NASA-CR-191397

View Generated Database
Final Report
NAS7-1066

Submitted to:

Dr. Neville Marzwell, JPL Technical Monitor
M/S 198-330
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109

and

Ms. Veronica Stickley, NASA Contrating Officer
NASA Resident Office
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109

Dr. James G. Downward
Innovation Associates, Inc.
P.O. Box 1306
Ann Arbor, MI 48106
(313)-995-9338

Table of Contents

Report Overview

Because KMS no longer has on its staff anyone capable of writing a final report for VGD, in January 1992, Innovation Associates informed NASA/JPL that they would write the final VGD report for KMS at no cost to NASA or KMS.

This document represents that effort and is the final report for the VGD project, NAS7-1066. It documents the work done on the project up to the point at which all project work was teminated due to lack of project funds. Up until the point when contract funds were exhausted, work on the VGD project was very promising and proceeding well. However, due to circumstances beyond the control of the Technical Staff on the VGD project, unanticipated Overhead and G&A rate increases occured at KMS during both 1990 and 1991. These increases, resulted in a) inadequate contract funding being available to perform the original statement of work, b) unexpected loss of staff performing the work, and c) the abrupt layoff of remaining project staff when a DOE-mandated cost accounting changes in mid 1992 caused Overhead and G&A rates to again increase thus exhausting remaining contract funds without prior warning.

Starting in January 1991, KMS Industries made good faith efforts fulfill its commitments to its R&D customers. Specifically, it attempted to shield its R&D contracts from the affects of the loss of a $16 million DOE ICF contract by requesting that NASA and other government agencies novate KMS Fusion's R&D contracts to KMS Advanced Products, Inc. and by maintaining separate overhead and G&A rates for the two companies. The novation requests dragged on without resolution for six months. At the end of March 1991, KMS suspended work on the VGD project because if the novation request was not granted, all available funds on the contract were already exhausted.

Section A
Project Accomplishments by Quarter

Quarterly Report #1

A general design for the VGD system was provided to the technical contract monitor to clarify the work that is to be performed under the VGD contract.

Quarterly Report #2

Methods for generating a sinusoidal projection pattern required by the VGD SURPHACE Camera are described. Techniques are reported for eliminated determental effects due to higher order harmonic errors in producing the sinusoidal slides.

Quarterly Report #3

A design is provided for a new ranging device, i.e, the SUrface Reconstruction by PHAse-shifted CosinEs (SURPHACE) Camera, which uses phase-shifted structured lighting to triangulate on the points of the surface of an object. The camera works with a simple light projector and a CCD camera and records registered range and intensity data.

Quarterly Report #4

Methods for generating a sinusoidal projection pattern required by the VGD SURPHACE Camera are described. Techniques are reported for eliminated determental effects due to higher order harmonic errors in producing the sinusoidal slides.

Quarterly Report #5

A more complete design of the VGD system is presented, together with a description of the calibration routines necessary to calibrate the VGD SURPHACE Camera and routines to manipulate surface data capture by the SURPHACE Camera. Two representations are presented for storing surface range/texture data, i.e., one in which surface data is stored as a set of pixel meshes, and the second in which data is stored as a polygon mesh.

1991 Unreported Work

The work on calibrating the camera was successfully completed and work was started in developing routines to patch together different views to obtain a complete surface representation. Because the staff working on these tasks quit with minimal notice, this has not been previously documented.

Section B
Project Task Status Review
(15-May-91)

Task I.1: Implement Stereo Reconstruction using Structured Lighting.

Proposal Objective:	To develop a simple approach for obtaining correlated range and texture data.
Work Accomplished:	KMS developed a phase-shift structured lighting system that uses a special projector with two stereo cameras to input texture and range data.
Significant Accomplishments:	KMS has developed good camera calibration routines and demonstrated feasibility of approach.
Technical Problems:	KMS had to move from a hand-held light-weight system to a tripod-held system because the combined weight of the cameras, the camera mounts, and the phase shift structured lighting assembly is too large for a the unit to be steadily held in the hand. When the project was terminated due to lack of funds, the system still did not provide the accuracy which was originally envisioned and range images were still noisy.

Task I.2: Investigate Stereo using Existing Lighting

Proposal Objective:	To develop a simple approach for obtaining correlated range and texture data without using projected lighting.
Work Accomplished:	KMS reviewed existing techniques to see where new approaches might be possible.
Technical Problems:	After reviewing existing techniques, KMS determined that the technical issues involved were too complicated for us to adequately address this problem and that there would be inadequate hours available on the contract for us to devote further effort to seeking a simple approach for correlating range and texture data without using projected lighting.
Effects on Project:	Little. This part of the task was a best effort.

Task II: Implement Software for Viewpoint Determination

Proposal Objective:	To develop a approach for determining the transform between different views.
Work Accomplished:	This work falls out from the camera calibration code developed for surface reconstruction. We still have to implement a small amount of code to complete the task.
Significant Accomplishments:	The KMS approach allows the camera viewpoints to be correlated (matched to each other) using existing surface features.
Technical Problems:	None known.

Task III.1: Implement Software for Manipulating VGD Database

Proposal Objective:	To develop an efficiently accessed (i.e., surface quadtree) representation for surface data.
Work Accomplished:	At the time contract work halted due to lack of funds, the database had not been implemented. KMS intended to provide a database that was simpler to implement than the surface quadtree databaase and which met the basic technical requirements.
Technical Problems:	KMS determined that there was inadequate labor time to implement the database as originally designed. Morover, the original database design did not necessarily provide a good representation for both surface and texture data.
Effects on Project:	Little effect, since the database that is to be used will play a similar role.

Task III.2: Inputting Raw Surface Data Into the VGD Database.

Proposal Objective:	To develop techniques for loading data into the VGD database.
Work Accomplished:	KMS was able to load both texture and range data.
Technical Problems:	None.
Significant Accomplishments:	The KMS approach allows texture/range data to be loaded.

Task III.3: Implement Software for Merging Subsurfaces.

Proposal Objective:	To develop code to merge surface data into one coherent surface.
Work Accomplished:	Designed, but not implemented.
Technical Problems:	There may be too little time left on the project to implement the needed code.
Effect on Project:	This Task must be finished to provide a working system.

Task III.4: Implement for Converting to IGES

Original Objective:

To develop code to convert database data to IGES format.

Work Accomplished:

IGES specifications were obtained. After reviewing the specifications, KMS determined that it was not technically feasable to represent VGD surface data in IGES format. While we could create a surface space frame and from there create line segments representing surface patches, the result would be an accurate representation of the data acquired by VGD.

The IGES standard was set up to provide an interchange format for line segments contained within 2-D CAD drawings. As such, standard IGES lacks primiteves for describing the 3-D surface features and information that VGD acquires, i.e. texture, brightness, 3-D coordinates of a surface area patch. JPL has created extensions to IGES to address some of the representation restrictions. However, staff determined that they did not have adequate time to pursue this approach.

KMS experienced a similar problem in trying to use IGES as an interchange media for transfering data from a 3-D solids modeling package on a Applicon CAD system to another system. Although the individual line segments of space frame drawings could be transferred, there was no way to transfer the information which tied the drawings into a 3-D shaded model of the object being designed.

Technical Problems:

The initial concept of converting the data into IGES was flawed by not understanding the severe limitations of the standard IGES format. Moreover, inadequate hours existed to try and address these limitations by investigating IGES extensions because of unanticipated 1990 overhead and G&A rate increases.

Effect on Project:

Little effect.

Task IV: Design and Implement Operator Interface

Proposal Objective:	To design an easy-to-use operator interface for manipulating surface data.
Work Accomplished:	Design complete.
Technical Problems:	Because of unanticipated 1990 overhead and G&A rate increases at KMS, staff determined that there would not be adequate time for implementing the user interface which was designed. As a workaround, the user will interact with the system using a command line interface.
Effect on Project:	While a reduction in operator ease-of-use is not desirable, the VGD system should be able to meet its technical objectives.

Task V: Design and Build Stereo-Camera System

Proposal Objective:	To design a hand-held, easy-to-use stereo-camera system.
Work Accomplished:	The camera and structured lighting system was built and is functioning. Data is being acquired with it.
Technical Problems:	The unit tends to run hot because the structured lighting system uses a projection lamp. This problem might be overcome by replacing the projection lamp with a laser diode. The camera itself is heavy and somewhat unwieldy. Although ultra-light weight cameras were obtained, the weight is greater than anticipated because of the weight of the camera mounting rail and the weight of the structured lighting assembly. The weight of the structured lighting assembly could also be significantly reduced using a laser diode as the illumination source.
Significant Accomplishments:	KMS developed a working structured lighting system.

Task VI: Procure Hardware

Proposal Objective:	Obtain hardware for efficient data capture and manipulation.
Work Accomplished:	All hardware obtained.
Technical Problems:	The CDA array processor still can not be used when the VAXstation 3520 is configured to use both CPUs. If the workstation is configured to use both CPUs, attempting to access the CDA array processor will crash the workstation. The problem has been traced to the driver for the CDA array processor. It does not support symetric multi-processing correctly. Repeated attempts were made to get CDA/Analogic to address this problem, but to date, CDA has not provided KMS with a driver which correctly supports the both processors in the VAXstation 3520.
Effect on Project:	Our inability to use the CDA array processor with the workstation as intended has hindered our ability to perform this contract. The speed at which we can manipulate data and test concepts just using the VAXstation is too slow. Because we have been unable to get CDA to resolve the problem, the time we spent tracking down the problem and trying to resolve it was wasted.

Section C
Quarterly Report #1: December 1989

C-1: Purpose and Scope of This Report

In conversations with NASA on December 4, 1988, we learned of possible applications of great interest to NASA to which VGD's technology might be applied. In particular, the three applications of greatest interest were: (1) autonomous docking between an OMV and a satellite, (2) ORU replacement by a telerobotic servicer, and (3) assembly and disassembly. These applications have the common thread that they all will necessarily employ model-based object recognition and tracking. The team at KMS that is working on this contract has much experience in this area; two of us have written our Ph.D. dissertations in these areas, and the rest of the team has done extensive research in these areas, as well as other areas of computer vision. In addition, we have won previous contracts to perform research in recognition and tracking. Therefore, we are very qualified to develop such a system. However, as originally proposed, the goal of the Phase II View Generated Database effort, is to develop a **model acquisition system.** This system would allow accurate 3-d models of 3-d scenes, or of individual 3-d objects to be constructed which is not as readily applicable to NASA's immediate needs as it might be.

However, in the event that NASA would consider a modification in the VGD statement of work which would slightly reorient the project's goals, KMS believes that it would be possible to redefine the VGD project to make it more readily applicable to NASA's mission. Consequently, part of the purpose of this report is to highlight what changes would be necessary to accomplish such a reorientation, if NASA determined such a change in project objectives was in the best interest of the government.

In order to clarify the changes required for reorientating VGD, we are providing an overview of KMS's approach to model-based object recognition and tracking followed by a synopsis of VGD as it was originally proposed in Section C-3. Section C-3 will also discuss how KMS could address NASA's applications in object recognition and tracking by slightly modifying the statement of work while maintaining the existing Phase II budget. Section C-4 provides a detailed description of KMS's tracking algorithm.

In addition to the primary purpose of this document, there are two secondary purposes. The first is to apprise you of KMS's software development process. The second is to provide a critical review of existing work in the areas related to VGD which is a task in the statement of work.

The heart of KMS's software development process is our software quality assurance guideline, which is outlined in Section C-5. This guideline insures that the delivered software system is well documented, easily maintainable, and contains the desired functionality.

Finally, in Section C-6 is a review of critical review of model acquisition systems. However, since we realize that NASA may have some interest in utilizing VGD's technology in object

recognition and tracking applications, we have included a critical review of object recognition methods as well in Section N.

C-2: Overview of KMS's Approach to Model-based Object Recognition and Tracking

KMS has considerable experience in building systems for model-based object recognition, transformation determination, and tracking. This section provides an overview of our approach to these problems.

KMS's approach to model based object recognition consists of three major phases:

(1) **Modelling phase:** model acquisition and representation,

(2) **Lock-in phase:** initial model or model sub-part identification and coarse viewing transformation estimation, and

(3) **Tracking phase:** fine viewing transformation estimation and hypothesis validity checking.

We have referred to phases phases (2) and (3) in "tracking" terminology as this is more suggestive of the functions they perform. However, each of these phases is largely isomorphic to a corresponding phase of the object recognition process. In a model-based tracking scenario, the lock-in phase refers to the problem of initial identification and view transformation estimation. Thus, in such a scenario, this phase must "lock-in" on which object (or objects) are present, and roughly estimate the viewing transformations so that the tracking phase can take over to accurately determine the viewing transformation, and update, i.e., track it, from frame to frame. In an object recognition and location scenario, the goal of the lock-in phase, as in the tracking scenario, is to provide initial hypotheses about the identity, position and orientation of the object that appear in an image of a scene. Also similarly to the tracking scenario, the position and orientation of each model hypothesis needs to be determined only roughly, because the function of the tracking phase is to refine the estimates of the viewing transform. In an object recognition scenario, the tracking phase would be augmented to perform some additional validity checking against the image to screen out possible false alarms.

The remainder of this section describes KMS's approach to these problems.

While phases (2) and (3) differ in some minor respects between a tracking scenario and an object recognition scenario, phase (1), the modelling phase, are identical in both scenarios. In either scenario, the modelling phase will be performed off-line, i.e., prior to the recognition phase or the tracking phase. If it were to be performed on-line, then VGD as it was originally proposed would be more appropriate. However, in the context of model-based object recognition and tracking, the modelling phase consists primarily of creating representations of 3-d objects that

expedite recognition and tracking. Conventional representations used by geometric modelers in CAD and CAM applications are unsuitable for vision applications. Thus, a critical part of the design of VGD is to represent the 3-d objects in a manner that expedites the tasks of recognition and tracking.

VGD will continue to concentrate on the modelling phase of the overall problem of object recognition, transform determination, and tracking. However, KMS will add a task to the existing task statement in our original proposal wherein we will investigate using these models in an existing object tracking and transform determination system. The remaining parts of this section are devoted to describing the modifications to each task of the original proposal, as well as describing the new task.

C-3: Reorienting VGD for Object Recognition and Tracking Applications

Based on conversations with NASA, we believe that model based object recognition, viewing transformation determination, and object tracking are of considerable interest to NASA. Therefore, we are offering to redefine VGD in a way that might be more responsive to NASA's needs. In particular, we could redirected the model aquisition effort away from its original emphasis on 3-d surface acquisition, detailed surface photometry measurement, world model maintenance and toward model aquisition and representation for object recognition, transformation determination, and tracking.

Therefore, Section C-3.2 provides a brief description of our approach to model acquisition and representation. This representation is much better suited to model-based 3-d object recognition and tracking than conventional CAD/CAM representations. Finally, Section C-3.3 provides a task-by-task description of the redirection that we could make (if directed by the contracting office) in the original six tasks. To further address NASA's needs, we could also add a task to incorporate the models created by VGD into existing object tracking software developed at KMS.

C-3.1: Synopsis of VGD as Originally Proposed

VGD provides the capability to accurately represent any real-world object or scene as a computer model. Such models include both an accurate spatial/geometric representation of surfaces of the object or scene, as well as any surface detail present on the object. Applications of such models are numerous, including acquisition and maintenance of world models for tele-autonomous systems, generation of accurate 3-d geometric/photometric models for various 3-d vision systems, and graphical models for realistic rendering of 3-d scenes via computer graphics.

The features and the benefits of VGD, as originally proposed, are listed in Table C-1.

Features of Proposed System	Benefits of VGD to NASA
Uses a hand-held stereo pair of solid state cameras.	Allows objects of arbitrary size, and location to be input into a computer model.
Uses a hand-held shuttered projection system that allows structured lighting to be projected onto a surface.	Allows accurate determination of the surface of an object even when the surface has little surface detail.
Uses structured lighting but can be adapted to using existing lighting.	Does not require a special environment in imaging the object.
Provides an innovative stereo matching algorithm.	Allows integration of a number of visual depth cues.
Registers different views of an object using natural and artificially placed surface markings.	Does not require that the object to be placed into a special jig or oriented by a sophisticated positioning device.
Provides a heirarchically organized database for combining subsurfaces of an object and for building a world model.	Allows rapid access to both the geometric and surface detail description of the object model.

Table C-1: KMS's View-Generated Database will provide a flexible tool for creating computer models of existing objects.

C-3.1.1: Perspective on VGD as Originally Proposed

VGD has many applications. A particularly important one is the capability to provide vision systems, tele-autonomous systems, with accurate geometric and photometric models of everyday objects -- objects that were not originally designed with CAD tools. State-state-of-the-art object recognition and tracking techniques require such models in order to recognize and localize objects. VGD allows such systems to operate on scenes containing any objects, not just the ones that have been created using a CAD system, thus circumventing the need to laboriously create such models. Another important application is the creation and maintenance of world models for tele-autonomous systems. VGD has the capability to create a 3-d model of any scene using its visual sensor apparatus. Known objects can be identified and their locations determined. Further, the 3-d position and structure of unknown surfaces can be determined and stored in the world model for use in obstacle avoidance and path planning.

C-3.1.2: Functionality of VGD as Originally Proposed

VGD's design provides

- a mobile stereo camera/structured light apparatus from which registered stereo and structured light images can be acquired;

- the capability to take multiple snapshots of a scene or object using the camera apparatus, and use them to produce a 3-d description of the scene or object. In particular,
  - calibration of the stereo cameras and the structured light system,
  - accurate determination of interframe transformations, first with manually placed features and later with existing features,
  - to determine coarse surface geometry using images of projected structured light that is registered with standard intensity stereo images,
  - recover detailed surface geometry and surface albedo of portions of the object corresponding to each view,
  - fuse the surfaces into a single world model or object model;

- an easy interface to existing modeling systems since
  - the 3-d model is represented both in planar patch format and surface spline format,
  - models may be stored in files adhering to standard 3-d modelling formats;

- a world-modeling database permitting
  - easy and efficient creation and maintenance of a world model,
  - searching based on geometric attributes such as surface structure and position,
  - searching based on user definable attributes;

- an easy to use, Macintosh-like operator interface that allows all facets of the system's operation to be controlled and monitored in a simple, intuitive manner;

C-3.2: Modelling 3-d Objects for Recognition and Tracking

The proper representation of objects is critical to VGD's success in recognition, location, and tracking. By using models, VGD can compute the edges, and other features, that should be visible from a particular viewpoint. The key to achieving ultimate real-time object recognition and tracking performance is in accomplishing this task in the most efficient manner possible. KMS has explored a particularly promising avenue for acquiring and representing models of objects. This object representation, called the Basri-Uhllman representation, or BU representation, allows edges and their positions and orientations, to be computed extremely quickly. Ultimately, the BU representation should allow implementation of real-time object recognition, transformation determination, and tracking.

Efficiently predicting the visibility, position, and orientation (i.e., the appearance of the edges) of edges on a smooth object is very difficult. In contrast, efficiently predicting the appearance of edges on non-smooth objects, i.e., objects possessing many creases (tangent discontinuities), is far simpler to accomplish. The reason for this is that rotation of an object with a crease will produce edges that are simply the projection of the new 3-d location of the crease. On the other hand, given two views of a smooth object, there will be two distinct sets of points on the object's surface, one corresponding to each view, that project to edges in each view of the object. Thus, predicting the appearance of edges in a view of a smooth object is a two step process. First, we must compute the set of points on the surface of the object that project to edges. This set of 3-d points on the surface of the object is the called the contour generator. After computing the contour generator, it must be projected to obtain the actual edge contours. In contrast, predicting the appearance of edges resulting from creases on a non-smooth object is much simpler. Since the 3-d positions of the points on the creases are known a priori, all that is necessary is projecting these crease points to predict the appearance of the edges.

The BU representation allows VGD to quickly predict the appearance of the edges visible from any viewpoint, even when the object is smooth. The following paragraphs provide an abridged description of the BU representation, and discuss its merits as an representation for model-base object recognition, transformation determination, and tracking.

The Basri-Uhllman (BU) Representation of 3-d Objects

The BU representation consists of several submodels, each of which allows accurate prediction of the appearance of edges over a portion of the viewing sphere. Each submodel, in turn, consists of a set of 3-d space curves that have the magnitude of the curvature vector associated with each 3-d point that comprises them. The curvature vector, r, is defined as $r = (r_x, r_y)$, where r_x is the radius of curvature of the 3-d surface of the object around the x axis, and, similarly, r_y is radius of curvature around the y axis. It can be shown that the direction of the curvature vector is perpendicular to the direction of the tangent to the contour [Basri Uhllman 1988]. Thus, r can be computed from its magnitude, which is stored in the representation, and the tangent direction, which can be computed from the projection of the 3-d space curves that are also part of the representation. This allows the surface of the object near the contours to be well approximated by

the circle of curvature about the axis of rotation. This approximation holds well over a large region of the viewing sphere. This approximation allows the contour generator to be efficiently computed using the approximation to the surface [Basri Uhllman 1988]. After computing the contour generator, it is projected to predict the appearance of the edges from the new viewpoint.

In order to build an model of an object using the BU representation, an edge image taken from a central view and four edge images taken from auxiliary views are required. The auxiliary views are obtained by positive and negative rotations about the x and y axes from the central view. The resulting changes in the positions of the points on corresponding edge contours between the central view and the auxiliary views allow the 3-d position of the contour generator in the central view, and the curvature vector r to be calculated at each point of the contour generator. As mentioned previously, information contained in the magnitude of r plus information contained in the contour generator contains sufficient to information to reconstruct r, and therefore to construct the approximation to the surface. Taken together, all of the contour generators in the central view comprise one submodel in the BU representation. This procedure must be repeated for each region of the viewing sphere that requires its own submodel.

Figure C-1 shows an example of one submodel of the BU representation applied to a Volkswagon (VW). The top VW view is the edges that are predicted by the submodel from the submodel's central view. The middle two VW's are the edges detected in images of real VW's that have been rotated 15 degrees in either direction from the central view of the submodel. The lower two VW's are the result of superposing the edge images in the middle row with the edges predicted by the submodel from the new viewpoints. As is evident, the submodel does a good job of predicting the appearance of the edges.

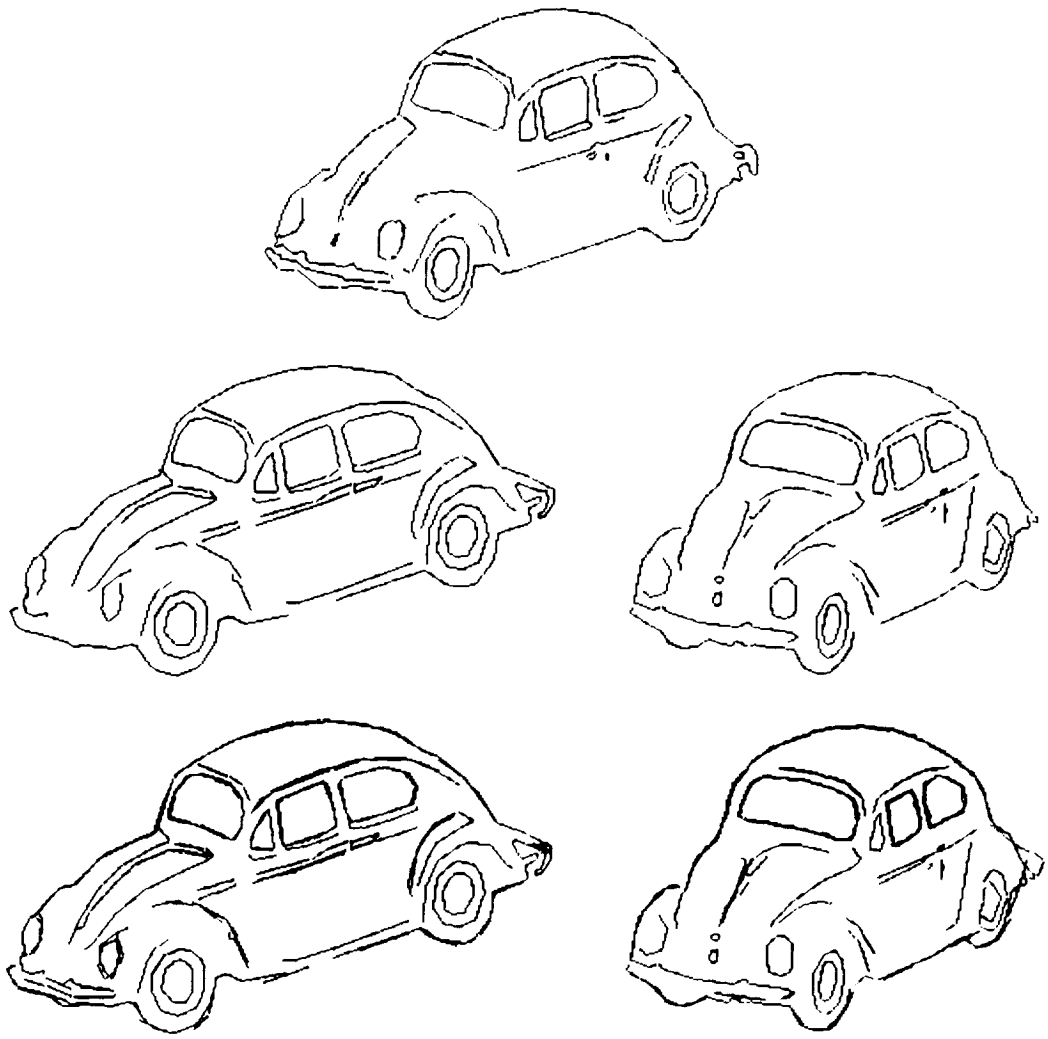


Figure C-1: Shown is an example of one submodel of the BU representation applied to a Volkswagon (VW). The top VW view is the edges that are predicted by the submodel from the submodel's central view. The middle two VW's are the edges detected in images of real VW's that have been rotated 15 degrees in either direction from the central view of the submodel. The lower two VW's are the result of superposing the edge images in the middle row with the edges predicted by the submodel from the new viewpoints. As is evident, the submodel does a good job of predicting the appearance of the edges.

The BU representation has several advantages. First, it allows internal edges to be modeled easily. Second, the method allows models of real objects to be easily built, as well as allowing easy conversion of objects that have been modeled on a conventional CAD system. Third, if the edge images used to create the model are taken under realistic lighting conditions, then the edges will be predicted by the model in a realistic manner. Finally, since the computations involved are so simple, predicting edges any viewpoint is extremely fast, easily accomplished in real time with readily available hardware..

C-3.3 Task by Task Description of Modifications to VGD

The changes that we propose to make in the existing tasks of VGD are concentrated in tasks I and III. The other tasks remain essentially unchanged. In addition, we are adding task VII. Table 2 summarizes the content of the changes. Sections C-3.3.1 - C-3.3.7 are devoted to describing the proposed changes in detail.

C-3.3.1 Task I: Implement Software to Construct Basri-Uhllman Submodels

The original thrust of this task was to take stereo images of a scene and use them to determine the 3-d structure of the scene, as well as the surface detail, such as markings and texture, present on the surfaces. Such surface models certainly contain enough information to be used by vision systems for object recognition and tracking, but the representation we originally proposed for VGD, i.e., a 3-d surface patch representation, is not well suited for these endeavors. Thus we will change this task to the aquisition and construction of Basri-Uhllman submodels. Section C-3.2 described how the BU representation is ideally suited to 3-d object recognition and tracking algorithms that use edges as features. Thus, we expect that the capability to acquire and create BU models will be very useful to NASA.

Section C-3.2.1 described the steps in creating a BU model. For each BU submodel, a total of at least five views are required. The set must be decomposable into sets of three views possessing viewing axes that are both coplanar and coincident at a single point. For this reason, we propose to modify the stereo/structured lighting sensing apparatus that was originally proposed by removing the structured lighting apparatus and adding a third camera (see Section C-3.3.5 "Design and Build Tricamera Sensor Apparatus). This will allow sets of three images satisfying the requirements stated above to be acquired simultaneously. The capability to acquire the three images simultaneously obviates the need to place the object in a jig or holder, making the system more versatile.

Task	Existing Task Description	Modified Task Description
1	Use stereo cameras and structured light apparatus to recontruct the 3-d structure and the surface detail of the evironment.	The BU representation is comprised of contours rather than surfaces. Thus, in this task we will build a single region representation of the BU representation from data acquired from the sensor apparatus (see changes to Task V).
2	Use features of the scene to register the views taken with the mobile sensor apparatus.	No changes to this task.
3	Possibly aided by the user, fuse the subsurfaces resulting from Task I into single surfaces using the view registration information from task II.	Combine several single view BU representations, generated in Task I, into a single multiview representation covering all possible viewpoints.
4	Implement an intuitive, easy to use, Macintosh-style operator interface.	No changes to this task.
5	Design and Build the hand-held stereo camera and structured lighting apparatus.	Since acquisition of models in the BU representation requires three images in a line, design and build a sensor apparatus consisting of three cameras, one at each end and one in the middle. No structured light will be used.
6	Support, maintain, and document VGD	No changes to this task.
7	Did not exist in the original proposal	Integrate objects in the BU representation into KMS's existing model-base, 3-d object tracking system.

Table C-2: A summary of the proposed changes to the tasks in VGD.

While the sensor apparatus permits the capturing of three registered images, at least five are needed to create one BU submodel. To accomplish this, the user will simply rotate the sensor apparatus and capture three more images for a total of six. The software implemented in Task II will allow the two trios of images to be registered, i.e., the transformations between the two views to be determined. Using the viewpoint registration information, the curvature information contained in each trio of images can then be integrated into the standard form of the BU submodel. In essence, recalling the terminology of Section C-3.2.1, instead of the ideal situation described there where r_x and r_y were determined directly due to the special configuration of the central and auxiliary views, we will obtain r_a and r_b, where a and b are arbitrary axes in the image plane. Given knowledge of the precise transformation between the first trio of images and the second, available from the software developed in Task II, it is a simple matter to derive r_x and r_y, and from this the magnitude of the curvature vector.

C-3.3.2 Task II: Implement Viewpoint Determination Software

This task remains the same as the original task.

C-3.3.3 Task III: Implement Software to Fuse Multiple Basri-Uhlman Submodels into a Single, Complete Model.

The viewpoint registration software developed in Task II allows several BU submodels to be combined into a single model. If enough correctly positioned submodels are used, then the appearance of edges from any viewpoint can be predicted.

It is relatively simple to fuse the submodels into a single overall model, given the availability of viewpoint registration data. The most difficult part of this task will be to determine where the submodels begin to erroneously predict the appearance of edges so that additional submodels may be added. Further, since we desire the minimal number of submodels providing a specified fidelity of prediction, we desire to add submodels to the optimal viewpoints in order to avoid redundancy. These measures will insure that the model provides accurate predictions over the entire sphere of views, while remaining efficient with storage.

C-3.3.4 Task IV: Implement Operator Interface

This task remains the same as the original task.

C-3.3.5 Task V: Design and Build Tri-camera Sensor Apparatus

As mentioned in Section C-3.3.1, the original stereo camera, structured lighting sensor apparatus will be changed to have three cameras. Referring to Figure C-2, two of the cameras will be located as originally proposed. The third will be located in the center of the apparatus, in the location of the sturctured light projector. This design will allow models of objects to be acquired

without the need to put them in a special jig or holder. This feature makes VGD much more versatile.

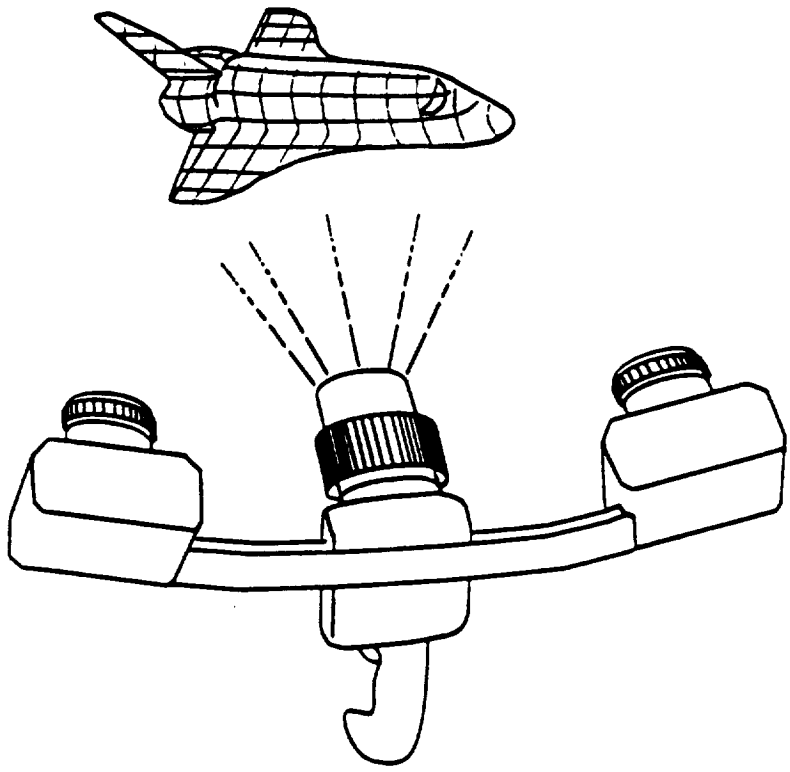


Figure C-2: Shown is a drawing of the original design of the sensor apparatus. There are cameras at either end, and a structured light projector at the midpoint. The revised design would be the same except that the structured light projector will be replaced by a third camera.

C-3.3.6 Task VI: Support, Maintain, and Document VGD

This task remains the same as the original task.

C-3.3.7 Task VII: Integrate BU Models into Existing Model-Based 3-d Object Tracking Software.

KMS has developed a powerful and fast model-based 3-d object tracking system. In this task, KMS will demonstrate the effectiveness of the BU representation by using such models to track real objects using our existing tracking software. Further, if NASA can provide us with real objects (or plastic models of these objects) we will create BU models of these objects, and then demonstrate tracking of these objects using our tracking software. We expect that the results of these experiments will show that real-time tracking of 3-d objects is well within the reach of readily available hardware using BU models coupled with KMS's tracking system.

We have provided a short description of the operation of KMS's tracking system in Section C-4.

C-4: Tracking and Precision Viewing Transform Measurement

During the first video frames of operation the tracking system will locate and lock in on the object or objects to be tracked. After an object is located, it is tracked, and continues to be tracked, until the object is lost from the field of view. At the start of each video frame, the tracking algorithm starts with the current estimate of the object's view parameters, available from the previous frame.

Using this estimate of the view parameters, the tracking algorithm adjusts the model to minimize differences between selected features of the object boundary and corresponding features of the model silhouette to obtain an optimum estimate of the object's view parameters.

Figures C-3 - C-7 display the step-by-step procedure used by the tracking algorithm to determine the view parameters of an object that is being tracked during a single frame of data.

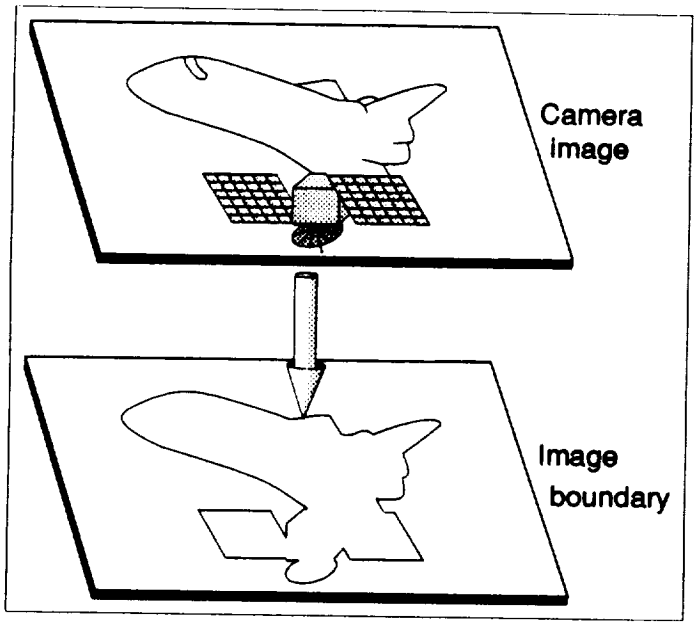


Figure C-3: The tracking algorithm consists of four steps. In the first step, after a video frame has been acquired, boundaries of the objects in the frame are located using an edge detection algorithm mapped onto a high-performance array processor.

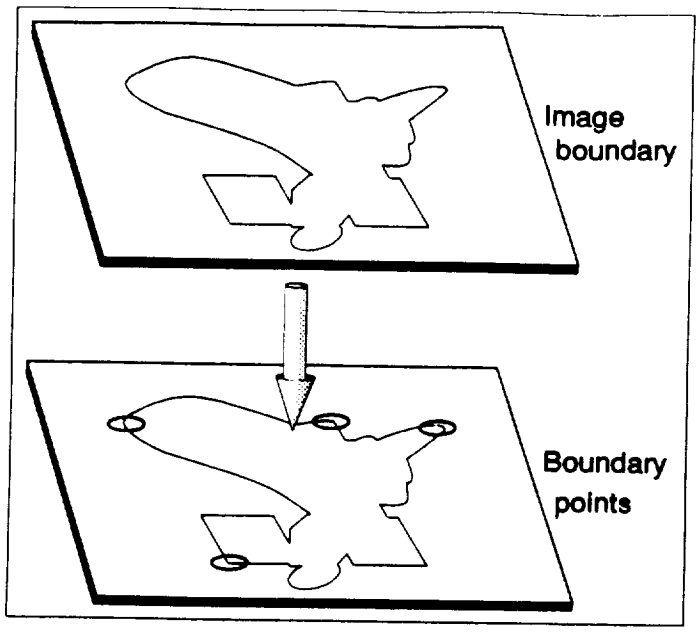


Figure C-4: In the second step, features are extracted from the edge boundaries. Features consist of pairs of points on the boundary at special locations, such as points of inflection or of high curvature, which are well localized and quickly found. Each feature possesses attributes that encode the shape, size, and location of the local support of the feature. The image features are stored in a special database, designed for fast retrieval.

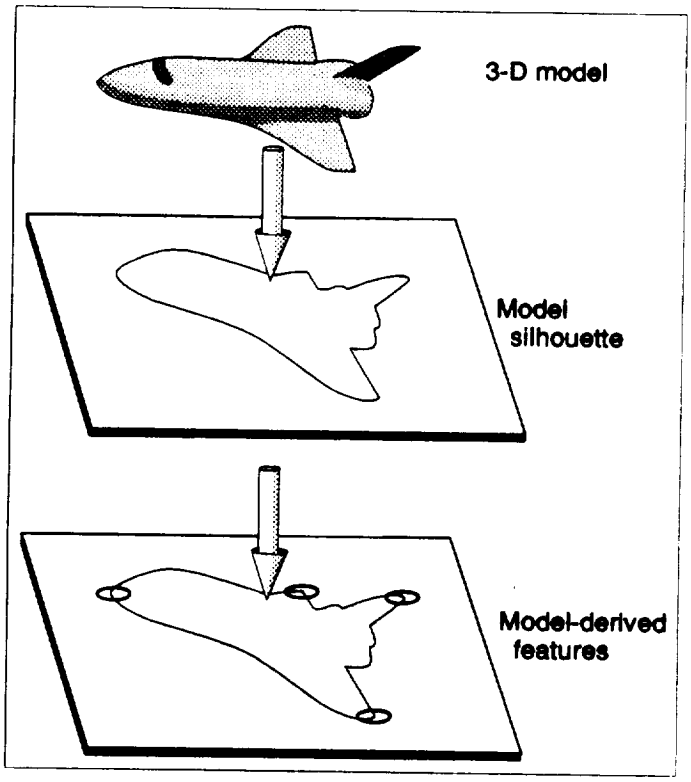


Figure C-5: In the third step, a silhouette is extracted from the model, where the model is scaled, positioned and oriented based on the current estimate of the object's view parameters. Features are again extracted.

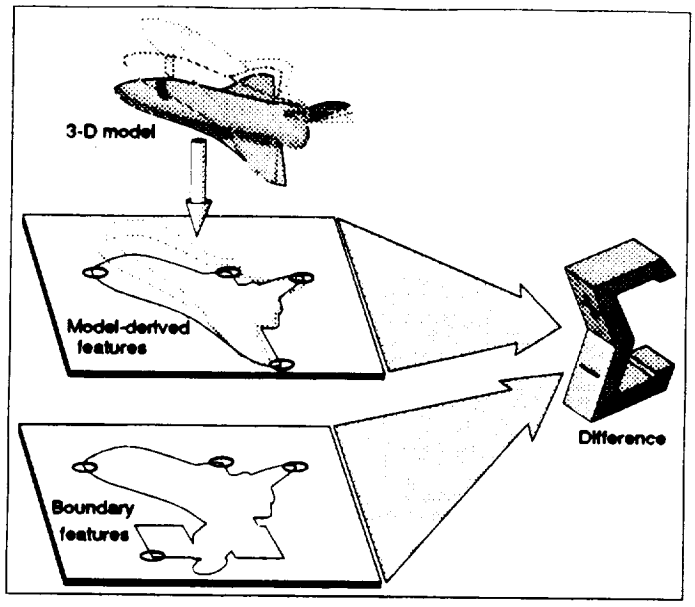


Figure C-6: In the fourth step, the differences between the features of the model silhouette and those of the object boundaries are computed. Steps three and four are repeated as the view parameters of the model are iteratively adjusted to minimize the differences between the features of the model silhouette and of the object boundaries.

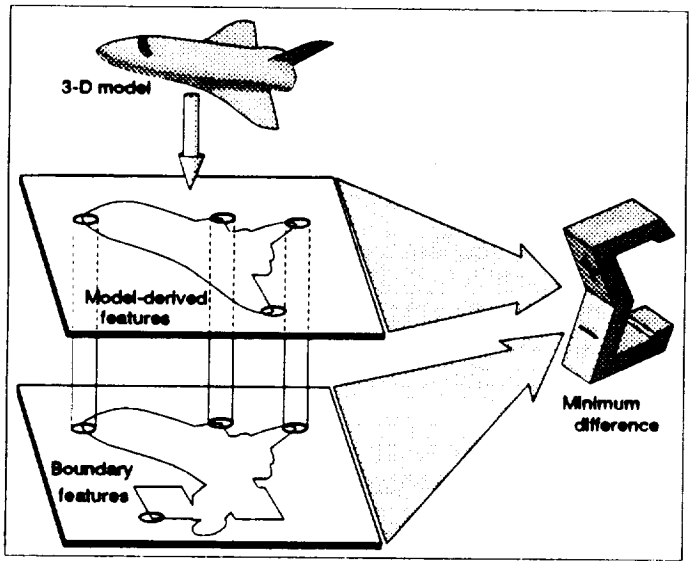


Figure C-7: When the differences are minimized, the view parameters of the model provide an optimal estimate of the object's view parameters. This estimate can be used to directly calculate the 3-space position and orientation of the object.

Figures C-8 and C-9 demonstrate actual results of the algorithm. In Figures 8a-d the tracking algorithm is lock in on the view parameters of the Space Shuttle starting from a initial, poor estimate of the view par The blue contour shown in 8a corresponds to the model silhouette generated from the initial estimate. Th contour (surrounding the Shuttle) corresponds to the boundary of the Shuttle. In Figures C-8(b-d) the al quickly readjusts the view parameters of the model until the model silhouette corresponds to the object boun

In Figures C-9(a-d) the tracking algorithm is used to lock in on the view parameters of a partially occlude Shuttle. Here, due to harsh lighting conditions, the nose cone of the Space Shuttle is missing. The blue cor 9b corresponds to the model silhouette generated from an initial, poor estimate of the view parameters. Th contour again corresponds to the boundary of the object. In Figures C-9(c-d) the tracking algorithm converges to the correct view parameters.

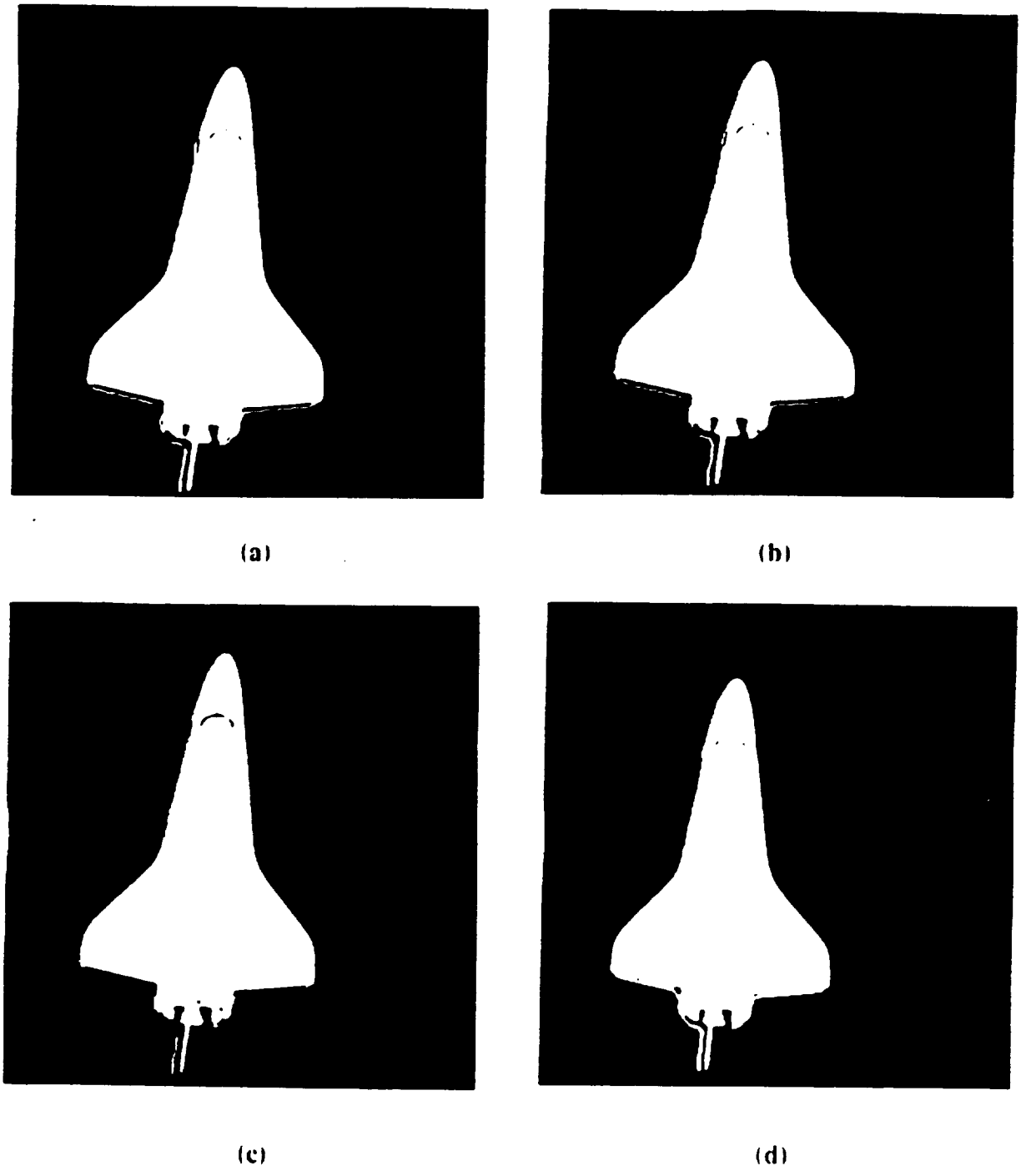


Figure C-8: **(a) Given a rough initial estimate of the view parameters of an object, the tracking algorithm determines a starting silhouette (blue contour) to be matched to the object boundary (yellow contour). (b)-(d) The algorithm then quickly locks in on the true view parameters of the object. Note that in (d), the tracking algorithm**

locates the shuttle boundary so exactly that the blue and orange contours overlap and are difficult to see except at the shuttle tail.

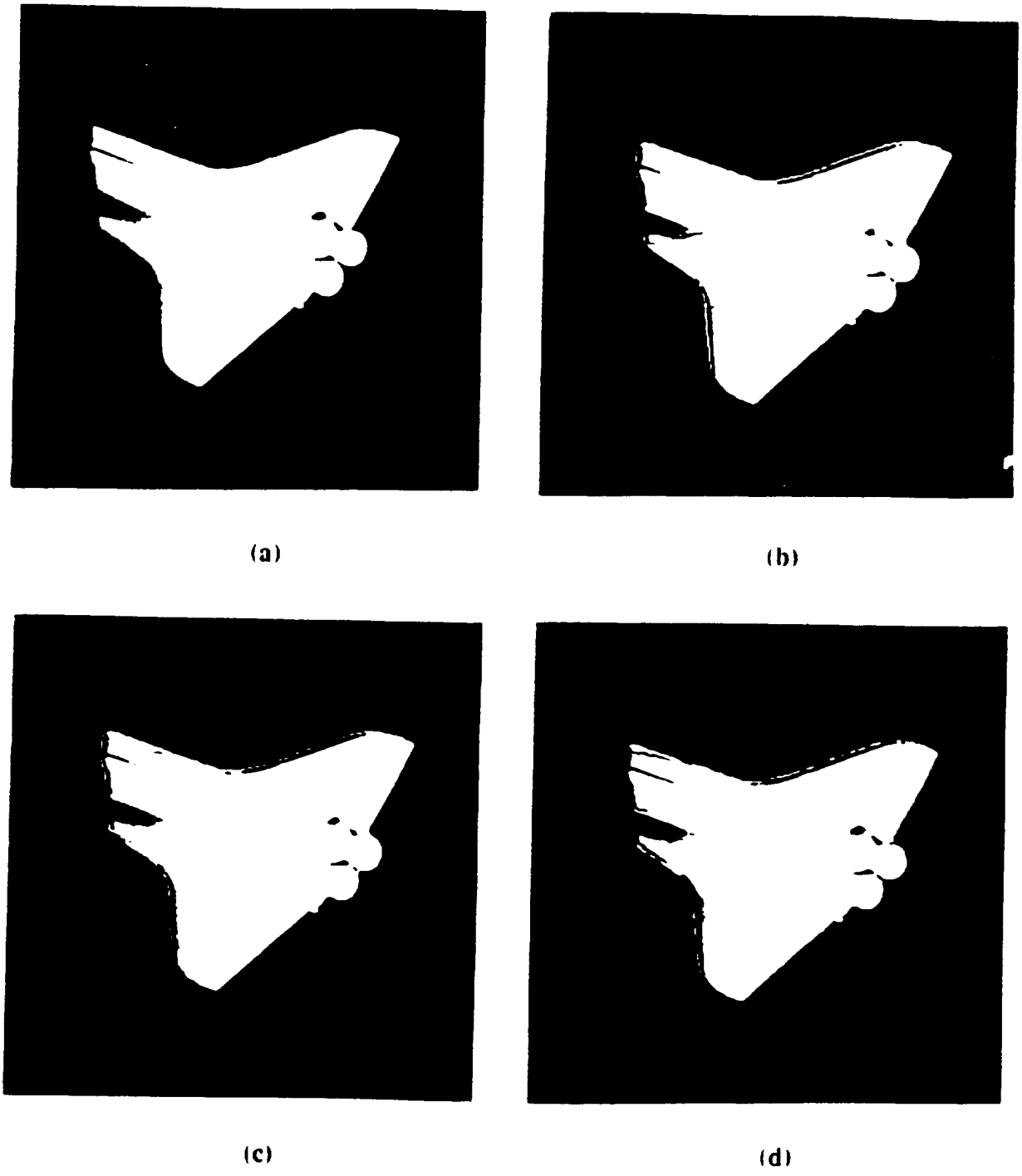


Figure C-9: **(a) Starting with an image of a partially occluded object (a), the tracking algorithm quickly estimates the view parameters of the object by, (b)-(d), iteratively matching the model silhouette (blue contour) to the object boundary (orange contour).**

C-5: KMS Software Quality Assurance Procedures

VGD's software development is proceeding through the project phases outlined in the KMS Software Quality Assurance Standards to ensure the development of high-quality, well-documented software meeting contractual requirements.

As a step toward insuring the continued quality of KMS software projects, an internal KMS project resulted in an update to the KMS Software Quality Assurance (QA) procedures. These procedures provide a formal framework within which high quality and cost-effective software will be developed at KMS. The KMS Software QA Standards incorporate recognized, industry-standard QA guidelines as published, for example, by ANSI and IEEE.

KMS felt that it was important to fully implement these updated procedures before proceeding with detailed development of the View Generated Database. All VGD project activities will be governed by these procedures. This insures the system which is delivered will adhere to the detailed "Data Item Descriptions" which are included in the contract.

The development of VGD is proceeding through the project phases illustrated in Figures 10 and 11. The current phase is the project plan. This will be completed when we have agreed upon the course that VGD is to take. When the plan is complete, a copy will be included in the next progress report. Based on the Project Plan outline, the **Requirements Definition** step is will be initiated. Throughout the VGD project, KMS will communicate detailed specifications and design decisions to the NASA via the regular quarterly reports, as indicated in the contract. As addenda to these reports KMS will include relevant documents produced in satisfaction of our QA procedures.

