output value even if the value may not be very large. On the other hand, if the present junction is possible, the prediction tester behaves conservatively; it reports the existence of new lines only if the edge detecting operator gives very large absolute values.

When a new line is found, a *line tracker* tracks it as far as possible and

extracts the whole line. The tracking terminates when the line disappears or when the physical type of the line changes (such as from concave to convex), or when it comes near to another line that has already been found.

A *straight-line adjuster* finds a sequence of straight lines which fit to the strings of points extracted by the line tracker. When a string is given, it first finds points with maximal curvatures (the curvature at a point on a string is calculated in the same way as proposed by Shirai [17]). The adjuster next divides the string at those points into substrings and finally replaces each substring with a straight line segment. If it fails in finding a well-fitting line, the substring is left for later analysis. The joints between two substrings as well as the end point of the original string are added to the drawing as new junctions.

Whenever a new line is added to the drawing, a *body partitioner* checks the conditions for the partitioning rules and, if one of the conditions is satisfied, it decomposes the line drawing into two smaller ones.

A *vertex-position corrector* revises a position of a junction whenever a new line is connected to the junction. The position is estimated as the weighted mean of intersections of all pairs of the lines incident to the junction, where the weights are chosen according to the lengths of the lines.

4.2. System Behavior

The processing of the system is divided into two stages: the preprocessing and the main processing.

In the preprocessing stage, the system first finds contour lines (i.e., obscuring lines and obscured lines) by tracking points where ranges show discontinuity, divides them into straight line segments, and locates junctions on the contours. Next it finds shadow-causing relationships and classifies the contour lines into obscuring lines and obscured lines. At this stage, shadow-causing junctions are also identified. Each connected region bounded by obscuring and obscured lines is called a body.

In the main processing stage, the experts which have been described in Section 4.1, work together to analyze the range picture using the following seven steps.

Step 1. If the descriptions of all bodies have been constructed, the system terminates the processing. Otherwise, the system picks up a body whose description has not yet been completed.

Step 2. If all the junctions of the body have been examined, the processing goes to Step 7. Otherwise, the system picks up a junction to examine next.

Step 3. The line predictor compares the junction with the dictionary, and proposes all predictions of missing lines around the junction. The prediction tester examines them to see if any lines are really there. If no lines are found, it puts the mark "examined" to the junction and the processing goes to Step 2. If old lines whose properties meet one of the predictions are found, the processing goes to Step 5. If new lines are found, the processing goes to Step 4.

Step 4. The line tracker extracts the strings of points which form the new lines. The straight-line adjuster locates new junctions on the strings, and divides them into substrings, each of which is approximated by a straight line segment.

Step 5. The lines which are extracted are connected to the present junction. The vertex-position corrector corrects the position of the junction using all the incident lines.

Step 6. The body partitioner examines the improved part of the body. If one of the conditions of the partitioning rules is satisfied, it decomposes the body into two and the processing goes to Step 1. Otherwise, it goes to Step 3 (in this case the junction is the same but its type has been changed).

Step 7. The system sweeps the body region with the edge detecting operator. If it finds a new line, it locates all junctions on it and goes to Step 2. Otherwise, it puts the mark "completed" to the body and goes to Step 1.

4.3. System Performance

The range pictures used as inputs to test the system are all from real scenes consisting of white polyhedrons. They are obtained by a rangefinder developed at Electrotechnical Laboratory [14]. Each picture consists of 200 by 240 points and the value of range $D(i,j)$ has nine bits of information.

Figure 16(a) shows a light stripe image of a scene with two blocks, one supporting the other, and Figures (b) through (f) show how the description of the scene is created progressively. The system, in its preprocessing stage, extracts the contour, finds junctions on it, and adjusts line segments to the contour. Junctions $J1$ and $J2$ are recognized as shadow causing junctions.

In the main processing stage, the system first picks the junction $J1$ and finds that $J1$ is impossible. The line predictor suggests a concave line in the left side of $J1$ (where the word "left" is used with respect to the reader) and the prediction tester actually finds the concave line. The line tracker follows the new line until it terminates at $J3$. Next the straight-line adjuster works with this line. Junction $J4$ is found on the new line (Figure 16(b)), and the resultant two portions of the string are approximated by straight line segments $J1J4$ and $J4J3$. The system next picks up the junction $J2$. It is an impossible junction and a missing line is predicted, but the old line $J3J4$ can be used to resolve the problem by merging $J2$ and $J3$. Then the condition for the partitioning rule is satisfied; that is, $J1$ belongs to the D type and $J2$ belongs to the S type, and they are connected by the concave lines $J1J4$ and $J4J2$. The body partitioner partitions the picture into two bodies, *BODY*1 and *BODY*2, (Figure 16(c)). The system picks up *BODY*1. When junction $J5$ is examined, a new convex line is found. The new line terminates at $J6$ and junction $J7$ is also located on the new line (Figure 16(d)). Junction $J6$ is joined to the neighboring junction on the contour. Another new convex line $J4J8$ is found from $J4$ (Figure 16(e)). Two impossible junctions $J7$ and $J8$ are joined together. Now, since all the junctions in *BODY*1 are examined and no new lines can be found, the description of *BODY*1 is completed. The

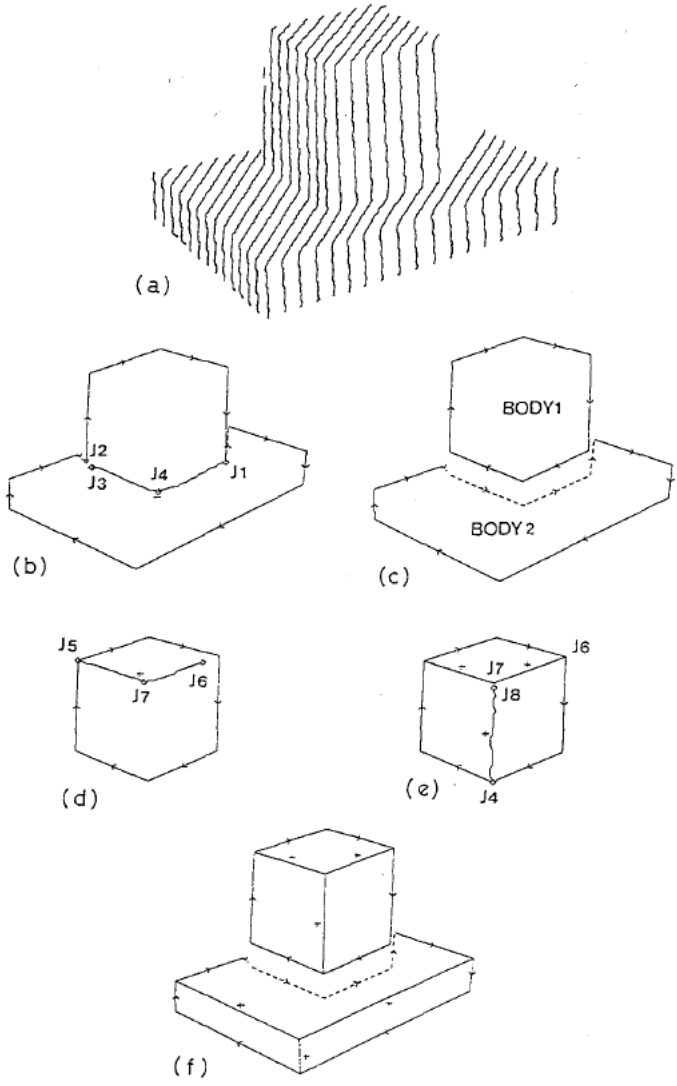


Figure 16: Example Scene 1

system picks up the other body *BODY2* and improves its description similarly. The final result is given in Figure 16(f).