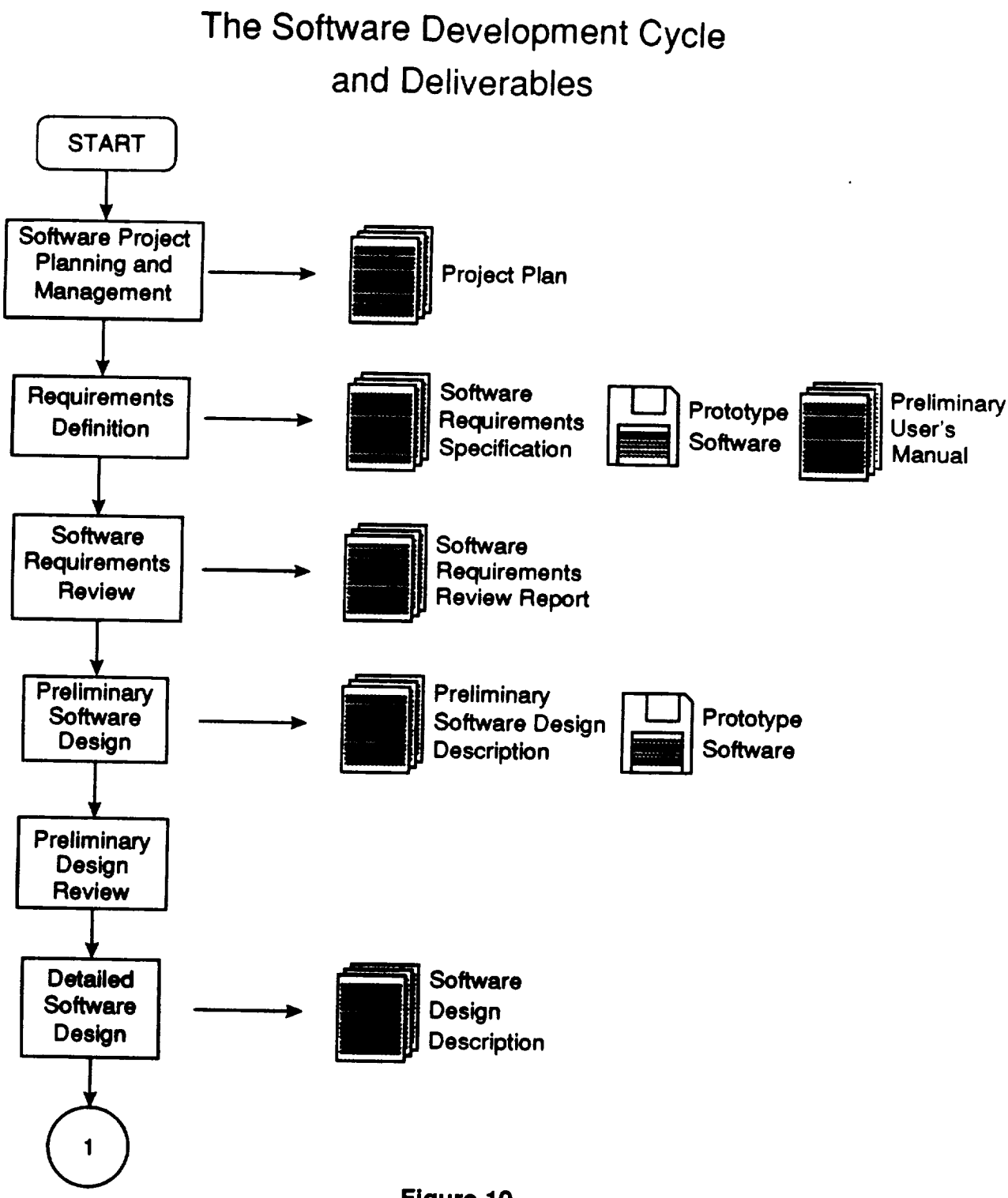

The Software Development Cycle
and Deliverables
START
Software Project Planning and Management
Project Plan
Requirements Definition
Software Requirements Specification
Prototype Software
Preliminary User's Manual
Software Requirements Review
Software Requirements Review Report
Preliminary Software Design
Preliminary Software Design Description
Prototype Software
Preliminary Design Review
Detailed Software Design
Software Design Description
1

Figure 10

The Software Development Cycle and Deliverables (cont)

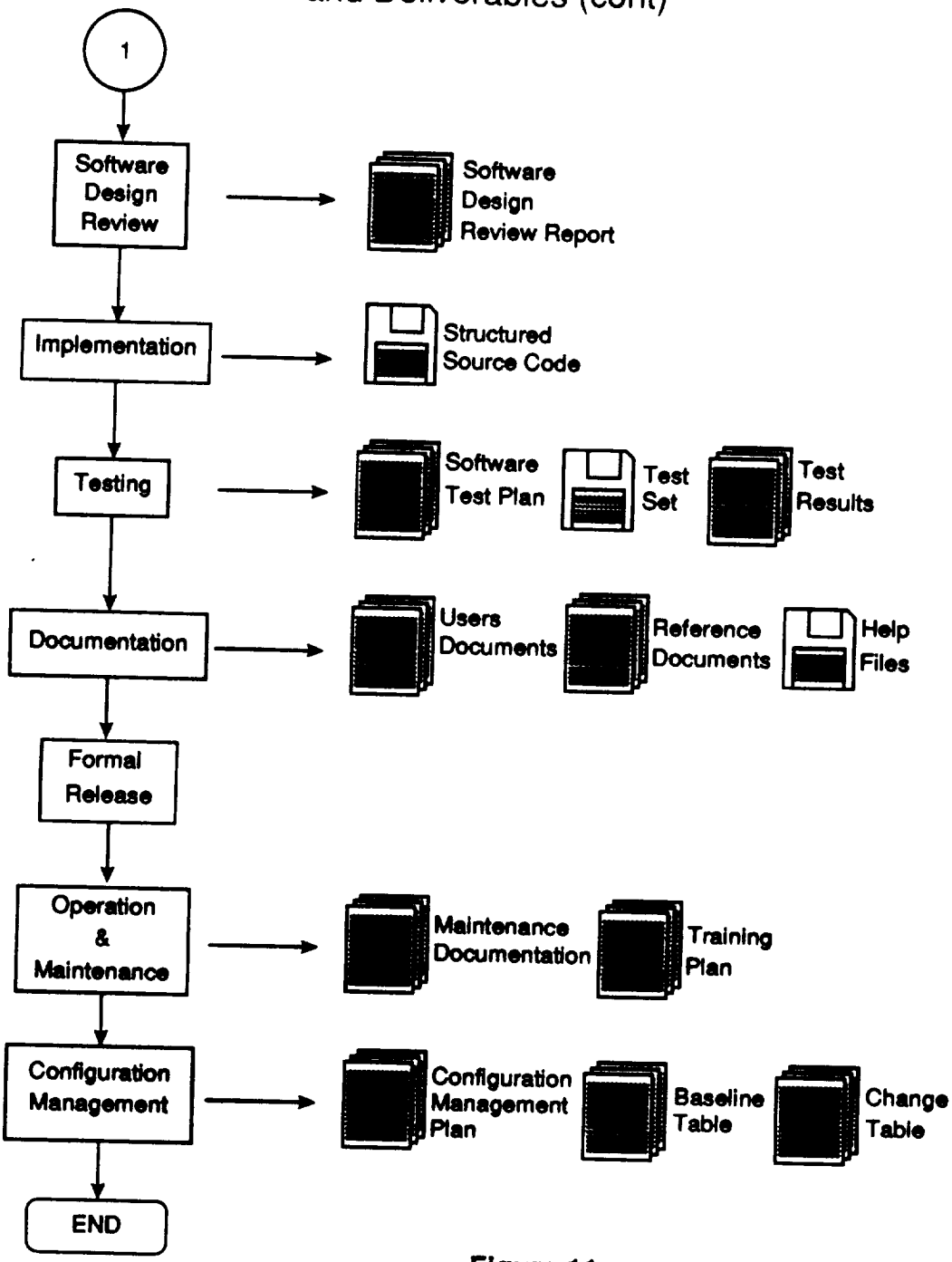


Figure 11

C-6: Review of 3D Model Acquisition Techniques

The modeling of three-dimensional objects using multiple camera views is an active research area in computer vision and is poorly defined at present. The goals of this research have ranged from rendering realistic environments for simulation, modeling of the world for intelligent robot navigation, acquiring three dimensional models for object recognition, to supplying model data for CAD and biomedical applications. Approaches for automatic model acquisition can be characterized along two basic data fusion methods 1) volume intersection of object silhouettes projected on the image planes, and 2) fusion of object surface patches reconstructed from multiple viewpoints. This section describes the various approaches that have been attempted to achieve these goals, and provide an outline of the respective research efforts classified along these two lines.

The different research efforts can also be described by the type of input data and the object representation scheme used in fusing the data. Intensity and range images are the two most commonly methods for inputting data. Intensity data is typically captured with video cameras, while range data is collected using either a laser-range camera, structure lighting, or stereopsis. The sensor used dictates the type of information that can be recovered. Structure lighting yields sparse but more easily registered range data points than dense range maps. Dense range maps are constructed slowly with a laser rangefinder that scans an object scan-line by scan-line. Conversely, intensity data can be instantaneously captured by inexpensive sensors. However, such images do not yield depth information directly. In addition, techniques such as stereopsis must be applied to recover range data.

To build up a complete model of an object, the data from different images must be first registered and then combined. Data has generally been registered using a) a fixture to hold an object at well-defined view angles, b) a fixed array of lights and cameras to capture the image data, or c) registration points on the surface of an object, where a sufficient number of points are shared between views to register the views. Multiple views of an object must be combined in order to capture the shape of a 3D object. Viewpoints can be constrained to 3 (orthogonal) or 13 positions to yield an efficient fusion algorithm. Fusion of data can be performed efficiently if the objects to be modeled are of some specific type, eg. polyhedral and cylindrical objects. Similarly, fusion can be simplified if object silhouettes can be obtained.

Finally, different representations have been used to model objects and for their reconstruction. The three basic categories have been volumetric, boundary, and wire frame models. Object models can be further organized by decomposing an object into modeling primitives. For example, volumetric models have been used that have volume elements, i.e. voxels, as their basic modelling primitives. Surface patches, on the other hand, are used to construct surfaces in boundary models. Lastly, coarse models can be derived by composing an object from generic primitives, such as cylinders and ellipsoids. In order to capture accurate surface detail, the VGD system will employ boundary models of surface patches for the 3D objects to be modeled.

C-6.1 Volume Intersection

A very popular data structure used for volume intersection are octrees. Octrees are 3D counterparts of quadtrees [Sam85], a hierarchical data structure for image regions. An octree is a true volumetric model representing a 3D object space, defined by a right-handed object coordinate system, to contain a $2^n \times 2^n \times 2^n$ array of volume elements called voxels. A voxel, the 3D equivalent of a pixel (picture element), is assumed to contain a homogeneous volume of material and can be either occupied or vacant. It also defines the resolution of the modeling system. The usual shape of a voxel is a cube although a rectangular parallelepiped could be used in a more general system.

The volume of the array of voxels is called a 3D voxel universe and it is aligned with the object coordinate system. It is assumed that all objects of interest are within this universe and remain there during all operations. The space outside the universe is assumed to be void of all objects. An octree is an 8-ary tree structure generated by a recursive subdivision of the modeling universe into octants until homogeneous blocks of voxels are reached. If an octant does not consist entirely of the same type of voxels, then it is further subdivided until homogeneous cubes, possibly single voxels, are obtained. The root of an octree is at level 0 and the voxels are at level n.

The generation of octree models of 3D objects consists of three steps. The first step generates conic octree volumes with the silhouettes of objects projecting from the image plane into the octree universe. The second step combines a sequence of such conic volumes into a model of the objects using simple volume intersection. The last step enhances an octree model by labeling individual objects, adding surface-normal vectors, and mapping intensity textures and other intrinsic properties on the surfaces of these objects.

After extracting the contour information and constructing the view cones, the volume intersection step can be done either in 3-space, the octree universe, or in the individual image planes. Other variations include the permitted number and flexibility of viewpoints during model construction. Restricting the possible viewpoints decreases the flexibility and resolution of the construction process but enables one to derive simple and fast algorithms for handling (simple) common cases. A variety of data structures for encoding an octree have also been derived for specific performance advantages in processing speed and storage overhead. These differences and their corresponding systems are highlighted below.

C-6.2.1 Review of Volume Intersection Systems

[Pot87] does the volume intersection by back-projecting the octree universe into the image plane. The silhouette of a perspective projection of a cube into an image plane depends on the relative position of the center of projection of the image and the coordinates of the six faces of the cube. The faces partition the 3D space of the cube into 27 half-spaces: 26 outside and 1 inside. The index number of a partition that contains the center of projection, relative to the current octree node, is used to find, in a lookup table, the number of silhouette vertices and their coordinates. The coordinates of the cube vertex are projected into an image plane using 12 lookup tables for

the object-to-image space transformations. The polygonal silhouettes of the objects, defined by the vertex coordinates, are then approximated by their bounding rectangles encoded as quadtrees. Each polygon is finally raster-scan converted for volume intersection, ie. the pixels inside the polygon are determined in scan line order and are individually compared with the contents of the (binary) image.

This is one of the more flexible octree schemes but the overhead in arbitrary back-projection of the cube and the individual raster-scan conversions is high. Moreover, it requires good camera calibration and viewpoint registration techniques to compute accurate object-to-image space transformation for each image.

Instead of performing the costly perspective backprojection, view cones can be projected into the octree universe and intersected directly in 3-space. [Nob88a,Nob88b] constructs polyhedral view cones for each image and projects them into the cube for volume intersection. If a view cone is not convex, it is defined as the union of partitioning convex cones. The cones are then used to check and classify the eight subregions of the parent cube as inside, intersecting and outside each convex cone. Cone unification rules are next used to do the same classification for the eight subregions with respect to each non-convex cone. Lastly, cone intersection rules are used to integrate the information on the subregions from the multiple view cones into the common region. A DF (Depth-First) representation [Kaw80] for linear encoding of the initial eight subregions is generated and recursively applied during model construction.

This octree building algorithm does not build a complete octree for each cone, but instead builds only a part of the octree within the common region. It is thus able to process an arbitrarily selected region in 3D space independent of all other region. The procedure is fast when restricted to polygonal silhouettes. More complex contours can first be approximated by linear segments but at the cost of processing a larger number of convex cones and introducing additional digitization inaccuracies. Accurate viewpoint positions are needed for the projections into the cube.

The above approaches are flexible but computationally expensive. An alternative is in restricting the possible viewpoints to yield simple and fast algorithms for handling specific cases. [Chi86a,Chi86b,Chi89] constructs the volume/surface octree from silhouettes obtained at three orthogonal views. Each occluding contour, encoded into the "principal" quadtree of the associated silhouette, is first fitted with a tension-spline [Sch66] to allow better approximations of the contour normals and hence more accurate surface information. With scaled orthographic projection along the viewing directions, each of the three "principal" quadtree sweeps out an oblique cylinder into the cube. Since sub-cylinders inside this volume may be identical, its exact octree can be replaced with a pseudo-octree that contains no identical subtrees. These orthogonal pseudo-octree are then intersected using simple tree traversal techniques to obtain the resultant model octree [Jac80].

This approach may be useful for simple objects that are symmetric along its three principle axes. The major disadvantage, on the other hand, is the inability to guarrantee that all significant features will be captured by the constructed model.

[Ahu89] considers a less restrictive case in using silhouette images obtained from any subset of 13 orthographic viewing directions. By restricting to these 3 "face" views, "6 "edge" views and 4 "corner" views, a simple relationship between pixels in the image and the octant labels in the octree is derived. The detection of intersections between the octree and the objects is thus replaced by a simple table lookup operation between a pair of views.

This approach provides more accurate information with the higher degree of reconstruction accurarcy but it still has problems. Discontinuities cannot be modelled, as evidenced by the approximation of object edges by rectangular steps. The proposed set of 13 viewpoints is not sufficient to build accurate models. The number of viewing samples needed for building an accurate model is arbitrarily large. The number of viewpoints necessary for reconstruction to within a desired accuracy depends on the viewing direction and the target object.

[Car85] proposes a solution for modelling discontinuities with the polytree model. It attempts to include more surface information with three extra classes of voxel structure: vertex cell, edge cell and surface cell. The result shows only negligible improvement. A forerunner of the octree is the "volume-segment" representation proposed by [Mar83]. It borrows scan-line techniques from computer graphics to construct a 3D object model from the bounding volumes carved out by the occluding contours. A wire frame with planar polygonal facets is then derived from the volume segments. This approach is slow, inaccurate and suffers from the usual problems associated with silhouettes. [Cap87] extends it to handle real images of parts and compares the constructed models with renderings of the original CAD designs.

Other less attractive volume intersection techniques have been proposed. [Che88] constructs a 3D model from a set of 3-view (orthogonal) type line drawings. Objects are restricted to polyhedral, cylindrical and composites of the two that are not rotationally symmetric objects. The object must be decomposable into subparts for individual reconstruction and then merged to form a Constructive Solid Geometric (CSG) [Man88] representation of the object. [Ide86] has an even more restricted system that reconstructs models of only polyhedral objects from a set of 3-view type drawings. [Sha86] uses simulated 2.5D sketches containing only depth information for their model construction but their volumetric representation and algorithms show no advantages over octrees. Similarly [Jiy86] describes an iterative 3D algebraic reconstruction technique with results for two synthetic test models under a variety of projections. The technique is, however, primitive compared to octrees.

C-6.2.2 Disadvantages of Octrees and Volume Intersection

Octrees are an elegant data structure which enables a simple implementation of volume intersection for model construction. This advantage is, unfortunately, offsetted by numerous problems.

One difficulty is that octrees can only provide coarse models of 3D objects. Astronomical storage and processing overhead are required to capture surface details accurately. Furthermore, the models are unstable and not invariant to rotation and translation. Thus, completely different octrees have to be constructed for slight changes in voxel resolution or object position.

A second problem with volume intersection, in general, stems from the dependence on silhouette information. First, using silhouettes requires reliable contour extraction and smoothing techniques. Second, silhouettes do not generally capture surface concavities and object self-occlusion effectively. Third, digitization effects during volume intersection often result in false boundaries in the final model.

There are possible remedies to some of these problems. For example, fine details can be generated with surface interpolation over the voxels with more (post)processing. The accuracy of this process depends, however, on the initial octree approximation. An alternative would be to use long sequences of frames to refine an octree. This, however, requires that the best views be selected, a difficult problem for unknown objects. Typically, arbitrary decisions are made on the trade-off between accuracy and computation, and the trade-off between coarse and fine resolution.

Not surprisingly, all experimentation with octrees to date has only been done with synthetic data.

C-6.3 Surface Patch Fusion

In general, surfaces can be expressed in an implicit or explicit form [Bol89]. An explicit form of a surface is the graph of a function of two variables. Let $f : U \subset \mathbf{R}^n \rightarrow \mathbf{R}$. The graph of f is defined to be the subset of $\mathbf{R}^{n+1}$ consisting of the points $(x_1,...,x_n, f(x_1,...,x_n))$ for $(x_1,...,x_n) \varepsilon U$, where U is an open subset of $\mathbf{R}^n$. The implicit form of a surface in $\mathbf{R}^3$ space is expressed as a set function $f : \mathbf{R}^3 \rightarrow \mathbf{R} \mid f(x,y,z) = \text{constant}$, where (x,y,z) are Cartesian coordinates of the surface points. The explicit form is a special case of the implicit equation and is sometimes referred to as a Monge patch. Another useful representation of a surface is the parameterized form. A parameterized n-surface in $\mathbf{R}^{n+k}$ ($k \geq 0$) is a smooth map $\phi : U \rightarrow \mathbf{R}^{n+k}$, where U is a connected open set in $\mathbf{R}^n$, such that $\partial \phi_p$ is non-singular (has rank n) for each $p \varepsilon U$ (which is called the regularity condition). A parameterized n-surface in $\mathbf{R}^n$ is simply a regular smooth map from one open set U in $\mathbf{R}^n$ onto another.

These surface patches are integrated together in an adjacency graph for the final representation of a 3D object. An adjacency graph is basically a bidirectional graph with each node representing a region and the links between nodes the region connectivity relationships. Such a graph is not, however, restricted to describing surfaces but can directly incorporate information and relations on edges, vertices (corners) or even generic primitives – all of which are derived from surface patches. Compared with volume intersection, the approach based on surface patches yields surface models of higher accuracy and detail in adition to allowing incorporation of surface reflectance properties during model reconstruction.

The standard model construction process has four major steps. The first step of segmenting the image into coherent regions can be driven initially by edge detection or region growing [Bal82,Hor86]. In the former method, the areas bounded by the extracted edge points are connected and tested for coherence as segmented regions. In the latter method, image pixels are grouped together on the basis of region coherence using intrinsic surface properties derived from differential geometry [Bes86]. In the second step, the surface patches derived using a least-

squares surface fit are either labelled with an unique symbol, or described in terms of parametric equations. The patches may be recursively combined to form larger entities as long as some coherence predicate is satisfied. The third step relates the various surface primitives in an adjacency graph. Information on the different properties of a surface, such as reflectance and texture, may also be included at each node. Lastly, integration of multiple views is done via the transformation of the graph constructed from a new view to the global coordinate system, the matching of corresponding nodes in the two view graphs, and the modification and/or insertion of surface primitives to refine the relational object description.

C-6.3.1 Review of Surface Patch Fusion Systems

Few past systems that follow the above general strategy based on surface patches meet with complete success. Although [Und75] may have proposed the first system that learns models of objects from multiple views, only results for a synthetic image of a single convex polyhedral object are demonstrated. To simplify the learning process, objects are restricted to be planar and convex and useful viewpoints that are known in advance. [Dan82] constructs a model from four image views for a single object. Partial results for nine sets of synthetic data are produced by an incomplete system. [Osh79,Osh83] consider multiple 3D objects in a single range image. Region growing at multiple, successively more abstract resolutions is used to generate an incomplete relational model.

[Asa87] segments the input images into spherical, cylindrical, or planar surfaces using shading analysis. A synthetic image of a single object consisting of cylindrical and planar Lambertian surface with constant albedo is projected orthographically. [Dou81] uses depth cues and object models to construct a representation of outdoor scenes. After coarse segmentation of the input image, a highly abstract semantic net relating the regions and object models is built for the environment. [Xie86a,Xie86b] outlines an expert system that constructs a relational model of a scene using artificial 2.5D sketches. No result/output is shown.

After extracting edges and lines from intensity images taken at different vantage points, [Sha77,Sha84] constructs an object model by using the concept of vertex cycles to match junctions and line segments between images. This approach stems from the various early works of Clowes, Falk, Guzman, Huffman and Waltz [Bar81]. [Yam88] considers only objects composed of hinges, slides and solids. The modeler learns the number of these features and the relationships between lines and vertices in image. Results are shown for a pair of compasses, essentially a 2D object, lying on an uncluttered tabletop. [Bak77] describes a scheme used in building models of 3D objects through binocular and motion parallax analyses. Curvature irregularities in the the region boundaries are then correlated to construct the 3D object model. [Yac75] analyzes images of objects taken from a nearly vertical direction by a TV camera. After simple thresholding and contour tracing of the object outline, the object model is constructed as a list of propeties.

[Vem86,Vem87] scans objects resting on a base plane with a laser. The range image is segmented into regions that are a collection of surface patches homogeneous in certain intrinsic surface properties. A straight line on the plane is captured in intensity images. It is used as a

calibration mark for calculating interframe transfromations. Simple averaging is then used to merge two overlapping views. [Lau87] constructs simple and incomplete relational models of (convex or non-convex) polyhedral objects. Jump boundaries are extracted from a 3D range image by gradient-based edge detection. The edge points are used to generate occluding and interior contours which segment the object from the image background and into coherent regions. The hemispheric histogram, a specialization of the Extended Gaussian Image, is used to extract surface orientation information in detecting corresponding regions in an image of multiple objects. Regions, however, can not contain holes and undersampling may occur with increased obliqueness of a facet.

[Her84a,Her84b,Her86] incrementally constructs a 3D model of complex scenes from multiple images. Stereo and monocular analyses are performed for image data collected at different viewpoints. Linear structures representing building boundaries are extracted from the images and combined with hypothesized new vertices, edges and faces, using task-specific knowledge on block-shaped objects in an urban scene. The edges and vertices from such analyses are then used to construct the 3D wire frames. The MOSAIC systems handles very complex scenes but in order to achieve success, a lot of assumptions and constraints have to be used: task-specific knowledge on urban scenery, trihedral polyhedra scene primitives, and limited linear 3D scene features of edges and corners. Due to the task complexity, the matching of junctions in the scene is slow and the scene interpretation is incomplete. Surface detail is provided as simulation by registering image regions with object surfaces.

[Ste86,Con86,Con87] constructs 3D models of polyhedra objects. The objects are placed on a turntable and scanned by a laser stripe. Wire frames are built from connected chains of curvature and step edges approximated by straight line segments. They are fleshed and adjusted before obtaining a least-mean planar fit of the surfaces bounded by viewplane cycles (closed set of edges and vertices). A refined model is produced by intersecting the view polyhedra from multiple views using a boolean intersection algorithm implemented in the GEO-CALC system [Bar86]. Surfaces are rendered with a Lambertian scattering model. This system is restricted by the need of placing the polyhedral objects on a turntable. No surface detail such as texture is modeled by the wire frame representation.

Instead of just surface patches, [Fer85,Fer88] take the surface patch representation a step further. A 3D object model from multiple views is derived in terms of volumetric primitives such as ellipsoids and cylinders. This is done by taking the intrinsic surface feature points, derived on the basis of surface principle curvatures, and grouping them together to form extremal and parabolic contours for parsing a surface into patches. 3D surface point correspondences for multiple views is done by normalized cross-correlation of the feature fields in the surface graphs. In matching candidate feature neighborhoods between views, the interframe transformations are recovered to construct a composite surface graph for the object. This composite surface graph, realized as a set of dynamically intrinsic images (DII), is used in the geometric inference process to identify the volumetric primitive associated with a patch (or set of patches) and to obtain the primitive's parameters. Mulitple primitive instantiations for the same subset of image data is resolved via shape similarity functions. Results are presented for a symmetric synthetic image and a range image of a statuette. The final models based on inferred volumetric primitives are

coarse and imprecise. Not only very little surface detail is captured, the modeling accuracy of the selected set of primitives is not obvious.

[Hu89] recovers 3D surface points for a scene containing multiple objects using structure lighting with a uniform grid. Either a striped image or a gray-scale image can be taken by merely switching the lights of the projector and the global coordinate system is fixed on the worktable. A calibration procedure computes the tranformation matrices M_C (worktable-camera) and M_P (projector-worktable) for computing the position of a surface point. After extracting the network of light stripes, geometric and topological constraints are used to hypothesize and test matches for solving the line labelling problem. The geometric and topological constraints are based on the uniqueness constraint (C_1) and the continuity constraint (C_2) of [Mar76]. Based on using these two sets of constraints for disambiguating surface solutions, [Hu89] presented five algorithms for 2D network extraction, single 3D point solution, single 3D network solution, network boundary extraction, and scene solutions. This approach is guided by three basic assumptions. First, objects in the scene are solid, static and opaque. Second, object surfaces are smooth and are of low order in the sense that the spatial frequency of surface undulation is less than the stripe frequency. Third, surfaces are much larger than stripe spacing so that a surface patch is covered by a multistriped network.

[LeM85]'s approach is based on the observation that the type of range data to be collected will differ with regard to sampling frequencies (in space and time) and resolution of range texture. A grid of horizontal and vertical lines including several dots is projected onto the scene. The dots are used as landmarks for initiating the line labelling process and "covers" the entire image. The camera and projector are separated by a vertical baseline so that range information is apparent in the distortions and discontinuities of the horizontal lines. The vertical lines are used to guide the extraction of the horizontal lines and to normalize the albedo variations. Grids can be designed with patterns of different thickness to be used for a multi-resolution range sensor. Associated with the finite thickness of the lines is an inherent "smoothing" of range texture. Higher frequency of range discontinuities cause the thinner projected lines to break up, whereas the thicker lines display very little distortion. A simple formula relates this smoothing of range texture to the thickness of the grid lines. In particular, the finite thickness of the lines imposes a maximum on the detectable range. After filtering the image with the Laplacian of Gaussian operator, a shrink and expand procedure is applied to extract the vertical lines. Row and column accumulators are used to locate the vertical bars and the grid intersections in the image. The albedos in the orginal gray level image is normalized by computing a local threshold based on a square neighborhood at each point. Labelling of the intersections is initiated with the location of the dots and completed by combining and interpolating among the initial labels. Disparity values are then finally assigned to the extracted and labelled intersections.

[Pot79] uses a pair of images containing a single object for such model construction. The object is pattern-illuminated by photogrammetric techniques and imaged under perspective projection inside a calibration fixture. Calibration marks on the fixture are extracted using the Laplacian operator and the Hough transform [Bal82]. The marks are used in computing the camera transformation matrix. The grid patterns are extracted in the form of straight lines and cubic curves using scan-line-to-vector conversion. Points and curves are then matched between images on the basis of the topology of the projected grid network to generate the 3D model. An inverse

mapping is used to reconstruct the matched cycle in 3D space using a least square-error technique. Nodes adjacent to this initial cycle are reconstructed and iteratively propagated in all directions using heuristic search methods. This process reaches quiescence when all the nodes are processed. This approach suffers from several problems. Surfaces are required to be photographed within a camera calibration fixture. The ten calibration marks have to be visible among the set of images. The surfaces cannot be transparent, highly reflective or totally black. Results are presented on the reconstruction of isolated surfaces using just polygonal shape approximations that contain no surface detail.

[Ver87] describes a method of obtaining 3D replica made of polyurethane foam from an arbitrary part of the human skin. An integrated system consisting of a photogrammetric stereo restitution system and a CAD system integrated with a NC 3D milling machine is used in constructing the replica. The depth values for a stereopicture of the frontal view of a human face is manually digitized. The surface is then fitted with B-spline functions which are used by the CAD system for the replica. [Duf88] scans the facial dimensions of a live human subject with a line of light from a low power laser under 5 seconds. The facial boundary is obtained by thresholding the range image and fitted with the longest possible line segments. Selected points in the depth map form vertices for the polygonal model of the face. Polygons are formed in such a way that where the surface details are complex, the polygons are small and where the surface is featureless, large polygons are generated. After hidden surface removal, texture mapping using Phong shading [Rog85] incorporates surface details onto the model facets.

[Sat86a] uses passive stereo techniques to measure the shape of statues on the Easter Islands from images taken at multiple viewpoints. For the solving the correspondence problem, small mark seals are stck on the statutes during the day and structured light patterns are projected onto the statutes during the night. Partial sterreo mathcing is done by dynamic programming and the epipolar line search. The correspondence search is completed with manual pointing by a human operator to construct the final range map. Results for images at a single viewpoint are demonstrated. [Sat87] obtains range data using the Liquid Crystal Range Finder (LCRF) based on a nematic liquid crystal mask [Sat86b] and Gray coding [Ino84]. The 3D model of an object on a turnatable is then constructed from a set of such range images taken at multiple view. The global coordinates is calibrated with respect to the floor (z-plane) and the rotation angle is calibrated against a reference cube. The wraparound 3D data is derived from horizontally sliced contours. Experimental resutlts is shown for one statuette.

C-6.3.2 Advantages of Surface Patch Fusion

Compared with volume intersection, the surface patch fusion approach yields surface models of higher accuracy and detail. It allows direct incorporation of surface reflectance properties during model reconstruction. It minimizes necessary storage and processing overheads. And last, nut not least, the parametric form for representing surface patches is invariant to motion.

Currently, robust techniques exist for calibrating cameras, calculating interframe transformations, recovering range information via stereopsis and triangulation, computing and reconstructing 3D surface solutions, and fusing 3D surface data from multipe views. The best work in this area are

exemplified by the systems of [Pot79,Hu89,Sat86a] as described above. Although they are incomplete and experimental in nature, they show considerable promise. Undoubtedly, a automatic model acquisition system can be realized in the near future.

C-7 References

C-7.1 References for Sections C-1 - C-5

Basri, R. and Uhllman, S., "The Alignment of Objects with Smooth Surface," Int. Conf. on Comp. Vis. and Pat. Recogn., 482 (1988).

Gottschalk, P.G., "Machine Recognition of 3-D Objects in Intensity Images," PhD Thesis, Univ. of Mich., (1989).

Turney, J. L., "Recognizing Partially Occluded Parts," Ph.D. Thesis, Univ. of Mich., (1986).

Van Hove, P., "Silhouette-Slice Theorems," Tech. Rep. TR-764, MIT Lincoln Lab, Lexington, MA (1987).

C-6.2 References for Section C-6

[Ahu89] N. Ahuja, J. Veenstra, "Generating octrees from object silhouettes in orthographic views," IEEE Trans. on PAMI, 2:2, pp.137-149, Feb. 1989.

[Asa87] M. Asada, "Cylindrical shape from contour and shading without knowledge of lighting conditions or surface albedo," Proc. IEEE ICCV'87, London, England, pp.412-416, 1987.

[Bak77] H. Baker, "Three-dimensional modelling," Proc. IJCAI'77, pp.649-655, 1977.

[Bal82] D.H. Ballard, C.M. Brown, Computer Vision, Englewood Cliffs, NJ:Prentice-Hall, 1982.

[Bar81] A. Barr, E.A. Feigenbaum (Eds.), The Handbook of Artificial Intelligence, Vol. 1-3, Menlo Park, CA: W. Kaufman, 1981.

[Bar86] M.M. Barry, C.I. Connolly, G. Spradlin, J.R. Stenstrom, D.W. Thompson, GEO-CAIC System Methods Reference Manual Version 1.0, Internal Report of Computer Science Branch, General Electric R&D Center, 1986.

[Bes86] P.J. Besl, R.C. Ramesh, "Invariant surface characteristics for 3D object recognition in range images," CVGIP, 33, pp.33-80, 1986.

[Bol89] R.M. Bolle, B.C. Vemuri, "On 3D surface reconstruction methods," IBM Research Report, RC 14557, IBM Research Division, Apr. 1989.

[Cap87] V. Cappellini, R. Casini, M.T. Pareschi, C. Raspollini, "From multiple views to object recognition," IEEE Trans. on Cir. Sys., 34:11, pp.1344-1350, Nov. 1987.

[Car85] I. Carlbom, I. Chakravarty, "A hierarchical data structure for representing the spatial decomposition of 3D objects," CG&A, 5:4, pp.24-31, Apr. 1985.

[Che88] Z. Chen, D. Perng, "Automatic reconstruction of 3D solid objects from 2D orthographic views," Pattern Recognition, 21:5, pp.439-449, 1988.

[Chi89] C.H. Chien, J.K. Aggarwal, "Model construction and shape recognition from occluding contours," IEEE Trans. on PAMI, 2:4, pp.372-389, Apr. 1989.

[Chi86a] C.H. Chien, J.K. Aggarwal, "Computation of volume/surface octrees from contours and silhouettes of multiple views", Proc. IEEE CVPR'86, Miami Beach, Florida, pp.250-255, 1986.

[Chi86b] C.H. Chien, J.K. Aggarwal, "Volume/surface octrees for the representation of 3D objects," CVGIP, 36, pp.100-113, 1986.

[Con86] C.I. Connolly, J.R. Stenstrom, "Construction of polyhedral models from multiple range views," Proc. IEEE ICPR'86, Paris, France, pp.85-87, 1986.

[Con87] C.I. Connolly, J.L. Mundy, J.R. Stenstrom, D.W. Thompson, "Matching from 3D range models into 2D intensity scenes," Proc. IEEE ICCV'87, London, England, pp.65-72, 1987.

[Dan82] C. Dane, R. Bajcsy, "An object-centered 3D model builder," Proc. ICPR'82, Munich, Germany, pp.348-350, 1982.

[Dou81] R.J. Douglass, "Interpreting 3D Scenes: A model building approach," CVGIP, 17:2, pp.91-113, Oct. 1981

[Duf88] N.D. Duffy, J.F.S. Yau, "Facial image reconstruction and manipulation from measurements obtained using a structured lighting technique," Pattern Recognition Letters, 7, pp.239-243, Apr. 1988.

[Fer85] F.P. Ferrie, M.D. Levine, "Piecing together the 3D shape of moving objects: An overview," Proc. IEEE CVPR'85, San Francisco, California, pp.574-584, 1985.

[Fer88] F.P. Ferrie, M.D. Levine, "Deriving coarse 3D models of objects," Proc. IEEE CVPR'88, Ann Arbor, Michigan, pp.345-352, 1988.

[Her84a] M. Herman, T. Kanade, "The 3D MOSAIC scene understanding system: Incremental reconstruction of 3D scenes from complex images," Proc. DARPA Image Understanding Workshop, New Orleans, Louisianna, pp.137-148, Oct. 1984.

[Her84b] M. Herman, T. Kanade, S. Kuroe, "Incremental acquisition of a 3D scene model fomr images," IEEE Trans. on PAMI, 6:3, pp.331-340, May 1984.

[Her86] M. Herman, T. Kanade, "Incremental reconstruction of 3D scenes from multiple, complex images," AI, 30, pp.289-341, 1986.

[Hor86] B.K.P. Horn, Robot Vision, Cambridge, MA:MIT Press, 1986.

[Hu89] G. Hu, G. Stockman, "3D surface solution using structured light and constraint propagation," IEEE Trans. on PAMI, 11:4, pp.390-402, Apr. 1989.

[Ide86] M. Idesawa, "3D model reconstruction and processing for CAE," Proc. IEEE ICPR'86, Paris, France, pp.220-225, 1986.

[Ino84] S. Inokuchi, K. Sato, F. Matsuda, "Range-imaging system for 3D object recognition," Proc. ICPR'84, Montreal, Canada, pp.???-???, 1984.

[Jac80] C.L. Jackins, S.L. Tanimoto, "Octrees and their use in representing 3D objects," CVGIP, 14, pp.249-270, 1980.

[Jiy86] X. Jiye, O.H. Kapp, "3D algebraic reconstruction from porjections of multiple grey level objects," Optik, 72:3, pp.87-94, 1986.

[Kaw80] E. Kawaguchi, T. Endo, "On a method of binary picture representation and its application to data compression," IEEE Trans. on PAMI, 2, pp.27-34, 1980.

[Kem88] K. Kemmotsu, Y. Sasano, K. Oshitani, "Model-based 3D vision system," SPIE: Expert Robots for Industrial Use, 1008, pp.40-47, 1988.

[Lau87] D. Laurandeau, D. Poussart, "Model building of 3D polyhedral objects using 3D edge information and hemispheric histogram," IEEE Journal Rob. Auto., 3:5, pp.459-470, Oct. 1987.

[LeM5] J. Le Moigne, A. M. Waxman, "Multi-resolution grid patterns for building range maps," Proc. VISION'85, Detroit, MI, pp.8.22-8.36, Mar. 1985.

[Man88] M. Mantyla, An Introduction to Solid Modeling, Rockville, MA:Computer Science, 1988.

[Mar76] D. Marr, T. Poggio, "Cooperative computation of stereo disparity," Science, 194, pp.283-287, 1976.

[Mar83] W.N. Martin, J.K. Aggarwal, "Volumetric descriptions of objects from multiple views," IEEE Trans. on PAMI, 5:2, pp.150-158, Mar. 1983.

[Nob88a] H. Noborio, S. Fukuda, S. Arimoto, "Construction of the octree approximating 3D objects by using multiple views," IEEE Trans. on PAMI, 10:6, pp.769-782, Nov. 1988.

[Nob88b] H. Noborio, S. Fukuda, S. Arimoto, "A fast algorithm for building the octree for a 3D object from its multiple images," Proc. IEEE ICPR'88, Rome, Italy, pp.860-862, 1988.

[Osh79] M. Oshima, Y. Shirai, "A scene description method using 3D information"," Pattern Recognition, 11, pp.9-17, 1979.

[Osh83] M. Oshima, Y. Shirai, "Object recognition using 3D information," IEEE Trans. on PAMI, 5:4, pp.353-361, Jul. 1983.

[Pot79] M. Potmesil, "Generation of 3D surface descriptions from images of pattern-illuminated objects," Proc. IEEE Conf. Pat. Reg. Image Proc., pp.553-559, 1979.

[Pot87] M. Potmesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images," CVGIP, 40, pp.1-29, 1987.

[Rog85] D.F. Rogers, Procedural Elements for Computer Graphics, New York, NY: McGraw-Hill, 1985.

[Sam85] H. Samet, "The quadtree and related hierarchical data," ACM Comput. Surveys, 16:2, pp.187-260, 1985.

[Sat86a] K. Sato, H. Yamamoto, S. Inokuchi, "3D shape measurement of megalithic statute -MOAI -," Proc. ICPR'86, Paris, France, pp.675-677, 1986.

[Sat86b] K. Sato, H. Yamamoto, S. Inokuchi, "Tuned range finder for high precision 3D data," Proc. ICPR'86, Paris, France, pp.675-677, 1986.

[Sat87] K. Sato, S. Inokuchi, "Range-imaging system utilizing nematic liquid crystal mask," Proc. ICCV'87, London, England, pp.657-661, 1987.

[Sch66] D.G. Schweikert, "An interpolation curve using a spline in tension," J. Math. Phy., 45, pp.312-317, 1966.

[Sha77] R. Shapira, H. Freeman, "Reconstruction of curved-surface bodies from a set of imperfect projections," Proc. IJCAI'77, pp.628-634, 1977.

[Sha84] R. Shapira, "The use of objects' faces in interpreting line drawings," IEEE Trans. on PAMI, 6:6, pp.789-794, 1984.

[Sha86] A. Sharma, S.A.R. Scrivener, "Constructing 3D object models using multiple simulated 2.5D sketches," Intl. J. Man-Machine Studies, 24, pp.633-644, 1986.

[Ste86] J.R. Stenstrom, C.I. Connolly, "Building wire frames from multiple range views," Proc. IEEE Conf. Rob. Auto, San Francisco, California, pp.615-620, Apr. 1986.

[Und75] S.A. Underwood, C.L. Coates, "Visual learning from multiple views," IEEE Trans. on Comp., 24:6, pp.651-661, Jun. 1975.

[Vem86] B.C. Vemuri, J.K. Aggarwal, "3D model construction from multiple views using range and intensity data," Proc. IEEE CVPR'86, Miami Beach, Florida, 1986.

[Vem87] B.C. Vemuri, J.K. Aggarwal, "Representation and recognition of objects from dense range maps," IEEE Trans. on Cir. Sys., 34:11, pp.1351-1363, Nov. 1987.

[Ver87] J.S.M. Vergeest, J.J. Broek, A. van Voorden, "Body surface measurement and replication by photogrammetry and computer aided design," Engineering in Medicine, 16:1, pp.3-8, 1987.

[Xie86a] S. Xie, T. Calvert, "Incremental construction of 3D models from a sequence of framed views: matching partial objects," Proc. Graphics/Vision Interface '86, Vancouver, B.C., Canada, pp. 300-306, May 1986.

[Xie86b] S. Xie, T. Calvert, "Constructing 3D models of a scene from planned multiple views," Proc. SPIE: Intelligent Robots and Computer Vision, 726, pp.233-239, 1986.

[Yac75] M. Yachida, S. Tsuji, "A machine vision system for complex industrial parts with a learning capability," Proc. IJCAI'75, pp.819-826, 1975.

[Yam88] K. Yamamoto, K. Sakaue, H. Matsubara, K. Yamagishi, "MIRACLE-IV: Multiple Image Reconstruction system Aiming Concept LEarning - Intelligent Vision," Proc. IEEE ICPR'88, Rome, Italy, pp.818-821, 1988.

[Yam86] H. Yamamoto, K. Sato, S. Inokuchi, "Range imaging system based on binary image accumulation," Proc. IEEE ICPR'86, Paris, France, pp.233-235, 198

Section D
Quarterly Report #2: March 1990

D-1. Introduction

In light of our clearer understanding of NASA's needs, we will proceed with the development of VGD as originally proposed. The work that was done this quarter, which was necessarily limited due to uncertainty with regard to the approach that KMS was to take. However, this uncertainty has now been resolved, and we understand that the originally proposed approach is most appropriate. Therefore, KMS will proceed full speed with the development of VGD according to the original proposal.

Since the last quarterly report, the hardware required for the project has been ordered. In addition, parts of the initial, tentative software requirements specification (SRS) for VGD has been completed. We will periodically include parts of the SRS as it becomes more complete for review by NASA. The remainder of this report includes the part of the SRS that has been completed for VGD thus far. This section of the SRS is a general description of VGD from the users' perspective. Shortly, KMS will complete the functional description portion of VGD's SRS. This section describes the functions that must be implemented in order to achieve the behavior described in the general description that has been included in this report.

D-2. VGD SRS: General Description of the VGD System

D-2.1: Perspective on VGD

VGD has many applications. A particularly important one is the capability to provide vision systems, such as that proposed in STORS, and tele-autonomous systems, with accurate geometric and photometric models of everyday objects -- objects that were not originally designed with CAD tools. State-state-of-the-art object recognition and tracking techniques require such models in order to recognize and localize objects. VGD allows such systems to operate on scenes containing any objects, not just the ones that have been created using a CAD system, thus circumventing the need to laboriously create such models.

D-2.2: Functionality of the VGD System

VGD design provides

- A hand-held stereo camera/structured light apparatus from which registered stereo and structured light images can be acquired;

- the capability to take multiple snapshots of a scene or object using the camera apparatus, and use them to produce a 3-d description of the scene or object. In particular,
  - calibration of the stereo cameras and the structured light system,
  - accurate determination of interframe transformations, first with manually placed features and later with existing features,
  - to determine coarse surface geometry using images of projected structured light that is registered with standard intensity stereo images,
  - recover detailed surface geometry and surface albedo of portions of the object corresponding to each view,
  - fuse the surfaces into a single world model or object model;

- an easy interface to existing modeling systems since
  - the 3-d model is represented both in planar patch format and surface spline format,
  - models may be stored in files adhering to standard 3-d modelling formats;

- a world-modeling database permitting
  - easy and efficient creation and maintainance of a world model,
  - searching based on geometric attributes such as surface structure and position,
  - searching based on user definable attributes;

- an easy to use, Macintosh-like operator interface that allows all facets of the system's operation to be controlled and monitored in a simple, intuitive manner;

D-2.3: Using VGD

The complex operations being performed by VGD can be as visible or invisible as the user desires. Most operations are accomplished using sequences of menu choices, or mouse clicks within images or graphical representations displayed on the screen.

There are several phases, or modes of operation in VGD. Some need only be used infrequently, such as the calibration phase. Others will be used extensively, such as the scene/object reconstruction phase. In particular, the primary user modes in VGD include

- frame acquisition
- calibration:
  - stereo camera calibration,
  - structured light projector calibration;
- frame registration;
- surface reconstruction;
- surface fusion;
- world model/database manipulation;

D-2.3.1: Frame Acquisition

Two types of frames can be acquired: a stereo pair of images, with and without structured lighting. The **frames** menu allows these types of frames to be acquired. The **frames** menu has four entries:

- **existing light stereo**: marks this option for the next acquire command;
- **structured light stereo**: marks this option for the next acquire command;
- **frame parameters**: sets the default frame parameters, such as the dimensions and position of the frame within the camera's field of view;
- **acquire frame**: acquires the frame, using the settings of the previous three selections (using defaults if not set), and stores them in viewable buffers of the user's specification.

Since acquiring a frame of structured light stereo uses all of VGD's image acquisition capabilities, this case will be described first. Acquiring a frame of structured light requires that (ideally) two simultaneous, registered stereo pairs be acquired: one with existing light and the other with existing light plus structured light. The structured light pattern can then be extracted simply by differencing the images. Since VGD's camera apparatus cannot simultaneously

acquire two such frames, the effect is obtained by acquiring two images in rapid succession, 1/60th of a second apart and differencing these images. VGD will direct this to occur, and save the result to a buffer if the **structured light stereo** option is chosen, followed by the **acquire frame** directive. If the **existing light stereo** option is chosen, then VGD will capture only a stereo pair with existing light, and save it to a buffer. If both are chosen (default), all of the above are saved in a viewable buffer.

The **frame parameters** option allows the user to set what size images will be aquired, as well as what portion of the camera's field of view will be used.

When both existing light stereo and structured light stereo pairs are captured, then VGD automatically associates these pairs as being taken from the same viewpoint.

D-2.3.2: Calibration

Calibration appears as a main menu having two entries:

- **Stereo:** takes an existing light stereo pair of a calibration card, prompts the user for correspondence between the patterns on the card in each stereo pair, and then generates a calibrated camera model.
- **Structured light:** Takes a structured light stereo pair of a flat surface and uses this to generate a calibrated model of the structured lighting acquisition process.

Choosing **stereo** allows the user to calibrate VGD's stereo camera apparatus. Calibration of a stereo camera consists of creating a camera model for each of the two cameras, and a precise determination of the relative position and orientation of the two cameras.

After choosing **stereo** from the calibration menu, the user is prompted to acquire a frame of existing light stereo. The frame must be taken of a **calibration card.** The calibration card consists of several ruled lines on a rigid, flat card. The card is placed in both cameras' fields of view. VGD graphically displays this frame, and directs the user to select as many corresponding pattern elements as possible (sets of intersecting lines are currently being used) between the two images. The user does this simply by clicking on corresponding pattern elements. After the user notifies VGD that he has made all possible correspondences have been made, VGD creates a calibrated model of the stereo cameras.

Choosing **structured light** allows the user to calibrate VGD's structured light projection apparatus. When this option is chosen, VGD directs the user to take a structured light stereo pair frame of a flat surface. VGD then directs the user to designate the areas of each image that consist of the calibration surface. This does not need to be done exactly; simply using the mouse to circle a valid area in each image is sufficient. VGD uses the valid regions to create a calibrated model of the structured light projector.

D-2.3.3: View Registration

In order for VGD to be able to fuse portions of the scene surfaces that it has reconstructed, it must register the frames taken of the scene. Registration consists of finding a transformation between the views. Ultimately, VGD will register views automatically, and will only require user assistance if problems are encountered. Before reaching this state, VGD will require users to select corresponding image features from each view.

In the initial version of VGD, manually placed features (in the form of removable, stick-on marks) will be used. Later, existing features will be used. The user will be presented groups of existing-light stereo pairs, and be asked to click on corresponding features in each view. VGD will present its hypothesis about which features correspond, both in the stereo pairs and in the larger scope of the set of views. Thus, the user interacts mainly to correct VGD's errors, if any.

In order to register a set of views, the user first selects the frames that he wishes to register. After these frames have been selected, VGD displays them in reduced, icon-like form (64x64) so that the user can quickly examine a large number of frames at once. The user may arrange the iconized frames on the screen in any manner he desires. VGD provides a set of zoom/reduce functions that allow the frames to be enlarged when accuracy is required, and reduced to conserve space otherwise. The user may then designate correspondences between features in distinct frames, or between the images in a single frame. VGD will protect the user from accidentally putting features into correspondence that are not visible from both images of a stereo pair. VGD allows the user to display groups of designated corresponding features by highlighing them in color. Further, VGD allows the correspondences to be edited in the case errors were made.

After the user is satisfied with the accuracy of the correspondences between features, VGD registers the views by determining the 3-d transformation between the world coordinate system and the frame coordinate systems. After the 3-d transformations have been computed, VGD associates the transformation with each frame for possible later use.

D-2.3.4: Surface Reconstruction

Reconstruction of a surface requires that the user has already taken a set of views of a scene/object and registered the views. It also requires both an existing light stereo pair and a structured light stereo pair of each view.

Surface reconstruction is initiated by choosing the **Reconstruct surface** item under the **Build** menu. When this item is selected, VGD first asks the user for a set of frames to use as data. For each frame of the selected set, VGD reconstructs those portions of the 3-d surface structure and photometry of the scene that can be reliably computed.

D-2.3.5: World model manipulation

Once surfaces have been reconstructed from frames, they become part of a **world model database** that allows operations to be performed on the component surfaces. In the absence of additional information, VGD adds a surface to its currently existing catalog of the surfaces in the

world model. In many cases, however, VGD is able to determine likely points of segmentation between objects, and also is able to determine which portions of objects should be fused into a single, larger surface, which may then be grouped with other surfaces into objects in the world model. VGD provides editing and viewing capability of surfaces and objects stored in the world model database, so that if VGD makes any errors, they can be corrected by the user.

D-2.3.5.1: Surface Fusion

Once a set of surfaces have been reconstructed, they may be fused into a smaller number of other 3-d entities. This is done by choosing the **Fuse surfaces** option under the **Build** menu.

D-2.4: Constraints on VGD

VGD is limited by the available processing power as well as the degree of user input that it requires. VGD achieves its high level of functionality by using advanced, computationally intensive algorithms, particularly during the surface reconstruction and fusion phases of the system. This constraint can only be relaxed by running the algorithm on a more powerful machine. As for the limitations imposed by user assistance, VGD will require less user assistance as it matures, approaching full automation.

Section E
Quarterly Report #3: June 1990

E-1. Summary of Progress

The surface reconstruction phase of VGD is progressing rapidly now due to the discovery of a new approach to surface acquisition which also holds a great deal of promise as a general purpose range camera. This technique is based on projection of phase-shifted, sinusoidal structured light patterns onto the scene. As described in Section 2.1 this allows the geometrical structure of the surfaces in the scene to be separated from photometric properties of the surfaces in the scene. This permits surfaces to be acquired quickly and easily. The primary engineering challenges with this approach have been fabrication of accurately sinusoidal transmission gratings and accurate calibration of the apparatus. However, these problems are on the verge of being solved. Section 2.1 contains a description of the theory of the new technique, how KMS is implementing it, and some preliminary surface data acquired using the technique.

Work on surface merging is also well underway. The central problem here is accurate determination of the viewpoint. We have nearly completed implementation of the software for accomplishing this. The software for the other aspects of surface merging, such as conversion of the pixel-based surface format into a format more amenable to manipulation, and such as the accurate combination of geometric and photometric data from separate surfaces into a single surface, has been designed. Work is just beginning on the implementation of these phases of the task.

E-2. Technical Status

E-2.1 Surface Reconstruction

E-2.1.1 Problems with Existing Surface Reconstruction Approaches

KMS has found previous approaches to the problem of surface reconstruction unsuitable to VGD. We discuss the reasons for this in the following paragraphs.

Passive sensors, i.e., sensors that use only ambient illumination, employ binocular stereo,[1,2] trinocular stereo,[3] epipolar motion stereo[4], or axial motion stereo[5] to derive range maps. These methods are extremely compute intensive, subject to error, and provide only sparse measurements of range data that must be interpolated later using even more compute intensive algorithms.[6]

Active sensors, i.e., sensors that produce their own illumination of the scene, include image laser radar systems, structured lighting approaches, and fresnel diffraction systems. Image laser radar systems use time of flight,[7,8] amplitude modulation,[9,10] and phase modulation.

Time of flight sensors determine range by measuring the travel time of light from the system to the object and back. The problems with this approach they are useful only for objects at large distances. The time delays for light travel over short distances are difficult to measure. In methods using amplitude and phase modulation the intensity or phase of laser light is modulated at a fixed frequency, bounced off the target object, detected by a photodiode, and phase-compared to the original output signal to determine a relative range. If the modulation frequency is chosen so that the object falls within one cycle of modulation, absolute depth can be determined. Problems with these approaches are that they are too bulky and power hungry for space-based robotic applications. For example, the lightest system of this type, built by Odetics, weighs 33 lbs and uses 42 W of power. In addition, many of these systems use complex scanning mechanisms that can be easily damaged.

Structured lighting approaches project a known pattern onto an object and interpret depth to the object by triangulating on the pattern. Patterns include points, lines, grids, circles, crosses, stripes, binary coded patterns, and random texture[11-20]. The problem with many of these approaches is that they require solving the correspondence problem, i.e., determining which projected feature, point, line, cross, etc., corresponds to which image feature. Once this problem is solved, the range measurements, as in passive methods, are only obtained at sparsely distributed points, except for the the coded binary pattern methods, which provide denser information. However, the binary coded patter approaches have difficulty separating the pattern from the surface detail of an object.

Projection moiré interferometry[21,22] is variation of structured lighting in which a fine grating pattern is projected onto an object and the pattern is read back through a second grating to produce moiré fringes. If the setup is properly aligned, the fringes will correspond to contours generated by the intersection of imaginary planes parallel to the camera system and the object under study. Single moiré images, however, do not uniquely characterize the surface.

An improved approach to moiré is phase-shifted moiré[23] in which the moiré pattern is shifted in phase by manipulating the projection grating. The result is that the phase of the moiré pattern is shifted in phase. Combining a set of phase-shifted patterns allows the underlying range data to be uniquely determined. Problems with phase-shifted moiré are that the grating frequencies are fixed and, therefore, the sensitivity of range measurements is also fixed. In addition, the return pattern has a underlying grating pattern superimposed on the moiré pattern which represents a form of noise that must be filtered.

E-2.1.2 KMS' Solution for VGD

Our investigation into this problem has led KMS to an ingenious solution. The technique uses a new type of structured lighting wherein phase shifted cosine patterns are projected using high-quality sinusoidal slides onto the surfaces of the scene and using a CCD camera to image them. We call this new technique SUrface Reconstruction by PHAse-shifted CosinEs (SURPHACE).

In SURPHACE, projecting a known pattern onto a surface allows the range of the points be triangulated by measuring the distortion of the pattern caused by the geometric properties of the surface. Unfortunately, the photometric properties of a surface also affect the amount of light

that is returned to the camera, and, thereby contribute to the distortion of the projected pattern. With a single image, it is impossible to separate the affects of geometric and photometric properties without a great deal of knowledge about the lighting and surface properties.

Using multiple images and a technique borrowed from interferometry, known as phase-shifting, the distortions in the pattern caused by the geometric properties can be isolated from those caused by surface photometric properties. Specifically, the slide projects a sinusoidal pattern with a fixed spatial frequency ω_0. Referring to Figure E-1, the pattern returned to the camera has the original carrier, ω_0, modulated by a phase function, $\Phi(x,y)$. The phase function $\Phi(x,y)$ measures the distortion of the return pattern due to the geometric properties of the imaged surface.

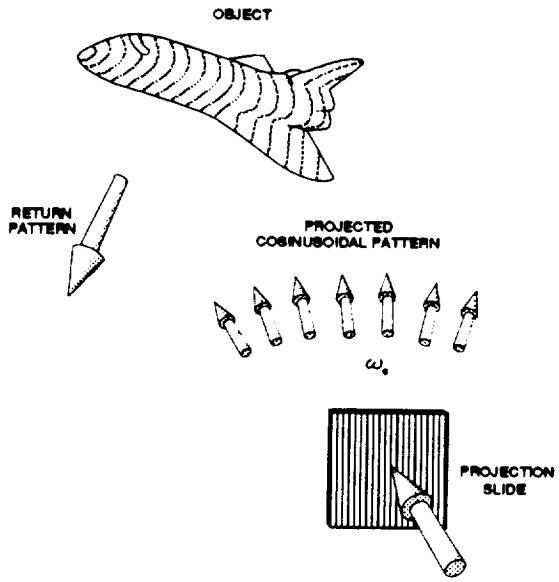


Figure E-1: **SURPHACE obtains range by projecting three sinusoidal patterns that have been phase shifted from each other triangulates on the resulting distorted phase function to determine the range to the surface.**<ps=12

The intensity, $I_R(x,y)$, of the pattern returned to the cameras can be modelled as modulated sinusoid:

$$I_R(x,y) = I_A(x,y) + I_B(x,y) * \cos(\omega_0 * x + \Phi(x,y) + \phi_i),$$

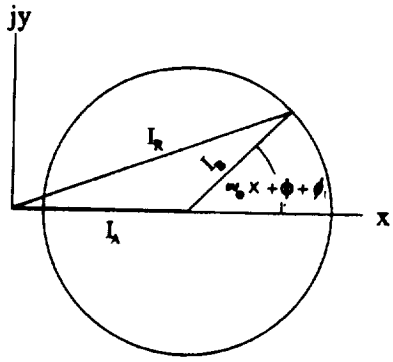
where $I_A(x,y)$ represents variations in the intensity due to background, or ambient lighting, and $I_B(x,y)$ represents the variation in contrast of the cosine pattern due to the photometric properties of the objects imaged. The term ϕ_i represents a phase shift that will be introduced into the pattern by moving the slide in the projector.

We may solve for Φ as follows. For each pixel at location (x,y) the return intensity can be represented by the real component of a phasor I_R,

$$I_R = Re(I_R) = Re(I_A + I_B * (\cos(\omega_0 * x + \Phi + \phi_i) + j \sin(\omega_0 * x + \Phi + \phi_i)))$$

(temporarily suppressing the x and y arguments in the notation).

The phasor I_R can be schematically represented in the complex plane as,



where I_A represents an unknown offset to the center of a circle in the complex plane, I_B represents an unknown radius of the circle, and $\omega_0 * x + \Phi + \phi_i$ represents an unknown offset angle on the circle.

Since the center of the circle is confined to the real axis, therefore possessing two degrees of freedom, two measurements, shifted in phase from each other, suffice to determine the circle. A third measurement, shifted in phase from the previous two, determines the end point of the phasor on the circle. Since the circle is specified by I_A and I_B, and the location of the point on the circle specifies Φ, three phase shifted measurements are all that is necessary to determine I_A, I_B and Φ. In fact, Φ can be determined by

$$\Phi(x,y)|_{2\pi} = \tan^{-1}((\Sigma\, I_{Rj}(x,y)*\sin(\phi_j)), (\Sigma\, I_{Rj}(x,y)*\cos(\phi_j)))$$

where $I_{Rj}(x,y)$ corresponds to the j^{th} phase-shifted image and j takes on values from 0 to 2, where $\phi_j = 2\pi j/3$. This operation can be performed with a table-lookup operation for an extremely efficient conversion of the phase-shifted images to phase data.

Because multiples of 2π are lost when the phase is encoded in the cosine functional, $\Phi(x,y)$ is extracted as *raw* data with phase values within the range 0 to 2π. In this case the raw phase is "unwrapped" from its 0 to 2π range by finding the locations at which artificial jumps, i.e., phase jumps due to the artificial restriction of the phase to the range 0 to 2π, and adding in an offset function that has the jumps of 2π with the opposite sign resulting in the desired unwrapped, continuous phase function. In practice, noisy data can cause phase jumps to be lost or to occur in the wrong locations. This problem can be solved trivially by measuring the range data with two different spatial frequencies and combining the two sets of data, which will have different phase jump locations, to unwrap the phase data.

Finally, the range data can be recovered easily from the phase data. For example, for the imaging configuration shown in Figure E-2,[15] the phase function $\Phi(x,y)$ is inversely related to the range, Z, from the camera to the imaged surfaces,

$$Z = (a_c * X_c)/(\Phi(x,y) * p),$$

where a_c and X_c are defined in Figure E-1, and p is the pitch of the grating in centimeters per radian.

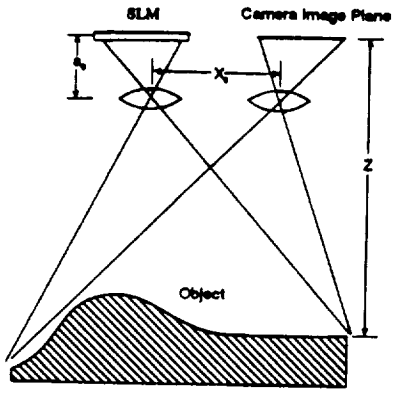


Figure E-2: For a suitable optical configuration the desired range information has a simple relationship to the phase function measured by the SURPHACE camera.

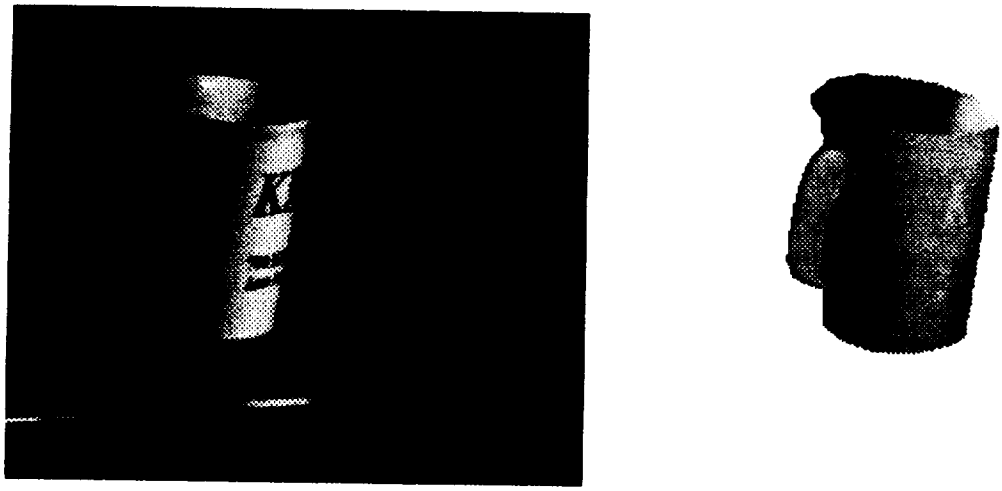


Figure E-3: SURPHACE obtains range by projecting three sinusoidal patterns that have been phase shifted from each other, such as the low frequency, single-cycle pattern shown in (a), and triangulates on the resulting distorted phase function to determine the range to the surface, as shown in (b) where the range has been measured with respect to a plane that is tilted with respect to the camera's viewing direction.

E-2.1.3 Design and Fabrication of the SURPHACE Projector

The projector for SURPHACE will consist of a high-intensity light projector with a micrometer-mounted slide holder that will permit accurate phase shifting. The design for the projector is nearly complete. Prior to the completion of the projector, KMS has been using a standard slide projector with reasonable success, although some errors are introduced due to the limited phase-shift accuracy in the current setup.

We have fabricated the slides by exposing computer-generated images of cosine pattern on an image recording device. However, due to the nonlinear nature of the transfer function from the 0-255 image pixel value to the slide film's transmission coefficient, the patterns produced in this way by recording a perfect computer-generated sinusoid are not perfectly sinusoidal, as seen in the plot of such a slide's transmission coefficient, obtained using a microdensitometer, shown in Figure E-4. We have overcome this problem by (1) compensating for the nonlinearities in the transfer function to produce more perfectly sinusoidal slides, and (2) extending the analysis

above to account for the presence of the higher-order harmonics in the exposed pattern. We measure the strength of the harmonics by producing the FFT of the microdensitomideter data and determining the relative strength of the peaks in the magnitude of the FFT.

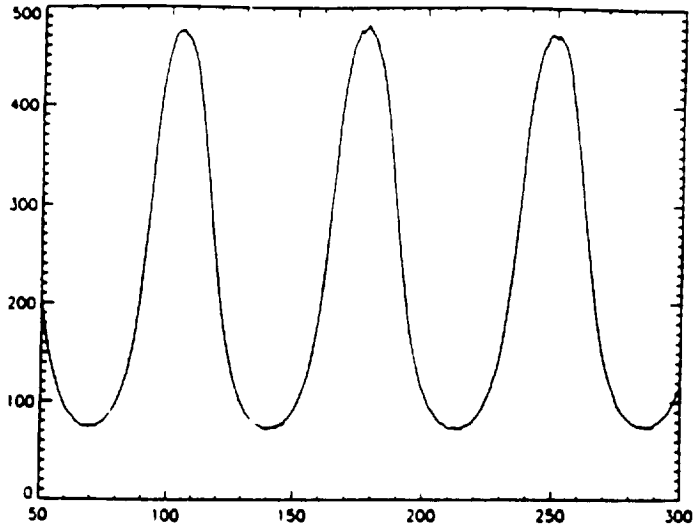


Figure E-4: **The microdensitometer measurement of a slide that was exposed with a computer-generated sinusoid, shown above, exhibits deviations from true a true sinusoid due to nonlinearities in the recording camera and the film.**

E-2.1.4 Calibration of the SURPHACE Apparatus

The accuracy of the SURPHACE technique depends, in large part, on how well the apparatus has been calibrated. Calibration consists of three parts (1) linearization of the camera's transfer function, (2) modeling of the camera's imaging geometry to enable correction of any non-ideal behavior, and (3) determination of the geometrical relationship between the camera and the projected pattern.

Aside from the assumption that the the light pattern being projected is sinusoidal, another assumption implicit in the SURPHACE approach is that the camera/digitizer is linear with respect to the incident light intensity. To the degree that these two assumptions hold, the SURPHACE method can obtain an accurate raw phase map. Recall that the raw phase map is the part of the projected sinusoidal function that implicitly contains the range data and is the result of separating the effect of the varying range to points in the scene from the effects caused by fluctuation in the reflectance of the surface and the intensity of the reflected light throughout the scene.

Linearizing the camera/digitizer transfer function can be accomplished by using calibrated neutral density filters to measure the response of the camera. KMS uses such measurements to linearize the response of the camera. Our experience in performing this procedure on several cameras has shown that CCD cameras are quite nonlinear, especially at low intensity levels, and must be corrected.

The camera/digitizer deviates from ideal behavior in other ways than being nonlinear. KMS will model these effects and measure them in calibration procedures so that they can be accounted for. Issues that have been examined include:

- Cameras often low-pass filter video signals to improve viewability but, in the process, create pixel ghosting effects in the horizontal direction.
- Distortion of pixel locations by imperfect lenses. The most prominent of these effects is radial lens distortion.[24]
- Variations in the sampling frequency and the horizontal retrace digitization startup timing leads to inaccuracies in the horizontal scale factor and the horizontal positioning of the image.
- Misalignment of the sensor array with the axis of the lens system, resulting in the middle pixel not being the true center of the optical system.

KMS is adapting a camera calibration method described in the literature[24] to create a calibrated model of the camera for VGD. KMS estimates that this approach will lead to an improvement in the accuracy in the SURPHACE approach by at least an order of magnitude.

Thus far, we have examined issues relating to the calibration of the camera and projector individually. The final calibration is to determine their geometrical relationship as a whole. Determining geometrical relationship amounts between the camera and the projected light pattern amounts to determining the transformation from the camera's coordinate frame to the projector's coordinate frame. This problem can be solved if the transformation between the camera and a plane in the scene is known. Knowledge of this transformation permits us to use a phase map of this known plane to determine the position of the projector.[25]

The position and orientation of the plane is determined by taking images of a known pattern of lines ruled on a plane and performing a least squares fit between the pairs of 3-d points that define the pattern and the 3-d plane that passes through the focal point of the camera and the imaged lines.

E-2.1.5 Surface Data Validity Assessment

While we have found that the vast majority of the data produced by the SURPHACE technique is valid, some of it of it may be poor or invalid. There are three sources of invalid data: (1) surfaces in the scene that the camera sees that are not illuminated by any light source; (2) very black regions that absorb so much light that there is not enough variation in the illumination between phase shifts to accurately determine the raw phase; and (3) specular regions that are so bright that the camera is saturated and, therefore, do not vary according to the model discussed in theory, yielding poor range data.

The three sources of invalid data listed in the previous paragraph are all characterized by very small variations in the pixel values between phase shifts: case one will have no variation, cases

two and three will have little variation. Therefore, one method for detecting this invalid data is to simply eliminate pixels that do not have sufficient variation among phase shifts. We believe that this simple approach will identify 95% or more of the invalid pixels.

E-2.1.6 Optimizing the Yield of Valid Data

While there is nothing we can do to improve the yield of valid range points from regions that have no illumination, we have the means to improve the yield of valid data from low reflectance regions and specular regions. Increasing the dynamic range of the camera by taking images at more than one f-stop is one way to accomplish this. As can be seen in Figure E-3, where the intensity and SURPHACE range images of a specular coffee cup are shown, the SURPHACE approach is not affected either by markings or specular points so long as the camera is not saturated and has the sensitivity to detect variations in the phase. Therefore, KMS is investigating taking phase-shifted images at more than one f-stop and merging the valid range pixels. This will maximize the yield of valid low-reflectance pixels, obtained when the camera's stop is set the most open, and maximize the yield of valid specular pixels, obtained when the camera's stop is the least open.

E-2.2 Viewpoint Determination

One view of an object does not provide sufficient information to produce a complete model of all surfaces of an objects. Thus, the object's surface must be captured from several different views, and the data taken at each view must be merged. To properly merge the data, data at different views must be *registered*. Research in this area includes the work of Potmesil[26], Dane[27], and Henderson.[28,29] In Potmesil's work, surfaces were registered by "sliding" overlapping portions of the surfaces onto top of one another and determining an optimal fit. This approach, however, does not work well since surfaces can match at many different positions. With Dane and Henderson's approaches the relation between views must be explicitly defined. To provide maximum flexibility, VGD allows surface data obtained from any viewpoint to be assimilated into the overall surface model.

Permitting surface data acquired from any view requires that the relative transformation between the viewpoints be determined precisely. We accomplish this by using *landmark features*, i.e., features that can be easily identified in the surface data from several viewpoints. We are currently investigating ways to automatically select and detect the landmarks. Currently, landmarks are hand-selected.

We describe the location of such landmark features by position vectors that obtained from the surface data provided by the SURPHACE sensor. If the position of a landmark feature i in viewpoint one was represented by the vector p_i, and the position of a corresponding feature in a second viewpoint was represented by vector r_i, then the two positions would be related by a rotation, R, and translation, t. These are the quantities that must be determined in order to determine specify the geometrical relationship between viewpoints. Mathematically, the relationship is

$$r_i = R\, p_i + t.$$

Now, if differences are taken between points p_i and p_j and the corresponding points r_i and r_j to produce difference vectors, p_{ij} and r_{ij}, then,

$$r_{ij} = R\, p_{ij}.$$

By representing the rotation operator as a quaternion, a simple solution can be obtained for determining the rotation between the viewpoints. Specifically in a quaternion representation, the above relation could be written as,

$$r_{ij} = q_R \bullet p_{ij} \bullet q_R^*, \quad \text{subject to } q_R \bullet q_R^* = 1,$$

where "•" indicates quaternion multiplication, and where the quaternion, q_R, and its conjugate, q_R^*, contains four variables that specify the axis and angle of rotation.

When more than three points, p_i, are used, the problem is overspecified and can be solved with least squares methods,

$$\min q_R^t\, B\, q_R, \quad \text{subject to } q_R^t q_R = 1,$$

where q_R now represents a 4x1 matrix of quaternion coefficients and B is a 4x4 matrix with terms involving the difference vectors, p_{ij} and r_{ij}. This problem is an eigenvalue problem and its solution is the smallest eigenvalue of the matrix B, normalized to 1. Once the quaternion rotation, q_R, is found the rotation matrix R can be determined from q_R. In addition, once R is determined, then t can be found by backsubstitution.

E-2.3 Surface Merging

As described above, in VGD the surfaces of an object captured from a number of views. The views are selected so that the surfaces, reconstructed from the SURPHACE camera data, overlap. Because the surfaces are registered, the geometric relationship between surfaces can be established, thus allowing the surfaces to be correctly positioned with respect to one another.

The geometric surface data captured by the SURPHACE camera is in the form of a range image. While the range image provides the distance of the captured image from the SURPHACE camera, it is not an efficient format for storing a geometric representation of an object. To provide a representation the range data will be approximated by a polyhedral surface. For example, Boissannat and Faugeras[30] have developed an algorithm for the polyhedral approximation of a surface from range data.

The photometric surface data captured by the SURPHACE camera is in the form of an intensity image. Because the intensity image represents a projection of the surface detail of the object onto the SURPHACE camera image plane, to obtain the true surface detail, the intensity image must be backprojected onto the object surface. Several issues must be addressed in

backprojecting. Since the surface is approximated by a polyhedral surface, then backprojecting is simply a problem of determining how to map the intensity data onto the facets of the polyhedral surface.

The simplest solution is to project a polygon facet of the surface into the intensity image to determine the polygonal area that corresponds to projected facet and to warp the intensity data with the projected polygon (using a standard image warping algorithm) so that it maps onto the surface facet. One problem with this approach are that a surface facet may have a high angle of tilt with respect to the intensity image plane, and, therefore, the backprojected data will be rather poorly mapped onto the surface. Since the surface will be recorded from multiple views, this problem can be reduced by backprojected surface data from a view which has the smallest tilt angle with respect to the facet to be mapped. Another solution would be to used data from a number of views and to determine the most consistent surface detail, given the data from all views.

If the surfaces are approximated by polyhedral surfaces, a combined surface can be constructed by merging the polyhedral surfaces. However, problems with this approach are (1) the overlapping areas of two surfaces will not necessarily be approximated by the same polyhedral approximation, and (2) surfaces that overlap may not have the same surface detail.

One solution to these problems would be to determine an optimum "seam" at which to merge the polyhedral surfaces and trimming the excess surface data beyond the seam. This may require creating new surface facets along the seam and remapping the surface detail onto these new facets.

E-2.4 3-d Object Tracking Using Surface Data

KMS has developed an innovative object recognition algorithm that is based on a newly discovered approach to object pose determination.[31] One of the unique aspects of the KMS recognition algorithm, called Recognition by Iterative Spring Energy Reduction (RISER), is that it applies to range imagery and intensity imagery equally well. To provide a demonstration of the practical application of the range data produced by VGD's the SURPHACE camera, as well as the 3-d models produced by VGD, KMS will test RISER on VGD range images and 3-d models.

In the RISER algorithms, 3-d model data representing objects that may appear in a scene, is matched to the range image of the scene. Efficiently matching models of objects to range data is equivalent to efficiently recognizing and tracking objects. Although range images, unlike intensity images, directly provides the geometry of the scene, the problem of identifying objects is nevertheless quite difficult.

Specifically, a number of problems must be solved. For example, the algorithm must be able

- To work with noisy, incomplete sensory data; parts of objects are often occluded, distorted, or out of the field of view.

- To identify object descriptors that are both selective, i.e., they are not too common among the objects to be recognized, and significant, i.e., the probability that they appear at random in the scene is small.
- To handle translation, rotation and scaling of the object to be tracked or identified.
- To efficiently compare surfaces of models with the surfaces of range data, without an early commitment to a match.
- To form matches based on a consensus of a large amount of data rather than relying on cues provided by a few, likely error-prone, features.
- To robustly converge to the correct match between model and range data.

During the development of its object recognition algorithms, it has relied on many insights gained from developing algorithms for 2-D and 3-D object recognition systems.[33],[31] In particular, the following techniques that were developed for the STORS[32] algorithm, which recognizes 3-D objects in 2-D images, are equally valid for identifying objects in range data.

To work with incomplete data, the algorithm must use local data. In the STORS algorithm the contours of an object were represented as a collection of overlapping segments, which formed local neighbors of data. With this representation, large sections of the contours could be missing or distorted without affecting the operation of the algorithm. Analogously, in the RISER algorithm, both the model and range data are represented as a collection of overlapping surface patches. We desire to keep the patches as small as possible to lower the probability of the patch being occluded.

To construct unique descriptors, the algorithm uses widely spaced sets of spatially local patches. Because each patch's data is defined over a small neighborhood, the patches has similar attributes, especially on smooth objects. In the RISER algorithm, descriptors consist of pairs of surface patches. The patches in descriptor are chosen from different regions of the surface contour and are not necessarily spatially close or overlapping. Descriptors of this type have a number of geometrical attributes, such as the angles between the normals at the center of each patch, the principle curvatures at each patch, etc., that allow them to be differentiated from each other.

The descriptors must be made invariant to translation, rotation, and, if absolute range data is not available, to scale. We may accomplish this if by choosing the attributes of the descriptors properly. In the RISER algorithm the attributes of a descriptor are chosen relative to a local coordinate frame defined by the two surface patches. This makes the descriptors invariant to translation and rotation. We can make the descriptors invariant to scale by fixing the ratio of the radius, r, that defines the local surface patch neighborhood, to the distance between patches, d. If absolute range data, such as that provided by VGD's SURPHACE camera is available, the descriptors do not need to be scale invariant.

To efficiently match surfaces and yet avoid early "lock in" on a possibly erroneous match, the algorithm uses an approach of gradual commitment. As in the STORS algorithm the

descriptors of the model are, at first, compared to all of the descriptors of the range data. To speed up this initial comparison, the descriptors of the range data is stored (as in the STORS algorithm) in a kd-tree,[34] indexed by the attributes of the descriptor. If n is the number of range descriptors, retrieving all descriptors having a fixed range of attributes is O(log n). If m model descriptors are compared to the n range image descriptors, the total complexity of comparison is O(m log n). This is far more efficient than correspondence approaches[35,36] which have high order polynomial or even factorial order runtime complexity.

To obtain a consensus of the data, the RISER sets up a set of pseudo forces between similar model and range descriptors. In effect, this is like connecting a 6-dimensional spring (3 translational dimensions and 3 rotational dimensions) between the descriptors, with a spring constant proportional to the descriptor's similarity. The spring attempts to pull corresponding the model and range descriptors into alignment.

After pseudo springs have been connected to between the model and image descriptors, the algorithm minimizes the sum of weighted pseudo spring potential energies. This is equivalent to allowing the spring forces to reposition the model to lower the total potential energy. To obtain a robust convergence to a correct solution, i.e., a correct match between the model and range data, the RISER employs the robust statistical approach of reweighting the spring constants. This is equivalent to checking the pseudo springs after at the pseudo energy has been minimized and cutting or weakening those springs that are overstretched.

Robust statistics is a discipline that has found considerable use in eliminating the effect of bad data in performing a fit of a parametric model to data. For example, in estimating the pose of an object, an residual error metric ε on the parameters of the object's model is minimized. This metric is the sum of the norm $\rho(\bullet)$ of the errors between the poses of the different descriptors. The poses of the descriptors is defined by a six-dimensional vector Π = (q, p), where q is the quaternion vector that describes the orientation of a primitive and p is a spatial vector that describes the 3-space position of a descriptor. If the norm $\rho(\bullet)$ is, for example, the L_2 norm, then the error measure would be proportional to the square of differences of the six-dimensional Π vector of the model and range descriptors. An alternate L_1 norm would measure the absolute value in the differences in the Π vectors.

While the L_2 norm often obtains a better fit, problems with this type of norm are that when the error is very large for a given data point (i.e., a statistical "outlier"), the L_2 norm of the residual makes a hugh contribution to the sum ε, which in turn has a large effect on the view parameter estimate that the comparable effect using, for example, the L_1 norm. A measure of the influence of outliers is the ψ function, which in one-dimension is defined by $\psi(x) = d\rho(x)/dx$. (The ψ functions for the 1-dimensional L_1 and L_2 norms are shown in Figure E-5 (a) and (b).) As can be seen, the L_1 fit is "less sensitive" to to data sets contaminated with statistical outliers than the L_2 fit because the "influence" of these outliers as measured by ψ is constant rather than linearly increasing.

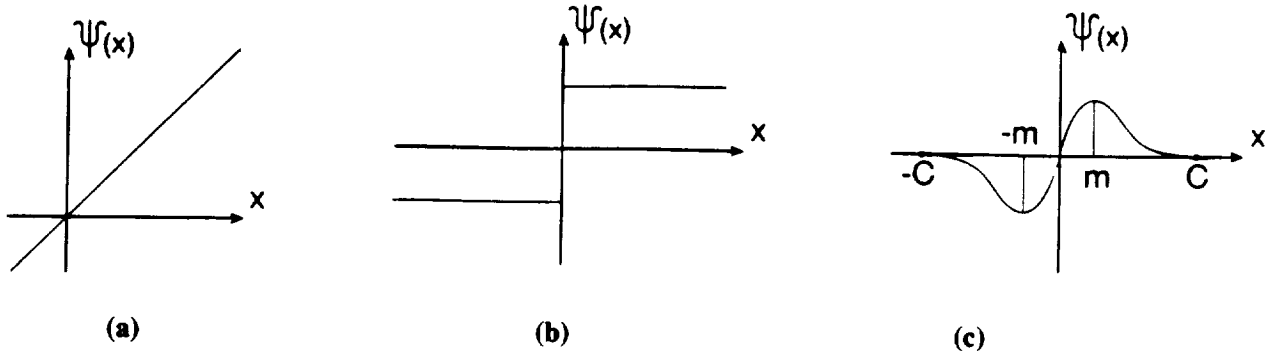


Figure E-5: **Above, (a) shows the ψ function of the L_2 metric, (b) shows the ψ function of the L_1 metric, and (c) shows a redescending ψ function with maximum points at m and -m, and cutoff c.**

It can be even more desirable to utilize so-called redescending ψ-functions,[37] such as the one in shown in Figure E-5 (c). Empirical studies show that the detailed shape of the ψ-function is not very important. The most critical features are the maximum ψ points, which measures gross error sensitivity, and the cutoff value, c, known as the finite rejection point.

The RISER object recognition and tracking algorithm uses a redescending ψ function to measure the degree of match between the model surface and the range surface. After the pseudo-energy is minimized at each iteration, the cutoff value is adjusted to a smaller value, and, essentially cut the contributions from matching descriptors that correspond to statistical outliers. This, as mentioned, is equivalent to cutting overstretched pseudo springs that tie the descriptors together.

In this way, although a consensus is used initially in determining a fit of the model to range data, the final fit will be least affected by bad data.

1. D. Marr and T. Poggio, "Cooperative Computation of Stereo Disparity," Science, 194, 283 (1976).
2. Y. Yakimovsky and R. Cunningham, "A System for Extracting 3D Measurements from a Stereo Pair of TV Cameras," Computer Graphics Image Processing, 7, 195 (1978).
3. Y. Ohta, M. Watanabe, and K. Ikeda, "Improoving Depth Map by Right-Angled Trinocular Stereo," Proc. Intl. Conf. Patt. Rec., pp. 519-521, (1986).
4. H. Baker and R. Bolles, "Epipolar-plane Image Analysis: A Technique for Analysing Motion Sequences," Proc. IEEE Workshop on CV, pp. 168-178, (1985).
5. R. Jain, S. L. Bartlett, and N. Obrien, "Motion Stereo Using Ego-motion Complex Logarithmic Mapping," IEEE Trans PAMI, pp. 356-369 (1987).
6. D. Terzopoulos, "Regularizations of Inverse Visual Problems Involving Discontinuities," IEEE Trans. PAMI, 8, pp. 413-424 (1986).
7. R. Ahola, T. Heikkinen, and M. Manninen, "3D Image Acquisition by Scanning Time of Flight Measurements," Proc. Intern. Conf. on Advances in Image Process. and Patt. Recog. (1985)
8. J. A. Ross, "Methods and Systems for 3D Measurement," U.S. Patent 4199253 (1978).
9. R. Terras, "Detection of Phase in Modulated Optical Signals Subject to Ideal Rayleight Fading," J. Opt. Soc. America, 3, 1816 (1986).

10. T. M. Quist, W. E. Bicknell, D. A. Bates, "ARPA Semi-annual Report: Optics Research," Lincoln Laboratory, MIT (1970).
11. J. R. T. Lewis, and T. Sopwith, "3D Surface Measurement by Microcomputer," Image and Vision Computing 4, 159 (1986).
12. J. L. Mundy and G. B. Porter, "A Three-dimensional Sensor Based on Structured Light," Three-Dimensional Machine Vision (T. Kanade, Ed.) Kluwer Academic, 3 (1987).
13. M. Potmesil, "Generating 3D Surface Models of Solid Objects from Multiple Projections," Ph.D Dissertation, Renssalear Polytechnic Institute, Troy, New York.
14. G. Stockman and G. Hu, "Sensing 3D Surface Patches Using a Projected Grid," Proc. Comp. Vision Patt. Recogn. Conf, 602 (1986).
15. G. Kinoshita, M. Idesawa, and S. Naomi, "Robotic Range Sensor with Projection of Bright Ring Pattern," Jour. Robotic Systems, 3, 249 (1987).
16. K. R. Pelowski, "3D Measurement with Machine Vision," Proc Vision '86 Conf., 17 (1986).
17. M. Asada, H. Ichikawa, and S. Tsuji, "Determining Surface Property by Projecting a Stripe Pattern," Proc. Inter. Conf. on Patt. Recog. 1162, (1986).
18. A. Rosenfeld and C. J. Tsikow, "High-speed Space Encoding Projector for 3D Imaging," Proc. of SPIE Conf. on Optics, Illumination, and Image Sensing for Machine Vision, 728, 146 (1986).
19. K. L. Boyer and A. C. Kak, "Color Encoded Structured Light For Rapid Active Ranging," IEEE Trans. Patt. Anal. Mach. Intell., 9, 14 (1987).
20. H. Schewe, and W. Forstner, "The Program PALM for Automatic Line and Surface Measurement Using Image Matching Techniques," Proc. Symp. Intern. Soc. for Photo. and Remote Sensing, 26, 608 (1986).
21. W. W. Macy, "Two-dimensional Fringe Pattern Analysis," Appl. Opt. 22, 3898 (1983).
22. M. Idesawa and Y. Yatagai, and T. Soma, "Scanning Moiré Method and Automatic Measurement of 3D Shapes," Appl. Opt. 16, 2152 (1977).
23. M. Halioua and V. Srinivasan, "Method and Apparatus for Surface Profilometry," U.S. Patent 4641972.
24. R. K. Lenz and R. Y. Tsai, "Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3-D Machine Vision Metrology," IEEE Tr. PAMI, 5, pp. 713 (1988).
25. P. G. Gottschalk, Research Notes (1990).
26. M. Potmesil, "Generating Models of Solid Ojbects by Matching 3D Surface Segments," Proc. of 8th Intern. Joint Conf. on Artif. Intell., 1089 (1983).
27. C. Dane and R. Bajcsy, "An Object-centered Three-dimensional Model Builder," Proc. of the 6th Intern. Conf. on Pattern Recognition, 348 (1982).
28. T. C. Henderson, "Efficient Segmentation Method for Range Data," SPIE Conf. on Robot Vision, 336, 46 (1982).
29. T. C. Henderson, "Efficient 3D Object Representations for Industrial Vision Systems," IEEE Trans. Patt. Anal. Mach. Intell., 5, 609 (1983).
30. J. D. Boissonnat and O. Faugeras, "Triangulation of 3-D Objects," Proc. 7th IJCAI, 658 (1981).
31. P. G. Gottschalk, "Machine Recognition and Attitude Estimation of 3d Objects in Intensity Images," Univ. Michigan Ph.D. Thesis (1990).

32. J. G. Downward, J. L. Turney, P. G. Gottschalk, and T. B. Ladewski, "Single-view 3d Object Recognition System", KMSF-U2280 (1989).
33. J. L. Turney, "Recognition of Partially Occluded Parts," Ph.D. Dissertation, Univ. of Michigan (1986).
34. P. G. Gottschalk, J.L. Turney, and T.N. Mudge, "Efficient Recognition of Partially Visible Objects Using a Logarithmic Complexity Matching Technique," International Journal of Robotics Research, 8, pp. 110-131 (1989).
35. W. E. L. Grimson and T. Lozano-Perez, "Localizing Overlapping Parts by Searching the Interpretation Tree," IEEE Tr. PAMI, pp. 469-482 (1987).
36. D. P. Huttenlocher, "3d Recognition of Solid Objects from a 2d Image," MIT Ph.D Thesis (1988)
37. F. R. Hampel, E. M. Ronchetti, and P. J. Rousseeuw, "The Change of Variance Curve and Optimal Redescending M-Estimators," J. Am. Stat. Assoc., 76, pp. 643-648 (1981).

Section F
Quarterly Report #4: October 1990

F-1. Summary of Progress

VGD has progressed in two important directions since the last report. The first is in overcoming of the major engineering obstacles in the path of developing of the SURPHACE camera, the full view, dense field range sensor described in the previous report. The second is the development of the user interface software. Much was accomplished in spite of the fact that relatively little effort was expended during the quarter.

We reported previously that the SURPHACE camera is based on a new method for rapidly acquiring surface data by projecting and phase-shifting sinusoidally varying intensity patterns. The patterns are produced by simply phase-shifting a slide with a sinusoidally varying transmission coefficient in a projector, as shown in Figure F-1.1. The existing theory for this technique relies on the pattern being close to perfectly sinusoidal. This, in turn, requires that the spatial variation of the transmission coefficient of the slides be accurately sinusoidal.

As of the last report, we had fabricated slides that had roughly 20% harmonic distortion. Harmonic distortion of this magnitude introduces a periodic error into the range images that is unacceptably high. We reported previously that coping with this problem was the primary engineering challenge we faced in the development of the SURPHACE camera. We also reported that we would attack the problem using a two-pronged strategy. First, we would continue to try to improve the quality of the slides. Second, we would attempt to extend the analysis to allow the use of non-sinusoidal periodic functions. KMS is glad to report that we have succeeded on both fronts. We describe the details of this work in Section F-2.1.

In addition to work on the range sensor, we have completed the software that will form the foundation of VGD's user interface. This is important because the calibration subsystem, the viewpoint determination subsystem, and the surface merging subsystem all require varying degrees of operator input. Implementing the basic operator interface was necessary in order to proceed with the implementation of other aspects of VGD. We describe this in greater detail in Section F-2.2.

Little effort was expended on VGD's other subsystems, i.e., the calibration subsystem, the viewpoint determination subsystem, and the surface merging subsystem. There were two reasons for this. First, the key personnel were heavily involved in several proposal writing efforts that greatly reduced the time they could spend on tasks related to VGD. Second, the user interface fell on the critical path of the development of these systems. Therefore, we postponed work on these tasks until the user interface was complete enough to permit work to proceed. We wish to stress that, considering the relatively small number of hours spent on VGD, much was accomplished. Moreover, we expect that work will continue at a normal pace for the remainder of the project.

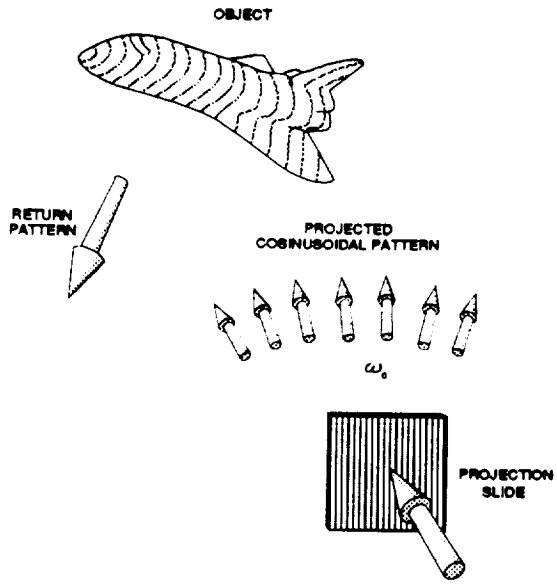


Figure F-1.1: SURPHACE obtains range by projecting three sinusoidal patterns that have been phase shifted from each other triangulates on the resulting distorted phase function to determine the range to the surface.

F-2. Technical Status

F-2.1 The SURPHACE Camera

We previously identified the goals of improving quality of the sinusoidal gratings on slides and extending the phase-shift analysis to non-sinusoidal functions as the key challenges on the way to developing the SURPHACE Camera. Solving either of these problems will make the SURPHACE camera a viable sensor for VGD. We have succeeded in both endeavors. While either of these techniques would suffice for the purposes of VGD, we expect that using them in combination will permit us to use the SURPHACE camera for high precision metrology and inpection tasks as well.

F-2.1.1 Fabrication of Accurately Sinusoidal Transmission Gratings.

The presence of harmonic distortion in the spatial tranmission function of a slide results in a correspondingly distorted projected light pattern. The measurements of the scene with the projected pattern, in the form of a set of phase-shifted images, are then also distorted. In turn, this leads to distortion of the raw phase function and the range map itself. For degrees of harmonic distortion less than about 20%, the resulting distortion of the raw phase function is roughly proportional to degree of harmonic distortion in the intensity of the projected pattern. The corresponding distortion induced in the range map is dependent on the geometry of the imaged surfaces and the imaging apparatus. However, under conditions of interest, the induced errors can equal the level of harmonic distortion.

In order to produce useful range maps using the current technique, the harmonic distortion of the transmission grating should be kept below 5%, and, if possible, smaller still. Figure F-2.1 shows a scan of the transmissivity of the best grating we were able to produce as of the last quarterly report. Figure F-2.2 shows a plot of the absolute magnitude of the discrete Fourier transform of this data. If we estimate the Fourier coefficient by measuring the area under the peak down to a

point that is half as large as the peak, then Figure F-2.2 indicates that the grating of Figure F-2.1 has distortion that is dominated by the second harmonic, and that the magnitude of the distortion is approximately 18%.

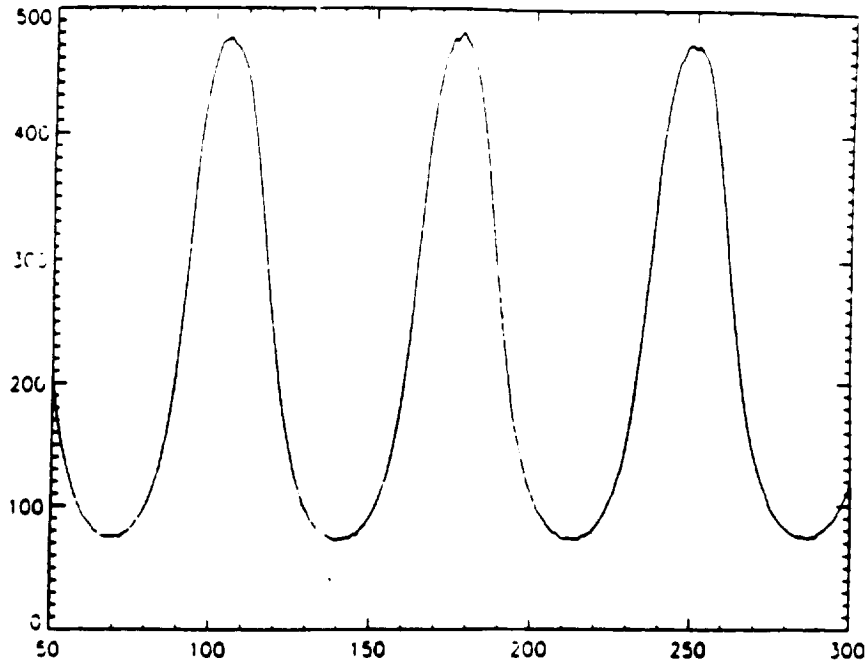


Figure F-2.1: The spatial variation of the transmission coeffient of the best, i.e., least harmonically distorted, slide that we had produced as of the last quarterly report. The transmission coeffiecient times 100 is plotted on the vertical axis versus distance on the horizontal axis. The entire scan covers roughly 3 cm.

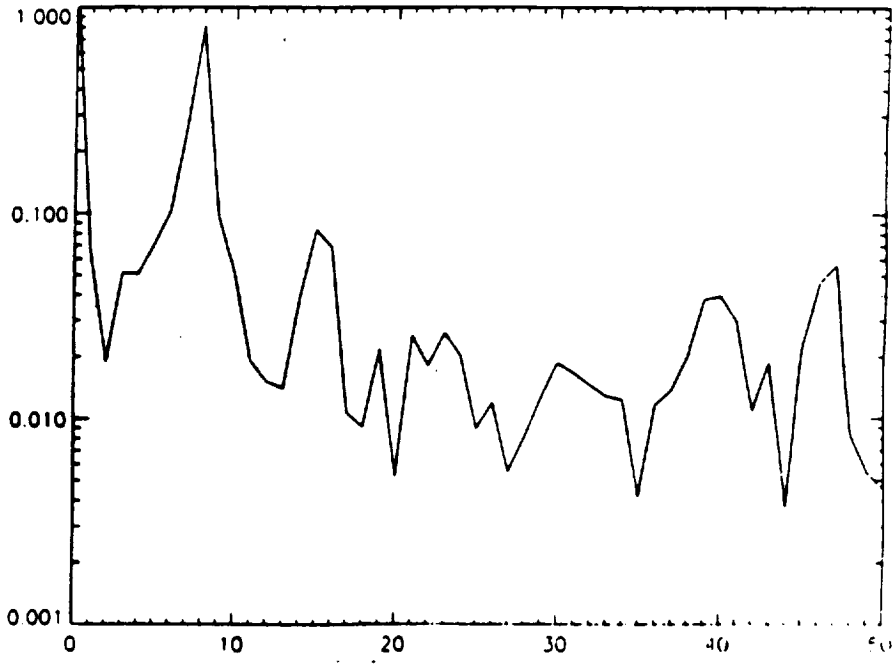


Figure F-2.2: Plotted is the magnitude of the discrete Fourier transform of the data plotted in Figure F-2.1. Measuring the area under the peaks of the fundamental and the second harmonic provides an estimate of the second harmonic distortion. This slide contains 18% second harmonic distortion.

Figure F-2.3 and F-2.4 are plots of the measured transmission coefficients of a low frequency grating and a high frequency grating. These are our best gratings to date. Figure F-2.5 shows the magnitude of the discrete Fourier transform of the data plotted in Figure F-2.4. From Figure F-2.5 we can see that the second harmonic is nearly completely absent, and third harmonic distortion, using the same estimation procedure as above, is 1.5%, which should be acceptable for the purposes of VGD.

F-2.1.2 Computing the Raw Phase from Non-Sinusoidal functions

The other approach to solving the problem of nonsinusoidal transmissivity is to extend the phase-shift analysis to the case of nonsinusoidal periodic functions. In the original phase-shift analysis, the intensity function is modelled as a biased pure sinusoid with phase distortion arising only from the geometry of the surfaces of the scene, i.e.,

$$I(x,y) = I_A(x,y) + I_B(x,y)\cos(\omega_0 x + \Phi(x,y) + \phi_i),$$

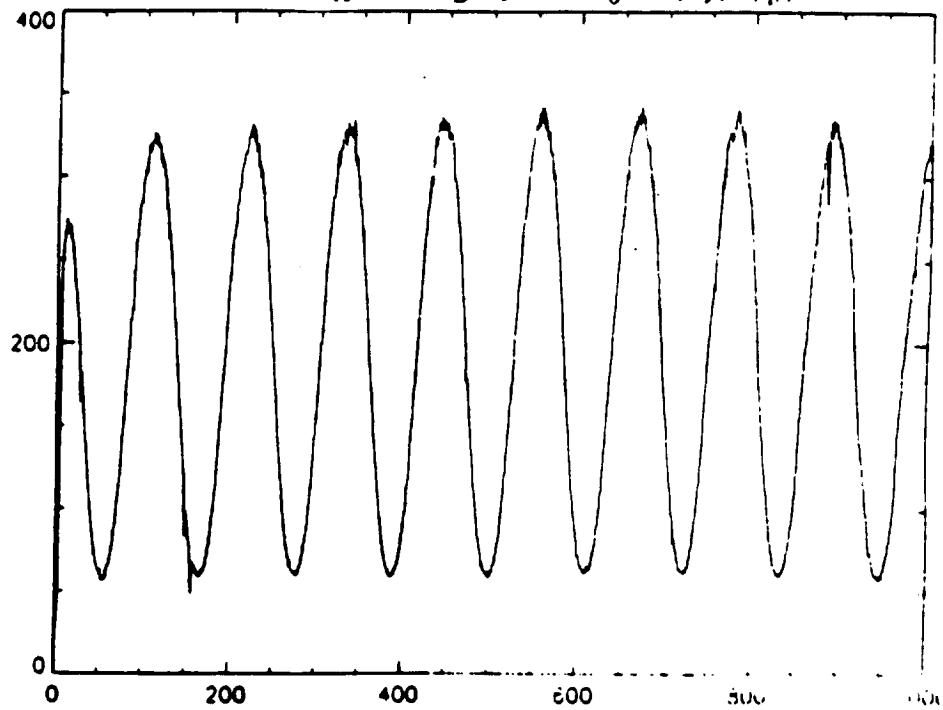


Figure F-2.3: **This is a plot of the transmissivity of a good low frequency grating. The interpretation of the plot is analogous to Figure F-2.1.**

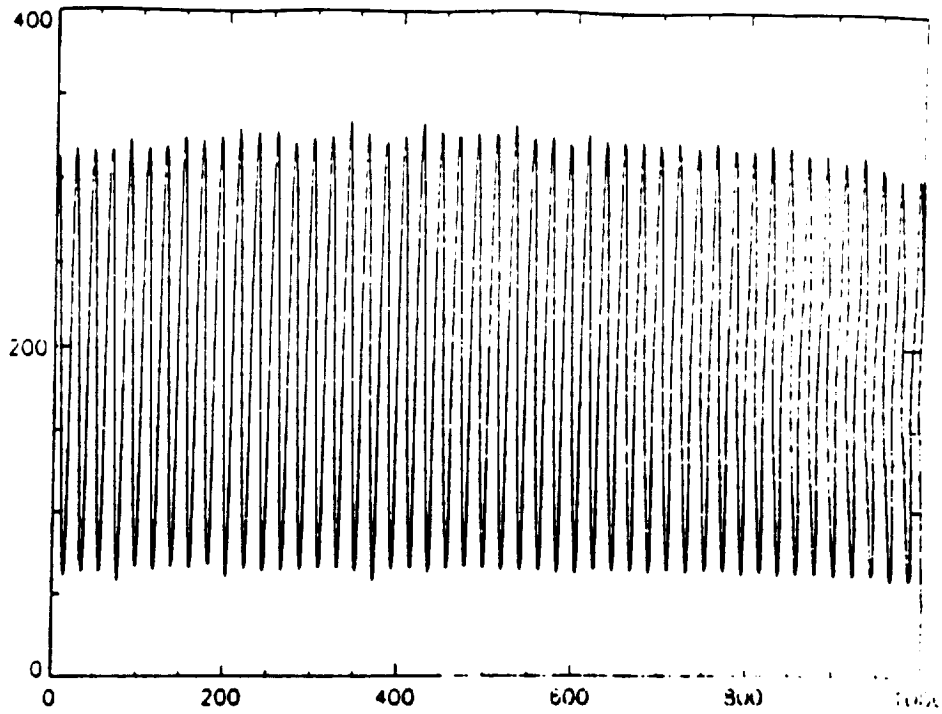


Figure F-2.4: This is a plot of the transmissivity of a good high frequency grating.

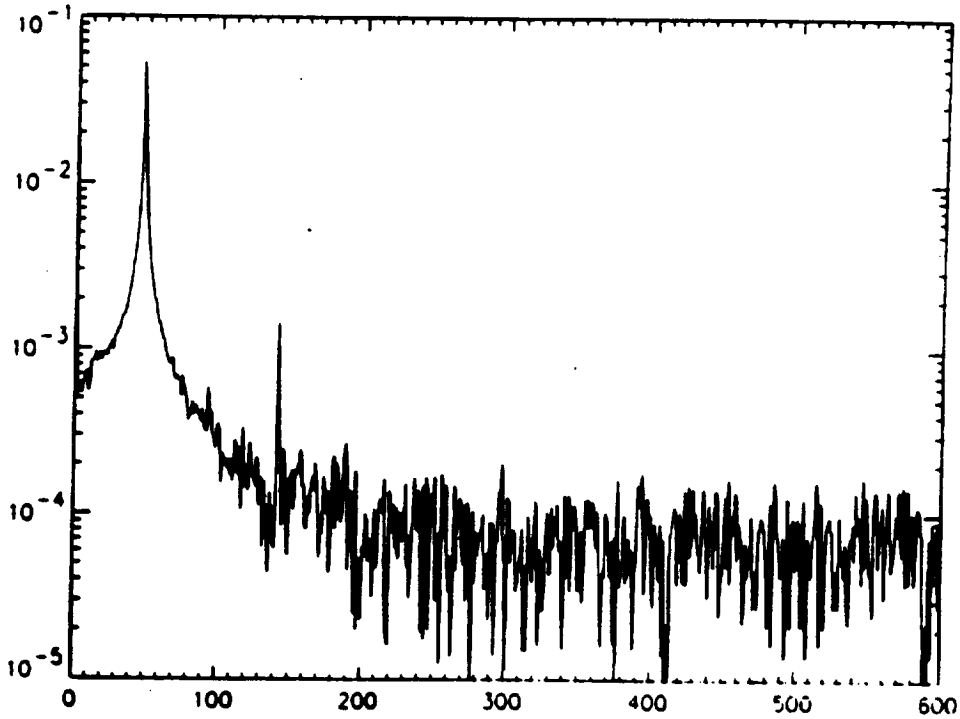


Figure F-2.5: This is a plot of the absolute magnitude of the discrete Fourier Transform of the transmissivity in Figure F-2.4. This plot makes evident the high quality of the grating. Specifically, the second harmonic is negligible while there is only 1.5% third harmonic distortion.

where $I_A(x,y)$ represents variations in the intensity due to background, or ambient lighting, and $I_B(x,y)$ represents the variation in contrast of the cosine pattern due to the photometric properties

of the objects imaged. The term ϕ_i represents a phase shift that will be introduced into the pattern by moving the slide in the projector. In this case, by judicious use of various trigonometric identities, we can show that the raw phase function, $\Phi(x,y)$, can be recovered through the simple formula

$$\Phi(x,y) = \tan^{-1}((2I_{\pi/2} - I_0 - I_\pi)/(I_\pi - I_0)), \quad (1)$$

where the index on I indicates the angle of the phase shift in degrees. For speed, the inverse tangent can be stored in lookup table.

If the grating is periodic, but not necessarily sinusoidal, the intensity function must be modelled by

(2)

where the a_i are the Fourier coefficients of the function, and n is the number of terms being kept in the Fourier expansion of the function. We are assuming that the Fourier coefficents are known, since we can directly measure the transmissivity of the grating and perform Fourier analysis to obtain the coefficients. If we let

$$Q(x,y) = (2I_{\pi/2} - I_0 - I_\pi)/(I_\pi - I_0) \quad (3)$$

and insert Eq. (2) into Eq. (3), we obtain the modelled intensity quotient

$$Q(x,y) = f(\Phi(x,y)), \quad (4)$$

where $f(\Phi(x,y))$ is

(5)

However, we can measure Q directly from the phase-shifted images by using Eq. (3). Denote the measured value for Q as Q_m. Then, Eq. 4 can be written as

$$f^{-1}(Q_m) = \Phi(x,y). \quad (6)$$

It is not possible to obtain an analytic formula for f^{-1} However, we can use Newton's method to solve for $\Phi(x,y)$ as follows. We know that in the case where there is only one term in the expansion in Eq. (2), f^{-1} reduces to the inverse tangent function. Using this as a starting point, we can iteratively run Newton's method, gradually increasing the magnitude of the coefficients of the harmonics in the expansion until they attain their measured values. If the coefficients are increased slowly enough, the solution will always be the physically meaningful one. In this way, we can compute a table of values for f^{-1}. This needs to be done only once for each grating. The table can then be used over and over to quickly obtain the raw phase $\Phi(x,y)$ from the measured value of Q using that grating.

Figure F-2.6 shows a one-dimensional synthetic intensity pattern. This pattern has harmonic distortion of 20%. To within isolated discontinuities of 2π, the phase of this pattern should be perfectly linear. Figure F-2.7 shows the phase computed using the technique we have just described. Indeed, the raw phase plot in the figure is perfectly linear save for isolated jumps of 2π.

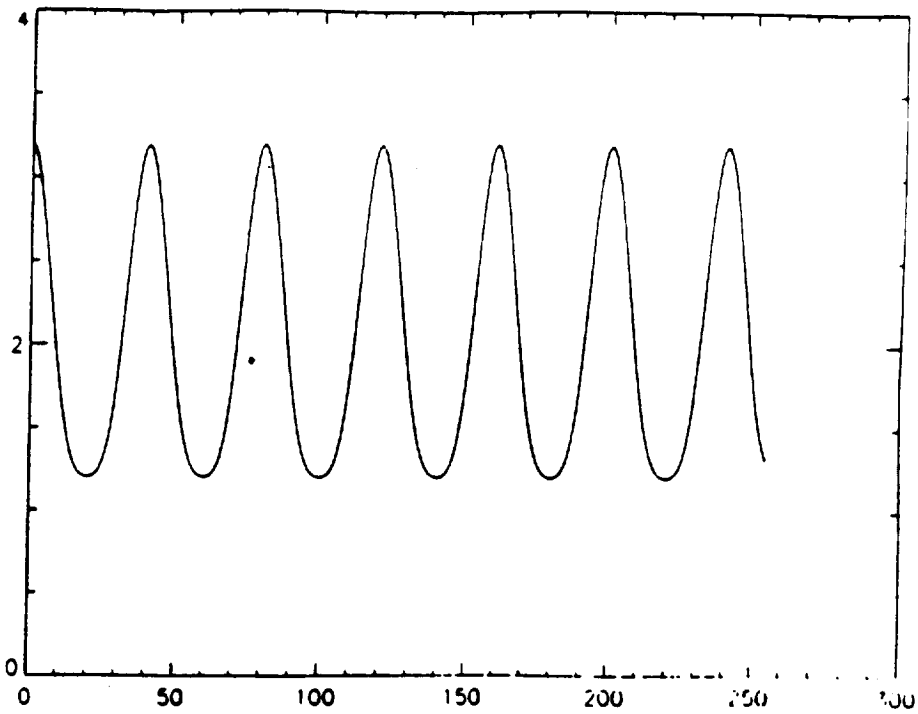


Figure F-2.6: Synthetically generated intensity profile with 20% second harmonic distortion.

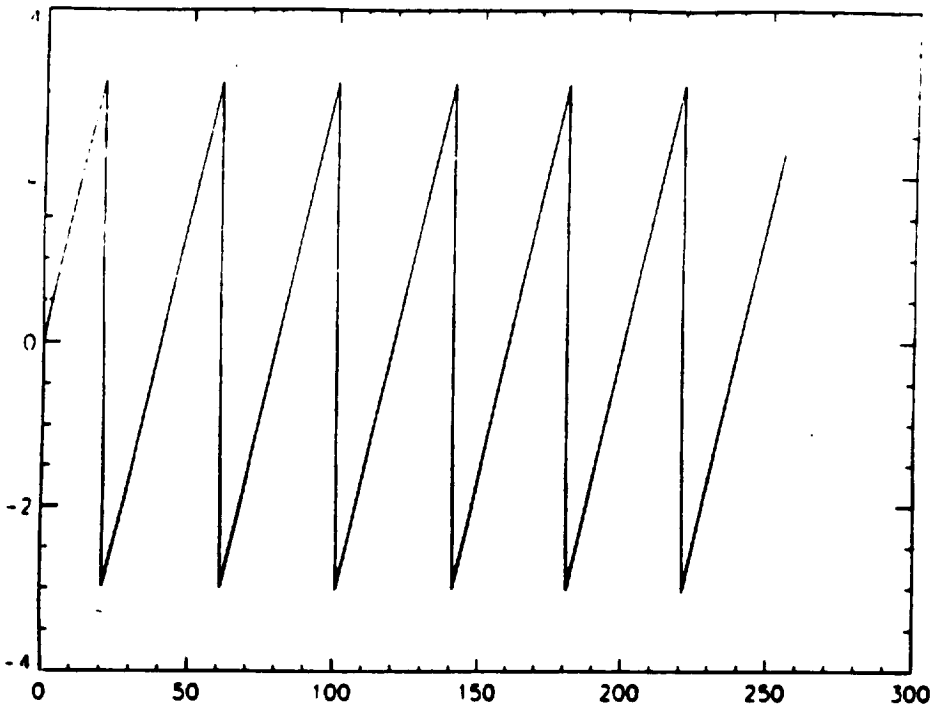


Figure F-2.7: Raw phase of the intensity profile computed using the technique described in the text for extracting the phase of a nonsinusoidal function. As expected, the phase of the pattern is linear except for jumps of 2π.

As we mentioned previously, either perfecting the slides or extending the analysis would have cleared the way for us to use the SURPHACE camera in VGD. Since, in fact, we have succeeded

in both endeavors, extremely accurate measurements should be possible with the SURPHACE technique by correcting for the small harmonic distortion present in even the best grating pattern.

F-2.2 User Interface Library

Many of the functions of VGD require the user to react to data that the system has gathered. For example, to calibrate the cameras, the user will take an image of a calibration card and the system will ask him to point to specific lines on the card. Similarly, the viewpoint determination subsystem requires user input, as does the surface merging subsystem. However, each of the subsystems has different requirements. Therefore, a general library of user interface routines for creating windows, displaying images, graphics and text, etc. has been developed in this quarter. The routines are built around the X windows standard, and are therefore very portable.

Section G
Quarterly Report #5: October-December 1990

At the end of 1990, several large scale computer reconfigurations were forced on KMS and apparently the original word processing sources and graphics used in this quarterly report could not be found in computer readable format. Consequently, this quarterly report has been incorporated without editing into the final report on the pages that follow this one.

1. Overview of the VGD Project

The primary goal of the VGD project is to develop a system that can capture computer models of real-world objects as 3-D textured-mapped surfaces. The VGD system will achieve this goal by (1) providing a range camera that can input registered range and texture data, (2) developing software modules that can merge the camera data into one coherent surface, and (3) developing software data structures and modules for efficiently displaying surface data.

One of JPL's major objectives in supporting the VGD system is to develop computer models that can be used in tracking 3-D objects. To address this objective, the models, provided by the VGD system, will be stored in a format that will allow the models to be quickly manipulated. This would allow, for example, a VGD model to be used in a closed loop estimator that, in turn, could be used to track the motion of a 3-D object in real-time [Gottschalk et al., 1989].

To build up models of objects, the VGD system will:

a) Capture registered range and texture surface data from a sufficient number of views to cover the entire surface of an object.

b) Determine the transformations between the views at which surface data is captured.

c) Merge the surface data into one coherent surface that is stored in a format that allows the data to be quickly manipulated.