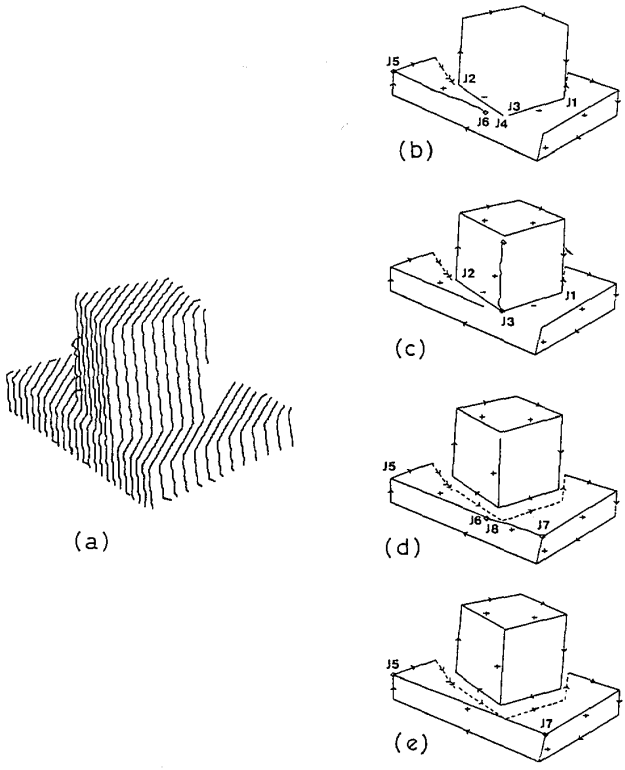


Figure 17: Example Scene 2

Figure 17(a) shows a light stripe image of another two-block scene. A vertex of one block lies accidentally on an edge of the other block, and a non-trihedral vertex occurs. The system conquers this difficulty as follows. A new concave line is found at $J1$ but it terminates at $J3$. Another concave line is found at $J2$ and it is traced to $J4$. The system finds a convex line at $J5$ and tracks it. However, the non-trihedral vertex obstructs the tracking of the new line at $J6$ (Figure 17(b)). When the system examines $J3$, $J4$ is merged to $J3$ and another convex line going upward is found at $J3$; $J3$ becomes a possible junction (Figure 17(c)). Now the partitioning rule is applied to the concave lines $J1J3$ and $J3J2$, and the scene is partitioned into

two bodies. When the junction $J7$ is examined, another part of the incomplete convex line is found; $J7J8$ in Figure 17(d). Junction $J8$ is joined to $J6$ and the resultant $J6$ is eventually deleted because the two lines $J5J6$ and $J6J7$ are collinear. In this way the complete line $J5J7$ is found as is shown in Figure 17(e).

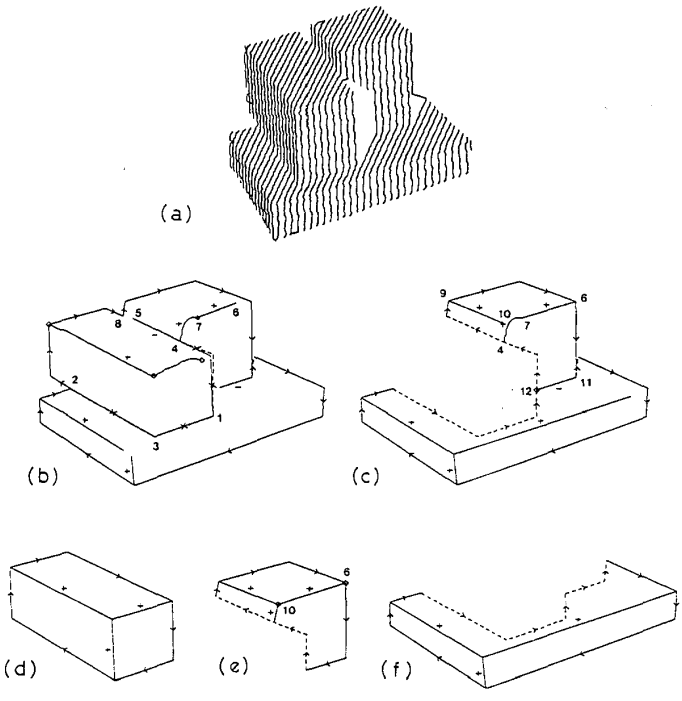


Figure 18: Example Scene 3

The third example is shown in Figure 18. The scene contains three blocks. The system finds two contours in this scene. One is a usual clockwise contour. The other is a counterclockwise contour surrounding a vacant hole found in the gap; that is, the contour $J1J4$ and $J4J1$ in Figure 18(b). When the system finds concave lines connecting $J1$ to $J2$, the body

partitioner cuts the body along the lines. However, the resulting body still remains connected. When the system examines $J4$, a new line $J4J5$ is found. Since $J4$ is still impossible, the system searches for another line and finds a convex line $J4J6$. On this line the system finds a new junction, $J7$ (though its position is not quite appropriate). Junction $J5$ is merged to $J8$ and the body is cut along the line $J4J5$ as shown in Figure 18(c) and (d). One of the resultant body is further partitioned along the line $J11J12$ into two portions shown in Figure 18(e) and (f). When the system finds a new convex line emanating from $J9$, the new line is followed to $J10$. The system next creates a new junction on the line $J4J7$, and the new junction is merged to $J10$. Then the straight-line adjuster replaces the two strings $J4J10$ and $J10J7$ with straight line segments, and since the two lines $J10J7$ and $J7J6$ are collinear, they are merged and the junction $J7$ is deleted as is shown in Figure 18(e).

5. Concluding Remarks

We have presented a range data analysis system using knowledge about vertex types. Since the junction dictionary suggests missing lines at each step of the analysis, the detection and organization of edges can be done efficiently and reliably. Moreover, the output is not merely a collection of lines but an organized structure of faces, edges, and vertices (that is, the surface patch description), which is useful for later processing such as object identification and manipulation.

The present approach can be extended in several directions. First, though we have restricted our objects to trihedral ones, our system can be applied to other objects with a slight modification. All we have to do is to replace the junction dictionary with the one containing all possible junctions in a new world. If the object world consists of a finite number of prototypes (which is often the case in industrial applications), the associated junction dictionary can be generated automatically. Further details may be found in

a paper by Sugihara [21].