Work performed during the last quarter has been directed toward developing the SURPHACE sensor (i.e., a *SU*rface *R*econstruction by *PHA*se-shifted *C*osin*E* sensor), described in the last quarterly. The SURPHACE sensor consists of (1) a structured lighting unit, that projects a sequence of cosine patterns of different phases onto the surface of an object of interest, and (2) two CCD cameras, that input and digitize the cosine patterns. Changing the phase of the cosine pattern by a number of known phase steps allows the range and surface albedo, i.e., texture, of a surface to be isolated from the background lighting and captured at each pixel. Moreover, using cosine patterns of different spatial frequencies (See Section 2.2) allows the SURPHACE sensor to extract the absolute range of surface points.

In addition, in order to merge surface data, work has also performed in determining ways of registering surface data taken from different views. In the current approach, surface markings that appear within two views will be used to cross register views (see Section 3). Registering can be further refined by manipulating surfaces so that, in areas where the surfaces overlap, the range and texture of surfaces is optimally matched (see Section 5.6).

To provide meaningful range data, the projector and CCD camera used in the SURPHACE sensor must be inter-calibrated (See Section 4). Work, therefore, has been performed in developing a calibration model for cameras, for projector/camera subsystems, and for the

camera/camera subsystem. Careful analysis has also been performed, for example, in linearizing the response of the CCD cameras for the individual pixel and across each scanline, to remove sampling artifacts, and to accurately determine the relative geometries of the projector and the cameras.

Effort has also been spent on determining methods for efficiently manipulating and displaying registered range and surface data. To simplify this task, the VGD system will use a model of a surface that consists of a set of *pixel meshes* [Williams, 1990] where each mesh, in turn, is made up of an array of range/texture pixels, that represent data captured from a particular view (see Section 5). With this data structure, surface data is kept at pixel resolution (as opposed to being stored in a polygon facet and then resampled when it is rendered in a scene, as is generally done). Using the pixel mesh representation, surfaces will be manipulated using standard warping operations to rotate (in-viewplane rotations), translate, and scale a surface. Tilting of a surface will be perform using a special scan-line tilting algorithm (See Section 5.1). To provide an integrated surface, the pixel meshes will be patched together along adjoining boundaries, and each pixel mesh will contain *hooks* that allow patched data from different views to be combined and displayed as one surface.

Finally, we described how the surface models input by the VGD system can potentially be used for tracking 3-D objects (see Section 6).

The following sections will describe the work that has been performed in more detail.

2. Capturing Registered Range and Texture Data.

2.1 Background

In review, the VGD System will use a phase-shift structured lighting system, i.e., the SURPHACE sensor, that acquires registered range and intensity data. Conceptually, the SURPHACE sensor projects a phase pattern onto a surface and inputs the phase pattern via CCD cameras. Because the phase pattern, is relatively unaffected by the surface properties of the object, and because a one-to-one correspondence can be found between the projected phase and the input phase, the system can triangulate surface points for each phase value (see Figure 1).

2.2 Capturing Phase and Albedo Data

In practice, the phase function, $\Phi_p(x,y) = \omega_0 * x$, projected by the SURPHACE sensor is encoded in a set of intensity patterns $L_{pi} = I_0 * \cos(\omega_0 * x + \phi_i)$ with a fixed spatial frequency, ω_0 and a phase offset ϕ_i. After projecting this set of patterns, the sensor measures the distortion of Φ_p caused by the surface in the form of a phase function, $\Phi^c(x,y)$ returned to the camera. The sensor images the phase function $\Phi_c(x,y)$ encoded as a set of intensity patterns,

$$I_{ci}(x,y) = I_b(x,y) + I_a(x,y)*\cos(\Phi_c(x,y) + \phi_i), \quad (1)$$

where $I_a(x,y)$ is the contrast of the pattern due to the photometric properties of the objects imaged, i.e., the albedo of the surface, and $I_b(x,y)$ represents variations in the intensity due to background, i.e., ambient lighting or shadows.

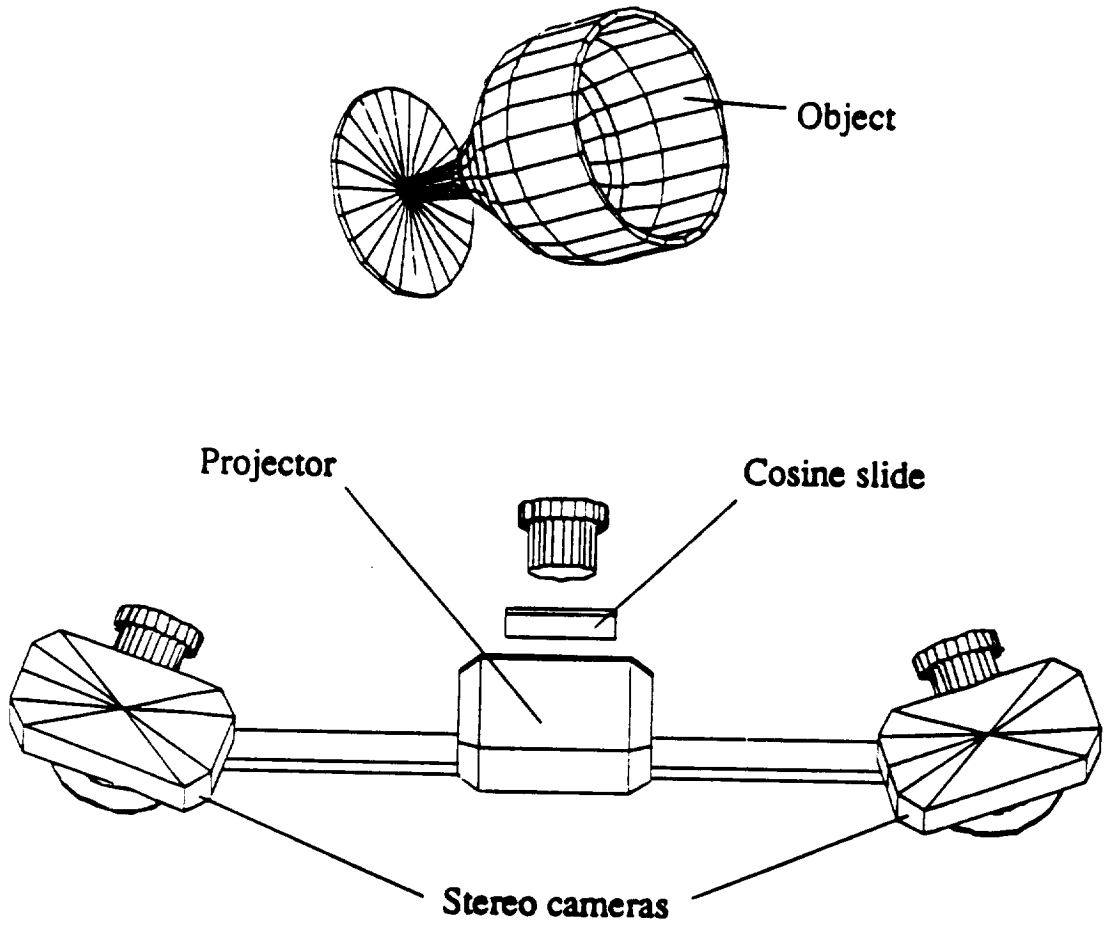


Figure 1: The SURPHACE sensor obtains range by projecting a phase pattern onto the surface of an object, by inputting the phase via a camera, and by setting up correspondences between the projected phase and the input phase.

The phase function $\Phi_c(x,y)$ is decoded from the intensity patterns (and, thus, isolated from the contrast $I_a(x,y)$ and background illumination $I_b(x,y)$ terms) by setting the phase-shifts ϕ_i to integer fractions of 2π (e.g., $\phi_j = 2\pi j/n$, j=0 to n-1 and n > 3), and by using an inverse Fourier series transform. Specifically, $\Phi_c(x,y)$ can be obtained from,

$$\Phi_c(x,y)|_{2\pi} = \tan^{-1}((\Sigma I_{cj}(x,y)*\sin(\phi_j)),(\Sigma I_{cj}(x,y)*\cos(\phi_j))) \quad (2)$$

Decoding $\Phi_c(x,y)$ can be performed efficiently with table-lookup. (Figure 2 shows an example of the phase obtained from a simple object.)

With a single cycle cosine pattern, the phase function Φ_p^s and the imaged phase function Φ_c^s would appear approximately as shown in Figure 3a. Because Φ_p^s and Φ_c^s can be inverted with respect to x (i.e., the functions $x_p^s = \Phi_p^s/\omega_0$ and x_c^s are single valued functions of Φ) a correspondence can be trivially found between the x_p^s and x_c^s for each phase value Φ.

Unfortunately, a single cycle pattern, in general, will not provide an accurate measurement of the range. This is because CCD cameras are typically capable of quantizing only 256 grays levels. Therefore, neighboring surface points of a slowing varying single cycle pattern may appear to have the same phase value, making it impossible to determine an exact correspondence between x_p^s and x_c^s for a particular value of Φ_c^s.

Figure 2: The above shows some preliminary results in obtaining phase data for a simple object, i.e., a coffee cup, using the SURPHACE sensor.

By using a multiple cycle projection pattern, however, (See Figure 3a) the SURPHACE camera can project phases over a much greater range, allowing a more accurate correspondence between the projected and imaged offset x_p^m and x_c^m. Unfortunately, for multiple cycle patterns, the phase function Φ_c^m imaged by the camera may no longer be directly invertible. This is because Φ_c^m, recovered modulo 2π, *wraps around* to 0 every 2π offset. This makes the inverse function $x_c^m(\Phi,y)$ multiple valued (See Figure 3b), and, as a consequence, makes the problem of setting up correspondences between $x_p^m(\Phi,y)$ and $x_c^m(\Phi,y)$ more difficult.

In addition, because the phase *jumps* that occur in Φ_c^m can be due to either wrap-around in the phase or can be due to physical surface jumps, there is no simple way to unwrapping the true Φ_c^m, with the multiple cycle pattern alone.

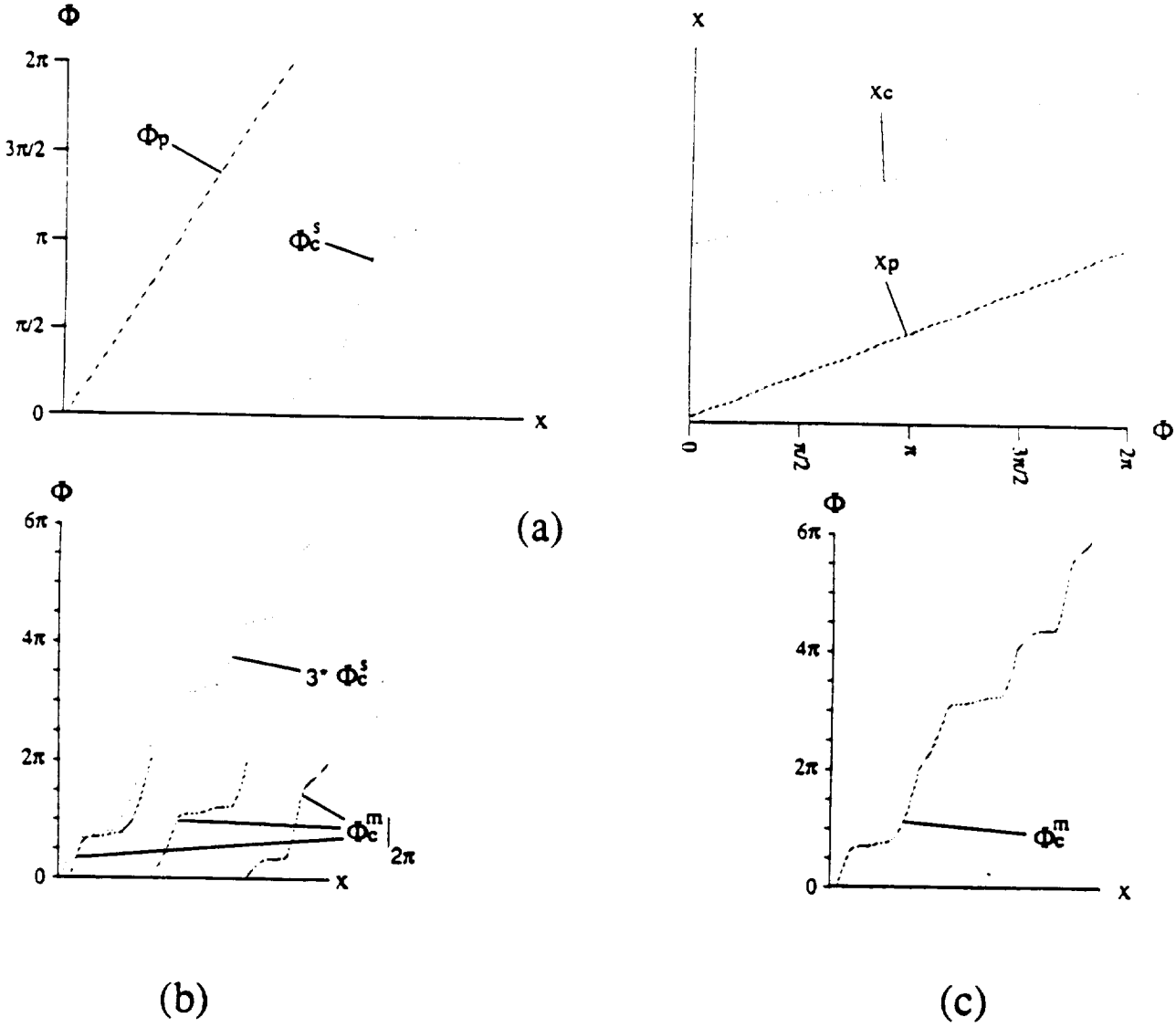


Figure 3: (a) For a single cycle pattern, the projected and imaged phase functions, Φ_p and Φ_c, are single valued and thus invertible. (b) With of a multiple cycle pattern, that can provide greater range accuracy, Φ_c is no longer invertible. By using Φ_c^s of a single cycle pattern as a guide, the multiple cycle phase function can be (c) unwrapped and made invertible.

To solve this problem, we take advantage of the invertibility of the single cycle frequency pattern by using the single cycle pattern, Φ_c^s, as a guide to unwrapping the multiple cycle pattern Φ_c^m (See Figure 3c), i.e., determining where to insert 2π offsets. This technique, however, requires that two sets of phase-shifted patterns be projected and imaged, but allows accurate range data to be captured.

To provide this capability, the current VGD system will use a set of calibrated cosine slides of two different spatial frequencies. These slides will be mechanically translated to provide

phase-shifted cosine patterns and will be interchanged to provide both single and multiple cycle patterns. Ideally, a spatial light modulator would be used to provide the phase-shifted single and multiple cycle patterns.

2.3 Triangulating to Obtain Range Data

After a correspondence has been established between points of equal phase, Φ, the range to a surface point corresponding to phase Φ can be determined using a accurate model of the projector/camera subsystem. For example, to model the projector/camera subsystem, we can use a pinhole camera model for both the camera and the projector (see Figure 4).

In the model we assume that the camera and projector have a coordinate frame with an origins at their respective focal points, and the two frames are related by an offset d and rotation R. For simplicity, we also assume that the projector and camera y coordinate axis are parallel, although this is not a requirement for the system to be calibrated. Since the pattern output by the SURPHACE projector does not vary in the y direction, the problem becomes a 2-D problem, where d can be treated as a two dimensional vector and R as a two dimensional rotation about the y axis with rotation angle θ, i.e.

$$R_{xx} = \cos(\theta) \qquad R_{xz} = -\sin(\theta)$$
$$R_{zx} = \sin(\theta) \qquad R_{zz} = \cos(\theta).$$

Given the above projector/camera geometry and given the x location of points in the projector and camera frame corresponding to phase Φ, i.e., $x_p(\Phi,y)$ and $x_c(\Phi,y)$ respectively, the distance to a point can be found as follows.

Assume that the imaging geometry is as shown in Figure 4. To simplify notation, let the ratios x_c/f_c and x_p/f_p be denoted simply by x_c and x_p, respectively. If the rotation R and translation d between frames are calibrated (see Section 4.2) then the distance of the surface point from the projector's focal point, z_p, and from the camera's focal point, z_c, must obey the relation

$$R(x_c * z_c, z_c) + (d_x, d_z) = (x_p * z_p, z_p).$$

Multiplying the individual terms in R, this relation can be solved for z_c to yield

$$z_c = (d_x - x_p * d_z)/((1 + x_p * x_c) * \sin(\theta) + (x_p - x_c) * \cos(\theta)), \qquad (3)$$

where θ is the rotation about the y axis of the camera coordinate frame with respect to the projector frame.

After z_c is found, the 3-D location of the surface point imaged at location $x_c(\Phi)$ can be determined from

$$p_s = (x_c * z_c, y_c * z_c, z_c).$$

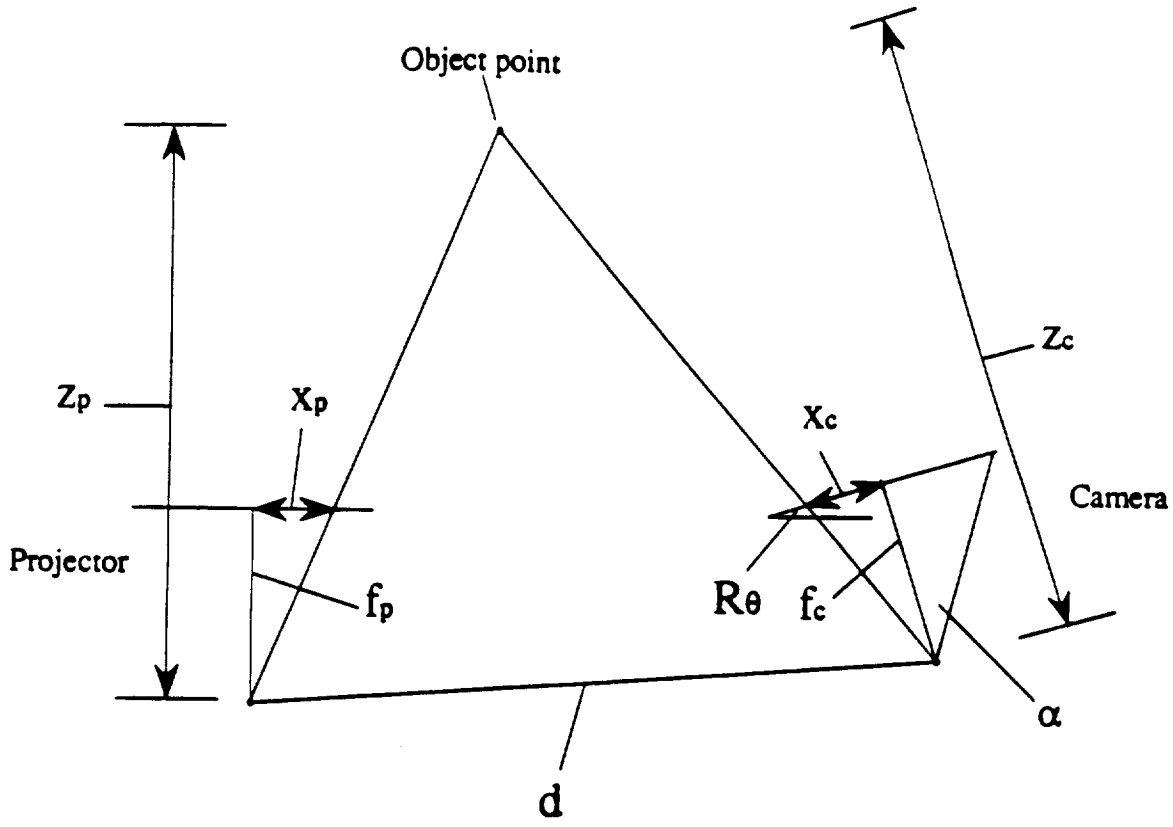


Figure 4: Given that the projector and camera y axis are aligned and given that the rotation R and translation d between local coordinate frames is known, then triangulation of a surface point with correspond phase in the projector and camera frames is straightforward.

Furthermore, if the projector and camera coordinate systems have the same orientation and only a x offset of d then, the equation for z_c can be simplified to

$$z_c(\Phi,y) = d/(x_p(\Phi,y) - x_c(\Phi,y)), \tag{4}$$

where the denominator x_p-x_c represents a normalized *disparity* term. The disparity term is similar to one that appears in the standard stereo camera equation, except that is this case the offsets x_p and x_c are normalized by their focal lengths.

From this equation, it becomes obvious that once the difference between the x offset in the projector and x offset in the camera coordinate systems are known for a particular projected phase value Φ, then the distance to the surface point imaged with phase value Φ, can be easily calculated.

Once the phase $\Phi_c(x,y)$ and $z(\Phi,y)$ are found a relative surface albedo can be determined from $I_a(x,y)$, if it is assume that the projected illumination falls off as the inverse of the distance to the surface point and returning from the surface point.

2.4 Range Sensitivity

Determining the range sensitivity of the SURPHACE sensor requires an analysis of how x_c and x_p affect z_c. For example, from Equation (4), a differential in z_c is related to differentials in x_c and x_p by,

$$dz_c/z_c = -(dx_p - dx_c)*z_c/d \tag{5}$$

where the percentage change in z_c due to dx_c or dx_p increases linearly with z_c and inversely with the offset distance d.

As described in Section 2.2, projecting a low frequency spatial pattern will not allow sensitive range measurements to be made. The solution to this is to project a high frequency cosine pattern onto the surface. However, the spatial frequency of the pattern, as viewed by the camera, is bounded by the sampling frequency of the camera. For example, assume that there are N pixels per scanline and that the camera model has a angle of view 2α (see Figure 4) where $-\tan(\alpha) \le x_c \le \tan(\alpha)$, then an upper limit on the spatial frequency that would be resolvable by the camera, would be $N/(2*f_c*\tan(\alpha))$. This means that the sensitivity of dz_c is limited by,

$$dz_c/z_c \le 2\tan(\alpha)*z_c/(N*d). \tag{6}$$

In other words the range sensitivity of the SURPHACE sensor depends on the spatial resolution of the camera for any given view angle, and on the distance between the projector and camera. As an example, for an object at 1 meter, with a view angle of 45•, and a 512x512 camera, the distance from the camera to the projector would need to be approximately 16 cm to obtain a measurement of the range to within 1%.

2.5 Potential Errors

Because the cosine pattern used to generate the phase-shift structured lighting has been produced optically, the errors in dx_p are minimal within the slide. On the other hand, since the cosine pattern is projected from a slide, errors in dz_c/z for any depth are linear in translational positioning errors, dx_p, in the slide

Even worse, there are cases in which phase data and hence the range data will be completely invalid, i.e., (1) surfaces in the scene that are input by the CCD camera but that are not illuminated by the projector, (2) regions that absorb all of the projected light so that there is not

enough variation in the illumination between phase shifts to accurately determine the raw phase, and (3) specular regions that are so bright that the camera is saturated and, therefore, do not vary according to the model discussed in theory, yielding poor range data.

The three sources of invalid data are fortunately all characterized by very small variations in the pixel values between phase shifts: case one will have no variation, cases two and three will have little variation. This makes it possible to mask out invalid data by simply eliminating pixels that do not have sufficient variation among phase shifts.

3. Registering Surface Data

One view of an object does not provide sufficient information to produce a complete model of all surfaces of an objects. Thus, the object's surface must be captured from several different views, and the data taken at each view must be merged. To properly merge the data, data at different views must be registered.

Permitting surface data acquired from any view requires that the relative transformation between the viewpoints be determined precisely. We accomplish this by using landmark features, i.e., features that can be easily identified in the surface data from several viewpoints. We are investigating ways to select and detect the surface landmarks. Currently, landmarks are hand-selected.

We describe the location of such landmark features by position vectors obtained from the surface data using the camera/camera, or stereo camera subsystem of the SURPHACE sensor. If the position of a landmark feature i in viewpoint one was represented by the vector p_i, and the position of a corresponding feature in a second viewpoint was represented by vector r_i, then the two positions would be related by a rotation, R, and translation, t. These are the quantities that must be determined in order to determine specify the geometrical relationship between viewpoints. Mathematically, the relationship is

$$r_i = R\, p_i + t.$$

Now, if differences are taken between points p_i and p_j and the corresponding points r_i and r_j to produce difference vectors, p_{ij} and r_{ij}, then,

$$r_{ij} = R\, p_{ij}.$$

By representing the rotation operator as a quaternion, a simple solution can be obtained for determining the rotation between the viewpoints. Specifically in a quaternion representation, the above relation could be written as,

$$r_{ij} = q_R \cdot p_{ij} \cdot q_R^{*}, \quad \text{subject to } q_R \cdot q_R = 1,$$

where "•" indicates quaternion multiplication, and where the quaternion, q_R, and its conjugate, q_R^* contains four variables that specify the axis and angle of rotation.

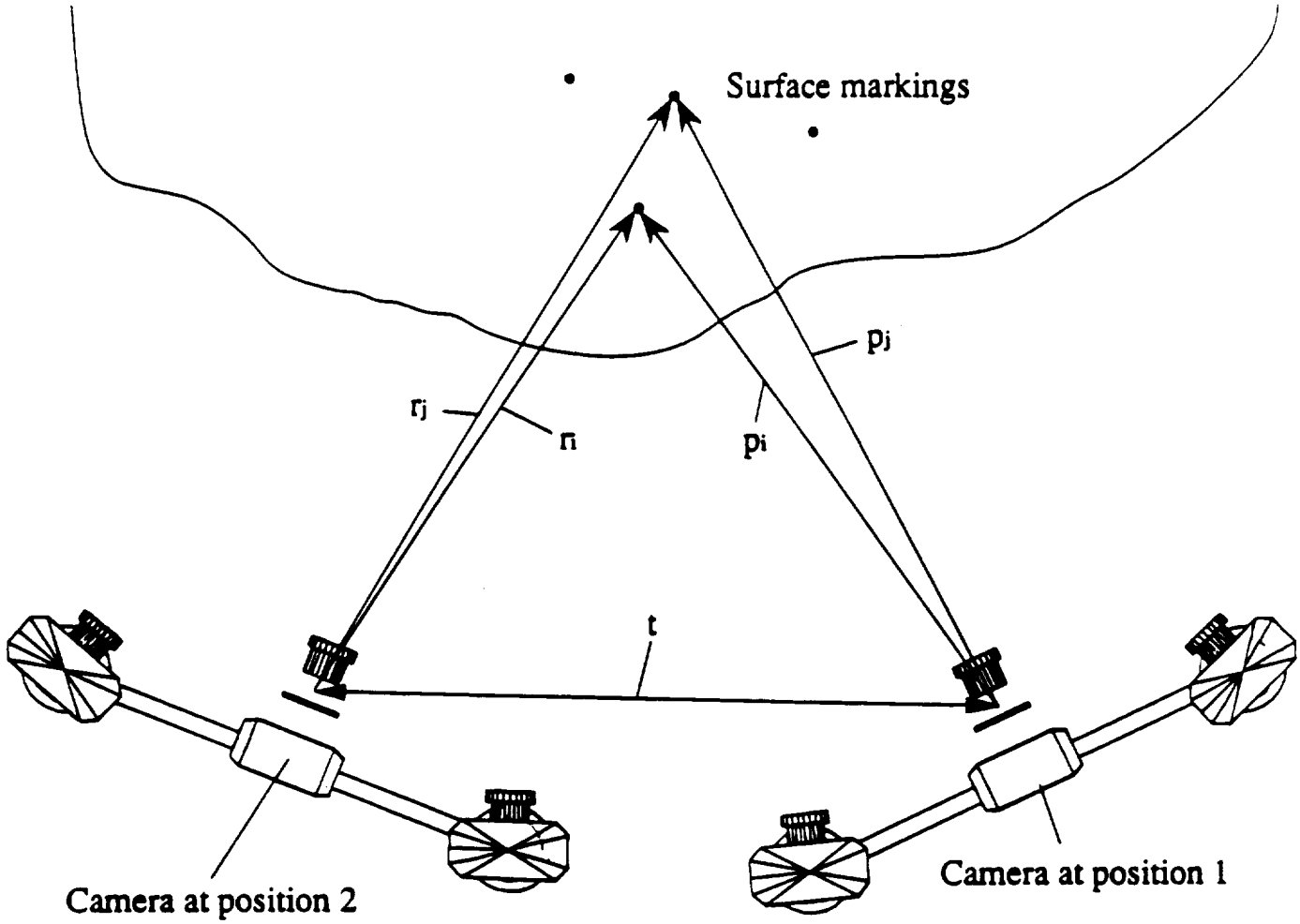


Figure 5: 3-D measurements of the landmark features will allow vectors, p_i and r_i to be made from different views. These vectors are related by the same transformation, (R,t), that can be used to transform the surface data into one frame.

When more than three points, p_i, are used, the problem is overspecified and can be solved with least squares methods,

$$\min q_R^t B q_R, \quad \text{subject to } q_R^t q_R = 1,$$

where q_R now represents a 4x1 matrix of quaternion coefficients and B is a 4x4 matrix with terms involving the difference vectors, p_{ij} and r_{ij}. This problem is an eigenvalue problem and its solution is the smallest eigenvalue of the matrix B, normalized to 1. Once the quaternion

rotation, q_R, is found the rotation matrix R can be determined from q_R. In addition, once R is determined, then t can be found by back substitution.

Another approach to registering surfaces without surface landmarks is use a device such as the 3-Space Tracker [Raab et al., 1979] to determine the 3-D pose of the SURPHACE sensor will it is acquiring data. The 3-Space Tracker uses a of three orthogonal magnetic coils mounted at a fixed location to transmit magnetic pulses through space and a set of three orthogonal pickup coils to receive the pulses and provide data that can be used to determine the absolute pose of an object with respect to the transmitter location. Although the VGD budget does not permit the use of a system of this type, it would be ideal for registered camera locations with surface markings.

Finally, fine-tuning of the registration between surfaces may be accomplished by correlating the surface data between views, and by positioning two surfaces, that are to be merge, to maximize the correlation between the range and intensity of the areas of surface overlap. If registration using surface markings proves to be too crude to accurately register surfaces, we will explore this technique (see Section 5.6).

4. Calibrating the SURPHACE Sensor

As described earlier, the SURPHACE sensor consists of a high intensity projector and two CCD cameras. The cameras are used individually with the projector to obtain surface data, and used in stereo to input the 3-D location of registration markings. This first capability is used to input surfaces and the latter capability is used in register surface data in order to merge surfaces into one coherent surface (see Section 5).

In any case, the problem of calibrating the SURPHACE sensor consist of three separate calibration problems, i.e., (1) calibrating a single camera, (2) calibrating a projector/camera subsystem, and (3) calibrating the camera/camera subsystem.

4.1 Calibrating a Single Camera

Calibrating a camera consists of linearizing the camera/digitizer transfer function and of modeling the camera's imaging geometry to enable correction of any non-ideal behavior.

4.1.1 Modeling the Camera

Linearizing the camera/digitizer transfer function was accomplished in two ways. First, there exists a gamma correction potentiometer on the CCD cameras that allows the camera signal to be partially linearized. Second, we found that pixels across a scanline, although having a linear behavior individually, had an uneven response. The response to the same light values was less at the beginning and end of the scanline than in the middle. Although the SURPHACE

camera only needs a linear response at each pixel to determine range, it requires uniformity in response over the pixels if surface albedo is to be properly measured. This problem will be solved by calibrating the response across a scanline and using table lookup to correct a response.

In modeling a single camera, a number of problems were examined and solved, include removing:

- Distortion of pixel locations by imperfect lenses. The most prominent of these effects is due to radial lens distortion.
- Variations in the sampling frequency and the horizontal retrace digitization startup timing that lead to inaccuracies in the horizontal scale factor and the horizontal positioning of the image.
- Misalignment of the sensor array with the axis of the lens system, resulting in the middle pixel not being the true center of the optical system.

To solve these problems a camera calibration scheme, described in the literature [Tsai, 1987], was adapted to create models of the CCD cameras used in the SURFACE Camera. The basic geometry of the camera model is shown in Figure 6. Imaging can as described in Tsai [1987] be viewed as required four steps:

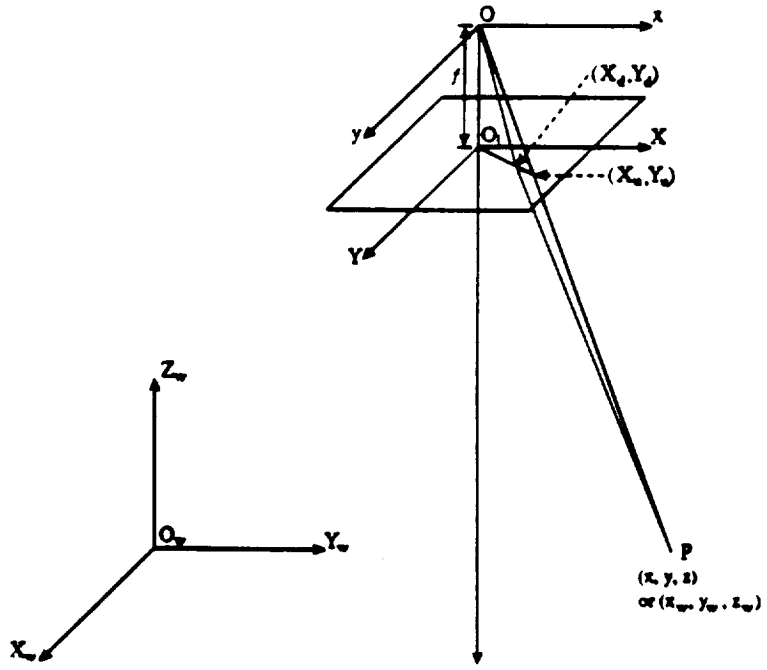


Figure 6: **The geometry of a single camera can be described reasonably accurately by a relatively few set of parameters. Radial lens distortion introduce some error into image measurements.**

1) A rigid body transformation from (x_w,y_w,z_w) to (x,y,z)

$$(x,y,z)^T = R(x_w,y_w,z_w)^T + T$$

2) A perspective projection with pin-hole camera geometry

$$X_u = fx/z$$
$$Y_u = fy/z$$

3) A radial lens distortion with second-order polynomial approximation

$$X = X_d = X_u(1+\kappa r^2)^{-1}$$
$$Y = Y_d = Y_u(1+\kappa r^2)^{-1}$$

with

$$r^2 = X_u^2+Y_u^2$$

4) A digitization of real image coordinate (X,Y) to computer image (frame buffer) coordinate (X_f,Y_f)

$$X_f = d_x'X+C_x$$
$$Y_f = d_y X+C_y$$

with

$$d_x' = s_x d_x$$

where

(x_w,y_w,z_w):	3-D coordinates of point P in 3-D world coordinate system
(x,y,z):	3-D coordinates of point P in 3-D camera coordinate system
(X,Y):	coordinates of the image coordinate system
(X_d,Y_d):	distorted 2-D image coordinates of P
(X_f,Y_f):	digitized 2-D image (frame buffer) coordinates of (X, Y)
(X_u,Y_u):	ideal undistorted 2-D image coordinates of P
(C_x,C_y):	image frame center
(d_x,d_y):	distance between adjacent sensor elements in X and Y direction, respectively
(d_x',d_y):	distance between frame buffer pixels in X and Y direction, respectively
s_x:	horizontal scale factor
f:	focal length

κ: radial lens distortion factor

d_x and d_y are available from the camera and frame grabber specifications, as are N_{cx} and N_{fx} for an initial rough estimate of s_x. We can assume zero distortion (κ=0) and unit focal length (f=1) at the beginning of the calibration procedure.

These parameters can be divided into two classes: intrinsic parameters and extrinsic parameters. Intrinsic parameters describe the imaging process within the camera, while extrinsic parameters describe the cameras relationship of the camera to the world coordinate system. Thus, R and T in the equations of the rigid body transformation are extrinsic parameters, while the remaining parameters, specifically f, κ, (C_x,C_y), and s_x, are intrinsic parameters.

4.1.2 Determining the Camera Parameters

Calibration of the video camera consists of determining the intrinsic parameters. However, since the transformation relative to any external calibration pattern must be determined, the extrinsic parameters must be determined as well. However, the effort spent on determining the extrinsic parameters is not wasted since we can use this information to calibrate the projector/camera and camera/camera transformations by determining the transformation between each of the SURPHACE subsystems' coordinate frames with respect to the same calibration scene. For example, determining the inter-camera transformation is all that is necessary to complete the calibration of the stereo apparatus.

The VGD algorithm for monocular video camera calibration rests on the following principle: given a 3-D calibration pattern consisting of line segments, if the camera is perfectly calibrated, the inverse image of a point on one of the lines comprising the calibration pattern in the computer's image frame buffer coordinate system (the (X_d,Y_d) plane), which is a ray emanating from the camera's optical center in the 3-D world coordinate frame, will intersect perfectly the corresponding line in the 3-D calibration pattern.

In reality, the parameters, intrinsic and extrinsic, will be only approximately known. Thus, the ray will not intersect the calibration pattern line. However, the better the camera model parameters are, the smaller the distance of closest approach between the ray and the line in the 3-D calibration pattern will be. This is shown graphically in Figure 7. If many points and several lines are used, the parameters resulting in the best calibration can be found by minimizing the sum of the distances between the rays and their corresponding 3-D lines in the calibration pattern.

Viewed from a mathematical perspective, the calibration process amounts to a nonlinear optimization. For this reason the robust unconstrained optimization procedure will be used to perform the optimization.

In using optimization, we needed to solve the key problem of avoiding optimization *traps* caused by singularities in the parameterization of the rotation R. A singularity is a point in the rotation parameter space where many parameter values map into the same rotation. When the optimization procedure nears a singularity, it may search out in a direction that has no effect on the rotation because of the singular values of the parameters. However, once this occurs, the optimization procedure will not change the values of the other parameters because, if it does, the rotation is likely to change rapidly, increasing the error metric, effectively trapping the optimization in the singularity.

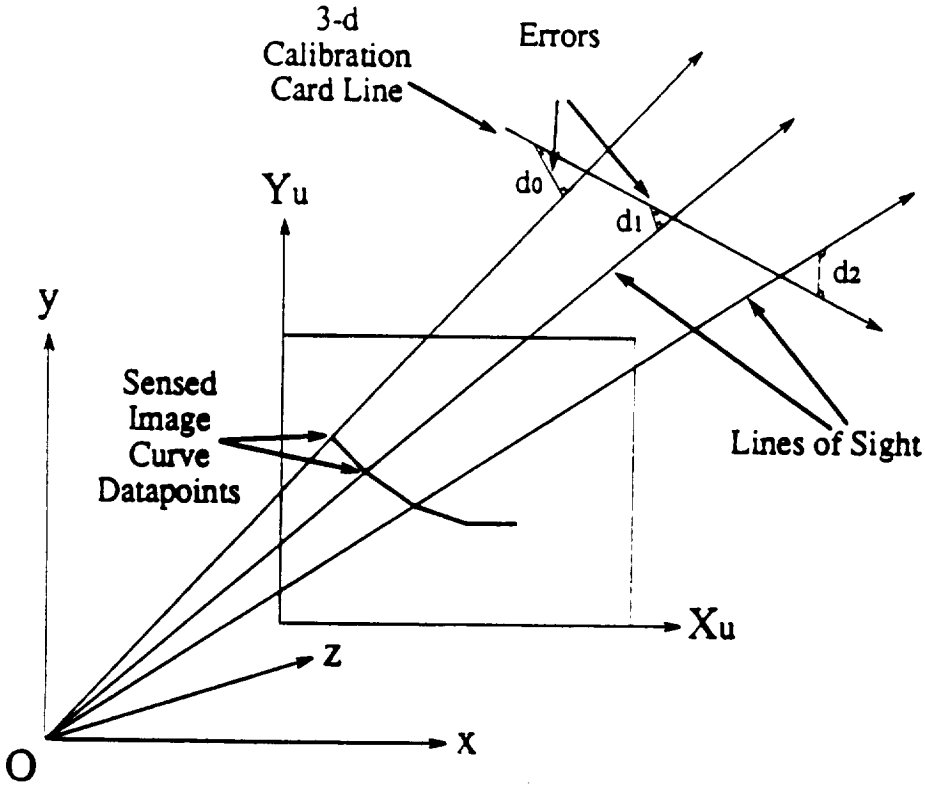


Figure 7: The camera model can be readily calibrated by using a straight line calibration pattern and by adjusting the camera model parameters until the distance between rays emanating from the focal point of the camera and the straight lines is minimized.

This problem was solved by parameterizing the rotation matrix using unit quaternions. Specifically, a quaternion representation has no singularities or the type described above, only the usual dipole symmetry that is unavoidable in any rotation parameterization (and which does not cause problems for the optimization procedure).

To achieve the best possible camera calibration, we must determine the location of the lines in the image of the calibration card to the highest possible accuracy. Conceptually, as described previously, the location of the points in the image that correspond to a particular line in the calibration card will be extended into 3-D space by a ray from the optical center of the camera

through the image point. Calibration will be effected by adjusting the camera parameters until the nearest approach of the ray and the 3-D lines forming the calibration card is minimized for a large number of such image points. Thus, it is critical that the location of the points that make up "lines" in the computer's frame buffer image (the "lines" are actually slightly curved do to the radial lens distortion) be determined with the highest possible accuracy.

The calibration algorithm allows the user to select a number of lines in the image by clicking the mouse at pairs of points near the image lines. The algorithm then searches perpendicular to the line the user provided, looking for image lines. When found, the location of the intersection points are determined to sub-pixel accuracy. This is accomplished as follows.

First, the image is resampled along the search line, using bilinear interpolation, to yield a 1-d signal of length n, where n may be user specified. We model the cross section of an image curve by a Gaussian pulse of width σ. The pulse is detected and accurately located using Boie and Cox's [1987] method. First optimal detection and localization masks are produced. The masks are correlated with the signal derived from the image to yield a detection signal and a localization signal. The points in the localization signal where a zero crossing occurs and the detection signal exceeds a detection threshold forms a detected point. Linear interpolation of the pixels on either side of the zero crossing yields the location of the pulse to sub-pixel accuracy.

Although Boie and Cox's algorithm uses hard thresholds, we have modified it to use a form of adaptive thresholding, thus permitting a wide variation in the contrast of the imaged lines without changing the user defined threshold value.

4.2 Calibrating the Projector/Camera Subsystem

In calibrating the project/camera subsystem, the primary problems are in insuring that the projector is outputting an accurate cosine pattern, and that the geometrical relationship between the projector and camera are accurately determined.

As described above the projector subsystem of the SURPHACE sensor consists of a high-intensity light projector with a micrometer-mounted slide holder that permits accurate phase shifting. A number of problems were encountered in manufacturing slides with accurate low spatial frequency cosinusoidal patterns, but this problem was solved when a vendor [Sine Patterns, 1990] was located who manufactures custom cosinusoidal slides of high accuracy.

The problem of modeling the projector is similar to that of modeling a single camera. To model the projector/camera subsystem, we will use a pinhole model for both the camera and the projector described in Section 2.3 (see Figure 8).

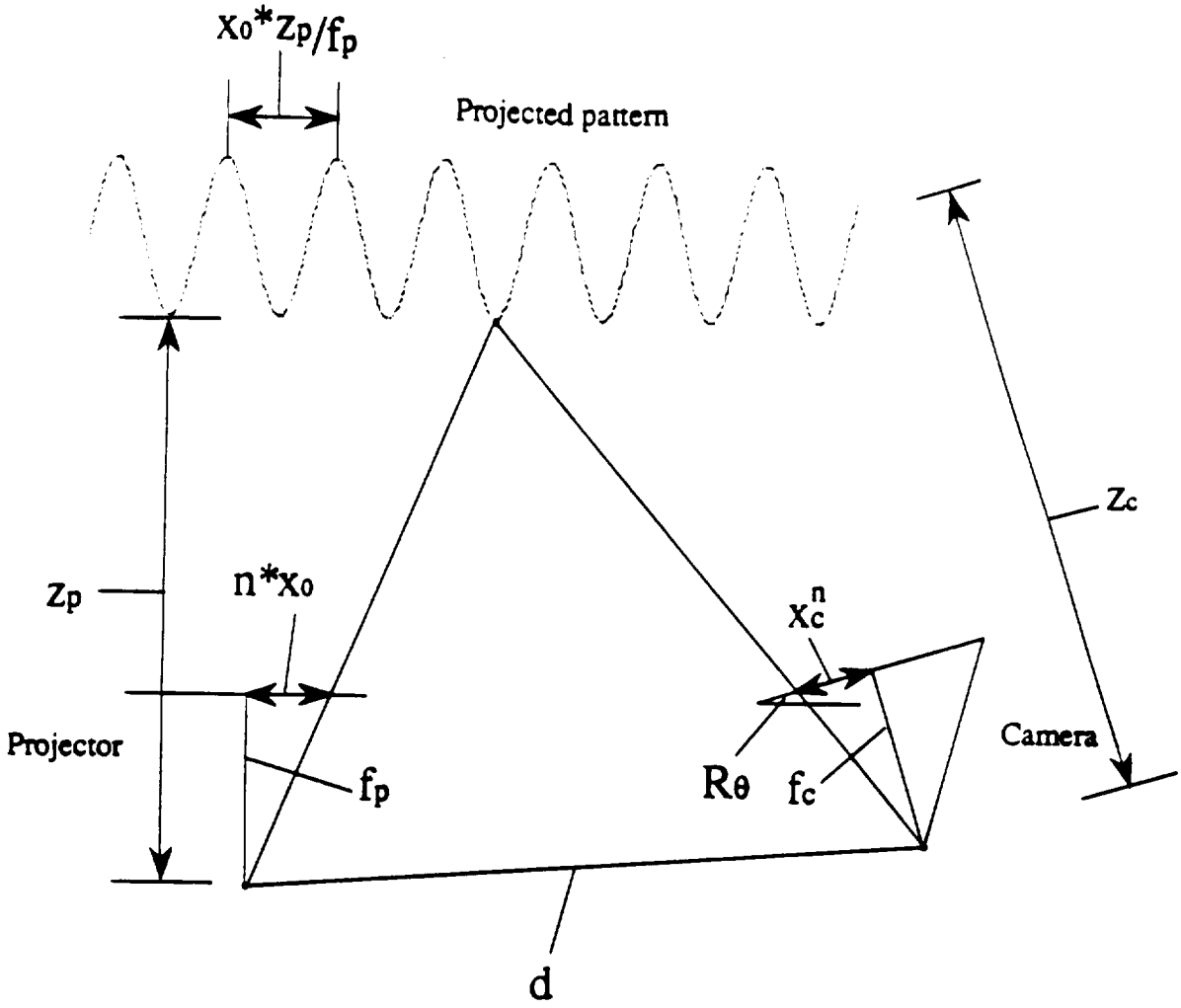


Figure 8: Assuming an ideal pinhole camera and projector system and assuming that the y axis of the projector and camera local coordinate systems are aligned, calibration of the projector/camera subsystem of the SURPHACE camera becomes simply the problem of determining the 2-D rotation R and the translation d between coordinate frames.

To calibrate the projector/camera system, we can place a screen at a distance z_p from the projector. If a pattern with equally spaced points, with x spacing x_0, has been projected onto the screen, then the spacing on the screen will be $x_0 * z_p / f_p$. Assume that one of these points is imaged at x_c^n in the camera. To simplify notation, let x_0 / f_p and x_c^n / f_c be denoted as x_0 and x_c^n respectively. A ray from the focal point of the camera through the point x_c^n can be described in the projector frame by,

$$R(x_c^n * z_c, z_c) + (d_x, d_z) \qquad (7)$$

where z_c is the distance from the focal point to a point along the line. Let z_c^n denote the parameter that must be selected so that the position along the line corresponds to position $nx_0 * z_p$, on the screen, i.e.,

$$R(x_c^{n}*z_c^{n}, z_c^{n}) + (d_x, d_z) = (n*x_0*z_p, z_p) \quad (8)$$

If this equation is used for two points at x offsets in the projector frame of $n*x_0*z_p$ and $(n-1)*x_0*z_p$ respectively, and if the resultant two equations are subtracted, the following relation is obtained,

$$R(x_c^{n}*z_c^{n} - x_c^{n-1}*z_c^{n-1}, z_c^{n} - z_c^{n-1}) = (x_0*z_p, 0). \quad (9)$$

This relationship represents three unknowns, the rotation angle θ, and the parameters z^n and z^{n-1}, not enough information to solve for θ. However, if this step is repeated for two points at offsets $n*x_0*z_p$ and $(n+1)*x_0*z_p$, one obtains the equation,

$$R(x_c^{n+1}*z_c^{n+1} - x_c^{n}*z_c^{n}, z_c^{n+1} - z_c^{n}) = (x_0*z_p, 0). \quad (10)$$

To set of equations in (9) and (10) can be solved for the angle of rotation, θ. After θ is found the vector d can be found from the previous relation involving d.

By using even more points, the system of equations become overspecified, and can be solved using least squares optimization for a more accurate final result.

Once the projector/camera system is calibrated, the surface points can be triangulated as described in Section 2.3 to obtain surface range.

4.3 Calibrating the Camera/Camera Subsystem

The purpose of calibrating the camera/camera subsystem is to provide a stereo camera system to be used in registering views of a surface. The system will input surface markings and determine the distance to the markings within each view and use this data to roughly find the 3-D transformations between views (see Section 3).

Even though the cameras may be calibrated with respect to the projector, they will not necessarily be calibrated with respect to each other. This is because the y axis in the camera and projector system were aligned and the projector output a pattern independent of the y axis. Therefore the two cameras may be out of calibration in the y direction.

Using stereo to input distances requires setting up correspondences between features in one image with features in a second image. This can be done quite easily if the features in one view are along the same scanline in the second view. This is called the *epipolar constraint*. To insure the the cameras in the VGD system satisfy the epipolar constraint, a calibration pattern of intersection lines will be projected by the projection system onto a screen perpendicular to the z axis of projector (using a similar geometry to that show in Figure 8). The location of the lines can be identified accurately using the algorithm described above, and, therefore, the location of the intersections of the lines can also be accurately described. By aligning the cameras so that

intersection points fall on the same scanlines within both cameras, and by knowing the calibration between the camera and projector, we will be able to easily determine the relative distance and rotation between camera frames.

5. Merging Surface Data into One Complete Surface.

As described above, the VGD SURPHACE camera will be used to capture the texture (i.e., detail) and range of the surface of an object for a set of digitized views. This data will be used to reconstruct partial surfaces of the objects. Since markings on the surface of an object will be used to establish the relationship between these surfaces by establishing the coordinate transformations between the views from which the surface data is captured, the final step in reconstructing a 3-D model of an object will be to use the relationships between surfaces to merge the partial surfaces into one coherent surface.

Some of the problems associated with merging surfaces are that,

- Errors in the view coordinate transformations will result in poor surface merging, with the result that surface texture will have discontinuities due to an incorrect overlap of the surfaces during merging.

- The surface data is captured as image data and must be converted to some form of graphical data structure for rendering.

- A technique must be devised from joining surfaces. This technique must not leave "gaps" in the merged surface.

To solve these problems we will investigate (1) approximate surfaces by a *pixel mesh*, (2) globally optimize the surface fit between surfaces of different views using a surface matching strategy that uses both range and texture data to adjust the view coordinate transformations, and (3) connect pixel meshes between views to obtain a coherent surface.

In the following we will present a discussion of each of these steps.

5.1 Converting Surface Data to a Pixel Mesh

In order to match surfaces from different views, the surface data must be mapped into a common frame and sampled on a common sampling grid. For example, the range data that is captured by the SURPHACE sensor will be sampled along rays that emanate from the focal point of the CCD camera and that intersect the surface at some point. To perform a comparison between surface data in two different views, surface data must be mapped between frames and sampled at the same sample points.

One possibility would be to approximate surface data by a polygon mesh [Foley and Van Dam, 1982], to map the texture data into the polygon facets, and to manipulate the surface data by manipulating the polygons. Then when two surfaces are compared, for example, during surface merging, the polygons and the texture data stored in the polygon facets would be resampled to obtain range and texture data at a set of sampling points.

A simpler approach, to be used in VGD, will be to sample the surface data in after it is orthogonally projected and to use standard image warping operations to manipulate the surface data. We will refer to this form of data as a *pixel mesh* in which conceptually each pixel of the resampled surface corresponds to a vertex of a faceted surface (see Figure 9). When surfaces are manipulated, surface data will be remapped, using image warping operations. For example, the orthogonally sampled surface data in view 2 is mapped into view frame 1 through a set of simple 2-D scale, rotation, and translational warping operations, as well as a new 3-D tilt operation.

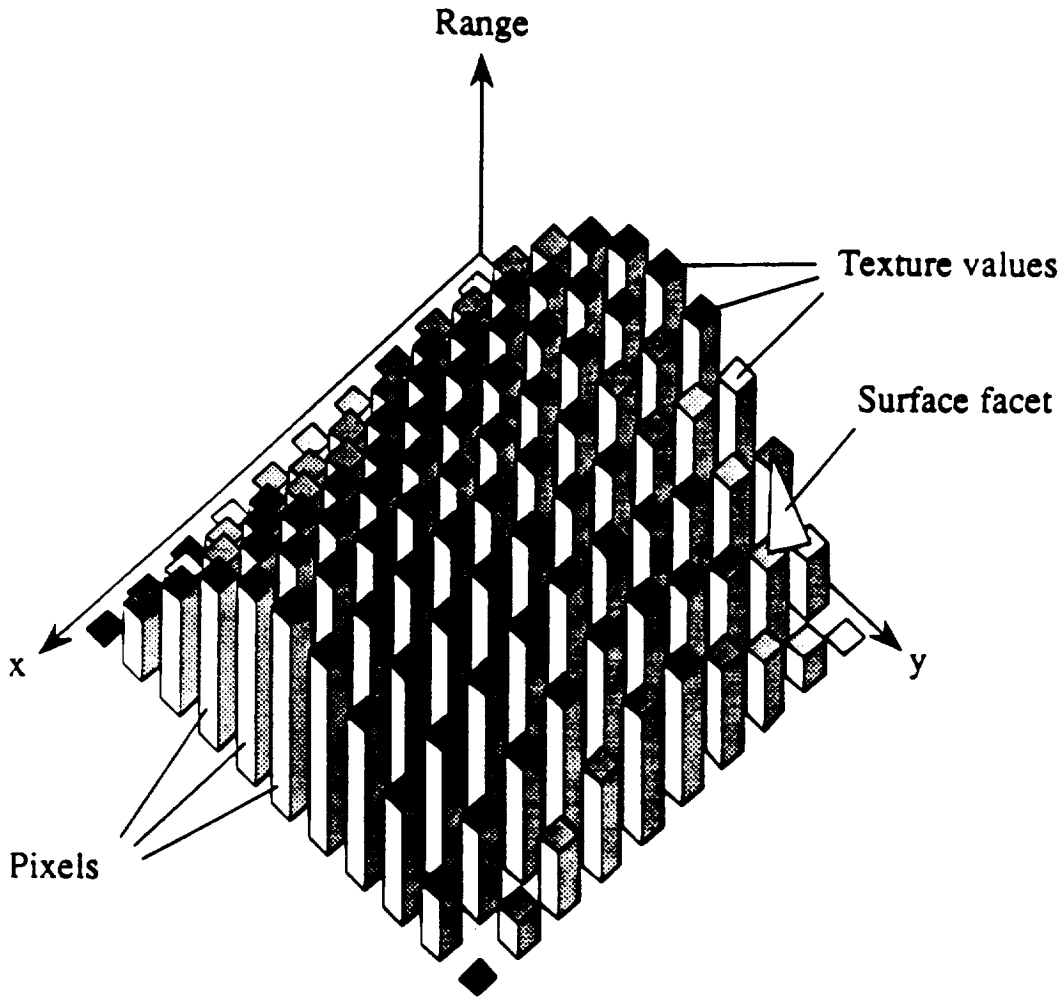


Figure 9: **Each pixel in the pixel mesh** is connected to its **nearest neighbors to form a** set of **surface facets. Both the range** and texture are **stored at the pixel locations.**

5.2 Manipulating the Pixel Mesh

To define our terminology, a 2-D rotation will consist of a rotation about the orthogonal view direction of the pixel mesh. A tilt, on the other hand, will consist of a rotation about an axis perpendicular to the orthogonal view axis (i.e., in the x-y plane in Figure 10).

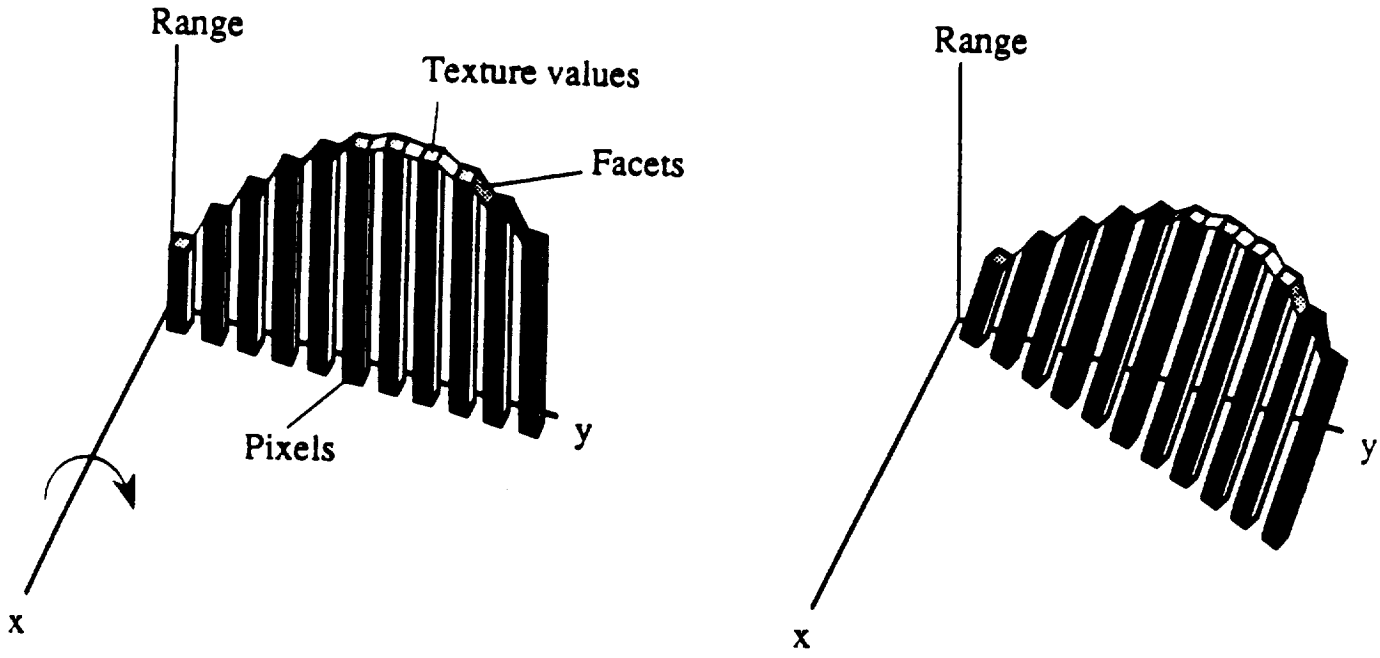


Figure 10: Rotations about axis perpendicular to the view-axis will be handled by a special tilt operation in which the data is tilted along a scanline. In tilting, facets between sample points are tilted and then scan converted into new range values at the original sample point locations, using a scan-line z-buffer.

2-D scaling, rotating, and translating a 2-D image can be performed efficiently on a pixel mesh using standard image warping operations [Catmull and Smith, 1980]. The only complication is in tilting a pixel mesh. This is because a pixel has a range value, and, therefore, when the surface is tilted, the pixels must manipulated to reflect the rotation. For example, a pixel mesh may fold over on itself, i.e., have locations that are multiply defined, due to the fact that pixel points having different range values now occupy the same x and y location. However, this type of operation can be handled using a scan-line z-buffer algorithm.

To perform tilting operations we will use a scan-line z-buffer algorithm, i.e., tilting of a pixel mesh range/texture image about an arbitrary axis will be done by first rotating the image about the z-axis (i.e., the axis perpendicular to the image plane) so that the tilt axis is aligned with the y-axis. The data is then tilted via a scanline algorithm and then rotated so that the tilt axis is its original direction.

When tilting about the y axis, each pair of sequential pixels in the pixel mesh can be treated as a linear line segment in a horizontal slice of a surface. To tilt a surface, each line segment in a scan-line is rotated about the y-axis and resampled at the points of the pixel grid. The values of the texture and range at the sample points can linearly interpolated from the texture and range values of the tilted line segment.

As in a standard z-buffer algorithm, if during the linear interpolation and sampling, the range value at a sample point has a value smaller than the value previously stored at the sample location, then the value will be assigned the current range and texture value. If, however, the range previously stored at the location is larger than the current value, the range and texture stored at the location is unchanged.

After each scanline of the image is tilted using this approach, the image is then rotated back to its previous orientation. This approach, thus, is simple and yet provides a general mechanism for tilting a range/texture image.

5.3 Comparing Pixel Meshes

Once a pixel mesh has been mapped from view 2 to view 1, it can be compared to the pixel mesh within view 1. Both the texture and range data will be compared using a texture and range metric respectively. Evidence of a match between two patches will depend on a weighted sum of the values returned by the texture and range metric. After the initial tilting operation, comparison of surface data is completely performed in 2-D.

5.4 Globally Optimizing the Surface Fit

Although views will be registered as described in Section 3, realistically, the transformations between views obtain using the describe approach will contain some errors. As a consequence, when the surfaces from different views are combined, discontinuities in the surface texture and range will appear in the combined surface along the boundary at which the surfaces are merged.

To minimize this problem, we will investigate the use of a global optimization approach in which, starting with the transformations obtained during registration, surfaces will be optimally fit together to minimize a global fit criterion. Fitting will be done by attaching spring-like forces between similar surfaces (see Figure 11), and by allowing the attached *pseudo-springs* to settle to a stable state. This approach will allow the transformations between view frames to be globally adjusted until the combined surface data has minimum discontinuity.

Specifically, using the initial view parameters established, surface data from different views will be matched in pairs. Consider the two view frames shown in Figure 11a. The approximate

overlap of the surfaces 1 and 2 can be found by mapping the region of valid data in view 2 into region of valid data in view 1 (see Figure 11b) using the transformation established during the registration step. 6-D pseudo-spring forces (with energy based on differences in coordinates and orientation) will be attached between similar regions within the two views. This technique will be repeated for all of the views that have overlapping coverage. Finally, the view parameters between views, i.e., the 3-D translations and rotations between views, will be adjusted until the sum of the total spring energy is minimized.

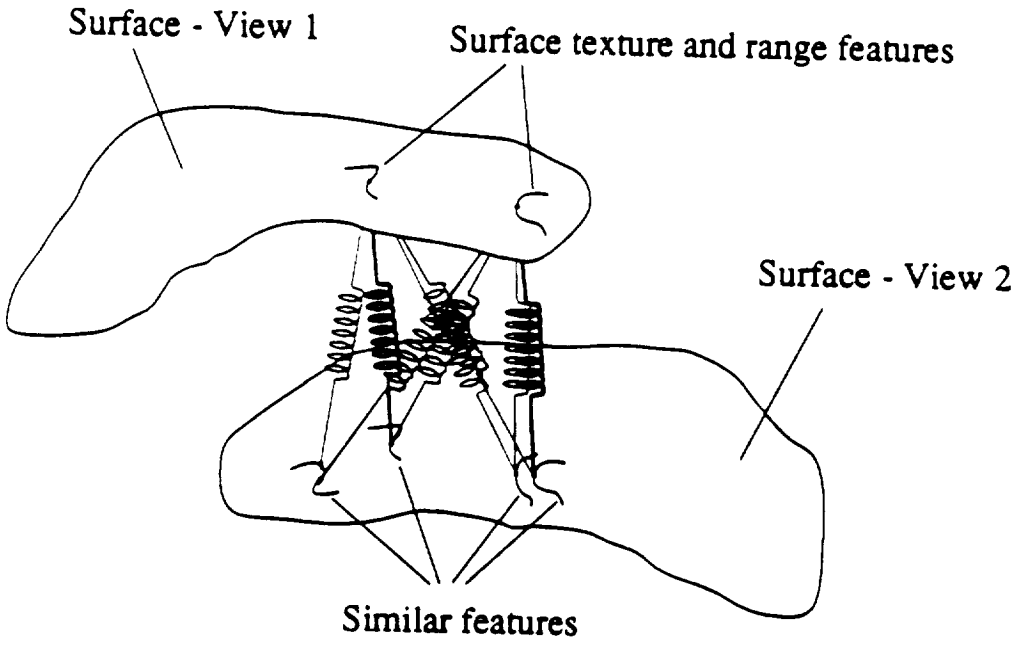


Figure 11: Surfaces will be compared and a set of pseudo-springs will be set up between the surfaces, where the springs will be attached between areas of similarity.

Using a robust statistics approach, springs that are overstretched will be cut and the energy of the remaining springs will be minimized. The technique of clipping overstretched springs is a method of removing outliers or false surface matches from the optimization process. Clipping of spring forces will be repeated until the surfaces are well matched globally, i.e., when there are no highly stretched pseudo-springs.

5.5 Segmenting the Surface into Best Estimate Regions

After a optimum set of transformations between views has been determined, an initial view will selected. For example, assume that view 1 is selected. All views that that share surface data with view 1 will then be mapped into view 1. Views that overlap view 1, but provide data outside view 1 will be marked in the view 1 image. When mapping surfaces from other views into view 1 some data will be lost due to foldover (i.e., self-occlusion). The location surrounding

the pixels at which this occurs, that is, the border of the foldover will be marked. This marked area represents the locations at which data must be patched in from other views to complete the surface data capture in view 1.

It is also possible to determine for each pixel of the range/texture images in view 1, whether or not data from another view might better represent the texture and/or range value at the pixel. For example, the normals of the surface (defined by the range data) can be used to determine the view that has the least oblique angle to a surface for a region of pixels. If this option is implemented, the region within view 1 that is better estimated by a different view will be masked out and surface data from the other view will used to patch the surface at this location.

After the surface from a particular view has been segmented into best estimate regions, it consists of a set of surface patches that are best described by a particular view. For example, in Figure 12, a region of the surface in view 1 (shown crosshatched) may be best described by surface data from view 2. This region is masked out in view 1 with the idea that data for the region will be supplied from view 2. In the special case, when surface data is visible in view 2, but is invisible to view 1, the visible data must be connected from view 2 into view 1 (See Figure 12).

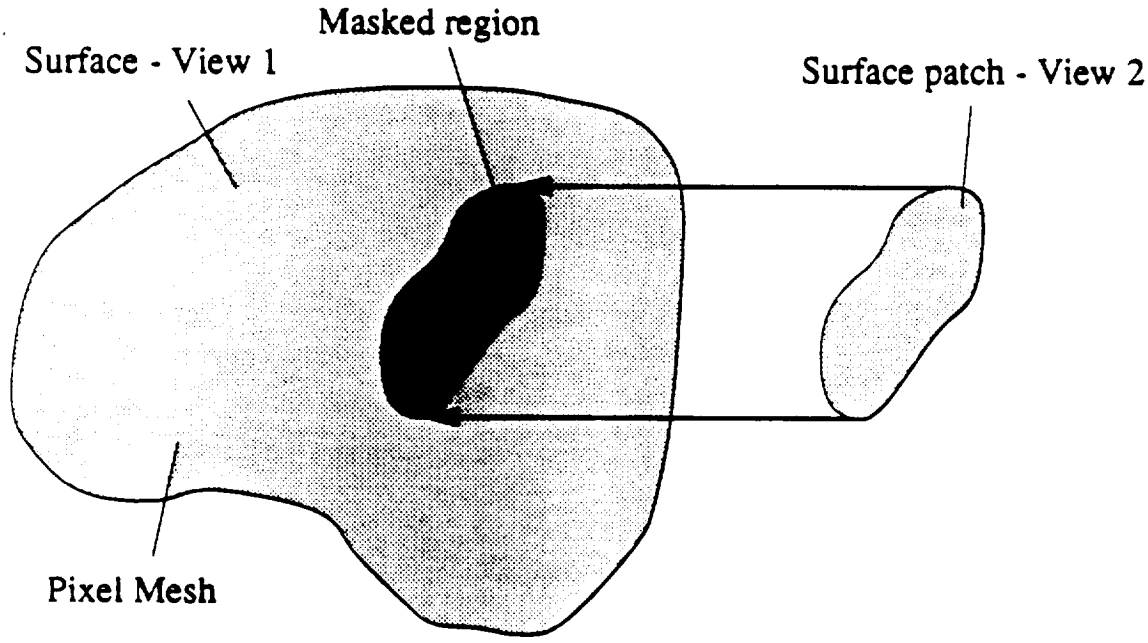


Figure 12: Pixel meshes will mask out areas that were badly captured or not visible from a particular view. They will also contain hooks into data from a different view.

To connect the pixel mesh in view 1 with other pixel meshes taken from alternate view, the boundary points of the pixel meshes of the alternate views will be mapped into view 1, and each pixel along the mapped boundary will be tied to the closest pixel point in view 1, thereby

connecting the external meshes with the mesh in view 1. This process will be repeated for all views until, the entire surface is connected.

Using this technique will allow the surfaces of an object to be recorded from arbitrary angles and to be interconnected to form one coherent surface. The technique, because it is done is piecemeal fashion, does not restrict the objects to any particular imaging geometry.

5.6 Assigning a Polygon Mesh to Surface

An alternative to a pixel mesh approximation of a surface would be to approximate the range data of the surface by a polygon faceted surface using as few polygons as possible. This can be done quite simply using one of several (non-optimal) algorithms [Faugeras et al. 1986, Schmitt et al. 1985]. Schmitt for example approximates essential a 2-D range image by a interactively generated triangular mesh. Once the polygon mesh is established, the texture associated with the pixels that belong to the surface approximated by a polygon facet would be mapped into the plane defined by the facet.

The advantage of this approach is that it would less storage to represent the surface of the object, and it would require less computation to manipulate a surface. A disadvantage is that the texture mapped into a facet is artificially distorted to fit onto the planar facets of the model and, thus may not model the texture of an object accurately. Another disadvantage is that surface texture of each facet must be separately warped to move with the facet. This would require either resampling the texture (as performed on Silicon Graphics Power Series workstation using MIP maps) or warping the surface texture for each individual polygon facet using a 2-D image warping operation.

For simplicity, the pixel-mesh approach will be implemented first.

6. Model for JPL to be Used in Tracking

In talking with the technical monitor, we found that one of goals of the VGD project should be determining a method of building models of objects that can be used in a vision database to locate and track objects. Therefore, one of the approaches that we have recommended in past reports for tracking objects requires a 3-D model of an object that can be continuously rotated, translated and scaled to provide the range and texture similar to that of the object being tracked. This model would be used in a closed-loop estimator, in the model would be continuously compared to an object to obtain an optimum match. The view parameter at the optimum match would then be used to estimate the view parameters of the object.

Two possible models are possible for this approach: one is an object-centered model and the second is a image-centered model. An object-centered model is a more common representation

in which an object is defined with respect to a coordinate origin fixed in the object, for example, a 3-D polygon-mesh representation. The image-centered model is described below.

6.1 Image-centered Model

In an image-centered model the distance to the model and its appearance are completely defined by a set of overlapping range/texture images. For example, Basri and Ullman [1988] have proposed a simple version of an image-based model (the B-U model), in which edge images are extracted for number of widely separated views of an object. As an object is tracked a system can switch to the nearest view to obtain an image of how the object should appear. The image at a particular view can be rotated, translated, and scaled within the image plane to maintain tracking. In order to reduce the number of views needed to track an object, data can be interpolated between views. In the B-U approach, for example, the local surface curvature at each edge point is extracted with the x, y, and z location of the point. Using this information, Basri and Ullman were able to accurately interpolate the appearance of edge pixels for views in between widely space sampled views.

One problem with the B-U approach, however, is that it does not deal well with the self-occlusion that occurs when an object is rotated. Edge data, initially, hidden behind a surface of the object in a sampled view, should suddenly appear as it becomes unoccluded. Data that is visible should disappear as it becomes self-occluded. The B-U algorithm fails to model this effect. As a result, a significant amount of edge data is not used in the B-U model. In addition, the B-U model does not model surface texture or range--data that could greatly improve object tracking.

An approach that models self-occlusion and carries texture and range data is the pixel-mesh model with warping described in Section 5.2. We will refer to this as a pixel-warp tracking model. In this approach, an estimate would be initially made for the view parameters of an object that is being tracked. This estimate would be used to select the sample view with the closest set of view parameters. Rotations about the z-axis, translations, scaling and tilt on the range/texture will be performed directly on range/texture images to cause the images to track a the range/texture of the real object.

The regions within the surface that are best estimated by other views will be *patched* into the image. More precisely, the patched regions will first be transformed into the current view and then transformed to matched the current view parameters (in fact, one combined transformation incorporation the transformation into the current frame and the transformation to the correct view would be applied to the pixel region) The texture and range of this surface can then be compared to those of the real object and view parameters will be adjusted to minimize the differences between these measurements.

An image-centered model of the type proposed has the advantages that the model captures the surface texture and range that would imaged by the sensors used in tracking. No unneeded

data is carried along. In addition, the surface texture provided by this model accurately portrays the actual texture of the object model.

6.2 Object-centered Model

An alternate approach will be to provide an object-centered frame model, as described in Section 5.9. Then when the surface is rotated, translated, or scaled during object tracking, only vertices of the mesh will need to be transformed to model the changing range data. However, the surface texture within each triangular facet will need to be warped to match the transformation of the model.

In the initial VGD system we will initially produce an image-based model of the object, since it is a simpler model to work with and easy for us to manipulate, given the hardware supplied by the VGD system.

7. Summary

In summary, current work on the VGD system has been to,

- Develop a sensor (i.e., the SURPHACE sensor) that can input registered range and texture data, using a phase-shift structured lighting,
- Calibrate the sensor to provide accurate measurements of the range data,
- Developing algorithms to approximately register surfaces, using surface markings,
- Determine a pixel mesh format to be used in storing the intermediate range/texture images, and allowing the data to be manipulated in refined the registration between surfaces.
- Define a model of a 3-D object which consists of pixel meshes patched together to form one coherent surface.

8. References

Basri, R. and S. Ullman, "The Alignment of Objects with Smooth Surfaces," Int. Conf. on Comp. Vis. and Pat. Recogn., 482 (1988).

Boie, R. A. and I. J. Cox, "Two Dimensional Optical Edge Recognition using Matched and Wiener Filters for Machine Vision," 1st Intl. Conf. on Comp. Vision, 450 (1987).

Catmull, E. and A. R. Smith, "3-D Transformations of Images in Scanline Order," Trans. of the ACM, SIGGRAPH, 279 (1980).

Faugeras, O. D. and M. Hebert, "The Representation, Recognition, and Locating of Three-Dimensional Objects," Intl. Jour. of Robotics Research, 5, 27 (1986).

Foley, J. D. and A. Van Dam, "Fundamentals of Interactive Computer Graphics," 506 (1982).

Gottschalk, P. G., J. G. Downward, and J. L. Turney, "VGD September-December 1989 Quarterly Report," KMS Rep. U2309 (1989).

Raab, F. H., E. B. Blood, T. O. Steiner, and H. R. Jones, "Magnetic Position and Orientation Tracking System," IEEE Trans. on Aero. and Elec. Systems, 15, 709 (1979).

Schmitt, F., H. Maitre, A. Clainchard, J. Lopez-Krahe, "Acquisition and Representation of Object Surface Data, Soc. of Photo-Optical and Instr. Eng., 602, 42 (1985).

Sine Patterns, 144 Fairport Village Landing, Fairport, N. Y. 14450-1885.

Tsai, R.Y. "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses," IEEE RA, 3, 4, 323, (1987).

Williams, L. W., "Performance-Driven Facial Animation," Trans. of the ACM, SIGGRAPH, 235 (1990).

Section H
1991 VGD Progress Report

During 1991, KMS failed to submit any progress reports. Just as the sixth progress report became due, development work on the project was suspended by KMS until it could be determined whether or not the contract would be novated. At that point, KMS notified NASA of the reason for suspending work. Following the suspension, The only authorized activities that continued past March 1991 were the preparation of "cost to completes" and preparation for a VGD project review.

Prior to project suspension, however, the project was on the verge of success. Dr. Gottschalk had demonstrated that the SURPHACE camera could be used to acquire surface profiles and developed the code necessary calibrate the CCD cameras in the SURPHACE Camera. This step is needed to remove radial lens distortion and to convert the x and y pixel locations into meaningful coordinates with respect to a camera model. We have also work out the details of how surface data can be manipulated in the VGD system. All hardware, with the exception of the array processor in multi-cpu mode was working and independant software modules had been developed which met the basic VGD project objectives.

Although Dr. Gottschalk had just been able to demonstrate that the camera worked, the results were not as accurate as he wanted. As a consequence, some time was spent in perfecting camera calibration techniques. At the point that work on the project was suspended, improved camera calibration techniques had been developed and Dr. Gottschalk was ready to start using the SURPHACE camera to acquire surface profiles.

Once this was done, the next main task was to acquire images from different angles and to merge these images together to generate a complete 3-D surface profile of the object being viewed. The last work done by Dr. Gottschalk was to develop prototype code for acquiring and merging images. This code was not tested prior to Dr. Gottschalk's departure.

Although from a technical standpoint, the project looked like it might succeed, the apparent lack of funding to complete the contract caused first one investigator (Arnold Chiu 4/29) and then the other (Dr. Gottschalk 7/10) to terminate their employment with KMS. When Arnold Chiu left, there was no time to transfer his information to Dr. Gottschalk. When Dr. Gottschalk left, he attempted to brief Dr. Jerry Turney on the status of the project. Unfortunately, Dr. Turney was activly working on another project, and all direct charge VGD work was suspended so there was no time available for him to document the work done by either Dr. Gottschalk or Arnold Chiu and when Dr. Turney was terminated in September 1991, no remaining staff in Technical Programs had any detailed technical knowledge of any of the work which had been done on the contract.

In the following sections, the work done by Dr. Turney and Dr. Gottschalk in calibrating the VGD sensor is presented. These sections were to be part of the six quarterly report which was never submitted. The text which follows is presented "as is" because the written inputs from Dr. Turney and Dr. Gottschalk for this report were never completed.

Camera Calibration

0.0 Abstract

{This section was not completed before Dr. Turney was terminated}

1.0 Background

{This section was not completed before Dr. Turney was terminated}

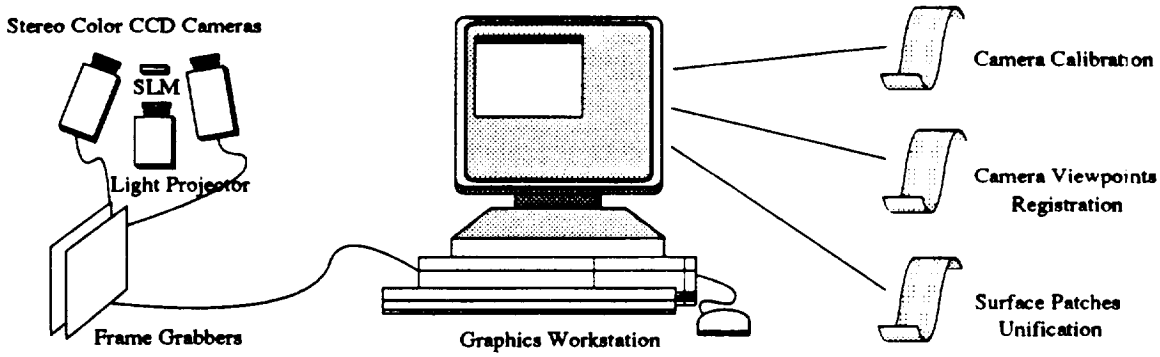


Figure. 1 The VGD Camera System in Operation.

The VGD camera was proposed to be a hand-held system for capturing arbitrary objects from multiple viewpoints. During the course of the project, the larger than anticipated weight of the system requires the VGD camera to be tripod mounted.

{MOREEEEEEEEEEE.....}
The following are Dr. Turney's notes, written to himself, which he intended to use in completing this section.

Provide background on VGD. Why do we need calibration? How does it fit in?

The camera calibration problem can be summarized as follows, similarly to Fischler & Bolles' description of the location determination problem [FiB81]:

> Given a set of "landmarks" ("control points"), whose locations are known in some world coordinate frame, determine the intrinsic and extrinsic parameters that transform each world point into an image point in the image coordinate frame.

(MOREEEEEEEEEEE.....)

The following are Dr. Turney's notes, written to himself, which he intended to use in completing this section.

Lots of people tackled this problem. List of major pitfalls. Just illustrative examples of each past methods.

In order to simplfy the camera calibration problem, previous researchers often makes use of various simplifications [Abidi & Chandra90] [Tsai87] [Faugauras & Toscani87] [Grosky & Tamburino90] [Fischler & Bolles81] [Ganapathy84] [Liu, Huang & Faugaras90] [Strat84] [Gremban, Thorpe & Kanade88], [Puget & Skordas90a-b] [Strat84]. There are a number of simplifications that generally lead to inaccuracies and imprecision in the final solution. First, R is linearized without maintaining its orthonormality. Second, no camera model (intrinsic parameters) is used. Third, intrinsic and extrinsic parameters are decoupled in their recovery. Fourth, special geomertic considerations are used in linearizing the equations, or providing restricted analytical closed-form solutions. Fifth, linear iterative methods are used after the problem is linearized with respect to incremental changes in the parameters. Sixth, R and T are decoupled and recovered independently.

[Chang & Liang89]'s approach considers time-varying camera parameters but it is based on [Tsai87]'s formulation. ([Kumar89] provides a similar critical examination.)

Decentering effect of the optical axis can be factored into R, but this would violate the orthonormality (?) of R. Fortunately, it has minimal effect.

Our algorithm does not make these assumptions. We do not need to be hand-guided [Faig75]. We maintain all constraints. We use non-linear optimization. It provides for simple single plane calibration, using coplanar sample points.

1.1 Overview

The following are Dr. Turney's notes, written to himself, which he intended to use in completing this section.Lots of people tackled this problem.

Bearing in mind the above pitfalls...

Overall VGD approach (algorithm).

MOREEEEEEEEEEE.....

This memo is divided into four major sections. Section 1.1 describes the VGD system requirements imposed on the VGD camera system in order to provide accurate 3D metrology of arbitrary objects. Section 1.2 outlines the general camera calibration problem and the mathematical formulation we have adopted. Section 1.3 summaries the pitfalls of previous approaches, the lessons on which we build the basis of our approach. Section 1.4 describes the

three new methods that we have implemented (is implementing?). Preliminary result with synthetic data are included (?). Section 1.5 discusses the related problem of viewpoint determination necessary for correlating the data from multiple arbitrary views. Section 1.6 provides a list of the cited references.

1.2 KMS Solution

MOREEEEEEEEEEE.....

The following are Dr. Turney's notes, written to himself, which he intended to use in completing this section.Lots of people tackled this problem.

Describe VGD setup with lines and user-assisted correspondence/line fitting/edge detection. The overall strategy for camera calibration is to non-linearly optimize all the extrinsic and intrinsic parameters of each individual camera.

Calibrate camera_1, calibrate camera_2, calibrate stereo baseline, calibrate projector (PHASOR Cam), related to viewpoints determination later; first two are monocular camera calibration (does not have to be coplanar data points)

Once the cameras (and projection grating) are calibrated as such, stereopsis is used to determine the base line of the VGD camera system. (Note cameras must be linearized as well. Other details such as distance between optical and camera center.)

The steps are 1) set up camera and calibration card (describe card); 2) image card and extract lines with user assistance in defining 2 points per line; 3) fit line to endpoints and sample "line" perpendicular to fit (better than edge detection [Canny8?] and then line fitting -- less fitting error to consider); 4) do non-linear optimization as described below to retrieve all extrinsic and intrinsic parameters simultaneously.

1.3 Calibration Model

Our calibration model consists of the intrinsic parameters, extrinsic parameters, camera model, optimization scheme...

MOREEEEEEEEEEE.....

1.3.1 Intrinsic Parameters

A set of physical intrinsic parameters form our sensor (camera) model. They are the focal length f, image frame center (C_x, C_y), horizontal scale factor s_x, and radial lens distortion factor κ. This simplified model of an imaging system is based on the pin-hole camera model with focal length f, while including effects from sensor sampling and lens distortion. A description of the lens optics according to thick-lens laws instead is given in [Sha89].

Both the image frame center and the horizontal scale factor are side-effects of the imperfection in a camera system's hardware timing for scanning and digitization. The horizontal scale factor s_x is actually equal to

$$s_x = f_c / f_f$$

where

f_c: pixel clock frequency of the camera
f_f: sampling frequency of the A/D-converter.

However, recovery of these frequencies [LeT88] is impractical for most applications. A rough estimate for s_x is

$$s_x = N_{cx} / N_{fx}$$

where

N_{cx}: number of sensor elements in X (scan-line) direction
N_{fx}: number of pixels in a sampled scan-line

But [LeT88] reported at least 4% difference between the above two methods. In our unified technique, we treat s_x as just another optimization parameter and recover it more accurately.

As for lens distortion, [Bro65] showed that the radial and decentering distortion of world points (x, y) into image points (x', y') can be corrected by using the equations

$$x' = x + x_m(K_1r^2 + K_2r^4 + K_3r^6 + \ldots) + [P_1(r^2 + 2x_m^2) + 2P_2x_my_m][1 + P_3r^2 + \ldots]$$
$$y' = y + y_m(K_1r^2 + K_2r^4 + K_3r^6 + \ldots) + [2P_1x_my_m + P_2(r^2 + 2x_m^2)][1 + P_3r^2 + \ldots]$$

where

$x_m = x - x_p$
$y_m = y - y_p$
$r^2 = [(x - x_p)^2 + (y - y_p)^2]^{1/2}$
y_p, x_p: coordinates of principal point
K_n: coefficients of radial distortion
P_n: coefficients of decentering distortion

[Bro71] pointed out that radial lens distortion is the predominant effect, and that the tangential and asymmetric components of decentering can be ignored. Thus, lens distortion can be approximated as a simple second-order effect in terms of r. [Tsa87] parameterized this as the radial alignment constraint (RAC) involving only one coefficient κ.

1.3.2 Extrinsic Parameters

The *extrinsic parameters* define the location determination problem, ie. determining the transformation of each point from the world coordinate frame to the image frame given known

intrinsic parameters. They are the six orientation parameters between the optical origin and the world origin. The rotation parameters can be represented in various ways, including: an orthonormal rotation matrix R of nine elements, three Euler angles (θ, ϕ, ψ), or a quarternion q. The translation parameters are represented as a vector T.

We elected to use the quarternion representation q for rotations. It is easier to enforce the unit magnitude of vector q than the orthonormality of matrix R. Moreover, the rotation singularities are located far apart at the dipoles of quarternion space, rather than being bunched together in Euler angle space clusters. A quarternion defines the direction ω and the magnitude θ of a desired rotation.

$$q = cos(\theta/2) + sin(\theta/2)\omega$$

This formula is used to construct the corresponding quarternion for an Euler angles triplet, where ω can be the inidividual unit vectors at the world origin or the optical center. q is a complex number with three imaginary parts, composed of a unit vector ω and the reals α, β, γ, δ.

$$q = \alpha + \beta\omega_x + \gamma\omega_y + \delta\omega_z$$

Its conjugate q^* is

$$q^* = \alpha - \beta\omega_x - \gamma\omega_y - \delta\omega_z$$

and its magnitude is

$$| q | = \alpha^2 + \beta^2 + \gamma^2 + \delta^2$$

The usual properties of real and complex numbers are preserved with quarternions except for commutativity of multiplication [Hor87]. Suppose we let

$$n_x^2 = n_y^2 = n_z^2 = -1$$
$$n_x n_y = n_z \quad , \quad n_y n_z = n_x \quad , \quad n_z n_x = n_y$$
$$n_y n_x = -n_z \quad , \quad n_z n_y = -n_x \quad , \quad n_x n_z = -n_y$$

Then if

$$p = p_0 + p_x n_x + p_y n_y + p_z n_z$$
$$q = q_0 + q_x n_x + q_y n_y + q_z n_z$$
$$r = r_0 + r_x n_x + r_y n_y + r_z n_z$$

we get the quarternion vector product

$$rq = (r_0 q_0 - r_x q_x - r_y q_y - r_z q_z) + n_x(r_0 q_x + r_x q_0 + r_y q_z - r_z q_y)$$
$$+ n_y(r_0 q_y - r_x q_z + r_y q_0 + r_z q_x) + n_z(r_0 q_z + r_x q_y - r_y q_x + r_z q_0)$$

We note that

$$rq \neq qr$$

The quarternion for a 3D point (p_x, p_y, p_z) is constructed as

$$p = 0 + p_x n_x + p_y n_y + p_z n_z$$

where (n_x, n_y, n_z) are the unit vectors along the xyz axes, respectively. The application of a rotation q to point p is then

$$p' = qpq^*$$

Note that $-q$ gives the same rotation

$$qpq^* = -qp-q^*$$

Moreover, the common rotation matrix R that takes p into p'

$$(p_x', p_y', p_z')^T = R\ (p_x, p_y, _z)^T$$

can be expressed in terms of unit quarternion q

$$R = \begin{matrix} q_0^2 + q_x^2 - q_y^2 - q_z^2 & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 2(q_y q_x + q_0 q_z) & q_0^2 - q_x^2 + q_y^2 - q_z^2 & 2(q_y q_z - q_0 q_x) \\ 2(q_z q_x - q_0 q_y) & 2(q_z q_y + q_0 q_x) & q_0^2 - q_x^2 - q_y^2 + q_z^2 \end{matrix}$$

[Hor87] proposed two constructions of the unit quarternion for an infinitesimal rotation $\delta\omega$. [PeW83] derived formulae for the magnitude, dot, cross, and scalar and vector triple products in quarternion space, that is, by considering only the imagainary parts of quarternions.

1.4 Camera Model

The basic geometry of the camera model is shown in Fig. 2. The imaging process is subdivided into four steps [Tsa87]:

1) Rigid body transformation from (x_w, y_w, z_w) to (x, y, z)

$$(x\ y\ z)^T = R(x_w\ y_w\ z_w)^T + T$$

2) Perspective projection with pin-hole camera geometry

$$X_u = f\ x / z$$
$$Y_u = f\ y / z$$

3) Radial lens distortion with second-order polynomial approximation

$$X = X_d = X_u (1 + \kappa r^2)^{-1}$$
$$Y = Y_d = Y_u (1 + \kappa r^2)^{-1}$$

with

$$r^2 = X_u^2 + Y_u^2$$

4) Digitization of real image coordinate (X, Y) to computer image (frame buffer) coordinate (X_f, Y_f)

$$X_f = d_x' X + C_x$$
$$Y_f = d_y X + C_y$$

with

$$d_x' = s_x d_x$$

where

(x_w, y_w, z_w): 3D coordinates of point P in 3D world corrdinate system
(x, y, z): 3D coordinates of point P in 3D camera corrdinate system
(X, Y): image coordinate system
(X, Y) or (X_d, Y_d): distorted 2D image coordinates of P
(X_f, Y_f): digitized 2D image (frame buffer) coordinates of (X, Y)
(X_u, Y_u): ideal undistorted 2D image coordinates of P
(C_x, C_y): image frame center
(d_x, d_y): distance between adjacent sensor elements in X and Y direction, respectively
(d_x', d_y): distance between frame buffer pixels in X and Y direction, respectively
s_x: horizontal scale factor
f: focal length
κ: radial lens distortion factor

d_x and d_y are available from the camera and frame grabber specifications, as are N_{cx} and N_{fx} for an initial rough estimate of s_x. We can assume zero distortion ($\kappa = 0$) and unit focal length ($f = 1$) at the beginning of the calibration procedure.

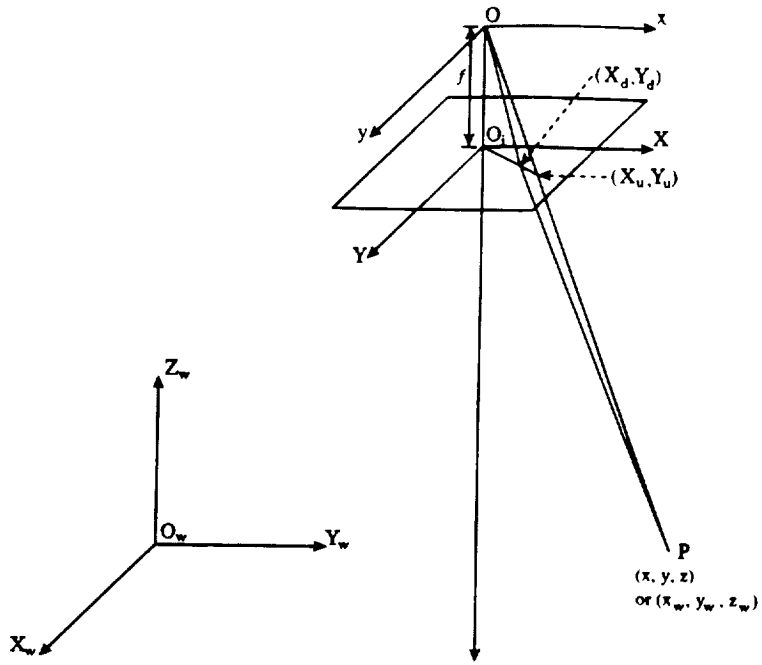


Figure 2. Pin-Hole Camera Geometry with Radial Lens Distortion

1.4.1 Monocular Camera Calibration

In order to account for the lens distortion, we need a relationship between the ideal and distorted image planes. According to the radial alignment constraint,

$$X_d / Y_d = X_u / Y_u \rightarrow X_d / Y_d - X_u / Y_u = 0$$

Using the radial distortion equations in our camera model and expressing Y_u in terms of X_u, this becomes

$$\kappa X_d (1 + Y_d^2 / X_d^2) X_u^2 - X_u + X_d = 0$$

We can solve this equation with the quadratic formula

$$x = (-b \pm \sqrt{(b^2 - 4ac)}) / 2a$$

Moreover, using L'Hopital's Rule on the quadratic solutions, as $lim\ \kappa \rightarrow 0$, we have $X_u = X_d$ and $Y_u = Y_d$ for the negative roots only. Thus, the projection of a distorted point (X_d, Y_d) onto the ideal plane (X_u, Y_u) is given by the equations

$$X_u = X_d[\{1 - \sqrt{(1 - 4\kappa(X_d^2 + Y_d^2))}\} / 2\kappa(X_d^2 + Y_d^2)]$$
$$Y_u = Y_d[\{1 - \sqrt{(1 - 4\kappa(X_d^2 + Y_d^2))}\} / 2\kappa(X_d^2 + Y_d^2)]$$

and (X_d, Y_d) is recovered from (X_f, Y_f) using the digitization equations in our camera model

$$X_d = (X_f - C_x) / s_x d_x$$
$$Y_d = (Y_f - C_y) / d_y$$

For the correct intrinsic and extrinsic parameters, a 3D line l in the camera coordinate system should lie on the projection plane M formed by its line image L in the ideal image plane (X_u, Y_u) (Fig. 3a). In other words, the perpendicular distance $|N|$ between l and M should be zero, where N is the surface normal of M. Each projection ray Q passes through an ideal image point (X_u, Y_u) at focal length f. So we have

$$Q = sU$$

where U is the unit vector along the direction $(X_u, Y_u, f)^T$. The unit normal is constructed as

$$N_i = U_i \times U_{i+1}$$

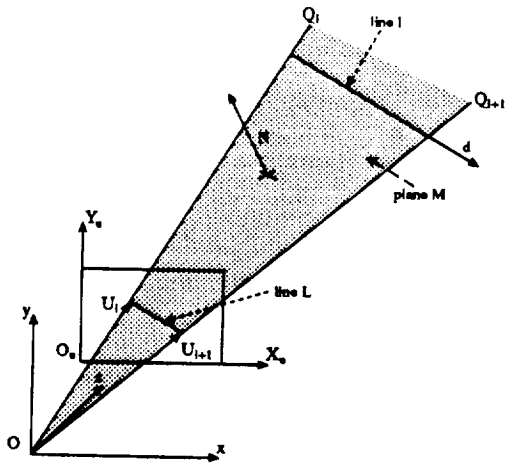


Figure 3. Projection Plane Constraint.

[LHF90] proposed to solve the location determination problem by minimizing the dot product between d, the direction vector of l, and N (Fig. 4a) using the constraint

$$N \cdot d = 0$$

However, the paramterization of l along its direction d generates a free-floating 3D vector that is not anchored at the optical center of the camera coordinate system. This is undesirable since the objective function can be minimized by merely bringing l to be parallel to M. In this case, both vectors N and d would be perpendicular but the perpendicular distance would be non-zero.

$$N \cdot d = 0 \not\Rightarrow |N| = 0$$

Thus, l may not lie on M but floats above or below it instead.

On the other hand, [Kum89] used a ray vector P going through the optical center to each 3D data point p (Fig. 4b). Since P can only be perpendicular to N when it lies on the plane M, we have

$$N \cdot P = 0 \Rightarrow |N| = 0$$

where

$$P = \mathrm{R}\, p_w + \mathrm{T}$$

so that the constraint is then

$$E = \sum_i \{ N_i \cdot ((\mathrm{R}\, p_{wi} + \mathrm{T}) / |\mathrm{R}\, p_{wi} + \mathrm{T}|) \}^2$$

where each image line L_i generates its corresponding normal N_i. The denominator is used to scale down the skewing effects of points that are farther away from the optical center, points that erroneously tend to contribute heavily to the rotation in proportion to their larger distances from the optical center.

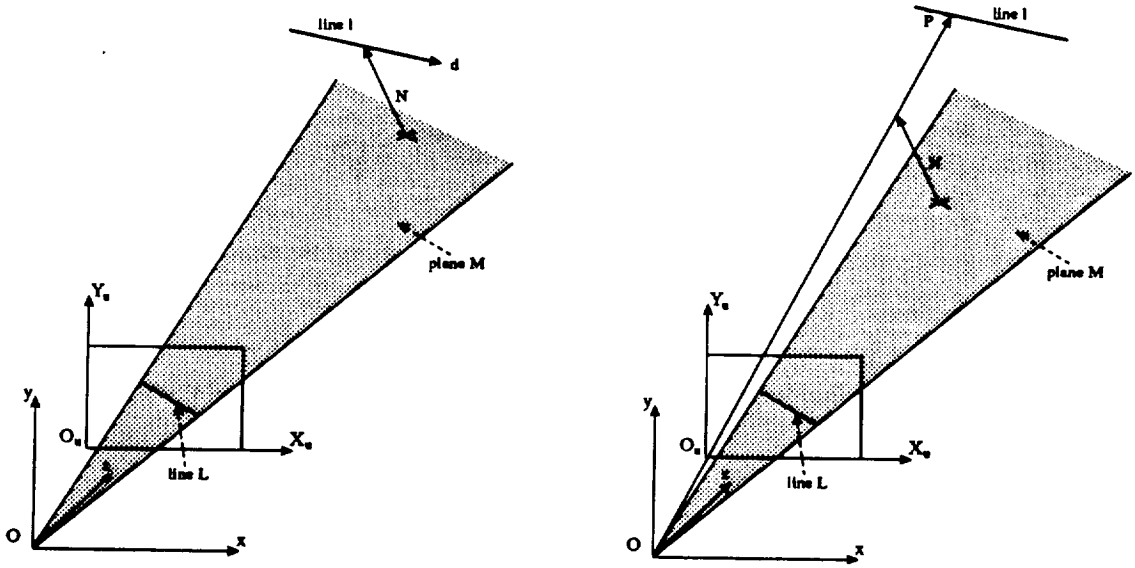


Figure 4. (a) Constraint used by [LHF90] (left) (b) Constraint used by [Kum89] (right)

In our case for camera calibration, we do not initially know the intrinsic parameters. Therefore, l would probably be a curve instead of a line in the ideal image plane and M would initially be a projection surface instead of a plane in 3D space. With radial distortion, the projection rays Q_i facet the projection surface M into multiple approximating planes M_i (Fig. 5). The extrinsic parameters basically control the rays P_i and shift them around in 3D space as the search proceeds in the multidimensional parameter space. The intrinsic parameters can be considered as warping the shape of M and shifting the projection rays Q_i and the facet normals N_i. As we get closer to recovering the camera model, this warped projection plane M flattens out correctly.

Since the projection surface M is faceted by the rays Q_i, we can recover the intrinsic parameters independently using the surface normals N_i of the facets M_i. When M is flattened out, the normals N_i would all be parallel, irrespective of the extrinsic parameters, since they are all anchored at the optical center. The cross product of two vectors is a measure of their parallelism

$$| x_1 \times x_2 | = | x_1 | | x_2 | \sin\theta$$

Each normal N_i is constructed from pairs of rays (U_i, U_{i+1}). With the associativity of cross products in mind, the projection facets constraint (PFC) for the intrinsic correction is constructed as

$$E_{PFC} = \sum_i \{ | N_i \times N_{i+2} | \}^2$$

To avoid the ill-conditioning introduced by the square root function used in computing the magnitude, we eliminate it by implementing the following equivalent objective function instead

$$E_{PFC} = \sum_i \{ u_{i,i+2,x}^2 + u_{i,i+2,y}^2 + u_{i,i+2,z}^2 \}$$

where

$$u_{i,i+2} = N_i \times N_{i+2} = (u_{i,i+2,x}, u_{i,i+2,y}, u_{i,i+2,z})$$

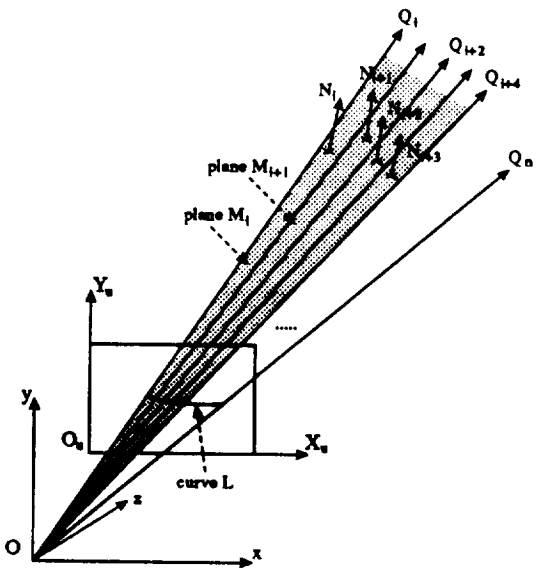


Figure 5. Distortion Correction using the Projection Facets Constraint.

Alternately, both intrinsic and extrinsic parameters can be recovered simultaneously. A plane *PPO* can be fitted to the 3D data points and the optical center (Fig. 6a). Since both triangular planes *PPO* and *M* meets at the optical center, their respective normals *PPN* and *N* would be parallel only when *PPO* lies on *M*. Using the cross product measure of parallelism and incorporating the intrinsic correction (E_{PFC}), the Plane-Plane (PP) constraint is constructed as

$$E_{PP} = \sum_i \{ \, | N_i \times N_{i+1} | + | N_i \times PPN_i | \, \}^2$$

where

$$PPN_i = P_i \times P_{i+1} / | P_i \times P_{i+1} |$$

Similarly to E_{PFC}, we implemented E_{PP} in terms of the sum of the squared magnitude components. For the PP and PFC constraints, we must make sure that the 3D data points p_{wi} and p_{wi+1} are spaced sufficiently apart. Otherwise, when they are far from the camera with large z values, the vectors P_i and P_{i+1} may effectively be parallel and the cross product value for their surface normal would be ill-conditioned.

Instead of considering the facets M_i, we can use the projection rays Q_i. Q_i can only be perpendicular to PPN when it lies on the plane PPO (Fig. 6b). Using the dot product measure of perpendicularity, the Plane-Vector constraint (PV) is constructed as

$$E_{PV} = \sum_i \{ \, PPN_i \cdot U_i \, \}^2$$

In the most general case, we deal directly with the individual 3D and image data points. The Vector-Vector (VV) constraint is constructed using the perpendicular distance between l and each Q_i (Fig. 6c). The line l is parameterized between points P_1 and P_2 along its direction d and its endpoint P_1.

$$l = rd + P_1$$

The direction of l is

$$d = P_2 / | P_2 | - P_1 / | P_1 |$$

We also know

$$Q = sU$$

Using vector analysis, the perpendicular distance between l and Q is

$$| N | = [\, -P_1 \, d \, U \,] / | d \times U |$$

However, we do not merely want to minimize the distance between arbitrary points on both line, but the distance between the endpoints of N_i, that is, p_{wi} and Q_i. Thus, we need to minimize r and s in order to get the correct minima for |N|.

Correct above to talk about scaling for skewing effects of farther away points.

v1 is ambiguous and can be any vector for t-parameterization p = v1 + t v2 (ie. r or s)

MOREEEEEEEEEE.....

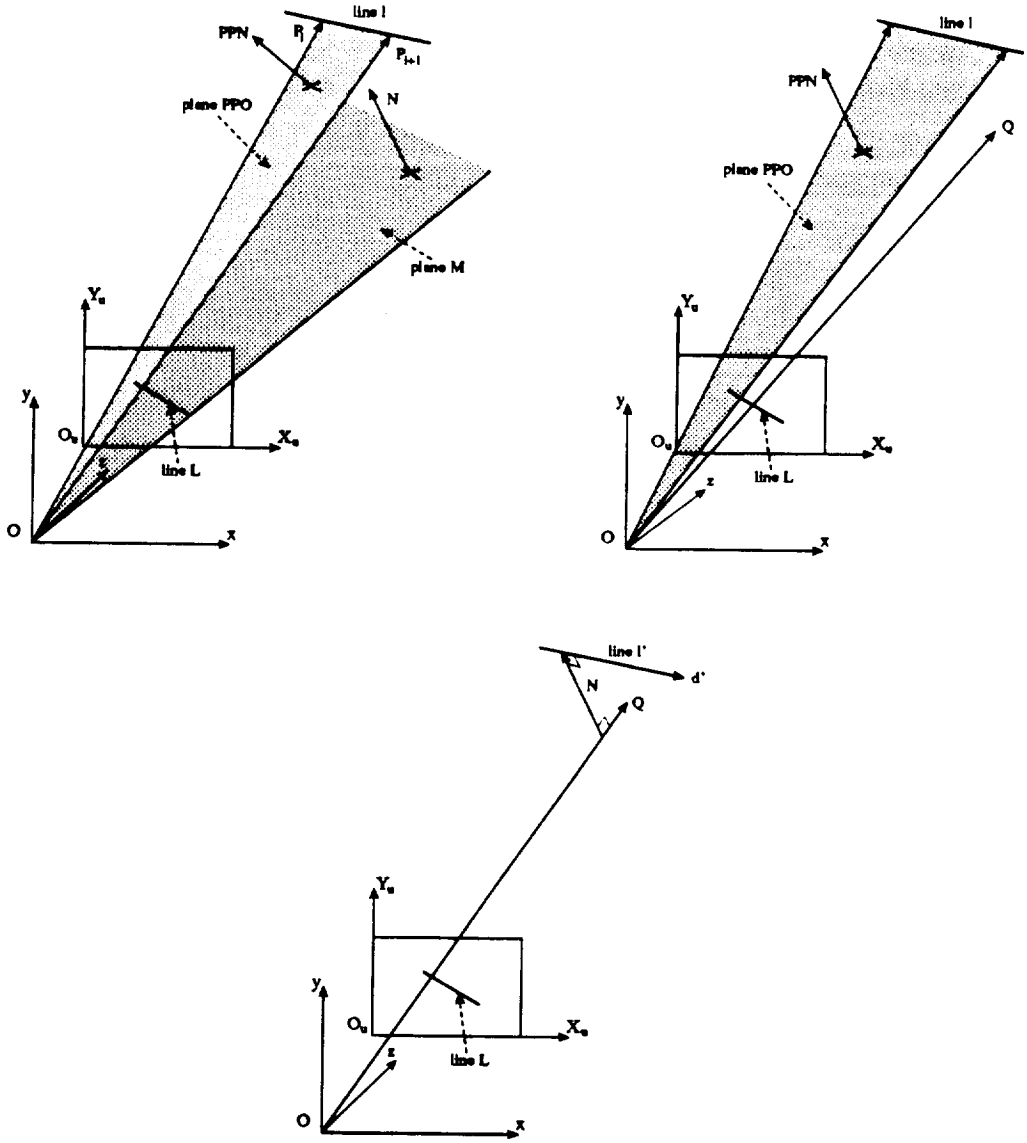


Figure 6. (a) PP Constraint (top-left) (b) PV Constraint (top-right), (c) VV Constraint (bottom)

In optimizing both the intrinsic and extrinsic parameters simultaneously, it is conceivable that the objective function can be satisfied by a feasible configuration of projection plane warping and ray vectors transformation; but not if sufficient 3D data points are provided with a calibration card such as the one shown in Fig. 6. The card has multiple sets of non-parallel intersecting lines so as to properly constrain all the degrees of optimization freedom. Moreover, it eliminates the position ambiguity of an image line *l* on the projection plane *M*.

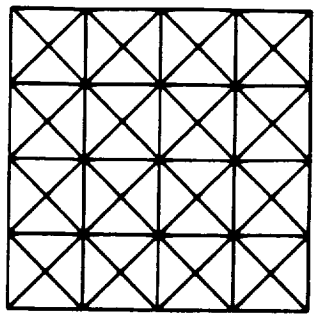


Figure 7. Calibration Pattern

The rotation matrix R is expressed as a function of quarternion q. In order to maintain the orthonormality of R, the constraint that q must be an unit vector can be expressed as

$$|q| - 1 = 0$$

[Hor88] proposed two forms of unit quarternion to iteratively ensure that q is an unit vector when constructing R. However, R "will still tend to depart somewhat from orthonormality due to numerical inaccuracies ... if many iterations are required." This is due to the incremental effects on R when adjusting the rotation through applying

$$R^{n+1} = Q^n R^n$$

To circumvent these problems, we use the standard Levenberg-Marquardt method for non-linear optimization [PFT88][Fle87]. Each estimate R^n is independently evaluated and the unit quarternion constraint can be added in as an extra term in our objective function. One possibility is to add the constraint in as a Lagrange-multiplier (eg. [Hor88]). To minimize $\rho(x)$, subject to the constraints $u(x) = 0$, this method reduces the problem to finding a stationary point of the function

$$\rho(x, \lambda)^* = \rho(x) + \lambda^T u(x)$$

However, this stationary point is neither a minimum or a maximum, so the typical minimization techniques will not work well [DaB74].

MOREEEEEEEEEE.....

The following are Dr. Turney's notes, written to himself, which he intended to use in completing this section.Lots of people tackled this problem.

how to express unit length as a vector constraint?

Instead, we enforce the constraint with a penalty function.... or barrier function

$$\rho(x, \lambda)^* = \rho(x) + k^{-1} u^T(x)\, u(x)$$

We iterate over k until q is close to an unit vector.
0 <= k(i) <= 1 for k(i+1) = k(i) + 0.1; or k=10,100,1000,...

1.4.2 Stereo Camera Calibration

The stereo camera calibration problem is very similar to the monocular case. We can use the same non-linear optimization idea to recover the relative orientation [Hor88] between the cameras. Alternately, since we have recovered the extrinsic parameters for each camera, then with some user assistance in matching corresponding landmarks in the two image frames, we can simply modify the R's and T's from monocular calibration as our solution.

1.4.3 Projection Camera Calibration

(see section by PGG2) -{This section was never written}

1.4.4 Viewpoint Registration

There are two approaches to viewpoint registration. First, we can use a tracking device such as the Polhemus 3DSPACE Tracker to monitior the position of the VGD camera system at the multiple viewpoints. Second, as in the stereo case, a geometric solution can be generated after some assistance by the user in solving the correspondence problem between the image frames.

1.5 Experimental Results

The following are Dr. Turney's notes, written to himself, which he intended to use in completing this section.Lots of people tackled this problem.

show results

compare synthetic data with real data!

Provide analysis?

Minimum number of lines consideration. Contrast [Kumar89] and [Horn87:book] arguments. [Tsai87] has ad hoc number of 60 sample points. Statistical considerations? Error analysis. Measurement analysis. Derive parameters from one card, and project expected lines onto a different card. No sub-pixel accuracy. In sythetic case, different set of data. Calibrate to a card. Different R and T, and try to recover that again. Show simulation error analysis, accuracy, and precision. Comparison with others?

1.6 References

[AbA73] Y.I. Abdel-Aziz, "Lens Distortion and Close Range Camera Calibration", Photogram. Eng., pp.610, 1973.

[AbC90] M.A. Abidi & T. Chandra, "Pose Estimation for Camera Calibration and Landmark Tracking," IEEE ICRA, pp.420, 1990.

[Bro65] D.C. Brown, "Decentering Distortion of Lenses," Photogram. Eng., pp. 444. 1965.

[Bro71] D.C. Brown, "Close-Range Camera Calibration," Photogram. Eng., pp.854, 1971.

[DaB74] G. Dahlquist & A. Bjorck, Numerical Methods, Englewood Cliffs, NJ: Prentice-Hall, 1974.

[FaT86] O.D. Faugeras & G. Toscani, "The Calibration Problem for Stereo," IEEE ICVIP, pp.15, 1986.

[FiB81] M.A. Fischler & R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," ACM CACM, 24, 6, pp.381, 1981.

[Fle87] R. Fletcher, Practical Methods of Optimization, New York, NY: Wiley & Sons, 1987.

[Gan84] S. Ganapathy, "Decomposition of Transformation Matrices for Robot Vision," IEEE ICRA, pp.130, 1984.

[GrT90] W.I. Grosky & L.A. Tamburino, "A Unified Approach to the Linear Camera Calibration Problem," IEEE PAMI, 12, 7, pp.663, 1990.