Secondly, our system can also be applied to curved surface objects. The greatest difficulty in dealing with curved surface objects is that curved lines appear in line drawings and consequently the number of possible junction types becomes very large. However, we can circumvent the explosion of the dictionary by replacing curved lines emanating from a junction with the straight lines tangent to them. This convention is also practical in the sense that the system need not distinguish between straight and curved lines, which is a difficult task especially when the data contain noises. Thus, curved surface objects can be dealt with by a little larger dictionary. The reader is referred to papers by Sugihara [18], [20] for further details.

References

[1] Agin, G. J. and Binford, T. O.
Computer description of curved objects.
In *Proceedings of the Third International Joint Conference on Artificial Intelligence*, pages 629-640. August, 1973.
Also appeared in IEEE Transactions on Computers, 1976, Vol. C-25, pages 439-449.

[2] Clowes, M. B.
On seeing things.
Artificial Intelligence 2:79-116, 1971.

[3] Faugeras, O. D.
New steps toward a flexible 3-D vision system for robotics.
Preprints of the Second International Symposium of Robotics Research. Note: also see In Proc. 7th Int. Conf. Pattern Recogn., pages 796-805; Montreal, Canada, July :222-230, 1984.

[4] Guzman, A.
Computer recognition of three-dimensional objects in a visual scene.
Technical Report MAC-TR-59, Ph.D. thesis, Project MAC, MIT, Mass., 1968.

[5] Herman, M. and Kanade, T.
The 3-D MOSAIC scene understanding system: Incremental reconstruction of 3-D scenes from complex images.
Technical Report CMU-CS-84-102, Carnegie-Mellon University, Computer Science Dept., 1984.

[6] Huffman, D. A.
Impossible objects as nonsense sentences.
Machine Intelligence 6.
Edinburgh University Press, Edinburgh, 1971, pages 295-323.

[7] Ishii, M. and Nagata, T.
Feature extraction of three-dimensional objects and visual processing in a hand-eye system using laser tracker.
Pattern Recognition 8:229-237, Oct., 1976.

[8] Kanade, T.
A theory of Origami world.
Artificial Intelligence 13:279-311, 1980.

[9] Marr, D. and Nishihara, K.
Representation and recognition of the spatial organization of three-dimensional shapes.
Proceedings of the Royal Society of London B200:269-294, 1978.

[10] Mino, M., Kanade, T., and Sakai, T.
Parallel use of coded slit light for range finding (in Japanese).
In *21st National Convention of the Information Processing Society of Japan*, pages 875-876. 1980.

[11] Nevatia, R. and Binford, T. O.
Description and recognition of curved objects.
Artificial Intelligence 8:77-98, 1977.

[12] Oshima, M. and Shirai, Y.
A scene description method using three-dimensional information.
Pattern Recognition 11:9-17, 1979.

[13] Oshima, M. and Shirai, Y.
Object recognition using three-dimensional information.
IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-5(4):353-361, July, 1983.

[14] Oshima, M. and Takano, Y.
Special hardware for the recognition system of three-dimensional objects (in Japanese).
Bulletin of the Electrotechnical Laboratory 37:493-501, 1973.

[15] Sato, S. and Inokuchi, S.
Range imaging by Gray coded projection (in Japanese).
In *1984 National Convention of the Institute of Electronics and Communcationa Engineers of Japan, Part 6*, pages 283-284. 1984.

[16] Shirai, Y. and Suwa, M.
Recognition of polyhedrons with a range finder.
In *Proceedings of the Second International Joint Conference on Artificial Intelligence*, pages 80-87. 1971.

[17] Shirai, Y.
A context sensitive line finder for recognition of polyhedra.
Artificial Intelligence 4:95-119, 1973.

[18] Sugihara, K.
Dictionary-guided scene analysis based on depth information, Report on Pattern Information Processing System.
Technical Report 13, Electrotechnical Lab, 1977.

[19] Sugihara, K.
Picture language for skeletal polyhedra.
Computer Graphics and Image Processing 8:382-405, 1978.

[20] Sugihara, K.
Range-data analysis guided by a junction dictionary.
Artificial Intelligence 12:41-69, May, 1979.

[21] Sugihara, K.
Automatic construction of junction dictionaries and their exploitation for the analysis of range data.
In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 859-864. 1979.

[22] Waltz, D.
Understanding line drawings of scenes with shadows.
The Psychology of Computer Vision.
McGraw-Hill, New York, 1975, pages 19-91.

PART III: 3-D RECOGNITION ALGORITHMS