[GTK88] K.D. Gremban, C.E. Thorpe, & T. Kanade, "Geometric Camera Calibration using Systems of Linear Equations," DARPA Image Understanding Workshop, pp.820, 1988.

[Hor87] B.K.P. Horn, "Closed-form Solution of Absolute Orientation using Unit Quarternions," J. Opt. Soc. Am. A, 4, 4, pp.629, 1987.

[Hor88] B.K.P. Horn, "Relative Orientation," DARPA Image Understanding Workshop, pp.826, 1988.

[Kum89] R. Kumar, "Determination of Camera LOcation and Orientation," DARPA Image Understanding Workshop, pp.870, 1989.

[LeT88] R.K. Lenz & R.Y. Tsai, "Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3D Machine Vision Metrology," IEEE PAMI, 10, 5, pp.713, 1988.

[LHF90] Y. Liu, T.S. Huang, & O.D. Faugeras, "Determination of Camera Location from 2D to 3D Line and Point Correspondences," IEEE PAMI, 12, 1, pp.28, 1990.

[PeW83] E. Pervin & J.A. Webb, "Quarternions in Computer Vision and Robotics," IEEE ICVIP, pp.382, 1983.

[PuS90] P. Puget & T. Skordas, "An Optimal Solution for Mobile Camera Calibration," IEEE ICRA, pp.34, 1990.

[PFT88] W.H. Press, B.P. Flannery, S.A. Teukolsky, & W.T. Vetterling, Numerical Recipes in C, New York, NY: Cambridge, 1988.

[Sha89] S.A. Shafer, "Geometric Camera Calibration for Machine Vision Systems," Manuf. Eng., pp.85, Mar. 1989.

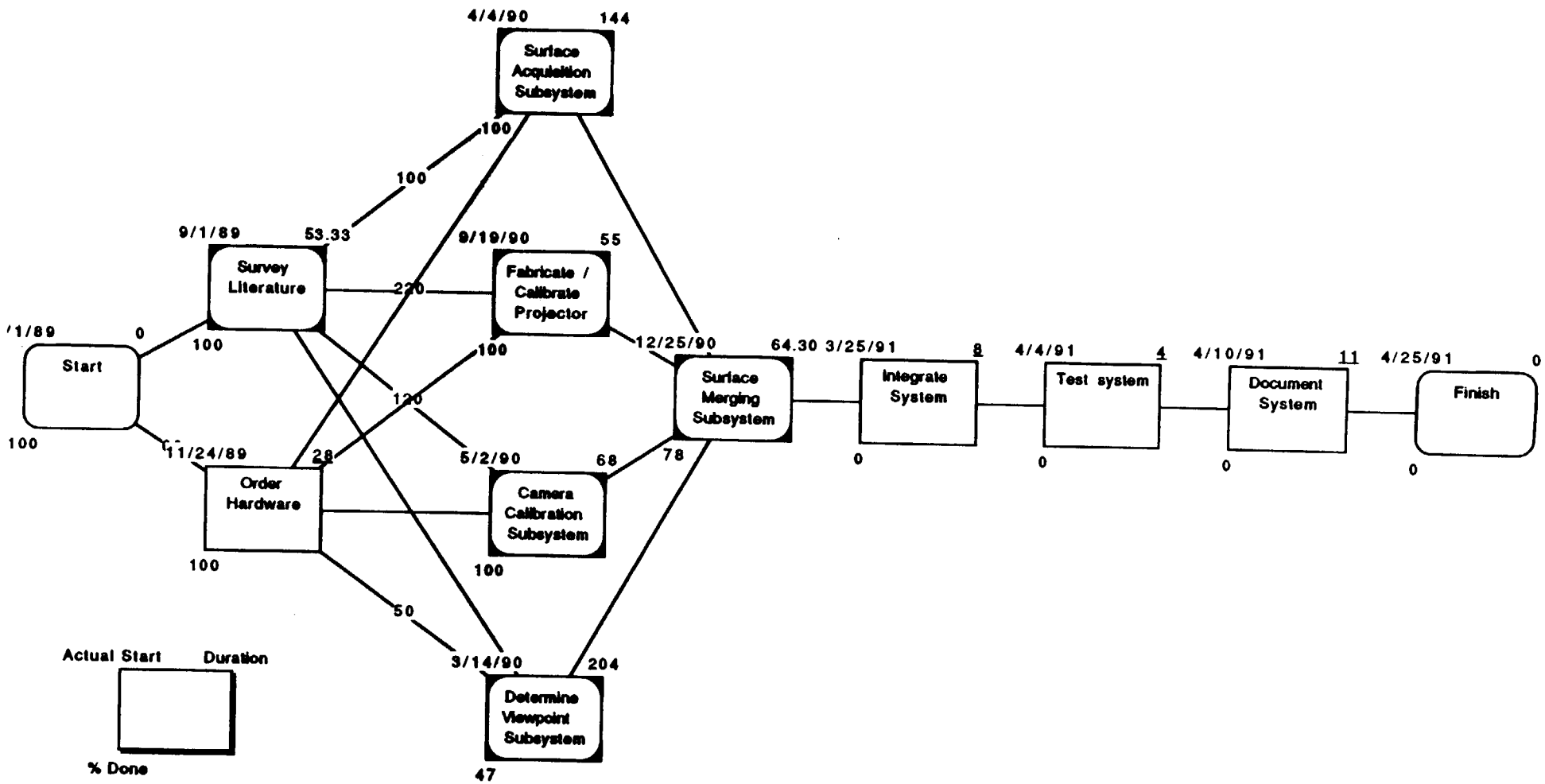
[Str84] T.M. Strat, "Recovering the Camera Parameters from a Transformation Matrix," DARPA Image Understanding Workshop, pp.264, 1984.

[Tsa87] R.Y. Tsai, "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses," IEEE RA, 3, 4, pp.323, 1987.

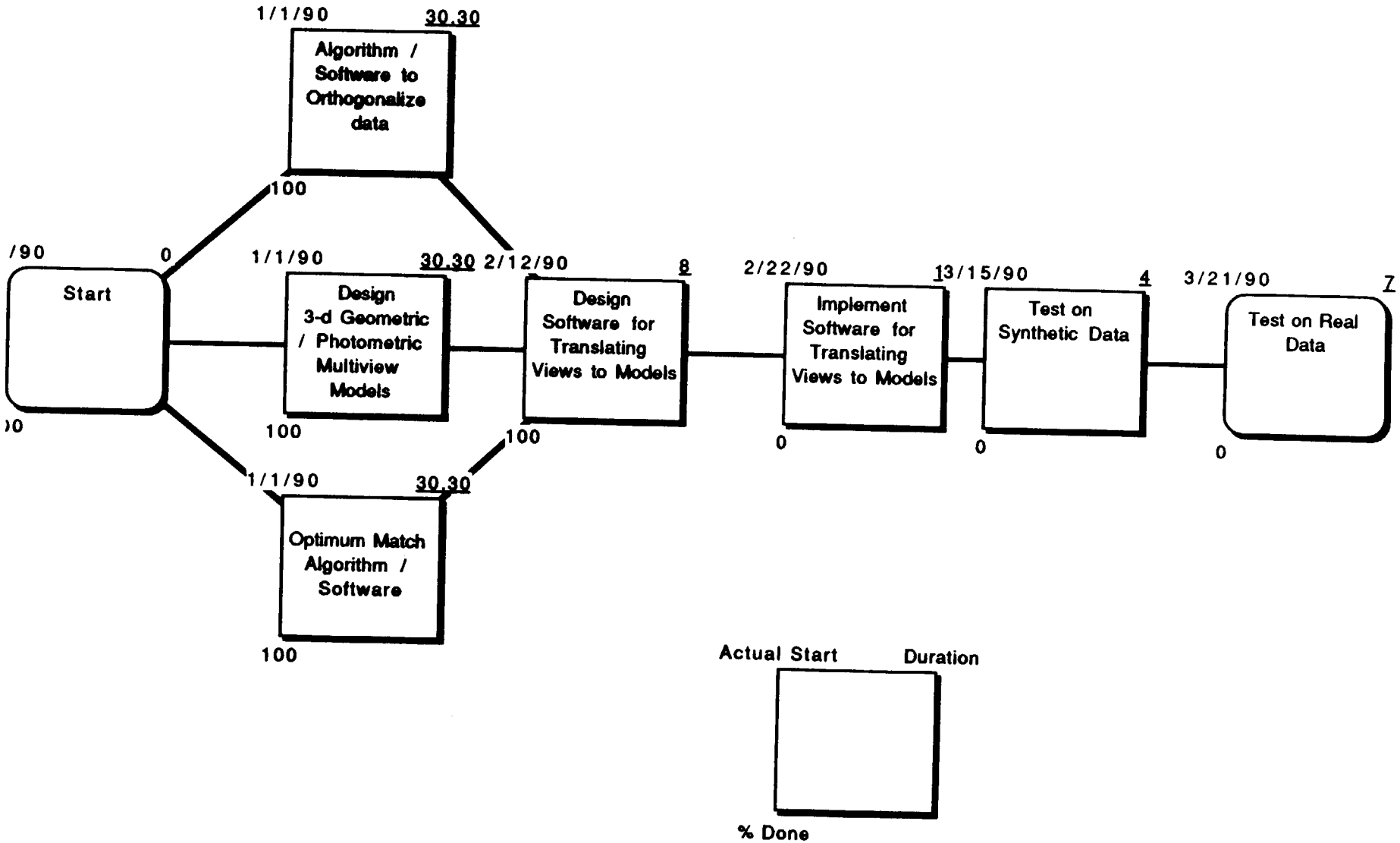
Section I
Detailed VGD Project Plans

On the following pages, the detailed flow of the work performed on the VGD project is schematically represented in MACproject format. Since the contract task numbers I - VI incorporate multiple elements of the task boxes shown in the activity breakdown, following the activity breakdown drawings are included drawings showing how the project tasks map onto the VGD sub-task matrix.

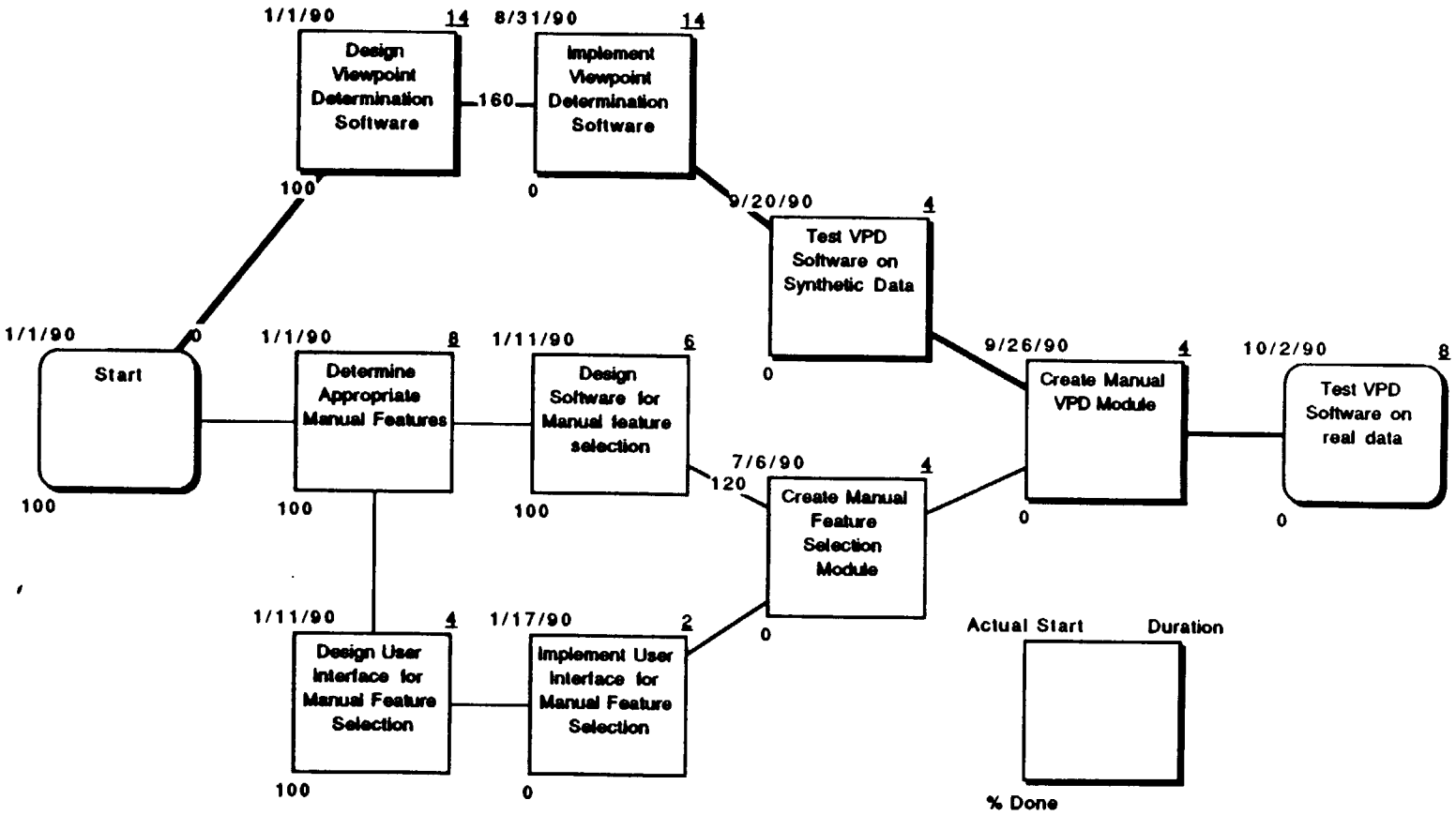
VGD Project Plan as of May 14, 1991



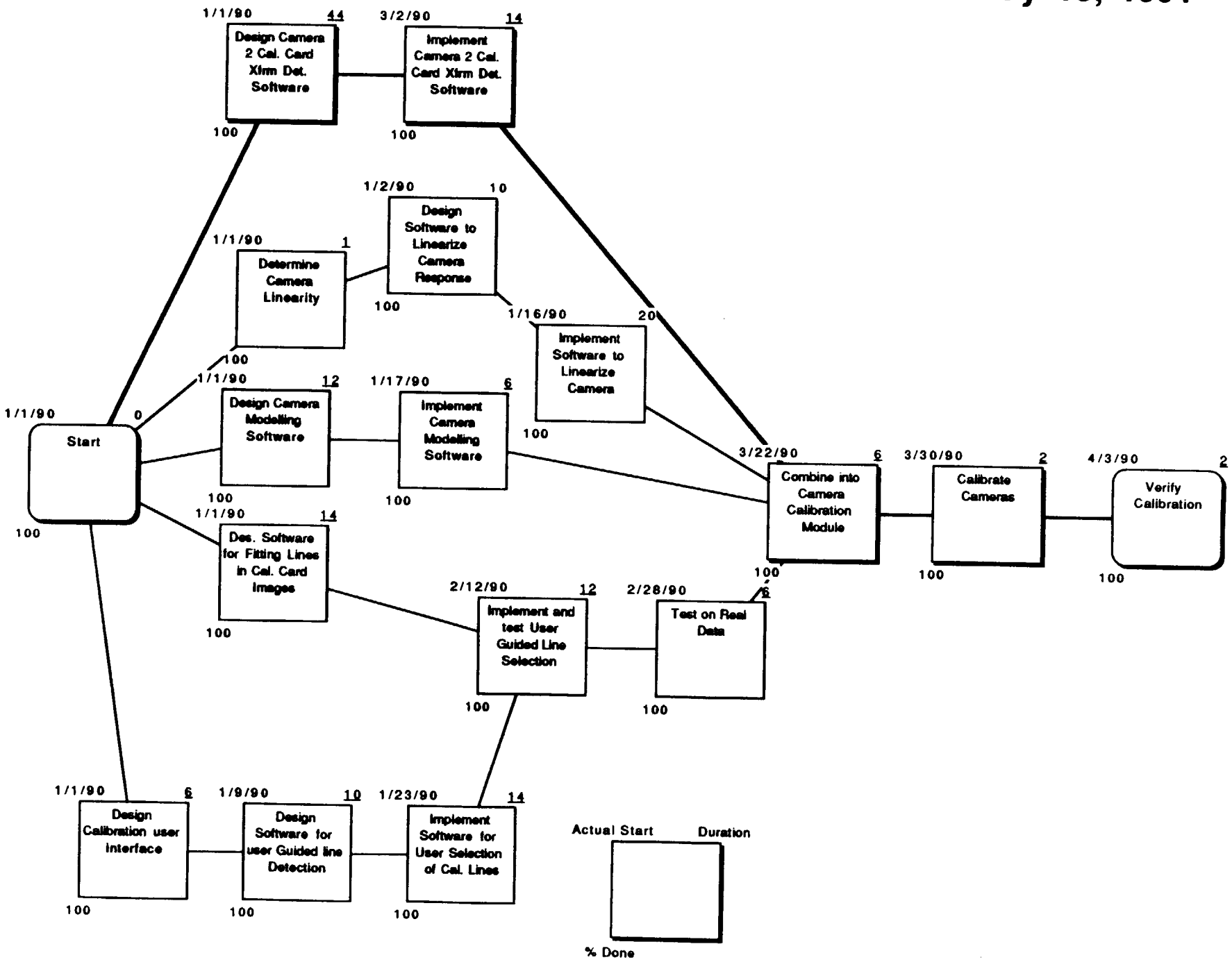
Surface Merge Task as of February 15, 1991



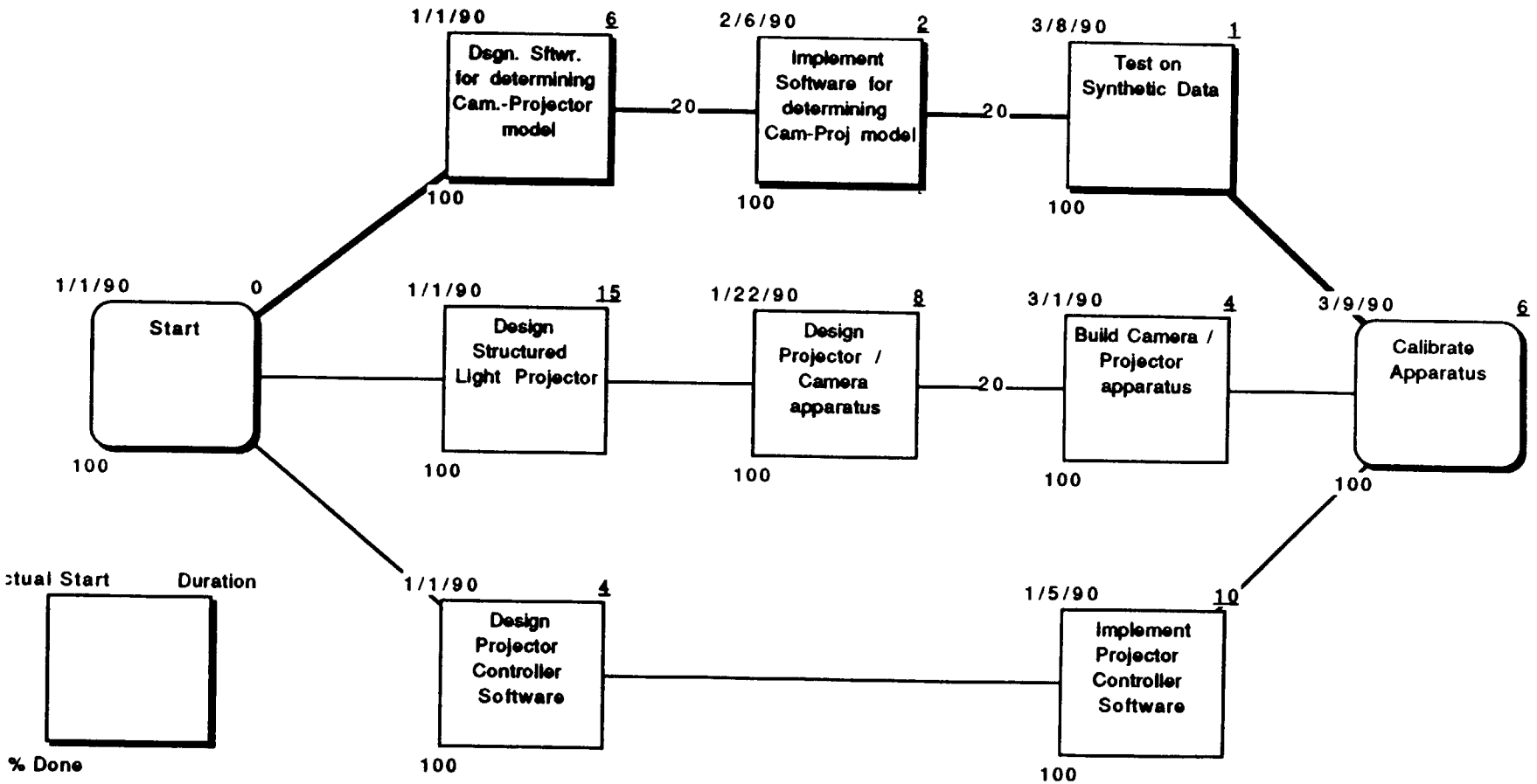
Viewpoint Determination Task as of February 15, 1991



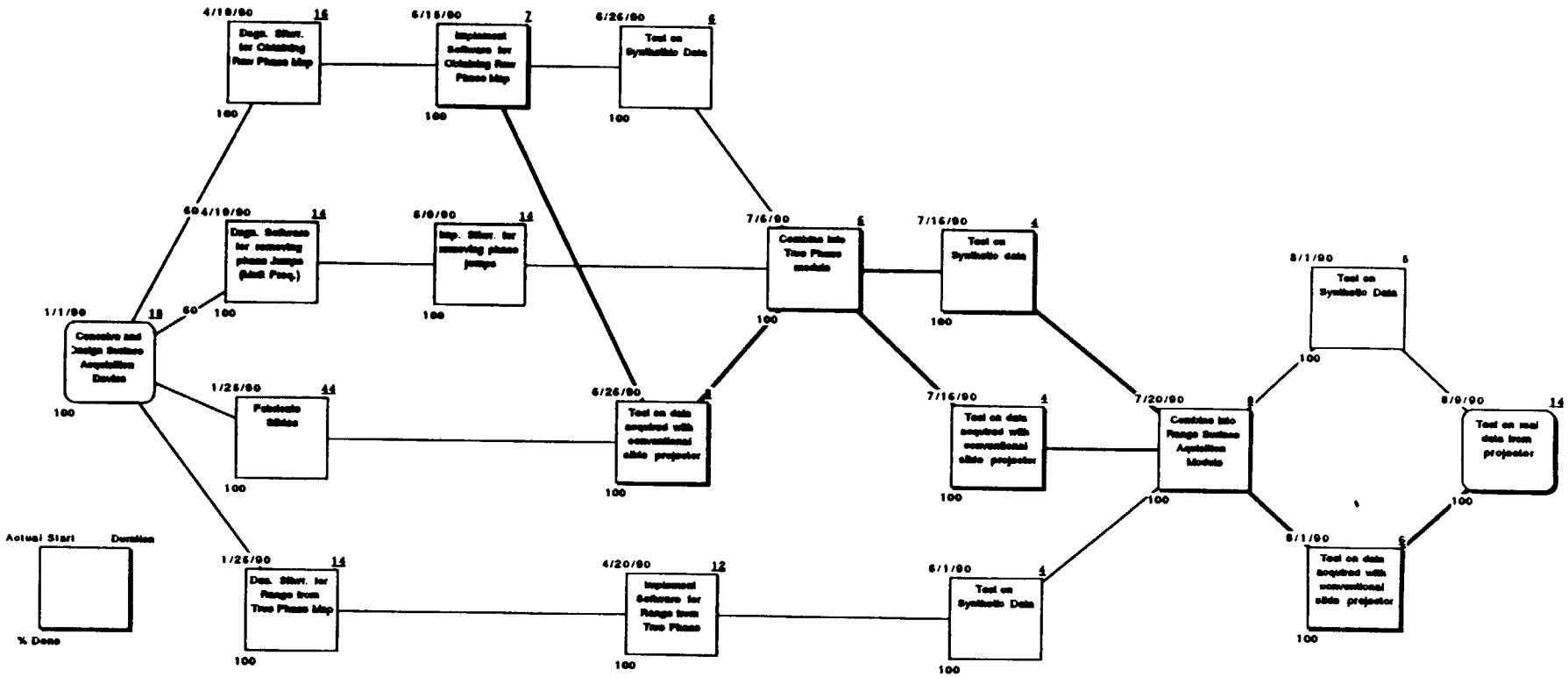
Camera Calibration Task as of May 13, 1991



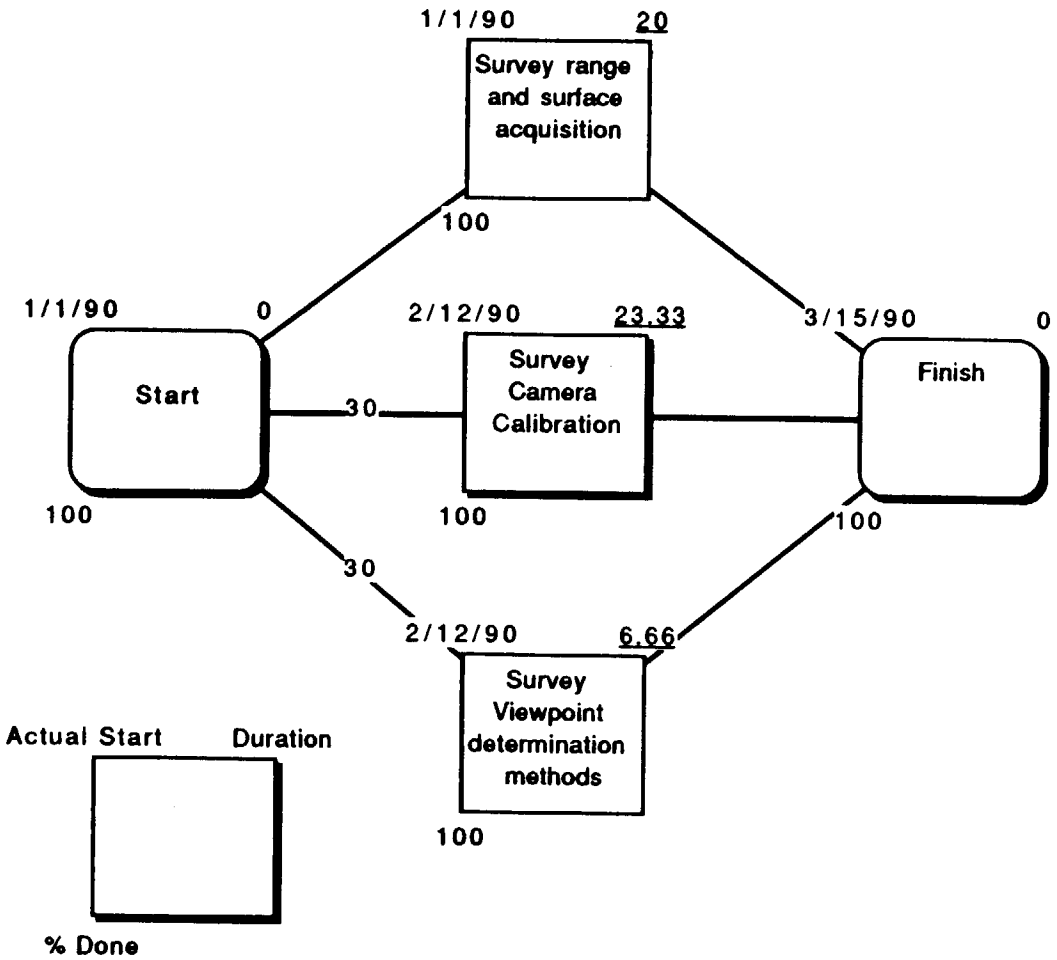
Fabrication/Calibration Task as of May 13, 1991



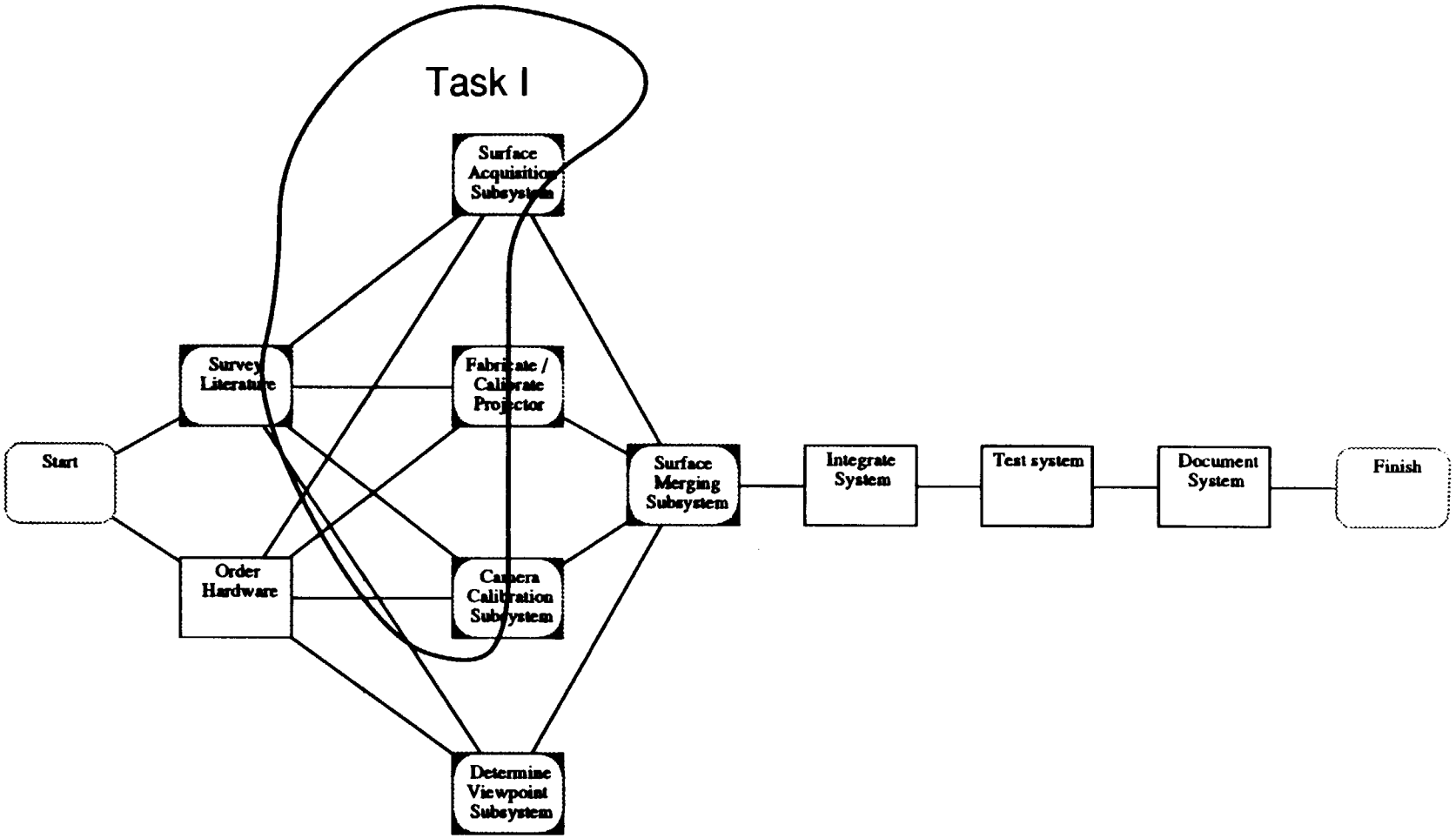
Acquire Surfaces Task as of May 14,



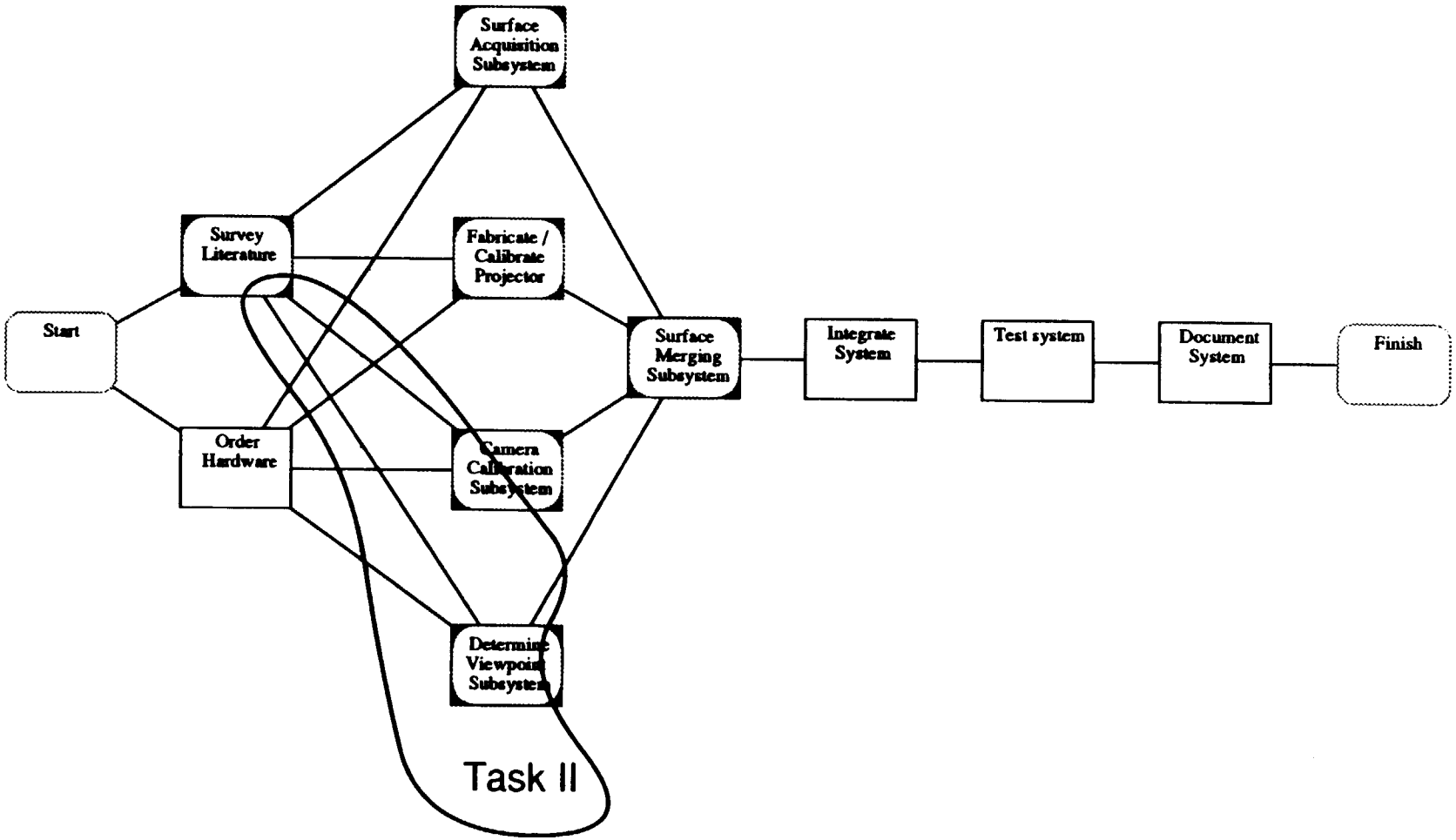
Literature Survey Task as of February 15, 1991



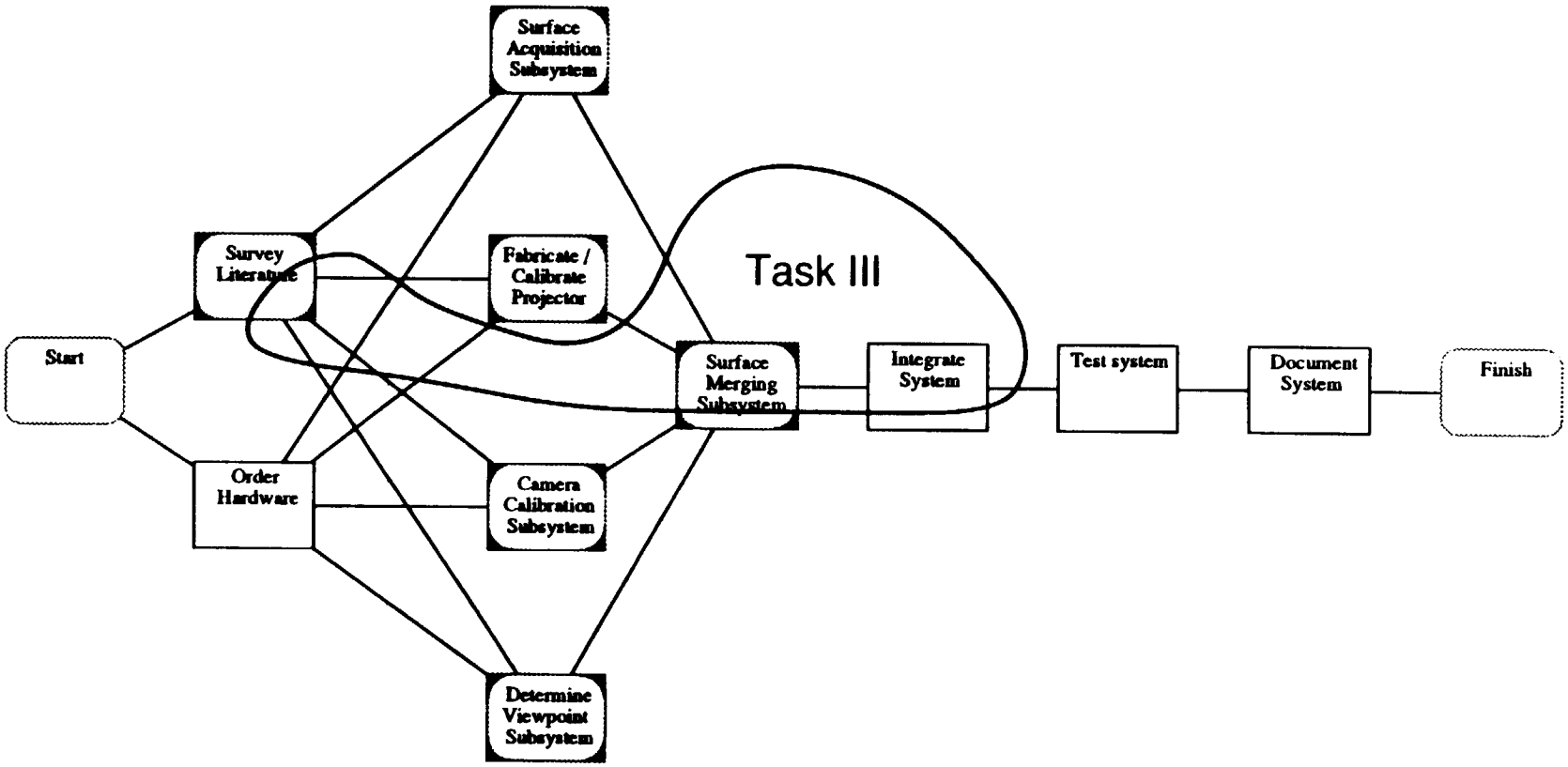
Task I: Implement Software for Stereo Reconstruction



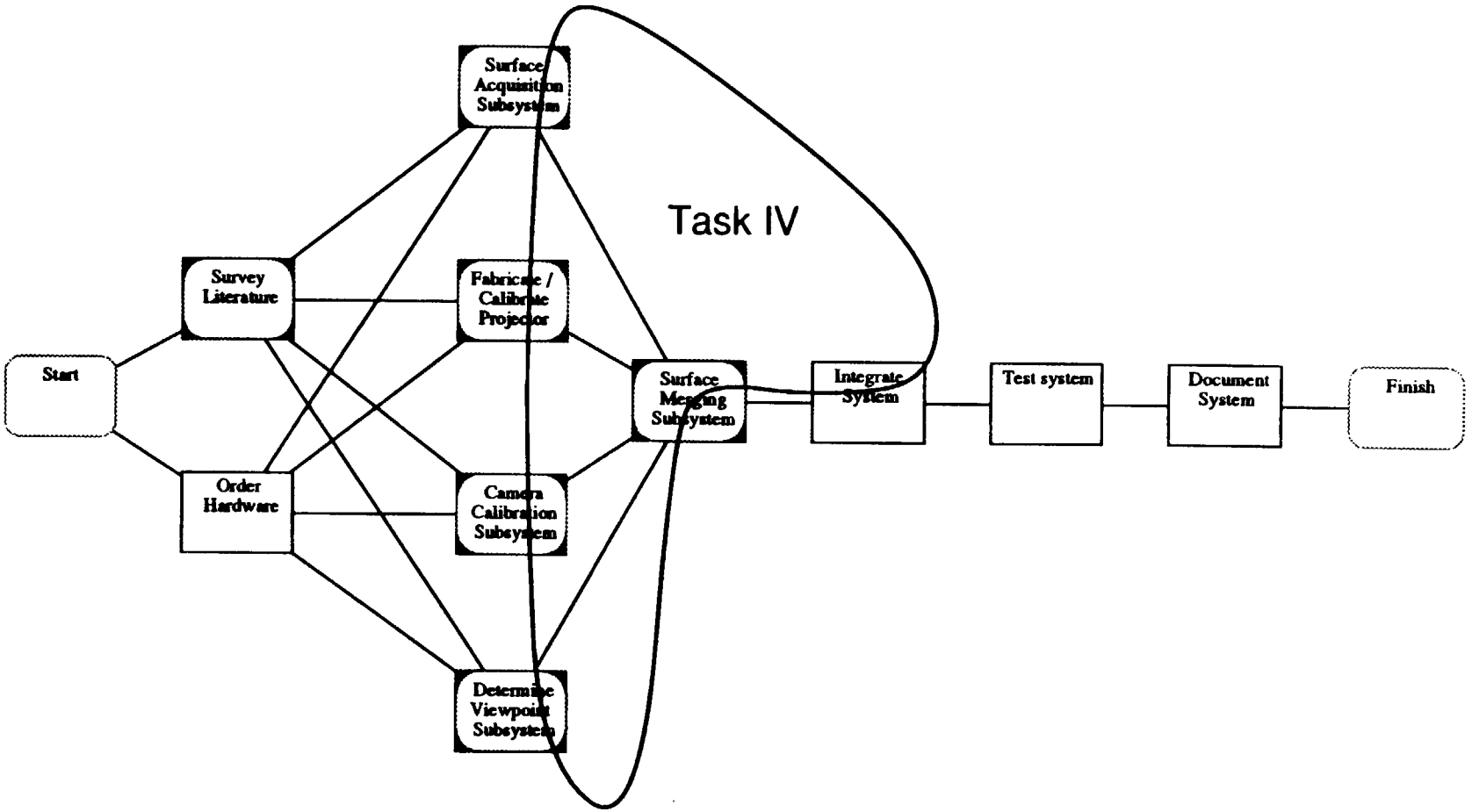
Task II: Implement Software for Viewpoint Determination



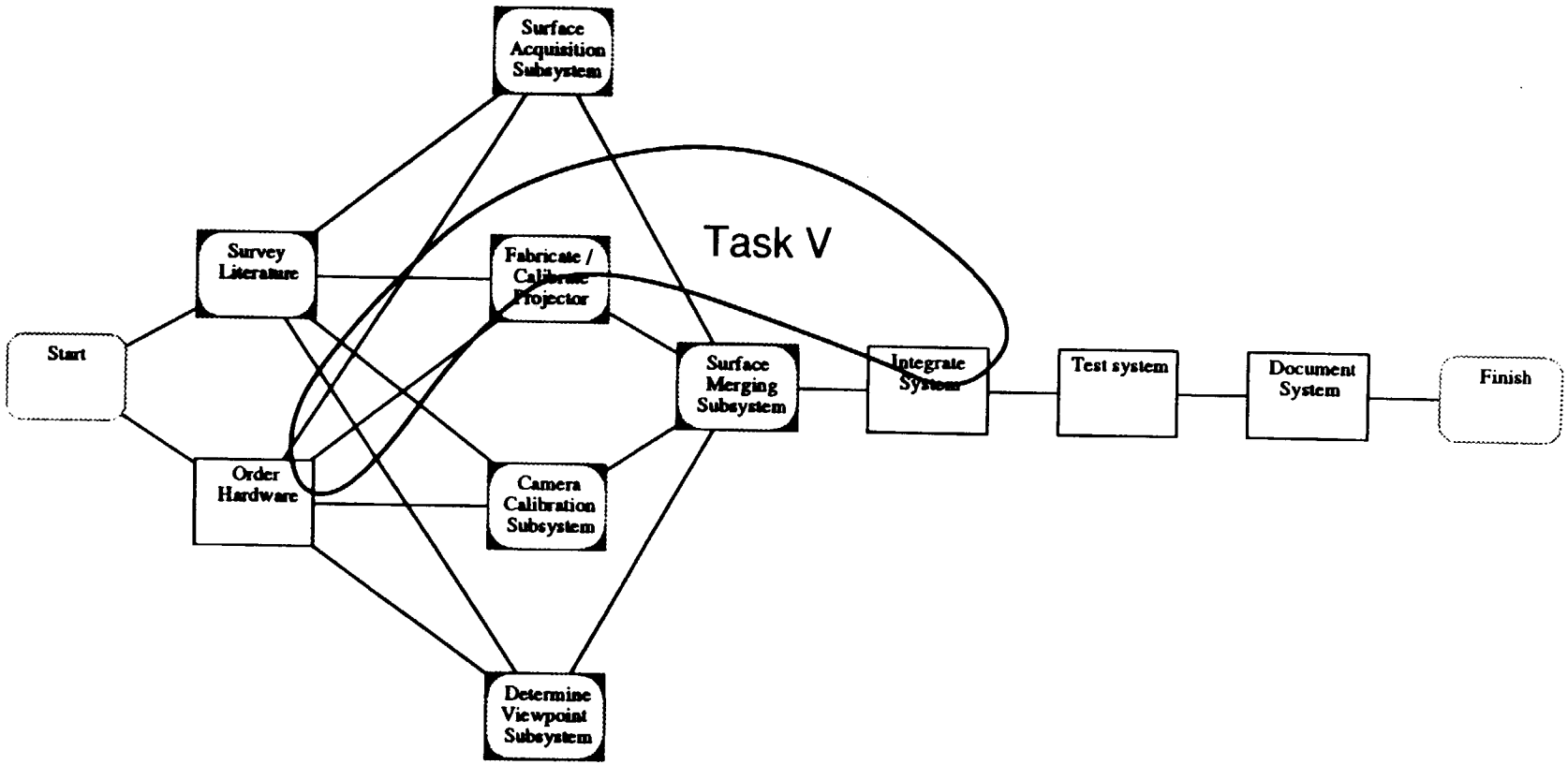
Task III: Implement Software for Storing and Manipulating Surface Data



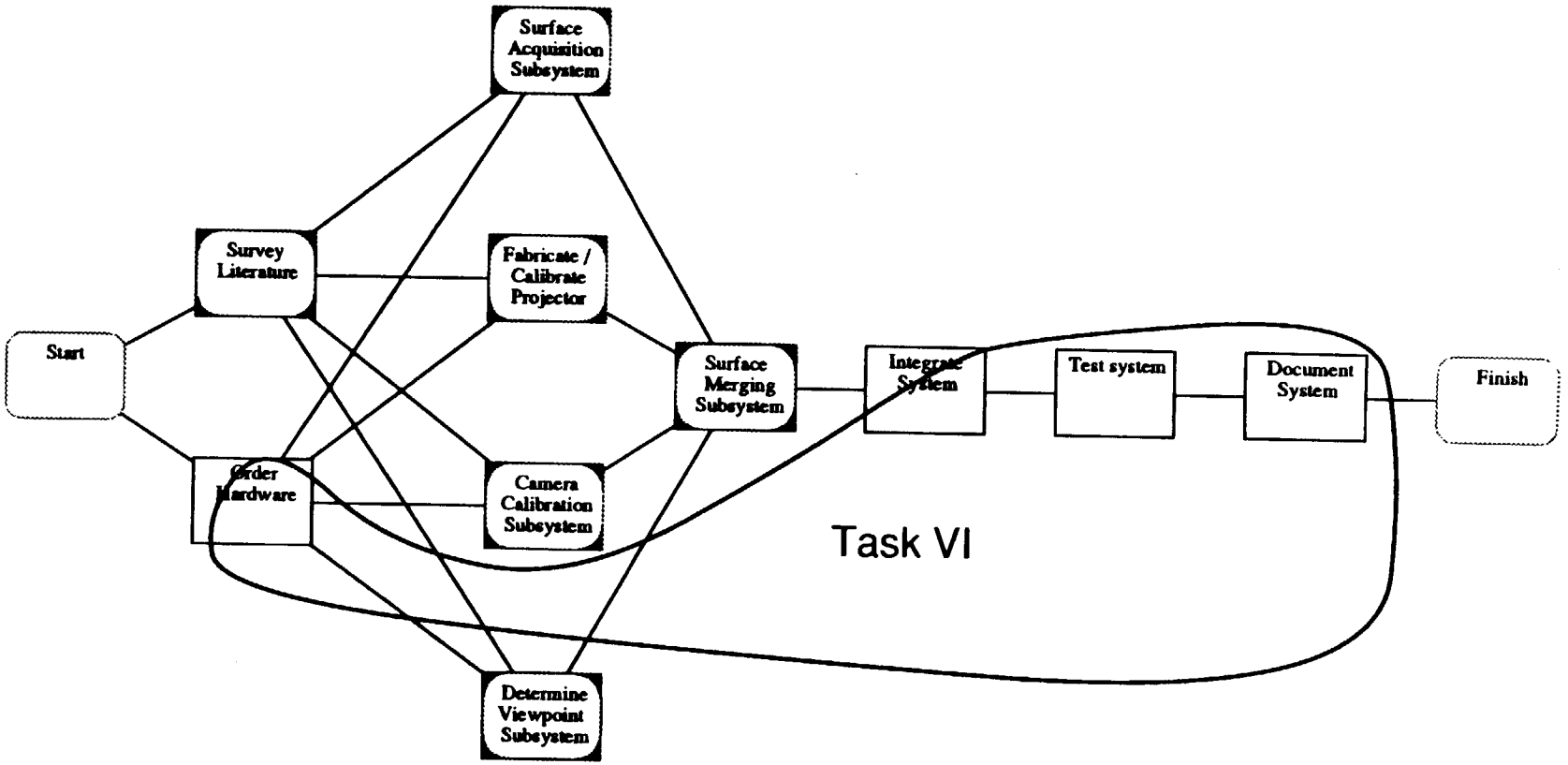
Task IV: Design and Implement Operator Interface



Task V: Design and Build Stereo Camera System



Task VI: Procure, Configure, Maintain, Test, and Document the VGD System Hardware; Test and Document System Software



Section J
February 1991 Progress Review Viewgraphs

The following pages contain the viewgraphs which were presented at KMS's February showing that a functional VGD system could be delivered to NASA given the hours remaining if

a) the contract was novated to KMS Advanced Products, Inc. and
b) the projected 1991 rates did not increase further.

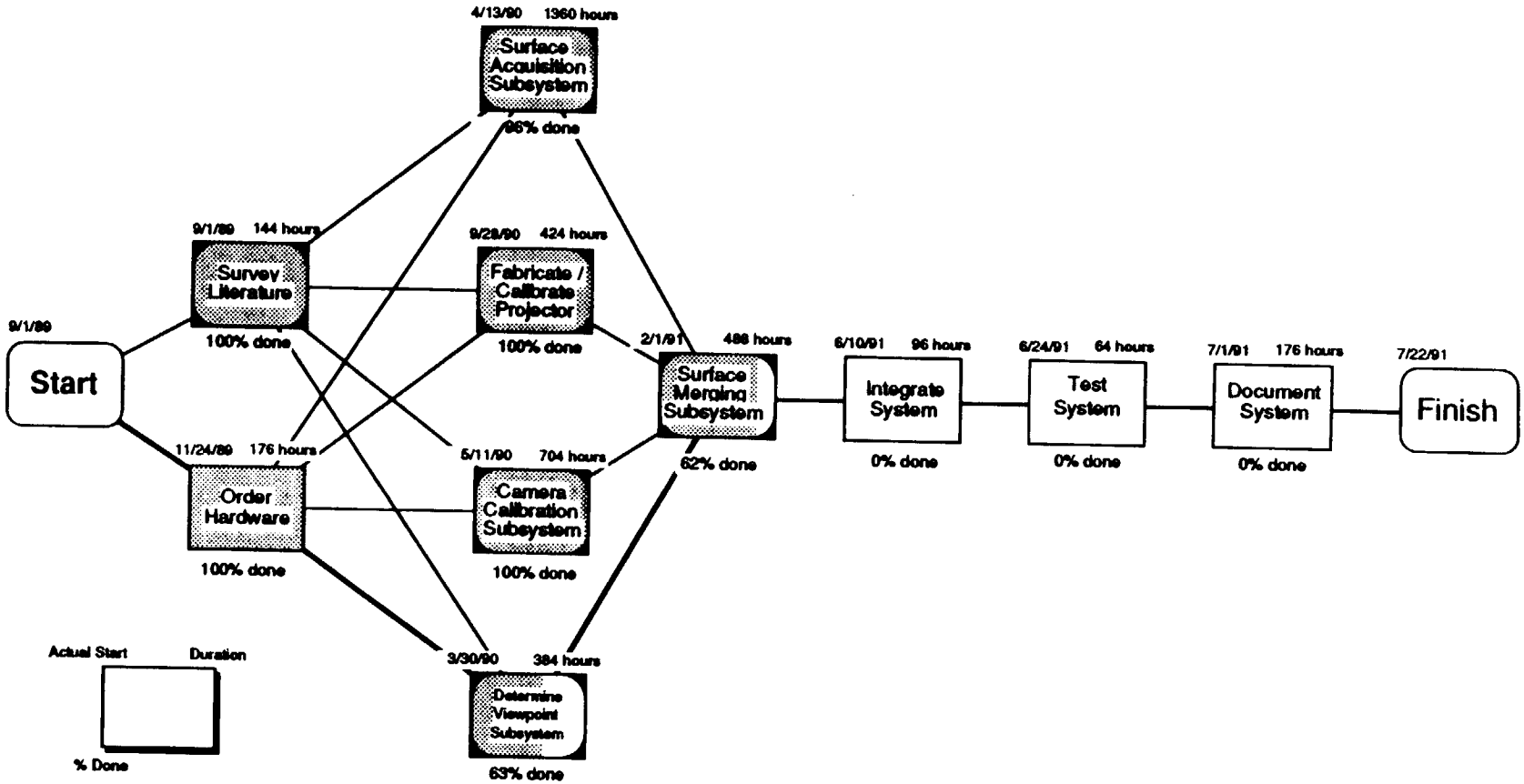
However, he did emphasize that certain convenience features which KMS wanted to incorporate in the system would have to be omitted because KMS's 1991 projected rates (assuming novation occured) indicated that there were too few hours remaining on the project.

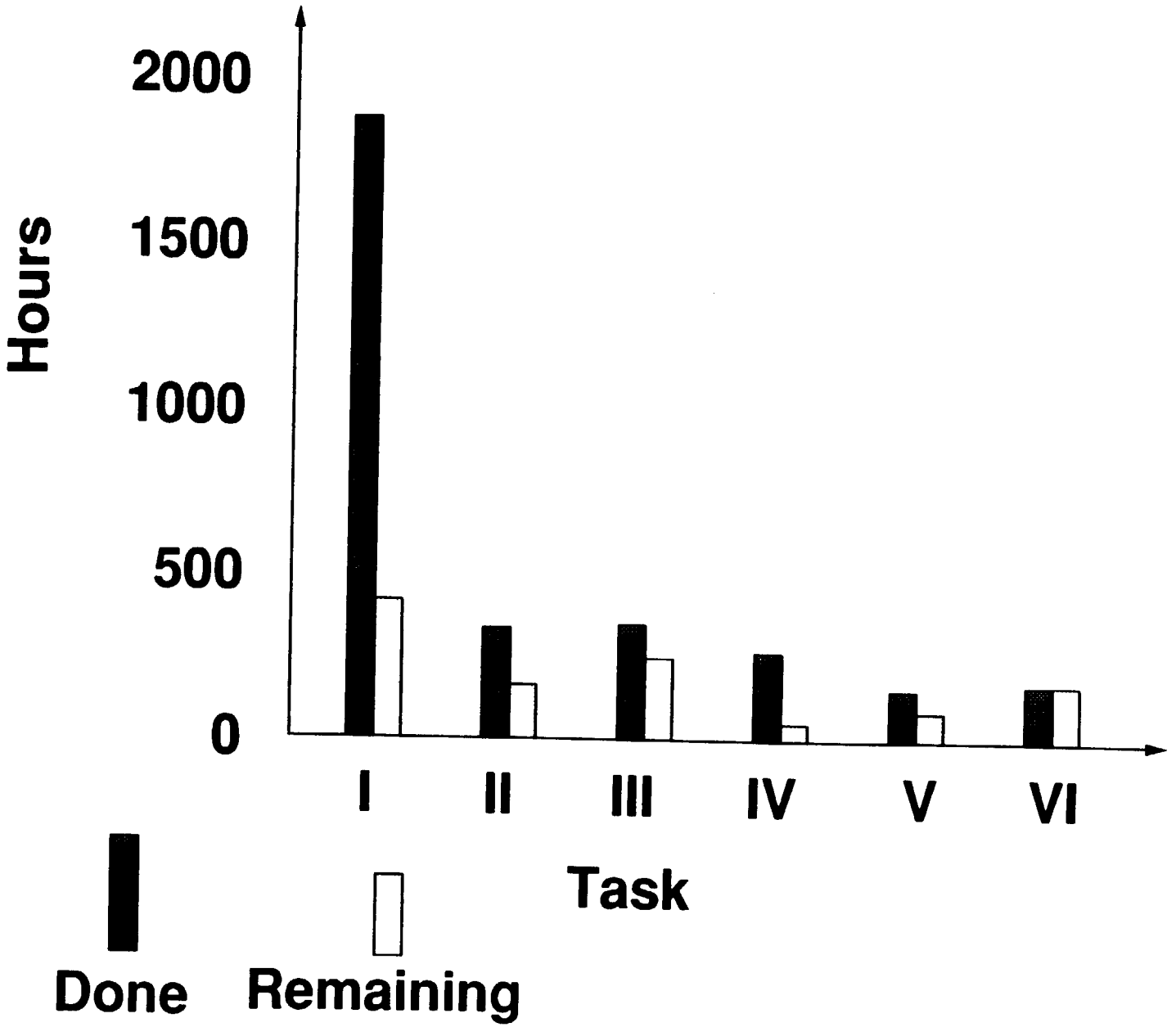
The primary feature in which KMS would be forced to deliver less functionality than we would like to was in the area of the user interface for VGD. Given the remaining budget, only a command line interface could be provided. However, all proposed functionality would be provided.

It should also be noted that as available hours decreased, the number of hour available for documenting the system had decreased.

At the end of the January review, Dr. Gottschalk and Arnold Chiu continued to work on various VGD tasks.

VGD Project Review
February 15, 1991
Hours and State of Completion




2000
1500
1000
500
0
Hours
I
II
III
IV
V
VI
Task
Done
Remaining

Cost To Complete

Hours spent: 3032
Hours remaining: 1104

Estimated hours to complete: 960
Budget slack: 144 hours

Section K
May 1991 Project Review Viewgraphs

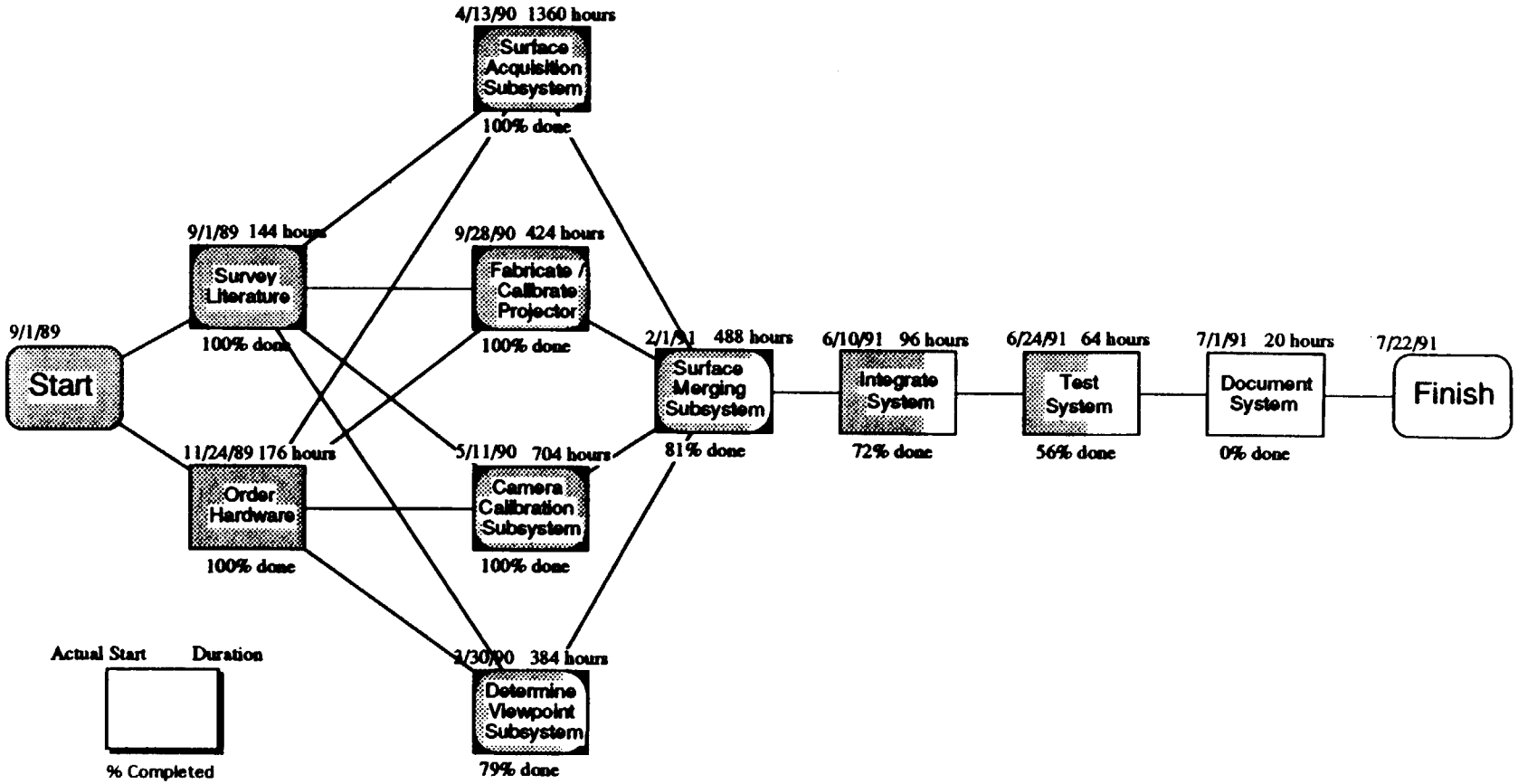
From January through March Dr. Gottschalk and Arnold Chiu worked feverously to try and accomplish the required tasks within the available labor budget. However at the end of March, KMS had suspended all work on the VGD contract because it was determined that contract funds were overspent if the contract was not novated from KMS Fusion to KMS Advanced Products.

The reason for this is that in order to try and shield the R&D contracts from the huge labor overhead rates associated with closing down the DOE contract (in excess of several thousand percent), the R&D staff working on the VGD project had been transferred from KMS Fusion, Inc. to KMS Advanced Products, Inc. As a consequence, if the contract stayed in KMS Fusion, Inc. an additional intercompany G&A cross charge would have to be applied.

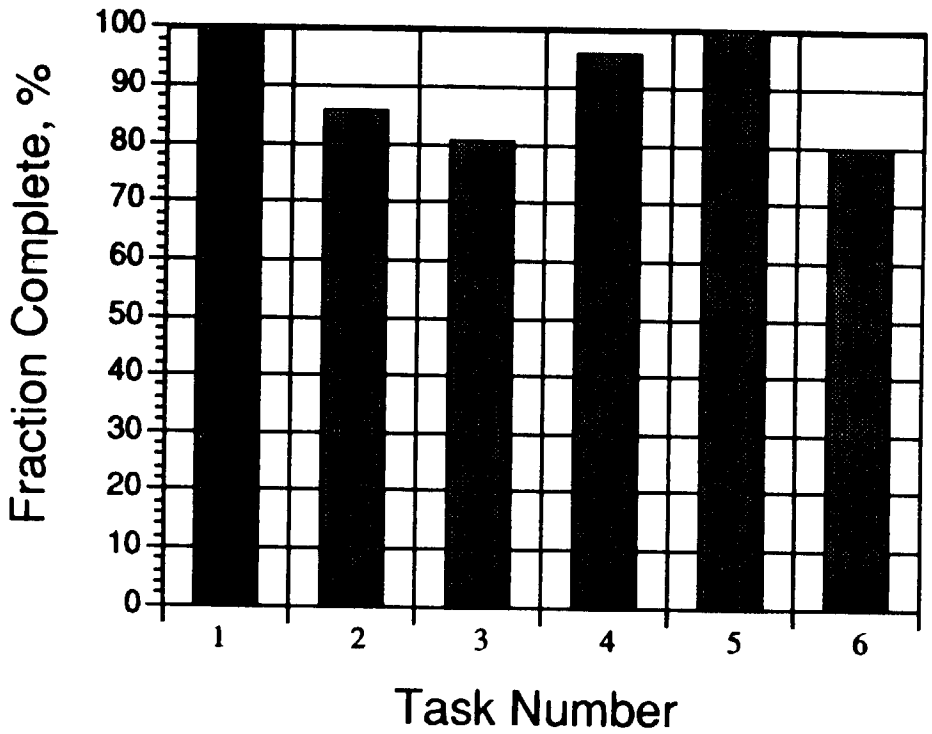
Shortly thereafter the suspension of work on VGD, Arnold Chiu terminated his employment with KMS on one day's notice. There was no time to transfer his work to Dr. Gottschalk.

In May, KMS held a project status review for VGD. The following pages contain the viewgraphs presented by Dr. Gottschalk at that status review. Please note that the project activity charts he presented show that by mid-May nearly all project tasks have been completed with the notible exceptions of integrating the software components into the final system and of documenting what had been done. At this point, Dr. Gottschalk was the only person knowledgeable as to the status of the project and what needed to be done.

VGD Project Review
Overall Project Status by Task
May 14, 1991



Fraction Complete by Task



Overall: 94% Complete

Cost to Complete

Hours Spent:	3865
Hours Remaining:	258
Hours to Complete:	247
Slack:	11

Section L
Delivered VGD Hardware

The hardware delivered with VGD includes:

- 1 Color VAXstation 3520 with 16 mbytes of memory, keyboard, mouse, manuals, cables, etc.
  Assorted cables and power cords
- 1 Expansion cabinet with cables.
- 1 Analogic/CDA Array Processor
- 1 SURPHACE camera assembly including
  - 2 miniature cameras and power supplies
  - 1 illumination system
  - 1 illumination slides
- 2 Data translation digitizer boards

Section M
Delivered VGD Software

The software developed on the VGD contract is on the directory DKA0:[VGD] on the VGD system. To access this software, the system manager for the system should log into the system account (SYSTEM) with the password VGDSYS. The system password should then be changed as required by site security policy.

The system is delivered with the following software licenses and distribution media.

1	VAX/VMS Operating System V5.4-2 License (VMS V5.4-2 VMSU2054, LATU2054 on TK50)
1	DVNETEND License for a VAXstation 3540
1	PHIGS Runtime License for a VAXstation 3540
1	VAXcluster License for a VAXstation 3540
1	DW-Motiv License for a VAXstation 3540
1	CDA Array processor software and library (MicroMSP Driver and Libraries V3.1 on TK50) (MicroMSP manual supplied)
1	Data Translation digitizer software driver (SP00231 V1.02 DT-IRIS tape on TK50) (Manual supplied)

The VGD software is on directory DKA0:[VGD]. Please note that the files on the [.MAC_FILES] directory structure are in Pacer_Link directory structure format which we believe to be compatible with the format used by either PacerShare, Alisa/Macintosh S/W, or DEC's Pathworks.

The files contained on this directory structure include:

```
Directory DKA0:[VGD]

CHIU.DIR;1          GOTTSCHALK.DIR;1    MAC_FILES.DIR;1

Total of 3 files.

Directory DKA0:[VGD.CHIU]

CALIB.DIR;1         HEAD.DIR;1          TEAPOT.DIR;1

Total of 3 files.

Directory DKA0:[VGD.CHIU.CALIB]

CALIB.C;31          CALIB.C;30          CALIB.EXE;39        CALIB.JOU;1
CALIB.OBJ;39        CALIB1.C;2          CLINK.COM;1         CMAKE.COM;2
DNLS1.DOC;1         DNLS1.FOR;2         DNLS1.OBJ;1         DUM.LIS;13
DUM.LIS;12          MACH.FOR;1          MACH.OBJ;1          TESTDNLS1.C;11
```

```
TESTDNLS1.OBJ;1      XERROR.C;2          XERROR.OBJ;1

Total of 19 files.

Directory DKA0:[VGD.CHIU.HEAD]

DSP.C;5              DSP.EXE;2           DSP.OBJ;3           MESH.DAT;1
PIC.DAT;1            README.;1

Total of 6 files.

Directory DKA0:[VGD.CHIU.TEAPOT]

BLEND.C;1            BLEND_VECTOR.C;1    BLEND_VECTOR.OBJ;1  COMPUTE_VIEW.C;1
COMPUTE_VIEW.OBJ;1   CURVE.C;1                               DISPLAY_CURVE.C;1
DISPLAY_CURVE.OBJ;1
DISPLAY_PATCH.C;1    DISPLAY_PATCH.OBJ;1 DUM.LIS;1           GLOBAL.H;1
INIT.C;2             INIT.C;1            INIT.OBJ;1          MAKEFILE.MMS;3
MAKEFILE.MMS;2       PATCH.C;1           PATCH.DAT;2         PATCH.DAT;1
ROTATE_3.C;1         ROTATE_3.OBJ;1      SIM_STER.H;1        SIM_STEREO.C;2
SIM_STEREO.C;1       SIM_STEREO.EXE;2    SIM_STEREO.H;1      SIM_STEREO.OBJ;2
SIM_STEREO_GLOB.H;1 SIM_STEREO_PROTO.H;1                     SURFACE.DAT;1
S_ERROR.C;1          S_ERROR.OBJ;1       TEAPOT.DAT;1        TEAPOT.EXE;1
TRANSLATE_3.C;1      TRANSLATE_3.OBJ;1   TRANS_3.C;1         VERTEX.DAT;2
VERTEX.DAT;1         VIEW.C;1            VP_TRANS.C;1        VP_TRANSFORM.C;1
VP_TRANSFORM.OBJ;1

Total of 44 files.

Directory DKA0:[VGD.GOTTSCHALK]

ACQUIRE.DIR;1        CALIBRATION.DIR;1   DT.DIR;1            LINES.DIR;1
PHASE.DIR;1          Q1FIGS.DIR;1        QUICKIES.DIR;1      QUICKRANGE.DIR;1
RANGEDATA.DIR;1      RAWPHASE.DIR;1      REPORT_GRAPHICS.DIR;1
SINESLIDE.DIR;1      SLIDEDATA.DIR;1

Total of 13 files.

Directory DKA0:[VGD.GOTTSCHALK.ACQUIRE]

ACQDISPLAYIMAGE.C;4  ACQDOACQUIRECOMMAND.C;24
ACQDOACQUIRECOMMAND.OBJ;12
ACQDOACQUIRECOMMAND.OBJ;11          ACQDOCENTERCOMMAND.C;3
ACQDOCENTERCOMMAND.OBJ;4            ACQDOCENTERCOMMAND.OBJ;3
ACQDONUMBERTOAVERAGECOMMAND.C;8     ACQDONUMBERTOAVERAGECOMMAND.OBJ;3
ACQDONUMBERTOAVERAGECOMMAND.OBJ;2   ACQDOSAVECOMMAND.C;17
ACQDOSAVECOMMAND.OBJ;6              ACQDOSAVECOMMAND.OBJ;5
ACQDOSETFRAMEGRABBERCOMMAND.C;3     ACQDOSETFRAMEGRABBERCOMMAND.OBJ;5
ACQDOSETFRAMEGRABBERCOMMAND.OBJ;4   ACQDOSWCCOMMAND.C;8
ACQDOSWCCOMMAND.OBJ;3
ACQDOSWCCOMMAND.OBJ;2               ACQDOSWSCOMMAND.C;8
ACQDOSWSCOMMAND.OBJ;3
ACQDOSWSCOMMAND.OBJ;2               ACQERROR.C;2        ACQERROR.OBJ;2
ACQERROR.OBJ;1                      ACQHANDLECOMMAND.C;9
ACQHANDLECOMMAND.OBJ;3
```

```
ACQHANDLECOMMAND.OBJ;2                    ACQUIRE.C;22          ACQUIRE.EXE;3
ACQUIRE.EXE;2        ACQUIRE.OBJ;5        ACQUIRE.OBJ;4         ACQUIRE.OPT;11
ACQUIRE_PROTOS.H;18 ACQUIRE_PROTOS.H;17 ACQUPDATEPADSIZE.C;10
ACQUPDATEPADSIZE.OBJ;3                    ACQUPDATEPADSIZE.OBJ;2
DO.COM;5             MAKEFILE.;55         TAPE.TIFF;1

Total of 42 files.

Directory DKA0:[VGD.GOTTSCHALK.CALIBRATION]

GEO.DIR;1            PHOTO.DIR;1

Total of 2 files.

Directory DKA0:[VGD.GOTTSCHALK.CALIBRATION.GEO]

GEO.EXE;11           GEO.H;47              GEO.OPT;13           GEOTEST.DIR;1
LSE_ERROR.LOG;1

Total of 5 files.

Directory DKA0:[VGD.GOTTSCHALK.CALIBRATION.GEO.GEOTEST]

ALL.FREE;2           ALL.FREE;1            C0.CARD;2            CAL0.CAM;1
CAL1.CAM;3           CAL1.CAM;2            CAL1.CAM;1           F0.FREE;11
HEXAGON.CARD;1       HEXAGON.TIFF;1        I0.CAM;21            I0.CAM;20
I0.CAM;19            I0.CAM;18             I0.CAM;17            LINES.TIFF;1
LINESA.TIFF;1        LINESC.TIFF;1         LS0.LSP;8            M0.CAM;82
M0.CAM;81            M0.CAM;80             M0.CAM;79            M0.CAM;78
M0.CAM;77            M0.CAM;76             M0.CAM;75            M0.CAM;74
M0.CAM;73            M0.CAM;72             M0.CAM;71            O0.OPR;26
ORIENT.FREE;1        ROT.FREE;2            ROT.FREE;1           S0.CAM;9
SYNTH.CAM;13         TR.FREE;1

Total of 38 files.

Directory DKA0:[VGD.GOTTSCHALK.CALIBRATION.PHOTO]

B0.TIFF;1            BORDER.TIFF;1         C0.TIFF;1            D0.TIFF;1
E0.TIFF;1            F0.TIFF;1             H0.TIFF;1            I0.TIFF;1
J0.TIFF;1            K0.TIFF;1             L0.TIFF;1            M0.TIFF;1

Total of 12 files.

Directory DKA0:[VGD.GOTTSCHALK.DT]

DT.H;16              DTERRORDETECT.C;1     DTERRORDETECT.OBJ;1 DTGRABIMAGE.C;34
DTGRABIMAGE.OBJ;7    DTINITIALIZE.C;28     DTINITIALIZE.OBJ;1   DTLIVEVIDEO.C;6
DTLIVEVIDEO.OBJ;2    MAKEFILE.;7

Total of 10 files.

Directory DKA0:[VGD.GOTTSCHALK.LINES]

LINES.C;50           LINES.C;49            LINES.EXE;2          LINES.OBJ;29
```

```
LINES.OPT;31          MAKEFILE.;23

Total of 6 files.

Directory DKA0:[VGD.GOTTSCHALK.PHASE]

P0.PHPARAMS;40        PH.H;22               PH.OPT;11

Total of 3 files.

Directory DKA0:[VGD.GOTTSCHALK.Q1FIGS]

AMBIG_ELLIPSE.DRW;1   BLOCK_W_VECTORS.PSP;2
SPHERE_W_VECTORS.PSP;3
STORS_PROP_II_2.DRW;1                       STORS_PROP_II_3.DRW;1
STORS_PROP_II_4.DRW;1                       STORS_PROP_II_5.DRW;1
STORS_PROP_II_6.DRW;1                       VOLKSWAGON.PS;1

Total of 9 files.

Directory DKA0:[VGD.GOTTSCHALK.QUICKIES]

CREATE_AND_WRITE_IMAGE.FOR;28               CREATE_AND_WRITE_IMAGE.OBJ;1
IMAGE_PROTO.H;8       MAKEFILE.;5           READIMAGEFILE.C;4
READIMAGEFILE.OBJ;1
SUBIMG.C;11           SUBIMG.DIA;4          SUBIMG.EXE;8          SUBIMG.JNL;1
SUBIMG.OBJ;2          WRITEIMAGEFILE.C;21   WRITEIMAGEFILE.OBJ;1

Total of 13 files.

Directory DKA0:[VGD.GOTTSCHALK.QUICKRANGE]

CAL.CAM;1             CAL.PROJ;11           MAKEFILE.;8           QR.C;46
QR.EXE;27             QR.H;10               QR.OBJ;27             QR.OPT;6
QRREADPROJECTORPARAMETERS.C;11              QRREADPROJECTORPARAMETERS.OBJ;11
QRREADSHORTIMAGE.C;5                        QRREADSHORTIMAGE.OBJ;6
QRWRITEDELTAGRAPH.C;2                       QRWRITEDELTAGRAPH.OBJ;5
QRWRITESWIVELPOLYMESH.C;23                  QRWRITESWIVELPOLYMESH.OBJ;21
TMP.C;1

Total of 17 files.

Directory DKA0:[VGD.GOTTSCHALK.RANGEDATA]

R0.DIR;1              R1.DIR;1

Total of 2 files.

Directory DKA0:[VGD.GOTTSCHALK.RANGEDATA.R0]

A.CAM;1               A.IMG;1               A.PROJ;1              A.TRUE;1
AVIEW.TIFF;1          A_000.TIFF;1          A_090.TIFF;1          A_180.TIFF;1

Total of 8 files.
```

```
Directory DKA0:[VGD.GOTTSCHALK.RANGEDATA.R1]

A.TIFF;1              C0H_000.TIFF;1        C0H_090.TIFF;1        C0H_180.TIFF;1
C0L_000.TIFF;1        C0L_090.TIFF;1        C0L_180.TIFF;1        C1H_000.TIFF;1
C1H_090.TIFF;1        C1H_180.TIFF;1        C1L_000.TIFF;1        C1L_090.TIFF;1
C1L_180.TIFF;1        CAL.CAM;1             CAL.PROJ;16           COKE0H_000.TIFF;1
COKE0H_090.TIFF;1     COKE0H_180.TIFF;1     COKE0L_000.TIFF;1     COKE0L_090.TIFF;1
COKE0L_180.TIFF;1     P0.PHPARAMS;41

Total of 22 files.

Directory DKA0:[VGD.GOTTSCHALK.RAWPHASE]

MAKEFILE.;17          PS000.PSDAT;1         PS090.PSDAT;1         PS180.PSDAT;1
PSIMAGES_TO_RAWPHASE.C;1                    RAWPHASE_$COMPUTE_PHASE.C;20
RAWPHASE_$COMPUTE_PHASE.OBJ;14              RAWPHASE_$PROTOS.H;2
SYNTH_RAWPHASE.DIR;1                        TESTRAWPHASE.DIR;1

Total of 10 files.

Directory DKA0:[VGD.GOTTSCHALK.RAWPHASE.SYNTH_RAWPHASE]

I.SYNTHDAT;9          MAKEFILE.;53          SYNTH_RAWPHASE_$MAIN.C;8
SYNTH_RAWPHASE_$MAIN.EXE;2                  SYNTH_RAWPHASE_$MAIN.OBJ;2
SYNTH_RAWPHASE_$PROTOS.H;1                  SYNTH_RAWPHASE_$READ_INPUT_DATA.C;7
SYNTH_RAWPHASE_$READ_INPUT_DATA.OBJ;2       SYNTH_RAWPHASE_$WRITE_DATA.C;11
SYNTH_RAWPHASE_$WRITE_DATA.OBJ;6

Total of 10 files.

Directory DKA0:[VGD.GOTTSCHALK.RAWPHASE.TESTRAWPHASE]

B000.PSDAT;1          B090.PSDAT;1          B180.PSDAT;1          FC.FCDAT;3
IDL.PS;6              IDL.PS;5              IDL.PS;4              MAKEFILE.;8
PS000.PSDAT;2         PS090.PSDAT;2         PS180.PSDAT;2         R.RDAT;2
TESTRAWPHASE.EXE;25   TESTRAWPHASE_$COMPUTE_PHASE.OBJ;11
TESTRAWPHASE_$MAIN.OBJ;7
TESTRAWPHASE_$PROTOS.H;4                    TESTRAWPHASE_$READ_DATA.OBJ;1
TESTRAWPHASE_$READ_FCOEFFS.OBJ;1            TESTRAWPHASE_$WRITE_PHASE_DATA.OBJ;1

Total of 19 files.

Directory DKA0:[VGD.GOTTSCHALK.REPORT_GRAPHICS]

REALVIEW_CONFIG.DRW;3                       REALVIEW_CONFIG.DRW;2
VGD_SURPH.DRW;4       VGD_SURPH.DRW;3       VGD_SURPH.DRW;2       VGD_SURPH.DRW;1

Total of 6 files.

Directory DKA0:[VGD.GOTTSCHALK.SINESLIDE]

RAMP.DIR;1            TRAPIXINIT.C;2        TRAPIXINIT.MMS;3      TRAPIXRAMP.C;13
TRAPIXRAMP.MMS;3      TRAPIXSINE.C;30       TRAPIXSINE.MMS;7

Total of 7 files.
```

```
Directory DKA0:[VGD.GOTTSCHALK.SINESLIDE.RAMP]

MAKEFILE.;6          RAMP.C;9              RAMP.EXE;2          RAMP.OBJ;7

Total of 4 files.

Directory DKA0:[VGD.GOTTSCHALK.SLIDEDATA]

MATRIX.DIR;1         SINEPATTERNS.DIR;1

Total of 2 files.

Directory DKA0:[VGD.GOTTSCHALK.SLIDEDATA.MATRIX]

C13_100_60_1.HDR;1  C13_100_60_1.IMG;1  C13_100_60_2.HDR;1  C13_100_60_2.IMG;1
C13_100_60_3.HDR;1  C13_100_60_3.IMG;1  C13_50_60_1.HDR;1   C13_50_60_1.IMG;1
C13_50_60_2.HDR;1   C13_50_60_2.IMG;1   C13_50_60_3.HDR;1   C13_50_60_3.IMG;1
C3_100_60_1.HDR;1   C3_100_60_1.IMG;1   C3_100_60_2.HDR;1   C3_100_60_2.IMG;1
C3_100_60_3.HDR;1   C3_100_60_3.IMG;1   C3_50_60_1.HDR;1    C3_50_60_1.IMG;1
C3_50_60_2.HDR;1    C3_50_60_2.IMG;1    C3_50_60_3.HDR;1    C3_50_60_3.IMG;1
C7_100_60_1.HDR;1   C7_100_60_1.IMG;1   C7_100_60_2.HDR;1   C7_100_60_2.IMG;1
C7_50_60_1.HDR;1    C7_50_60_1.IMG;1    C7_50_60_2.HDR;1    C7_50_60_2.IMG;1
C7_50_60_3.HDR;1    C7_50_60_3.IMG;1    CS.DAT;1            IDL.PS;3
IDL.PS;2            IDL.PS;1            SINGLELINE.FRM;1

Total of 39 files.

Directory DKA0:[VGD.GOTTSCHALK.SLIDEDATA.SINEPATTERNS]

HIGH0001.HDR;3      HIGH0001.IMG;3      IDL.PS;10           LOW0001.HDR;3
LOW0001.IMG;3       MED0001.HDR;3       MED0001.IMG;3       SP.FRM;1
SPHIGH.FRM;1        SPLOW.FRM;1         SPMED.FRM;1

Total of 11 files.

Directory DKA0:[VGD.MAC_FILES]

ACQUIRESURFACES.;1  ACQUIRESURFACES.LOG;1                   AFP_RESOURCE.DIR;1
CAMERACALIBRATION.;1                    CAMERACALIBRATION.LOG;1
FABRICATECALIBRATE.;1                                   SURFACEMERGE.;1
SURVEYLITERATURE.;1
VGD.;1              VGD.1;1             VGD.2;1             VGD.LOG;1
VGDHARDDATES.;1     VGDPLAN.;1                              VGDPROJECT.;1
VIEWPOINTDETERMINATION.;1

Total of 16 files.

Directory DKA0:[VGD.MAC_FILES.AFP_RESOURCE]

ACQUIRESURFACES.;1  ACQUIRESURFACES.LOG;1                   AFP_INFOFILE.DIR;1
CAMERACALIBRATION.;1                    CAMERACALIBRATION.LOG;1
FABRICATECALIBRATE.;1                   FILEBACK.AFP;1      FILEINFO.AFP;1
SURFACEMERGE.;1     SURVEYLITERATURE.;1 VGD.;1              VGD.1;1
VGD.2;1             VGD.LOG;1           VGDHARDDATES.;1     VGDPLAN.;1
```

```
VGDPROJECT.;1        VIEWPOINTDETERMINATION.;1

Total of 18 files.

Directory DKA0:[VGD.MAC_FILES.AFP_RESOURCE.AFP_INFOFILE]

A C Q U I R E S U R F A C E S . ; 1     A C Q U I R E S U R F A C E S . I O G ; 1
CAMERACALIBRATION.;1
CAMERACALIBRATION.LOG;1                  FABRICATECALIBRATE.;1
SURFACEMERGE.;1      SURVEYLITERATURE.;1 VGD.;1               VGD.1;1
VGD.2;1              VGD.LOG;1           VGDHARDDATES.;1      VGDPLAN.;1
VGDPROJECT.;1        VIEWPOINTDETERMINATION.;1

Total of 15 files.

Grand total of 33 directories, 438 files.
```

Appendix A
Literature Review

The literature survey originally included in the first quarterly report is included on the following pages for completeness.

Appendix A

Review of 3-d Object Recognition Techniques

Work on 3-d object recognition was launched by the seminal work of Roberts [Rob64]. Earlier work on object recognition dealt primarily with the recognition of 2-d patterns in images. Unfortunately, 2-d object recognition problem is fundamentally different from that of 3-d object recognition, consequently many approaches that work well for 2-d cannot be extended to 3-d. Roberts' work addressed many of the key issues involved with 3-d recognition and, for a first attempt, was remarkably complete. It will be enlightening to examine Roberts' work in some detail later in this review. For now, we note that his method uses a combination of viewpoint-invariant qualitative topological relations between features and quantitative, viewpoint-dependent interfeature relations to select possible models and determine their spatial pose relative to the viewer. It was recognized that one of the weaknesses of Roberts' algorithm was the low-level feature extraction and grouping modules. Reasoning that the state-of-the-art in the low-level aspects of vision would eventually catch up, studies of the higher-level aspects of the recognition process were undertaken in simplified artificial domains. Such domains, often referred to as *blocks world domains* [Guz69, Huf71, Wal72, Tur74, Kan78] are usually restrictive about the types of objects that are allowed. A common example is polyhedra.

In synthetic, blocks world-like domains, features and their relationships are assumed to be known precisely, circumventing the difficulties that Roberts encountered with his low-level modules. Thus freed from the difficulties of noisy and error-prone low-level data, researchers in blocks world-like domains found that topological constraints, which are more viewpoint invariant than most other image relationships, can be used to perform many image analysis tasks, including object recognition. Unfortunately, the state-of-the-art in the low level aspects of vision has never achieved the low error rates that would allow these methods to be used with real data.

While part of Robert's work was carried to unfruitful ends, the rest of Robert's work contains many of the essential components of a contemporary

3-d object recognition algorithm. It is interesting to speculate what the current state-of-the-art might be had the lead provided by Robert's been followed more fully.

1 Classification of Object Recognition Methods

As there are so many approaches to recognition, it is difficult to find any single taxonomy that fits all of them well. Perhaps the broadest distinction between methods is based on the relationship beween the sensed data and the object models. Later in this section, how this relationship influences the design of an object recognition algorithm will be discussed.

Recognition systems may also be characterized by the nature of their commonly held attributes. In particular, in this section, systems will be compared based on the nature of their *features*, their *object models*, and, most importantly, their *matching methods*.

1.1 The Relationship Between Object Models and Sensed Data

Object recognition algorithms are most obviously divided into two general categories based on the relationship of the sensed data to the models in the recognition system's vocabulary. One class consists of *matched dimension domain* (MDD) algorithms [1]. Such algorithms assume that the scene geometry can be sensed so that geometrical relations in the scene are isomorphic to geometric relations in the models. MDD algorithms include methods for recognizing 2-d objects from intensity images as well as methods for reconizing 3-d objects from range images. In both cases, the geometry of the sensed scene can be directly compared to the geometry of the models in the system's vocabulary. The other class of methods, which we will call *general domain* (GD) methods, consist of systems that assume that the sensors do *not* give explicit geometric information about the geometry of the scene. This class includes intensity-based 3-d object recognition.

Most object recognition methods in the literature are of the MDD variety. This is probably due to the fact that solving MDD recognition is simpler than solving GD recognition. In a matched dimension domain, relative geometrical

[1] This terminology was introduced in [Hut88].

relationships existing between parts of the models are preserved in the sensed data, to within noise and visibility constraints, regardless of the viewpoint and pose of the object in the scene. By contrast, accomplishing recognition in general domains is more difficult since the geometry of the scene is not directly available from the image, and must be deduced indirectly.

MDD methods fall into two categories: those that are extensible to general domain recognition and those that are not. Our focus is on the solution of the object recognition from intensity images, which is a case of a general domain problem. Thus, we will concentrate on those methods that contribute to understanding or solving such problems. We will briefly examine the techniques that are not extensible to the general domain, primarily to gain understanding of what makes them inextensible.

1.2 Anatomy of Machine Recognition

Object recognition algorithms may be classified according to the particular nature of their commonly held attributes. In particular, recognition algorithms can be compared by the nature of their:

- **models:** how objects are represented to facilitate recognition.
- **features:** how the sensor data is transformed and grouped into representations that facilitate recognition.
- **matching:** how the space of scene instances is searched for explanations of the sensor data.

We will be most concerned with comparing recognition algorithms on the basis of matching, since this is one of the primary issues in the design of a 3-d recognition algorithm. However, the features and models that algorithms use often strongly influences the matching strategy. Therefore, issues associated with modeling and feature selection relevent to matching will be discussed where appropriate.

2 Previous Work in Object Recognition

Much of the remainder of this review examines and classifies previous work on object recognition. In many cases, classification is not obvious since many

object recognition methods combine elements of multiple matching strategies. In addition, there are far too many algorithms in the literature to discuss each of them in full detail here. In order to do a broad survey yet benefit from the insights resulting from detailed inspection of previous work, we will: first define several archetypical matching methods, then discuss one or two examples of each archetype in detail, and, lastly, briefly discuss other, similar, methods. Approaches that have elements of more than one archetype will be discussed where they fit best.

GD object recognition algorithms (i.e., not MDD algorithms) must search the space of scene instances. MDD recognition methods, on the other hand, exploiting the isomorphism between the sensor data and the models, need not search this space. Instead, they may search the space of image-feature to model-feature correspondences, looking for consistent sets of features among the model-features and the image-features. GD recognition approaches usually employ one of the following six archetypical matching paradigms:

- transformation clustering
- hypothesize and verify
- predict-observe-backproject
- backprojection
- global feature-based
- optimization-based

MDD approaches are a more varied lot. Many of them fall into the above catgories, but many do not. Those that do will be mentioned in the appropriate section. Those that do not will be collected under the heading "Miscellaneous".

2.1 Transformation Clustering and Hough Methods

The viewpoint consistency constraint implies that all the visible features on a rigid object must be consistent with projection from a single viewpoint. Clustering methods were among the first to employ the constraint, athough Lowe

[Low87b] is responsible for its recent appelation. The way that clustering methods use this constraint is by computing the object's feasible viewing transformations and then attempting to locate any clusters in viewing parameter space among the feasible transformations. Feasible transformations are typically computed by forming a set of correspondences between enough image-features and 3-d model-features to yield solution for a unique (or almost unique) set of viewing parameters that consistently projects the 3-d features onto the corresponding image features. Thus, most clustering methods employ object-attached features. Once the feasible sets of viewing parameters have been calculated, objects are recognized by locating clusters of feasible viewing parameters. In such systems, clusters are powerful evidence for the existence of an object in a scene since a cluster indicates transformation that maps many 3-d model features near to image features. The probability of such an event occurring accidentally, especially for large clusters, is small.

Once possible clusters have been identified, recognition is usually based on the properties of the clusters. Typically, the largest or largest few clusters are accepted, or the cluster size is required to be larger than some threshold cluster size.

In a clustering system, computation of feasible sets of viewing parameters is usually the least difficult part of the algorithm. Many possible solutions exist, differing in such details as the number of correspondences necessary to determine the viewing parameters, the number of degrees of freedom allowed in the transformation, and the nature of the features. For example, in the case of weak perspective, where there are six degrees of freedom, [Hut88] gives a solution using three pairs of simple points, whereas a single pair of more complex features called vertex pairs suffices [TM87].

The most problematic aspect of recognition using clustering is the location of *significant* clusters of feasible sets of viewing parameters in a high-dimensional parameter space. Often, the statistical properties of the distribution of the feasible viewing parameters is difficult or impossible to determine, precluding the use of well-understood probabilistic methods. In such cases, various non-probalistic techniques are used. These include variations of the *k-means method*, projection onto lower dimensional subspaces, and the *Hough transform*.

The k-means method is a simple iterative strategy for finding clusters in an n-dimensional vector space. The assumption is made *a priori* that there are exactly k clusters, which is a weakness. The n-d input vectors are divided into k groups, and prototypical values for each of the k classes is computed, often

the centroid of the vectors in the class. The vectors are then redistributed to the class whose prototype is nearest, according to some distance metric. This process is iterated until changes in the prototypes are no longer significant.

One of the problems with this approach is that a distance metric must be defined in the parameter space, which often consists of mixtures of rotational and translational parameters. Additionally, proximity in the parameter space may or may not imply similarity in projected shape.

Another method for finding clusters in high dimensional spaces operates by projecting the feasible points onto lower dimensional subspaces, and searching for clusters there. This is advantageous since finding clusters in lower dimensional spaces is easier than finding them in higher dimensional ones. However, the clusters in lower dimensional spaces could result from the accidental coincidence of points along the dimensions of the projection, and therefore the validity of such clusters should be verified.

The final generic clustering technique that we will discuss is the *Hough transform*. The idea of the Hough transform is to quantize the parameter space into uniform buckets and count the number of feasible parameter vectors in each bucket. Each bucket will tend to contain similar viewing transformations. Buckets with large occupancies will correspond to clusters, which, in turn, indicate likely instances of an object in the image.

There are a number of problems with Hough based methods. This is evident from the analysis in [Hut88]. When the number of "good" features is large compared to the number of "bad" features, i.e., features not resulting from an instance of an object in the image, then Hough methods work well. This is because the peaks in the array of bins is easily located: they are not submerged in a ocean of bad transformations. Unfortunately, in complex images, most of the features do not correspond to instances of the objects being sought. In this case, the probability of large false peaks in the bin array is large, and it becomes difficult to distinguish them from true peaks. One way to combat this problem is to reduce the number of features by making them more complex. Since the features are more complex, there are fewer of them, helping to reduce the number of false peaks. Unfortunately, it also reduces the number of features on the model that can contribute to a true peak in the bin array. In addition, since complex features tend to be less spatially localized, the chance that a feature may be occluded is increased. Further, the size of the bin array becomes astronomical when the dimension of the parameter space is more than four or so. In such cases, projection to lower dimensions becomes a necessity, which

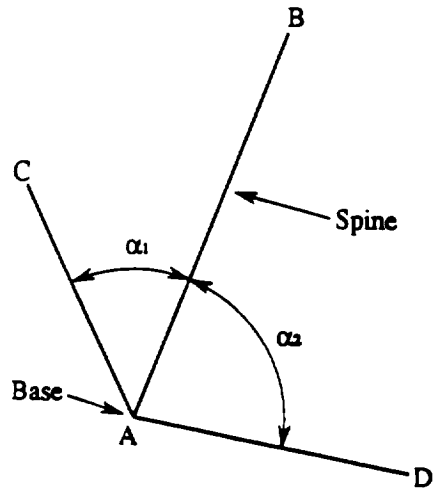


Figure 1: A vertex pair feature. AB is the spine, with A as the base vertex and B as the auxiliary vertex.

further aggravates the problem of random peaks.

Thompson and Mundy [TM87] provide a good example of using Hough clustering to recognize 3-d polygonal objects in intensity images. Features consist of "vertex-pairs", shown in Fig. 1, which come in 2-d and 3-d varieties. The 2-d variety is confined to a plane. Both kinds consist of a "*spine*" which joins two vertex points. One of the vertices is defined to be a "*base vertex*" which has two other edges incident on it, in addition to the spine. The other "vertex" is the point at the other end of the spine. In 3-d, the vertices consist of the vertices of a 3-d polygonal model. The 3-d vertices project to 2-d vertices, and, as shown in Fig. 1, the 2-d vertex pair is characterized by the angles α_1 and α_2 between the spine and the edges incident on the base vertex, as well as the spine vector. How such vertex pairs are segmented from the image is not discussed.

For a given 3-d vertex pair, there is a unique weak perspective transformation that projects the 3-d vertex pair to a given 2-d vertex pair. Thompson and Mundy compute this projective transformation using the quaternion technique of [FH83] which allows an algebraic solution. However, to speed the computation of the transformation, the computation is split into in-plane translations and rotations, and out-of-plane rotations. For each 3-d vertex pair in the model polyhedron, the authors compute a table of the out-of-plane rotation parameters versus the values of the angles in an observed 2-d vertex pair, α_1 and α_2. There is no

entry if the 3-d vertex pair is not visible. The set of tables for all possible 3-d vertex pairs comprises the model of each object. Thus, when a 2-d vertex pair is detected in the image, possible out-of-plane rotation parameters can be quickly computed. The remaining in-plane rotation, translation, and scale parameters can be easily determined by aligning the spines and vertices of the 2-d and projected 3-d vertex pairs.

The clustering used by [TM87] is Hough-based, with the 6-d transform parameter space being decomposed into a 2-d space of out-of-plane rotations, a 1-d space of in-plane rotations, and finally, a 3-d scale/translation parameter space. Each correspondence between a 2-d vertex pair and a 3-d model-derived vertex pair yields a set of transformation parameters. Entries are first made in the 2-d out-of-plane rotation bin array, with bins representing two degree increments in α_1 and α_2. The bin array is scanned for peaks indicating clusters, and these peaks are re-histogrammed in a 1-d array for the in-plane rotation. Clusters detected in this 1-d table are further clustered in the 3-d space of translation/scale. This final clustering is done using a variation of the k-means method described above. A cluster in the final histogram with more than three assignments is considered a correct match.

Results were good, although the tests were done on images where the object(s) to be recognized possessed the vast majority of the features. As mentioned earlier, this type of image is the type the clustering algorithms perform best on.

Thompson and Mundy also describe a nearly identical approach to recognition in range images [CMST88]. The Hough-based matching scheme is identical, and most of the novelty resides in segmenting vertex pairs from range images. Strangely, the authors continue to use 2-d vertex pairs even though the range data provides true 3-d features that could be compared directly with the model features rather than though their projections.

Hough-based recognition methods are further beset by problems not mentioned so far. In particular, choosing the bin size is difficult. Making the bins large reduces storage requirements. Additionally, and more importantly, the chance that the cluster resides wholly in one bin is enhanced, improving the chances of detecting the cluster. On the other hand, since the bins are larger, the chance of large random peaks is larger as well, making detection of true clusters more difficult. Also, since the parameters of the bin are typically used to estimate the pose of the object, a large bin size reduces the accuracy of the estimate of the pose of the object. Making the bin size smaller improves the accuracy of the estimate of the pose but reduces the likelihood that the cluster

will fall into a single bin, making detection more difficult. Clearly, if there were no errors in the calculation of the feasible transformations, a very small bin size would be preferable as all the points in the true cluster would fall in a single bin while the chance of large random peaks would be vanishingly small, simplifying detection, and improving the estimate of the pose. However, since there is always error in the transformation parameters, there is an *optimal* bin size that depends on the statistics of the error and the transformations that do not belong to the true cluster.

Often, the optimal cluster size for detection is too large for precise pose estimation. In order to overcome this problem, [SDH84, SHD84] discuss an *iteratively subdivided* Hough procedure for finding clusters of feasible transformations in a 5-d parameter space. Detection is done at a large bin size that is good for detection and then the detected clusters are rebinned into smaller an smaller bins until the cluster begins to fragment into multiple bins, providing better pose estimation.

In a similar vein, [LHD88] describes a recognition method that locates clusters in a full 6-d parameter space. Features are 2-d and 3-d triangles. The vertices of the 2-d triangles are generated by intersections of linear contours in the image and the vertices of the 3-d triangles consist of the vertices of the polyhedral model. Corresponding the points of a 2-d and a 3-d triangle leads to a nearly unique solution for a feasible transformation. The feasible transformations are clustered in a 3-d bin array that uses only the translational parameters, as they can be computed very quickly. Peaks are located by examining a 3×3 neighborhood and suppressing nearby peaks that are likely to be fragmentations of true peaks. Then, peaks in a histogram of translational parameters are detected. Finally, rotation parameters of the transformations are computed and checked for consistency. Visibility of the model triangle pairs can be determined from the rotation parameters, and these constraints are also applied to filter transformations. A further heuristic is applied to determine which of the 3 possible image-triangle to model-triangle vertex correspondences is most likely to be correct. Final acceptance of a cluster is based on how closely the projection of the consistent features in the cluster match with actual image features. Once a cluster has been accepted, a final fit of the model triangles to their corresponding image triangles is done using a least squares measure, yielding good accuracy for the estimate of the pose of the object.

Stockman and Esteva [SE85] describe a similar transformation clustering technique. In this case the transformations are constrained, and result in a 3-

d parameter space. Correspondences are formed between pairs of 2-d feature points and 3-d model points, which, in the 3-d transformation space, allow the pose to be determined uniquely. Feasible transformations are computed for all possible pairings of image and model features; unlike [TM87, CMST88] visibility constraints are not applied. Clusters are then detected among the feasible transformations. Although not explicitly stated, it appears that a variant of k-means is used to find clusters. Another method, described in [FHK+82], is very similar.

A series of papers and reports by Lamdan, Wolfson, and Schwartz [LSW88b, LW88b, LSW88a, LW88a] describe an interesting clustering based algorithm for recognizing 3-d objects from intensity images. An earlier paper [KSSS86] describes a similar approach for 2-d objects. At the heart of these methods is a representation scheme that the authors call "geometric hashing". A key component of the "geometric hashing" approach is the existence of a representation of the features that is invariant to the 2-d transformations that the features undergo as the object's pose changed in 3-d space. For example, object-attached features on flat, rigid objects imaged under weak perspective undergo a *affine* transformations as the viewing parameters are varied. If primitive features, such as corner points, are described solely by their position in the image plane (not by additional descriptors such as orientation, curvature, etc.), then a set of three such points defines a *basis set* that forms a coordinate system in which all the remaining feature points can be described. As shown in Fig. 2, affine transformation of the points does not affect their coordinates in the basis-set coordinate system. Of course, this is only true if the same basis set is used. Similar representations can be defined for object-attached features on rigid 3-d objects under weak perspective as well; four points are required in this case.

Given existence of such invariant representations of feature points in terms of a multipoint basis-set, the "geometric hashing" approach simply represents *every* feature point in terms of *every* possible basis-set and stores each point's coordinates in terms of each possible basis-set in a hash table data structure. The "hashing" is done by representing each coordinate by a binary number and truncating the low order bits, leading to a discretization of the coordinate space into hypercubes, and then using the resulting binary string as a key into a hash table. The buckets in the hash table are, therefore, representations of various hypercubes in the feature space. The hash table is loaded with all possible representations of all possible features for each model in the system's vocabulary. It is possible that a hypercube may contain more than one feature,

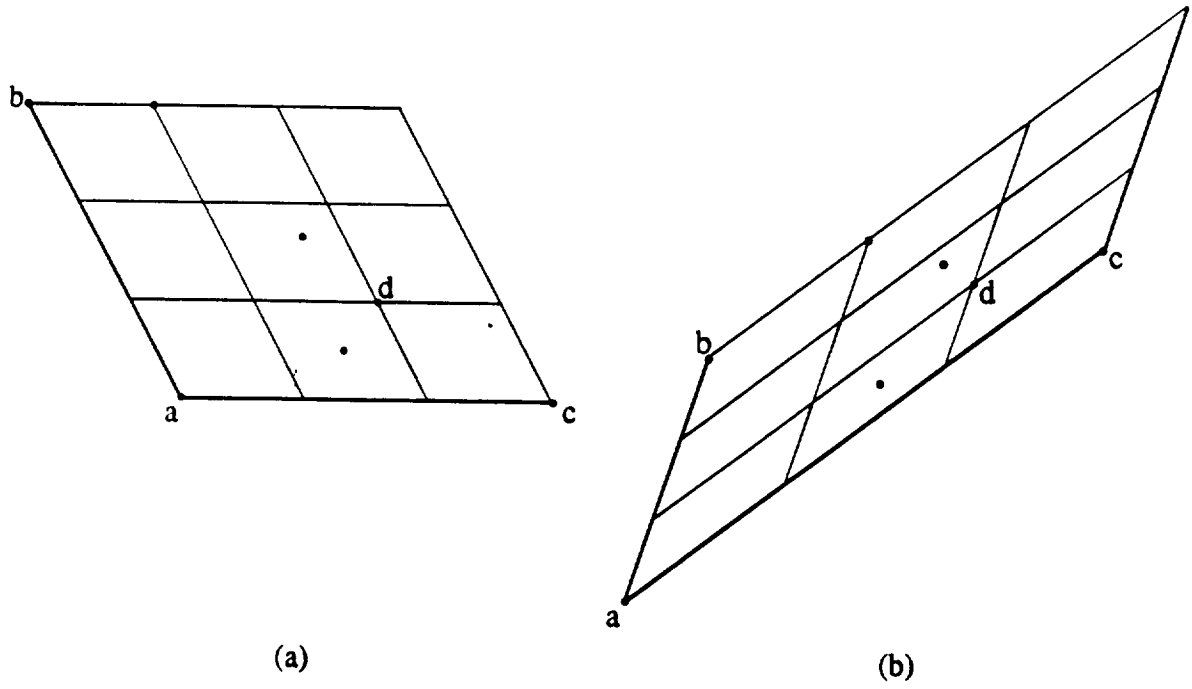


Figure 2: Demonstration of invariance of Lamdan *et al*'s feature representation to affine transformations. Above, (b) is the result of applying an affine transformation to (a). In both (a) and (b), the feature points **a**, **b**, and **c** form a "basis" coordinate system that has the property that the coordinates are invariant to affine transformations. The other feature points, such as point **d**, can be represented in terms of these invariant coordinates. For example, the coordinates of **d** are (2,1) in both (a) and (b) in spite of the affine transformation between them.

especially if the size of the hypercubes is large.

Recognition proceeds by picking a possible basis (three points) in the image and computing the coordinates of all the other feature points in the image in terms of it. Each hypercube that contains an image feature is retrieved. Records, consisting of ¡model, basis¿ pairs, are constructed. For each such record in the hypercube, increment a vote counter for that record. If any particular ¡model, basis¿ record scores a large number of votes, then it a matching candidate.

The correspondence between the image basis and the model basis provides a unique solution to the transformation parameters. Thus, while the transformations are not explicitly represented in this method, they are implicitly coded in the representation. The "voting procedure" is thus voting for a discrete set of possible transformations that are induced by the choice of the image basis points. For this reason the method should be considered to be a transformation clustering approach. The candidate matches and the associated transformations are then retrofitted using least squares and employing any additional, close matching points in the image. Finally, a verification is done between the boundaries of the model and the edge contours in the image.

The method of "geometric hashing" suffers from a few difficulties. First, use of the term "hashing" is somewhat misleading because it implies constant time access of the records. Since only the hypercubes themselves are indexed in the hash table, strictly speaking, the access time is linear in the average occupancy of the hypercubes. The only way to reduce this is to make the quantization finer. However, this makes it more likely that a noisy feature will not hash to the correct hypercube, similar to the problems faced by Hough-based methods. Under poorer conditions than the high-contrast, backlit scenes reported, the method may break down. Further, extension of this method to true 3-d objects requires that the voting occur at all hypercubes intersected by a line in the feature space, which appears to be an inefficient operation. Finally, the combinatorics of this algorithm are unfavorable since voting must be done for every possible basis set in the image.

There are numerous examples of clustering approaches used to recognize 2-d objects from intensity images [MF75, Bal81, Seg83, BS85, KK85, TMV85, Hwa87, Ger88, SKB82], a matched dimensionality domain. With one exception [Hwa87], these methods employ Hough-based clustering exclusively. In 2-d methods, the transform space is at most 4-d, consisting of translation, rotation about the normal to the image plane, and 2-d translation within the image plane. In [MF75, Bal81] the pioneering work on applying the Hough transform to the

recognition of arbitrary 2-d shapes is described. In common with Hough-based approaches in the 3-d recognition domain, some of these methods attempt to overcome the problem of a high-dimensional parameter space by decomposing the transform bin array into lower dimensional bin arrays [BS85, Seg83]. The approach described in [TMV85] determines weights for each transformation based on a rigorously defined *saliency* measure of the features used to compute the transformation. This greatly reduces the number of random peaks in the bin array, though extending the notion of saliency to 3-d appears to be somewhat difficult. Another method [Ger88] "links" transform space and the feature space by performing, essentially, a *verification* of the clusters detected in transform space by matching predicted features in the image to filter false clusters from true ones.

2.2 The Hypothesize and Verify Paradigm

As we have seen, methods that recognize by clustering attempt to form hypotheses that have considerable global support. Such methods need many features to "vote" before a "consensus" is reached, and, with few exceptions, the clustering is the sole means of accumulating evidence for particular hypotheses. The *hypothesize and verify paradigm* (HVP), on the other hand, is less democratic. In the HVP, only features that can explain the image data in the locality of themselves are allowed to become valid hypotheses. Then, they may win overall by explaining a more global portion of the image data.

Typically, the first step in the hypothesize and verify paradigm is to construct a sparse representation of the image in terms of features. Ideally, such features are highly selective, i.e., model generated features are chosen so that the likelihood of them having the same attributes as an incorrectly matching image feature is very small. Unfortunately, there are practical limits to how selective features can be, as highly unique features tend to be more global in nature, in addition to often being too sparse. Given a reasonable choice of image features, one is chosen by some means, and the set of possible scene instances that could explain its existence, to within measurement error and noise, is computed. Typically, this set is represented by a number of individual scene instances that are treated as separate, competing hypotheses. This process is called the *generation* phase. Next, the existing hypothesis are *verified*. As mentioned in the preceding paragraph, each hypothesis provided by the generation phase is usually a single scene instance. This scene instance is then used make predictions that can be

used to perform a detailed check against the image. Based on how well the predictions match the observations, a decision is made as to whether the hypothesis is strong enough to be considered a valid recognition result. Typically, the representations of the predictions and the representations of the observations used in the comparison are more *complete* than the representations used during the generation phase of the HVP. That is, the representations contain more of the information that is contained in the model instance or the image.

The exhaustive approach to recognition would be to consider *every* model instance, to within an error resolution volume, as a hypothesis and to verify all of them, passing those that scored high enough in the verification. This is computationally untenable. The hypothesize and verify paradigm overcomes this by considering only those model instances that have at least a small bit of evidence in their favor; usually one, two or a few features that match image features. This filtering vastly improves the efficiency of the search.

In the hypothesize and verify algorithms come in a spectrum of varieties. Aside from differences in the types of features and representations that they use, these algorithms differ in two other respects:

1. how much effort is expended to generate a strong hypothesis, and

2. how features and hypotheses are ranked for further processing.

With respect to item 1, some algorithms have opted to expend the minimal effort generating hypotheses, typically creating many weak hypotheses that are then rejected, while the few strong hypotheses are passed. This approach insures that the correct hypotheses will be very likely to be among the set generated. This approach is robust, but tends to be slow since verification is usually rather expensive in comparison to generation. The other extreme, is to generate hypotheses that are rather strong, and, therefore, likely to be correct. Since the generation algorithm in this approach is discriminatory, it may not allow the correct hypothesis to pass on the basis of the limited information that is has available to it, thus missing a correct interpretation. Also, if taken to extreme, the effort spent generating a strong hypothesis may outweigh the cost of verifying it. Thus, this extreme tends to be less robust, and often just as slow as the other extreme. The optimum tradeoff falls somewhere between the two. [KJ86] describes a 2-d algorithm where the tradeoff is adjusted to minimize the overall time spent generating and verifying hypotheses under certain assumptions about

the way the hypotheses are generated. Unfortunately, such analysis cannot easily be extended to the 3-d domain.

With respect to item 2, the queue of features waiting to be processed can be ranked according to the likelihood that a given feature will generate a strong hypothesis. If the ranking heuristic is good, this will reduce the time it takes to locate objects in the scene. Similarly, hypotheses that have been generated but not yet verified can be ranked in the queue for verification, also reducing recognition time. Most HVP algorithms rank features in some manner. For example, a number of systems use model-independent feature grouping [Low87a, Chi89, Hut88, Jac87] to rank the features. It is less common for systems to rank the generated hypotheses before verifying them; usually they are verified immediately upon being generated. An example of a 2-d object recognition system that does rank hypotheses is described in [BC82]. In that system, as mentioned previously, considerable effort is spent to generate strong hypotheses. In that system, hypotheses are graphs of whose nodes are possible image-feature to model-feature pairings, and arcs represent mutually consistent pairings, and whose nodes are fully connected. These hypotheses are then ranked for verification by the size of the graph, which simply measures the number of features that have been matched. Chien and Aggarwal's system for recognizing 3-d objects [Chi89] also ranks the generated hypotheses. In their work, transformations are computed for each hypothesis from hypothesized correspondences between quadruples of image features and model features. The transformation is solved simply as a system of linear equations. In particular, orthogonality of the rotation matrix is not enforced. If the hypothesized correspondence is correct, then the computed transformation should have an orthonormal rotation matrix. The rotation matrix is not likely to be orthonormal if the correspondence is incorrect. To rank the hypotheses, Chien and Aggarwal examine the transformation associated with each hypothesis to determine how close its rotation matrix is to being orthonormal . The hypotheses with transformations containing rotation matrices that are nearly orthonormal will be processed first, and grossly non-orthonormal cases will be eliminated.

2.2.1 3-d Intensity-Based Hypothesize and Verify Methods

An excellent example of the HVP is the ORA (Object Recognition by Alignment) system described in [HU88, Hut88]. This system also typifies how object-attached features may be used to drastically simplify the computation of the

hypothetical viewing transformation under the HVP.

The essence of the ORA system is very simple. First, image features, consisting of triplets of points, are put into correspondence with model features, consisting of triplets of 3-d model points. For each correspondence, the ORA system "aligns" the model so that the projection of the triplet of model points comprising the model feature exactly corresponds to the triplet of points in the image feature. Of course, for each pairing of image and model features, there are three resulting possible permutations of image points and model points, and therefore three possible alignments. ORA uses the weak perspective imaging model, and therefore three image-point to model-point correspondences are sufficient to determine the viewing transformation to within a reflection across the image plane. Note that ORA's approach to solving for the transformation is an improvement over that in [Chi89] since the orthonormality of the rotation matrix is maintained. Thus, ORA does not consider any invalid transformations. After alignment, the hypotheses are verified. If a hypothesis is accepted, the image features that have been matched to it are removed from further consideration.

ORA employs curvature-based segmentation of image curves. ORA's creators argue strongly for the use of inflection points, or zeroes of curvature, and line segments as features. They give two reasons for this: first, inflections and line segments are preserved when a space curve is projected, and second, they argue that there is no psychological evidence for high-curvature points over inflection points, citing Lowe's cat [Low85] as a counterexample to Attneave's [Att54]. Oddly, they go ahead and use high-curvature points anyway. All of the features, are assumed to be object-attached, otherwise the procedure of assuming 3-d to 2-d correspondences and calculating the alignment transform breaks down. As a result of the use of object-attached features, ORA's vocabulary consists of planar or polyhedral objects. The image features are ranked according to how likely they are to be part of the same object in the scene. The heuristics used are:

1. Points are likely to belong to the same object if they appear on the same edge contour, or if they belong to two contours that are likely to have come from the same object.

2. Two contours are likely to be from the same object if their endpoints are in close proximity.

3. Two contours are likely to be from the same object if the relative intensity

on either side of each contour is comparable and the contours form a compact geometric shape.

Verification in ORA is a two stage hierarchy: the initial stage is cheap computationally but eliminates many false matches, while the second stage is slower but more accurate. The initial stage operates by checking the endpoints of curve segments in the projection of the model with segment endpoints in the image. Endpoints are either inflection points, high-curvature points, or endpoints of linear segments. Both the position of the points and the direction of the tangents through them are required to be within an error tolerance before an endpoint can be said to match. If more than half of the endpoints visible in the model match with endpoints in the image, then the hypothesis is passed through the initial stage of verification. The detailed verification procedure compares all of the visible contour segments with nearby image segments. If enough of the predicted boundary is near to an image boundary, the hypothesis is passed.

The ORA system is typical of 3-d HVP methods in that it employs an analytic formula to yield the transformation that maps some minimal number of 3-d model points onto 2-d image points. For any such method to work, the features must be object-attached. Some work has been done to extend ORA to smooth 3-d objects [BU88, SU88]. However, it is unlikely that the alignment method, in its present form, can be extended to recognize smooth 3-d objects.

Lowe's SCERPO system, described in [Low87b, Low87a, Low85], is also a HVP approach. It differs from ORA, which came later, in several respects. First, SCERPO is only able to recognize polyhedral objects since object-attached line segments constitute its feature set. Lowe's work is based heavily on the idea of "perceptual grouping" which is essentially the ability of a vision system to group features based on a model-independent measure of their "perceptual significance". Perceptual significance is really the same as non-accidentalness, i.e., the likelihood that a configuration of features observed is not the result of a visual accident. By making several assumptions about the stochastic properties of the distribution of detected line segments in the image, Lowe is able to come up with an analytic heuristic measuring the perceptual significance of several types of relations between line segments: proximity, parallelism, and colinearity. These significance measures work on pairs of line segments. Significant pairs are further grouped into significant clusters by noting the pairs that share segments.

The "perceptual grouping" process in SCERPO can be carried out equally well on the 3-d model segments as on the 2-d image segments. Hypothesis

generation then proceeds by simply matching 3-d groupings to 2-d groupings with the same number of segments, with the groupings with the most segments being processed first since they are the most "significant", although such groupings will lead to a large number of possible permutations in the correspondences of image line segments to model line segments. Image groupings with the same number of segments as model groupings are matched, and the viewing transformation is determined using the correspondences of the individual line segments (three or more uniquely determines the transformation), yielding initial hypotheses. The initial hypotheses are then verified. SCERPO verifies hypotheses by first using the initial hypothesis to find all image line segments that match sufficiently well to line segments predicted by the model. These image-segment to model-segment pairings are then used to find a least-squares fit between the projected model-segments and the image-segments using a multi-dimensional Newton-Raphson algorithm. Note that this differs greatly from the tracking method of VGD due to the fact that explicit correspondences are made between single pairings of 3-d model line segments and 2-d image line segments. VGD's tracking method maintains a fuzzy degree of correspondence between all possible pairings. In addition, the features in VGD's tracking method are not object-attached. Recently, the fitting technique used by SCERPO has been extended to handle parameterized models [GL87]. Finally, after the least squares fit, the predicted model-segments are again matched to image-segments, and if more than ten matching pairs result, the hypothesis is accepted.

The recent work of Chien and Aggarwal [CA87, Chi89] follows the HVP. Similarly to both the ORA and SCERPO systems described above, Chien and Aggarwal's system detects features in the image (sets of four consecutive "corner-like" features from the edge contours) and forms hypothetical correspondences between these quadruples of image points and quadruples of 3-d model points. This allows the transformation parameters to be solved. As mentioned earlier, Chien and Aggarwal do not enforce the orthonormality of the rotation matrix portion of the transformation, in contrast to ORA, and therefore many inconsistent hypotheses are generated. These are weeded out by applying the orthonormality constraints.

Chien and Aggarwal's algorithm employs 2-d feature points and 3-d model feature points. The 2-d feature points are simply high-curvature points. The 3-d points are chosen by the finding 2-d high-curvature points in three orthogonal "principle views" of the object, and then determining the intersections along the lines of sight from the different views to yield 3-d feature points. Since the

3-d high-curvature points are assumed to project to 2-d high-curvature points, these features are object-attached. The features are ranked by the heuristic that postulates points with higher curvatures as more likely to be reliably detected than those with smaller curvatures. After hypotheses with valid transformations are found, they are then verified. Verification is a direct comparison of contour shape using a polar representation of the contour with the contour centroid as the origin. This measure does not work for occluded objects. A method for doing verification for occluded objects is discussed, but no results are given. It appears that parts of the hypothesis generation algorithm require knowledge of figure-ground segmentation as well, rendering this algorithm very weak for practical scenes. In its favor, the verification method could easily be fixed, using a method such as the one in ORA, or the one used by VGD.

All of the methods described above owe a large debt to the seminal work of Roberts [Rob64], which also followed the HVP. Like SCERPO, Roberts' system used perceptual groupings. In Roberts' system, the groupings consisted of polygons about vertex points in the image. These image polygons were topologically matched to polygons derived from the model. This is also similar to SCERPO's topological matching of perceptual groupings, though SCERPO's implementation is more robust. Following the topological matching, Roberts' system computed hypothetical transformations in a manner similar to the systems described above, though the particulars are most similar to Chien and Aggarwal's system. Roberts's method does not enforce orthogonality of the rotation portion of the transformation, forcing him to patch it up in an *ad hoc* manner. The result of computing the transformation is a set of hypotheses, which are then verified. Verification consists of computing the mean-square error in the projected model points that were members of the original topological match and the corresponding image points.

Other HVP approaches to recognition of 3-d objects from intensity images tend to be very similar to the systems described above, though they may be less complete. For example, the method described in [Whi88] goes through some representational gymnastics in a feature space called "vertex space" to come up with features, called "key features", also known as triangles, that are matched to image features. After matching, a viewpoint hypothesis is generated. As usual, the model is then projected, verification is done by seeing if enough predicted vertices match observed vertices. Similarly, [PD87] describes a hypothesize and test algorithm that uses a representation called an "asp", consisting of the deformations that the triangles in the polyhedral model undergo as the viewpoint is

continuously changed. Features in the image, which are polygons, are compared with the features in the "asp" to see if there are any matches. If there are, a viewpoint hypothesis results. The "asp" is really a form of multiview model. In fact, the features used are object-attached, and therefore, a multiview model is superfluous since viewpoint hypotheses can be computed directly. Details of the verification method are not provided, nor are any results.

A set of two papers by Hansen and Henderson [HH87, HH88] describes a HVP approach called "recognition by strategy trees". The method works with polyhedral objects, and represents them as "strategy trees". A strategy tree consists of a model at its "root". A set of "level one features", and a "corroborating evidence subtree" constitute the body of the tree. The level one features are the "strongest set of view-independent features chosen for their ability to permit rapid identification of an object and its pose". Given an image feature, the matching method finds the model's "level one features" that have similar attributes. These features are used to find the pose of the model, presumably in a manner similar to the other methods described above, or by using visibility constraints. For each such "level one match", a "corroborating evidence subtree" is evaluated. Essentially, this is a verification procedure. Details are sketchy, but the verification seems to check predicted features against observed features and then perform a fit, as in SCERPO. Following that, a detailed boundary check, as in [BC82], is done. The most interesting aspect of this approach is the freedom to tailor the features used in the generation and test modules to optimize performance for each object.

The method of Sato *et al* [STT87] is interesting because it uses a connectionist network both to filter competing hypotheses and to fit the model to the data. Features are based on line segments. Hypotheses are initially generated by pairing "L" junctions detected in the image data with "L" junctions in the model, and using the hypothetical correspondences to solve for initial viewing parameters. Verification consists of finding "clusters" of compatible hypotheses in the graph of compatibility relations. Each image feature will generate many possible hypotheses seeking to explain it. A "compatibility" measure is defined, and a Hopfield network [HT85] is used to simultaneously adjust the compatibility between hypotheses, yielding clusters of compatible hypotheses. Note that the clusters exist in the graph of compatibility relations, not in viewing parameter space. Each cluster indicates a recognized object. The information in the hypothesese comprising the cluster are then used to refine the estimate of the viewing parameters.

Fisher [Fis83] describes a HVP method that is unique in that it employs *region* based features in contrast to the *edge-based* features used by most other intensity image recognition systems. However, estimating the pose of a model from region data is difficult. Fisher's technique is to correlate the cross sections of model surfaces to segmented image regions, ultimately solved by optimizing a weighted distance measure between the projected model surface and the image region. Hypotheses are represented in a frame-like manner, with slots for transformation parameters, observed surfaces, and instantiated subassemblies. After generation, many of the surface slots and subassembly slots have not been filled. A phase of attempting to fill the slots is entered, implemented by a rule-based system. The initial phase, described earlier in the paragraph, can be called to instantiate subassemblies as necessary. Once all slots are filled, the complete hypothesis is verified. Verification consists of reestimating transformation parameters, checking if projected model surfaces substantially cover the image regions assigned to them, and finally, checking if the predicted boundary matches well against the observed boundary.

The primary weaknesses in Fisher's algorithm include the method for estimating the initial viewing parameters using correspondences between model surfaces and image regions, and the requirement that the segmentation of the image be very good. Hand segmented images were used in to obtain the results shown in the paper. Strengths of the algorithm include its use of hierarchical models and hierarchical matching. This helps to improve the efficiency and robustness of the algorithm.

2.2.2 MDD Hypothesize and Verify Methods

Many 2-d recognition systems have been reported that use the HVP. Some of these methods exploit the symmetry that exists between the model domain and the image domain, with the result that they are difficult or impossible to extend to the 3-d case. A example of such a method is the well-known "local-feature-focus method" (LFF method) [BC82]. This method was designed on the premise that verification is very expensive, so generating good hypotheses (i.e., ones likely to be correct) is important. Hypothesis generation is done via a graph search in a graph where the nodes represent possible image-feature to model-feature pairings, and arcs between nodes represent compatible pairings. Features are circles and corners. Roughly speaking, consistency is determined by the absence of competition for the same features, as well as agreement of

relative poses between image features and model features as represented by the pair of nodes in the graph. The graph is the searched for the largest sets of mutually consistent pairings of image features and model features. In order to improve the effiency of the NP graph search, nodes which contain hand-selected "focus features" are used to focus the search for maximally connected subgraphs. The largest such set is used to determine the viewing transform, a rotation and translation, by aligning corresponding features. This hypothesis is passed to the verification module, which directly compares the transformed model boundaries to the image boundaries using probes that are perpendicular to the model boundary. Dark-to-light transitions are positive evidence, light-to-dark and all-light transitions are negative evidence, and all dark transitions are neutral. This scheme assumes that the relative brightness of objects and background is known *a priori*.

A method that is similar in many respects to the LFF method called "3DPO" is described by Bolles *et al* in [BHH83, BH86] 3DPO attempts to recognize 3-d objects in range images. 3DPO's hypothesis generation apparatus is very similar to that of the LFF method described in the preceding paragraph. The features differ, being based on range discontinuities. They include circular arcs and linear segments, along with information about the surfaces on either side of them. As in LFF, the modeling system allows the user to label certain features as being particularly selective, as in the "focus features" of LFF. The viewing parameters are computed by successively constraining the possible transformation parameters as each image feature is matched to a model feature. In this regard, the hypothesis generation portion of 3DPO is an instance of the predict-observe-backproject recognition paradigm discussed in the following section. Verification consists of a comparison between the predicted range image that the model would produce with the actual data.

Rearick *et al* [RFC88] describe a method based on a connectionist network (as distinct from a neural net) that, they argue, is fundmentally different from any "model-based" method. The method is region based, and classifies regions into three types: *hons* (Japanese for long, thin objects like pencils) , *cusps, and loops*. The regions are detected using a modified medial axis transform [Hea86] and classified according to the topology of the skeleton and the width of the region about the skeleton. Relationships between the regions are used to recognize objects. Each object has a customized network that "filters" out sets of regions whose relations are not sufficiently similar to corresponding model relations. In fact, the network is really doing nothing more than a clique-finding

procedure similar to that hypothesis generation portion of the LFF method, one of the methods that Rearick *et al* say they are so different from. There is no explicit verification procedure.

Part of VGD is based on work described in [GTM89, GTM87]. The method is for recognizing 2-d objects, but the hypothesis generation portion of it has been incorporated into VGD with few modifications. The features, called "CPN's", are vectors that efficiently encode the shape of edge boundaries in the neighborhood of high-curvature points. Using training images, models of objects are constructed that consist of both the sparse CPN representation and a complete x, y, and slope-angle (θ) versus arclength representation. The models are stored in a special vector-associative memory, implemented with a k-d tree [Ben75] indexed by the five descriptive parameters of the CPN's. The associative memory allows a model containing a feature that matches an image feature to within a user-specifiable tolerance to be retrieved in $O(\log N)$ time, where N is the total number of model CPN's stored in the memory. To the author's knowledge, this is the most efficient hypothesis generation method in the literature most of which are linear at best. Hypotheses are generated by querying the associative memory for models possessing features that are similar to the image feature, and then aligning the model feature with the image feature. These hypotheses are then verified in a hierarchical fashion. First the percentage of CPN features predicted are compared to those detected to reject many possible hypotheses. Then a detailed boundary check similar to, but more general than, the one used by the LFF method is performed.

The GROPER system [Jac87] uses a hash table to implement an associative memory for hypothesis generation. The indexing is done on quantized versions of five parameters that describe the geometric relationship between two pairs of line segments. This approach has the usual problems associated with quantizing the parameter space (see the discussion of clustering based methods, and in particular, [LSW88b, LW88b, LSW88a, LW88a] above). Verification consists of checking how many detected line segments are close to predicted line segments. GROPER is not unique because of its generate and test modules, but rather in how it uses model-independent grouping to reduce the number of hypotheses that are generated. Edge grouping is done on the basis of relative proximity and orientation similarity. The number of hypotheses generated was reduced by nearly a factor of 400 by the inclusion of the grouping module. An improvement in accuracy was also noted, primarily because the verification module in GROPER is weak, and the relatively powerful grouping module assisted the

verification module.

Perkins [Per77, Per78] represents both the image boundaries and the model boundary in terms of "concurves", sequences of line segments and circular arcs. Concurve sequences are matched in a correlational manner. If a model has a subsequence of concurves that matches a subsequence of an image concurve, then a transformation is determined, and a hypothesis is created. The transformation is determined by aligning the matching concurves in tangent-slope-angle versus arclength space, and the hypothesis is checked in a manner similar to the LFF method, differing only in that the presence of an edge pixel with the proper direction is positive evidence, and there is no negative evidence. This is superior to the LFF verification technique as it requires no *a priori* assumptions about the illumination and reflectance of objects relative to the background.

Methods that rely on correlation of the portions of the model boundary with the image boundary for matching or pose determination, as in Perkins method above, cannot easily be extended to the general 3-d recognition. This is because the shape of the model boundary may change drastically with viewpoint, requiring that a correlation be performed for many points viewing parameter space. Since correlation is often time consuming, performing a correlation for a large number of viewpoints would be very slow. In effect, the boundary representation is too complete to allow efficient hypothesis generation.

Another method that uses correllation to generate hypotheses is described by Knoll and Jain in [KJ86, KJ87]. In this example, portions of the model boundary are correlated in Cartesian space. If the segments match well enough, the translation and rotation necessary to align the features are determined, and a hypothesis is created. Verification is nearly identical to that in the LFF method. What is unique about this algorithm is that the features are chosen to minimize the total recognition time under the assumption that the model features and the image boundary is searched in a linear fashion. Other correlational approaches to 2-d recognition are described in [Fre77, DKZ79, YMA80, BBR83].

Ettinger [Ett88] extends the work of Knoll and Jain in the direction of improving recognition time. He describes how employing a sub-part hierarchy can markedly improve recognition time complexity. The features employed by his system are those of the "curvature primal sketch" [AB86]. The features are contained in a scale hierarchy where the coarsest level of features is used to generate the initial hypotheses about the subparts of an object. Subpart hypotheses that have enough support in the form of more matching features at finer resolutions of the scale hierarchy can generate full object hypotheses. These, in turn, can

direct the search for other subparts. The full object hypotheses are hierarchically verified through the subparts.

The method described in [TFF88] is a HVP method that has been designed with parallel implementation on the Connection Machine [Hil87] in mind. Features are line segments and corners (corners are line segments whose endpoints are near to their intersection). Since the transformation space is 3-d, each image-corner to model-corner correspondences allow a model transformation to be computed, and a hypothesis to be generated. Since all hypotheses are generated in parallel, an initial level of verification is performed by clustering the hypotheses in the model transform parameter space. Clusters indicate hypotheses with a large degree of mutual support. A multiscale Hough-like method is used to do the clustering. From the clusters, an aggregate transformation is determined, and a refined hypothesis is generated. Verification consists of comparing transformed model line segments with image segments.

2.3 Predict-Observe-Backproject Paradigm

As its name implies algorithms based on the predict-observe-backproject paradigm, or POBP, has three basic steps that are iterated. First, *predictions* are made about what may be observed in the image. The predictions are based on the models and the current state of the algorithm. Typically, the type, attributes, and pose of features are predicted. Next, the image is examined for *observed* data that match closely with the predictions. Assuming such a match is found, the implications of these matches are "backprojected" into the space of scene instances, resulting in one of two outcomes: either the scope of feasible scene instances is narrowed by the additional constraints induced by the match with the current set of feasible scene instances, or, the induced constraints result in an inconsistent solution. When an inconsistent solution is generated, POPB algorithms typically backtrack to a previously visited feasible solution, i.e., one containing a non-empty set of scene instances. This loop is iterated, usually resulting in the tree-structured search that is the hallmark of the POBP.

2.3.1 Backprojection

The heart of the POBP is *backprojection*, the propagation of constraints induced by a match between a predicted feature and an observed feature. Strictly speaking, any algorithm that uses a match between model features and image features

to calculate a viewpoint is doing a form of backprojection, though not necessarily in its fullest sense. Therefore, all transformation clustering approaches and many HVP methods employ a limited form of backprojection. However, backprojection in its full sense requires a probability distribution on the space of scene instances. For example, suppose that an image feature has been assigned to match a predicted model feature. Assume that these features consist of triplets of primitive feature points. Further, for simplicity, assume that the features can be considered to be object-attached. Therefore, the features each possess six attributes, specifically, the x and y coordinates of each of the three primitive feature points comprising each compound feature. Thus, under weak perspective, a particular correspondence between the image feature and the object feature will yield a unique solution, up to reflection though the image plane, for the pose of the model that causes the predicted features to coincide with the observed image features. That is, if the observed feature's attributes are known with full certainty, then a point in the observed feature's attribute space maps to two points in the scene instance space, as shown in Fig. ??(a). In reality, however, a feature's attributes are never known with certainty due to noise and various other distortions. Rather, there is a probability distribution on the feature attributes. Through the mapping from feature attribute space to scene instance space, a probability distribution on the space of scene instances is induced. Denote this probability distribution as $P(s|\mathbf{f}^i \equiv \mathbf{f}^m)$, where s is a scene instance (which may be the null instance), $\mathbf{f}^i$ is the image feature, and $\mathbf{f}^m$ is the predicted model feature, and $\equiv$ indicates that $\mathbf{f}^i$ matches $\mathbf{f}^m$. Figure ??(b) illustrates the probabilistic definition of backprojection. This definition can be carried further to the case of n matching image and model features. In this case, the distribution is

$$P(s \mid \mathbf{f}^i_{j_1} \equiv \mathbf{f}^m_{k_1}, \mathbf{f}^i_{j_2} \equiv \mathbf{f}^m_{k_2}, \ldots, \mathbf{f}^i_{j_n} \equiv \mathbf{f}^m_{k_n}). \tag{1}$$

Explicitly calculating these conditional probability distributions has never been attempted owing both to the inherent intractability of the problem as well as to the fact that the distribution is dependent both on the details of the model and types of features used. What is typically done is to divide scene instance space into regions $\mathcal{R}$ and $\neg\mathcal{R}$ such that $P(s \in \neg\mathcal{R} \mid \mathbf{f}^i_{j_n} \equiv \mathbf{f}^m_{k_n}) < \epsilon$, and work with these regions rather than the probability distributions themselves. Figure ??(c) shows an example of such regions. Following Cass [Cas88a, Cas88b], in the following paragraph, we will refer to a region $\mathcal{R}$ as a *match region* because all

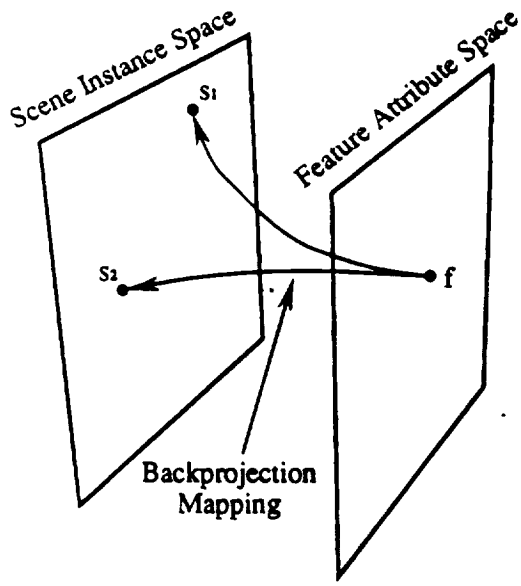


Figure 3: An illustration of noiseless backprojection. The vector of measured attributes of a projected model feature is **f**. In this example weak perspective is assumed, and, as explained in the text, features consist of triples of 2-d image points and 3-d model points. In this case, perfect, noiseless measurements are assumed. Thus, measured values for **f** imply that the model can have two possible poses. Therefore, **f** *backprojects* to two points s_1 and s_2 in scene instance space, as shown schematically.

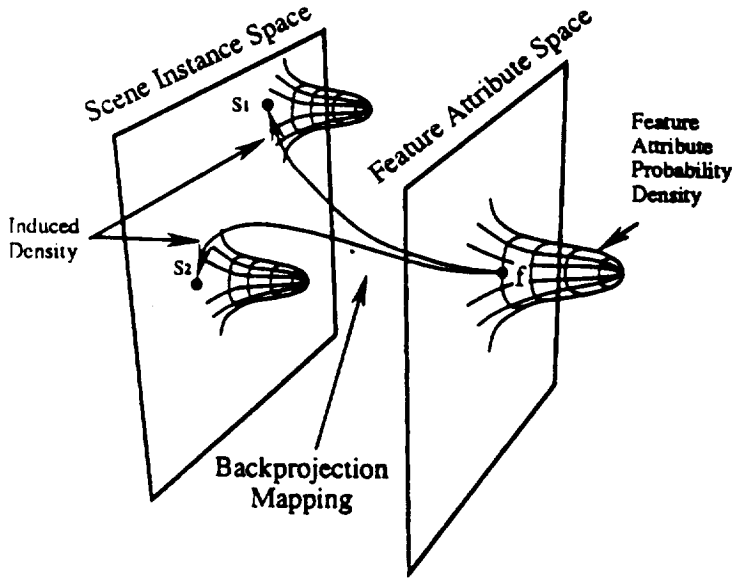


Figure 4: An illustration of backprojection in the presence of measurement error. As in the case of Fig. 3 above, the vector of measured attributes of a projected model feature is **f**. In this case, the measurement is not assumed to be perfect, i.e., there is a probability distribution on **f**. Backprojection of this distribution then induces a probability distribution on the space of scene instances.

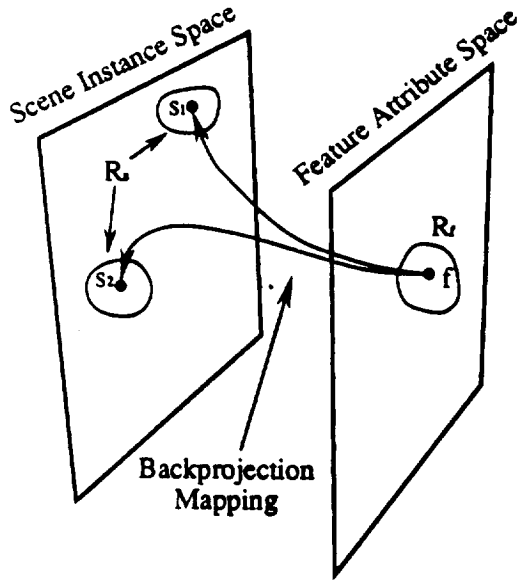


Figure 5: An illustration of a common approximation to full backprojection. The situation is as in Figs. 3 and 4. Typically, full back projection, as illustrated in Fig. 4, is replaced by a simple approximation, as shown above. Region R_f, containing most of the probability density of f, backprojects to region R_s, which, in turn, contains most of the induced probability density. Most methods that perform backprojection work with such regions, or further approximations to them.

scene instances $s \in \mathcal{R}$ project $\mathbf{f}^m$ to within a neighborhood of $\mathbf{f}^i$ defined such that the undistorted value of $\mathbf{f}^i$ falls in the neighborhood with probability $1 - \epsilon$.

Backprojection alone is the basis of only one recognition algorithm that we know of [Cas88a, Cas88b]. This algorithm bears a superficial similarity to the transformation clustering based methods discussed previously, but actually differs in important respects. The work addresses 2-d recognition: rigid image-plane rotations and translations of models are allowed. The viewing parameter space for this method is therefore three dimensional. However, the idea is general, and could be easily extended to recognition of 3-d objects from 2-d images using appropriately chosen object-attached features.

Features in this algorithm consist of evenly spaced points on an image or model boundary, along with the orientation at each point. Due to the isomorphism between the image domain and the model domain, the representation of model and image boundaries and their derived features are identical.

For each correspondence between an image feature and a model feature , a match region is defined; i.e., regions where the projection of an of a model feature will result in a predicted feature whose attributes are within a neighborhood of an observed feature. The size of the neighborhood models the amount of distortion caused by the imaging process. In [Cas88a, Cas88b], match regions are simply cylinders in the 3-d viewing parameter space. The dimensions of the cylinder are chosen so that the transformed feature lies within a neighborhood of the undistorted image feature with high probability. Consider, for every pairing of an image feature with a model feature for a particular model, the intersections of all of the match regions. Let every non-null intersection of two or more match regions be called an "intersection volume". Each intersection volume constitutes a hypothesis that the model appears at a transformation contained within the intersection volume with probability $1 - \epsilon$. Clearly, the best such volumes are likely to be those that are the result of the intersection of a large number of match regions, as such intersections volumes are comprised of the transformations that place a large number of features near to the true image feature with high probability. For each model, the intersection volumes that are comprised of the intersection of greater than a certain number of match regions are the hypotheses considered to be valid recognition results.

This algorithm is simple, elegant, and highly parallelizable, as was demonstrated by its implementation on a Thinking Machines Corp CM-1 Connection Machine [Hil87]. Match regions were represented by a uniformly sampled grid

of points inside each region so that the intersections could be accomplished efficiently in parallel. A more refined version of the method, also described in [Cas88a], uses approximations to the conditional probability distributions discussed previously to give even more accurate results.

2.3.2 The Predict Observe and Backproject Paradigm

Having examined backprojection, the heart of the POBP, in some detail, we can now complete describing the POBP.

Prediction can also be viewed as *forward projection*. Given a probability distribution on the error in an observed feature, backprojection induces a probability distribution on the set of scene instances. In contrast, in the case of prediction, or forward projection, the probability distribution on the set of scene instances induces a distribution on the attributes of a feature. This probability distribution can be used to match observations during the *observe* step of the cycle. That is, it allows us to assess the probability that a predicted feature falls within a neighborhood of an observed feature in feature-attribute space. If the probability is large, then the POBP may assign the predicted feature to match the observed feature. The implications of this match are then found by backprojecting the measurement error distribution of the observed feature by recomputing the conditional probability given in (1). Algorithms employing the POBP typically maintain a measure of the current goodness of the solution. One measure is the volume, in scene instance space, for a given model, that contains a sizable portion of the conditional probability distribution. If the volume of such a region goes to zero, the solution can be considered inconsistent, and backtracking is usually prescribed. If, on the other hand, a small region containing much of the conditional probability exists, it is likely to be correct. Thus, the POBP usually takes the form of a tree search, with each node representing an additional hypothesized match between a predicted feature and an observed feature. If the measure of the goodness of the solution increases above a threshold, then an object is recognized.

As mentioned previously, backprojecting the implications of a feature match is difficult for the case of 2-d data and 3-d objects. This may explain why there are few reported methods for using the POBP to recognize 3-d objects in 2-d images. In matched dimensionality domains, by contrast, full-fledged backprojection can be approximated in such a way that it becomes trivial. Thus, there are a larger number of tree structured methods in matched dimension

domains that fall into the category of POBP than in general domains. However, the backprojection portions of these algorithms are somewhat atrophied. These will be mentioned in later paragraphs.

Perhaps the two best examples of algorithms that employs the POBP for recognizing 3-d objects in 2-d intensity images are the methods of Goad [Goa83] and ACRONYM [BGB79, Bro81, Bro83, CCL84]. We feel that Goad's approach is better overall. In particular, Goad's method is actually has been shown to recognize 3-d objects whereas ACRONYM has never been shown to work on true 3-d scenes. Further, ACRONYM, while it has much to contribute, has serious flaws that would prevent it from ever becoming a practical system. The problems with Goad's approach are of a more subtle nature, problems that VGD has been designed to overcome. Therefore, we will describe Goad's system as the archetype of the POBP.

The features used by Goad's approach are line segments. 2-d line segments are assumed to be the result of the projection of one of the edges of one of the polyhedral models. Thus, the method uses object-attached features.

Models consist of a list of the 3-d edge segments comprising a polyhedron. Each 3-d model segment has an associated list of facets on a partition of the viewing sphere from which it is visible, called a "visibility locus". The viewing sphere is partitioned into 218 view regions. The locii are represented as bit strings that denote the union of some of the 218 view regions. Each visibility locus for each model is precomputed since the locus does not change during recognition.

At any time during the course of a solution, Goad's system maintains a current "locus of visibility", denoted L, which is the current set of viewpoints that could account for the visibility of the currently matched image and model line segments. The locus L is Goad's approximation to the conditional probability density described above.

At the start of the algorithm, all possible model edges are considered candidates to match any model edge. Once the initial match is made, the visibility locus L is initialized to the visibility locus of the model feature (the first backprojection). Prediction is accomplished in two steps. First, an unassigned model edge whose visibility locus has a non-null intersection with L is selected. The second part consists of two cases: (1) assume the edge is visible; and (2) assume that the edge is actually invisible. At first glance, this may seem to be contradictory since, if L intersects the visibility locus of the currently selected model edge, then it should be visible. Actually, this is not so, since the true viewpoint

of the model may lie inside a portion of L that lies *outside* of the intersection of L with the visibility locus of the edge. This situation is illustrated in Fig. 6. In the case where the edge is assumed to be visible, L is updated to be the intersection of itself with the visibility locus of the currently selected segment. The position and orientation of the projection of the model edge relative to the projection of the initially matched model edge is then computed as the viewpoint ranges over the current visibility locus. The range of relative locations and orientations, plus the known location of the first edge (since it has already been matched), and some account for measurement error, provides bounds on the location and orientation of image edges that could match the projection of the model edge. If, after attempting to extend the match further, the algorithm finds that the current match is inconsistent with the assumption that the current model segment is visible then the algorithm assumes that the current segment must be invisible after all. It then restores L to its state before the visibility assumption was made, and then updates L to be the intersection of itself with the *complement* of the visibility locus of the currently selected segment. If this intersection is empty, the algorithm backtracks to a previous choice point. If there is an intersection, the algorithm chooses a new model segment and continues with a new prediction as in the visible case.

Observation is simply the process of checking the list of detected image segments for those whose position and orientation fall into the bounds predicted for the model edge we want to match. If any such image features exist, extend the hypothesis to include one of them as a match.

Backprojection refines the visibility locus of the current hypothesis, L, by restricting it to a smaller locus, say L', that is consistent with the measured pose of the image feature that was matched to the currently selected model feature. This is done by computing the pose of the current model feature at all points in L and retaining those that result in the projection of the model feature having the same relative pose to the initially matched edge as measured from the image segment.

A hypothesis is accepted if its "reliability" is greater than threshold, and backtracking occurs if L becomes null or if the "plausibility" of the match falls below another threshold. "Reliability" is similar to "perceptual significance" as defined by Lowe [Low87a], and is thus synonymous with "non-accidentalness". Details on the calculation of the reliability measure are not given. However, it appears to be done in a manner very similar to Lowe's method. "Plausibility" is a measure of the likelihood that the edge detector would have missed the edges

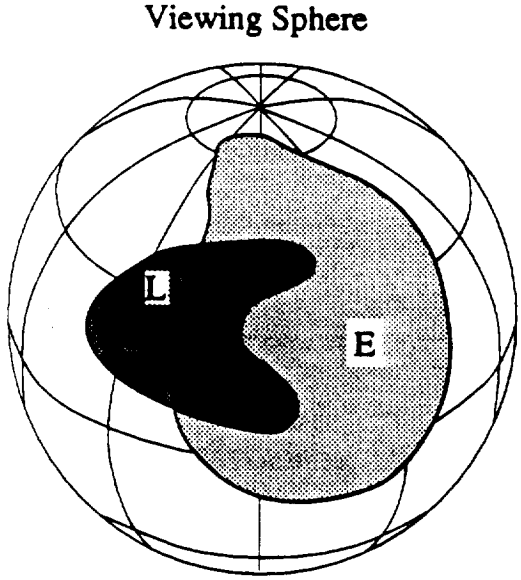


Figure 6: L is the visibility locus of the current solution, and E is the visibility locus of the model segment currently under consideration as a possible addition to the hypothesis. The dot in L denotes the correct viewpoint, under the assumption that the hypothesis so far is correct. Although E has a non-null intersection with L (the dark gray region), the edge is actually *not* visible since E does not contain the true viewpoint.

that were predicted to be visible in the current hypothesis. However, poor edge detection is not the only way that edges that are visible can fail to be detected; occlusion is another possibility. The algorithm does not take this into account, and thus is restricted to recognizing largely visible objects.

Since the relationships between the features are all relative angular relationships, the values of the viewing parameters not described by L, which include image-plane translation, rotation, and scaling, can be determined by examining any pair of edges. No fitting is done to improve the final estimate of the viewing parameters.

ACRONYM [BGB79, Bro81, Bro83, CCL84], an ambitious system that has been cited often in machine vision literature, also follows the POBP. The system is complex, and its description will be heavily abridged in this discussion. Objects are modeled via an "object graph" which consists of two subgraphs. The first is a subpart graph whose nodes are assemblies, and whose arcs are directed from complex to simple assemblies. The other subgraph, called the "restriction graph", allows generic classes to be represented by placing bounds on the dimensions, number, and relative poses of subassemblies. These two subgraphs are actually trees. The leaves of the trees are modeling primitives called "generalized cylinders". Generalized cylinders (GC's) are specified by a "spine" and a "sweeping rule". A GC is created by sweeping a cross-section, specified by the sweeping rule, along the spine of the GC. In the general case, the cross-section may change arbitrarily along the length of the spine, while remaining perpendicular to it. Thus, GC's are very flexible modeling elements. However, ACRONYM actually uses a small subclass of all possible GC's. ACRONYM allows GC's that have rectangular, hexagonal or circular cross-sections. Spines may be linear or circular; GC's with linear spines are allowed to have their dimensions linearly varied along the length of the spine, while those with circular spines must have constant cross-sections.

Perspective projection of ACRONYM's restricted class of GC's results in 2-d ellipses, trapezoids, and hexagons. The features that ACRONYM uses reflects this: ellipses and "ribbons" constitute ACRONYM's feature set. A ribbon is the 2-d analog of a a generalized cylinder. Ribbons are restricted to have linear spines and sweeping rules, implying that they are actually trapezoids. The trapezoids and ellipses are found by applying an edge detector then a linker followed by a line finder [NB80]. Ribbons and ellipses are detected from the line segment data, and become the nodes of the "observation graph". Arcs of the observation graph are relationships between features. Only connectivity between

ribbons was implemented. The matching in ACRONYM is based entirely on this ellipse-ribbon level of description.

Matching follows the POBP. At first, there are few, if any, constraints on the poses of the objects and their component subassemblies. Thus, the possible poses and shape attributes of the ellipses and trapezoids that could result from the projection of any of the object graphs vary widely, allowing many possible matches between predicted features and observed features. Later in the interpretation process, the constraints on the poses of the objects and their subassemblies reduce the variance in the attributes of the predicted features, reducing the number of observed features that are likely to match, as in Goad's method above. Predictions are placed in a "prediction graph" whose nodes consist of *features* and *bounds* on their attributes, or, recursively, other prediction graphs. Arcs describe relationships between the features, such as relative spine orientation and connectedness.

Observation consists of querying the observation graph for any features that match primitives in the prediction graph and that are consistent with any observed-feature to predicted-feature matches already in effect.

If an observed feature is assigned to match a predicted feature, then the attributes of the predicted feature (which may be quite loosely specified) are set equal to the observed feature. The implications of this match are *backprojected* in the form of tighter constraints on the pose of the objects and their subassemblies, reducing the possible ranges of the viewing parameters and the model parameters. The manner in which this is done is at the heart of ACRONYM, and, in this author's estimation, is one of the primary reasons that ACRONYM never worked on true 3-d images. All transformations are represented symbolically by products of primitive transformations, which, in turn, are parameterized by various angles and displacements. The forward projection and backprojection mappings are, therefore, described by very complex, coupled sets of symbolic trigonometric equations. A symbolic algebra manipulation system was employed to propagate the effects of fixing a set of feature attributes back into the object graph. The same system was used to propagate the new bounds on the attributes of the predicted features that result from the tightening of the backprojected constraints. The problem is that solving such systems of equations exactly is intractable, so the system was forced to make liberal approximations. The approximations could be so bad that the resulting upper and lower bounds are useless in constraining the search.

While a great deal of effort was put into ACRONYM to handle general 3-d

scenes, none of the results published have ever shown ACRONYM is able to recognize general 3-d objects. Most results show interpretation of aerial scenes from a fixed viewpoint, a problem that could be solved much more simply. Extensions were shown for a set of switch parts [CCL84], however, only stable poses were allowed. These situations reduce to 2-d recognition. Further, the use of a restricted set of GC's hurt the system, as well as staying at or above the level of ribbons and ellipsed in the level of data abstraction, preventing lower level information from strengthening or weakening interpretations.

Another POBP method for recognizing 3-d objects in 2-d intensity data is described in [BAM86]. Features in this method are corners and line segments. Large "blobs", or regions enclosed by these features, are also found. The search is tree-structured through the space of possible model-feature to observed feature pairings. Backprojection consists of using either a least-squares method or a method based on the ratios of the areas of blobs to find the transformation that best maps the model features onto the observed features. No effort is made to approximate the conditional probability density; a single point in transformation space is found, with the implicit assumption that some neighborhood of this point is also valid. The tree search is guided by an admissible heuristic that encodes information about the geometric mismatch between predicted features and observed features, as well as noise in the feature detection process. Results are given for images of wholly visible fighter planes.

There are two probable reasons for the relative scarcity of methods that use the POBP to recognize 3-d objects in 2-d intensity images. One is that performing backprojection is very difficult in all but the most simplified cases, even when object-attached features are employed. The second is that, empirically, it appears that reliable recognition of 3-d objects in 2-d intensity images requires that, at some level, the prediction be very *complete*. In contrast, most POBP-based methods employ a sparse feature-based representation of the predictions of the objects. They do this primarily to cut the combinatoric complexity, although the complexity of performing a tree search in an inhomogeneous feature space is also likely to be a factor. The incremental verification scheme of VGD is directed toward handling this problem, while retaining robust performance.

2.3.3 Matched Dimension Domains and the Predict Observe Backproject Paradigm

POBP-based methods that operate in matched dimension domains are more plentiful than methods that operate in general domains. This is partly because backprojection becomes simpler, amounting to nothing more than alignment of model features with observed image features to within measurement errors. In addition, less complete representations of 2-d predictions can be used and still result in a a robust approach, allowing a tree search in a homogeneous feature space to be used.

There are several well-known examples of such approaches. One of these is the method described in the following papers by Grimson and Lozano-Perez [Gri88a, GLP84, GLP85, GLP87]. This method is applied both to recognition of 3-d objects from sparse range data, as well as recognition of 2-d objects in 2-d intensity images. The approach for recognizing 3-d objects in range data 3-d approach is described; the 2-d case recognition case is completely analogous. Both models and images are represented in terms of their features, planar patches. The method searches an "interpretation tree" (IT), whose nodes consist of all possible strings of observed-feature to model-feature pairings. The length of the strings corresponds to the depth of the node in the tree. Observed features may also be paired with a "null face", indicating that the feature is spurious with respect to the current model. The pairings are first "filtered" by clustering the model-feature to image-feature pairs in a subspace of the full 6-d viewing parameter space using a Hough approach. Branches of the IT that have pairings that belong to the same cluster are regarded as more likely to represent valid interpretations. Early versions of the method [GLP84] used decoupled geometric constraints between pairs of observed features and model features to prune the IT, i.e., a branch is pruned if the latest set of two pairs of model features and observed features do not have the same relative geometric relationship, to within measurement error. Thus, the search never explicitly worked in viewing parameter space, and so there was no backprojection step. More recently, however, the authors used full coupled constraints that consist of determining the viewing parameters based on the features matched so far. This constitutes a simple form of backprojection, to prune the IT. If backprojection indicates that there is no set of viewing parameters that can map the model faces onto the observed patches to within measurement tolerances, then the branch is pruned. The search of the IT proceeds in depth-first fashion until the

interpretation becomes valid, or is pruned. An interpretation is considered valid if it explains a large enough portion of the area of the range image.

This method has been extended to work with curved objects in 2-d [Gri89, Gri88c] and parameterized 2-d objects [Gri88b]. There are a number of MDD methods that are very similar to the one described above. They include [Gre86, KK87, FH83, AFF84, AFFT85, AF86, MC88, PILSL88].

An interesting algorithm is described by Van Hove in [Hov87]. This method is very similar to the early version of Grimson and Lozano-Perez's algorithm wherein the branches of the tree search were pruned using pairwise geometric relations between features. The novel part of Van Hove's approach is that this idea is used for 3-d recognition from 2-d intensity images. This is suprising because such geometric relations are highly dependent on the viewpoint, and may take on a wide range of values over the viewing sphere. Such wide ranges on values of the attributes of the geometric relations reduce their pruning power. Grimson and Lozano-Perez's method needed only to account for measurement error, which it typically much smaller in magnitude than the variations due to changes in viewpoint. Since Van Hove's method is very similar to Grimson's method, and since it performs no backprojection during its tree search phase, it is discussed here rather than earlier in this section.

Van Hove's method employs pairs of linear edge fragments as features. The key to the method is a preprocessing step where each model is rotated to all possible views in a densely populated sampling of the viewing sphere. The range over which each feature's attributes vary is recorded and stored as part of the model. The space of image-feature to model-feature pairings is searched, as in Grimson and Loazano-Perez, without the use of Hough clustering as a heuristic. At each node, a model feature is allowed to match an image feature only if the image feature's attributes are within the allowable precomputed bounds of the model feature's attributes. If not, the branch is pruned.

Were the tree search the only means of finding valid interpretations, it is likely that Van Hove's method would not work well. As it is, the tree search is used only to *generate* good hypotheses for testing, somewhat in the spirit of Bolles and Cain [BC82]. During the hypothesis test, the features matched by the tree search algorithm are used to compute an estimate of the viewing parameters of the model. These, in turn, are used to generate a more complete prediction of the appearance of the model, which is then compared to the image to decide which hypotheses represent valid interpretations.

2.4 Global Feature Methods

A *global* feature is a feature whose attributes are calculated based on the entire region that an object occupies in an image. Typically, global features are *shape descriptors*, i.e., their attributes encode the *shape* of the region that the object occupies. Usually they are constructed to be invariant to image-plane translations, rotations, and scalings. Thus, ideally, their attributes change only when the shape of the object changes. Under weak perspective, the usual imaging assumption for such methods, this can happen only when the viewpoint changes.

By their nature, methods that rely on global features assume that an object has been accurately segmented from the background, leaving only the problem of *identification*. In certain domains, this is a reasonable assumption. In most domains, however, such segmentation is very difficult. Because of this, global feature methods are of little practical value. However, in their favor, unlike all other methods that have been examined so far, these methods do not employ the object-attached feature assumption. This follows because the region over which a global feature is calculated is the silhouette of the object, and, as shown earlier, any feature depending on the shape of the silhouette cannot be object-attached.

Since global features are not object-attached, analytic methods for determining the viewing parameters of the model given the values of the feature's attributes do not exist. Therefore, the mapping from feature attributes to viewpoints must be precomputed and stored in a form of *multiview model*. Indeed, it is a trademark of global feature methods that they employ some type of multiview model.

There are many types of global features. Some common examples are area, perimeter, compactness (ratio of area to perimeter squared), Fourier descriptors [PF77, Gra72] and Wigner distributions [JW84], and moment invariants [Hu62, Tea80, AMP84, TC88].

Matching in global feature methods is usually very simple; most of the effort goes into computing the features. The preprocessing stage creates a multiview feature representation for each model by calculating the feature vector for each model for a large number of views, usually approximately uniformly spaced over the viewing sphere. Most methods do this by graphically rendering the silhouette of 3-d CAD models of the objects, although some have simply taken images of a physical model over the viewing sphere [DBM77]. At recognition time, an image is processed by segmenting the object region and calculating the

feature vector from it. Then, the feature vectors stored in the multiview models are compared to the image feature vector using some distance measure on the feature space. The model and view with the most similar global feature vector to the image feature vector is taken to be the correct identification of the object. Typically, no further verification is done.

A good example of a global feature method is that described by Dudani *et al* in [DBM77]. The features used by this method are *moment invariants*. There are several flavors of moments and invariants. Dudani *et al* use the standard central moments and the invariants found by Hu [Hu62] using the theory of algebraic invariants. A moment is defined as:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q \rho(x,y)\,dx\,dy, \qquad p,q = 0,1,2,\cdots, \tag{2}$$

where $\rho(x,y)$ is an image function. In the method at hand, if $\mathcal{S}$ is the set of silhouette points in the image, then $\rho = 1$ if $(x,y) \in \mathcal{S}$, and $\rho = 0$ otherwise. The central moments are defined by

$$\mu_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x-\overline{x})^p (y-\overline{y})^q \rho(x,y)\,dx\,dy, \qquad p,q = 0,1,2,\cdots \tag{3}$$

where

$$\overline{x} = m_{10}/m_{00}, \overline{y} = m_{01}/m_{00}.$$

Central moments are invariant to translations of the silhouette. Using the theory of algebraic invariants, it is possible to combine the central moments so that they are invariant to image plane rotation and scaling as well. For example, the invariants of order two, i.e., $p+q=2$, are:

$$\mu_{02} + \mu_{20},$$

$$(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2.$$

Dudani *et al* used the seven lowest order invariants. Higher order invariants have been shown to be sensitive to noise [Wie83, AMP84, AMP85, TC88]. The feature vector was fourteen elements long: seven of the invariants were computed with $\mathcal{S}$ as the entire silhouette region (while the other seven were computed using $\mathcal{S}$ as only the points on the silhouette boundary.

A multiview model was constructed by taking images of the models to be recognized at 5° increments of the Euler angles parameterizing the viewing

directions, and, for each view, calculating and storing the resulting fourteen element feature vector.

Recognition consisted of taking the binary silhouette, calculating its 14 moment invariants, and searching the multiview models for the feature vectors that are most similar using the distance-weighted k-nearest neighbor rule [Dud76]. The viewing direction that yielded the most similar feature is reported as the system's recognition result. This system does not actually make a decision about whether the object is present or not, and so arguably, does not perform true recognition.

Other global, moment-based methods include those described in [RT89, TR87, BF86, Ree81, RPT85]. There are a number of methods based on normalized Fourier descriptors as well [WM80, WW80, WMF81, Kuh84, SD71].

2.5 Optimization-Based Methods

Optimization based approaches are superficially similar to VGD's tracking module. These methods usually generate some kind of global disparity measure based on the shape-disparity between curves or regions in the image and the rendering of curves or silhouettes derived from a 3-d model. The disparity measure is a function of the viewing parameters. Since the viewing parameters form a continuous space, and the disparity measures are themselves made to be smooth functions of the viewing parameters, continuous optimization procedures may then be used to search the space of scene instances for the minimum of the disparity measure. If the disparity resulting from applying the optimization procedure is small enough, an object is recognized.

While the approach is, in principle, sound, there are a number of difficulties that plague these methods. The most serious is the problem of local minima. Any continuous optimization procedure uses local knowledge of the similarity function to drive its search. Unfortunately, local information yields no information enabling a local minimum to be distinguished from the global minimum, thus there is no way for an optimization procedure to distinguish a local mimima from a global mimima. Therefore, these methods often become trapped in local minima of the disparity measure.

Another problem is the disparity function itself. The nature of the disparity measure impacts the number and severity of the local minima, as well as whether the method can find partial instances of objects. The problem of local mimima is often dealt with by starting the optimization at many different points in scene-

instance space, choosing the best result, and hoping that the best result is the global minimum. This is both time-consuming and unreliable. Recognition of partial objects is not possible with the reported methods.

While optimization-based methods have difficulties, they have the advantage that any features, not just object-attached features, can be used. If their other problems could be solved, they would be able recognize wider classes of objects than other methods. The tracking approach of VGD goes much of the way toward solving these problems, opening the door to this goal.

A recent optimization-based method possessing some parallels with VGD is described in by Stark *et al* [SEB88]. Models in this method are polyhedra and an associated *aspect graph*. As mentioned previously, an aspect graph is a set of topological equivalence classes over the sphere of viewing directions. In this case, the edges of the polyhedral models are rendered under perspective, with hidden lines removed, and the resulting views are grouped by the equivalence of the topology of the resulting set of 2-d line segments. Each such equivalence class is called a "cell". Associated with each cell is a set of viewing parameters that generate a "prototype" instance of the object. The essence of the method is to use these prototype model instances as starting points for optimization. The authors argue that the aspect graph provides a partition of viewing parameter space that will be likely to have separate local minima. Thus, the optimization is started from *every* prototype model instance, and it is constrained to remain in the cell during the optimization. The best result is then selected as the recognized object.

The idea of attempting to systematically isolate local minima is a good one, and is somewhat similar to the approach of VGD. However, VGD uses a feature indexed associative memory to retrieve *only* those views that have matching features rather than attempting to start from every possible stored view, or aspect, as this method does.

While there are some parallels to VGD the optimization portion of the algorithm is rather different. The system processes an image to extract a line drawing of the image, and, treating it as a graph with vertices as nodes and segments as arcs, selects a unique subset of the set of all possible circuits in the graph. These circuits are the "elementry" circuits of the graph, and correspond to the faces of the imaged polyhedron. Next, a Fourier descriptor representation of each such elementry circuit is computed. The figure of merit for the match is computed by projecting the model using the current set of viewing parameters (as obtained from the optimization algorithm), and Fourier descriptor features

are computed as for the image features. The algorithm examines all possible pairs of image circuit Fourier descriptor features with those derived from the model to determine the "best" match. Using this match, a "figure of merit" is reported (not described in the paper) that is used as the cost function. The cost function is optimized using the method of damped least squares.

Note that this method requires the entire object to be visible. In addition, the method works only for simple polyhedra, as complex polyhedra have intractably large aspect graphs.

There are a number of earlier examples of optimization-based methods. The method described in [WS82] uses the Levenberg-Marquardt method to optimize the squared distance between the Fourier descriptor representations of the image edge contours and the projection of a space curve. The approach described in [HW75] uses gradient descent to optimize a matching metric consisting of the sum of distances between corresponding points on the image edge contours and the silhouette outline. The system described in [MGA88] is interesting because it describes connectionist matching network, and, further, includes both a subpart hierarchy for models and a class hierarchy, expressed in the form of **isa** and **ina** links in the model network. Possible instantiations of model entities are connected to all possible matching model parts and subparts by matching nodes. The strength of the matching nodes are adusted to maximize the consistency of the match. The consistency of a match is computed using "consistency rectangles" which, essentially, describe the similarity of binary relations in the image to binary relations in the model.

Local minima are reported to be problematic in all of the methods described above, resulting in limited success. In all cases, continuous optimization methods were used. Discrete optimization methods, such as simulated annealing [Rut89] and dynamic programming [ATW88] could be used, but they are more inefficient than continuous methods. A better solution is to use additional information to provide good guesses as to the location of the global minima. This idea is at the heart of VGD, and opens the way to the recognition of occluded objects using general, not just object-attached features.

References

[AB86] H. Asada and M. Brady, "The curvature primal sketch," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, pp. 2–14, January 1986.

[AF86] N. Ayache and O. D. Faugeras, "HYPER: A new approach for the recognition and positioning of two-dimensional objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 44–54, January 1986.

[AFF84] N. Ayache, O. Faugeras, and B. Faverjon, "A geometric matcher for recognizing and positioning 3-d rigid objects," in *Proceedings of the SPIE Conference Intelligent Robots and Computer Vision (Vol. 521)*, pp. 152–159, 1984.

[AFFT85] N. Ayache, O. Faugeras, B. Faverjon, and G. Toscani, "Matching depth maps obtained by passive stereo," in *IEEE International Conference on Robotics and Automation*, pp. presented, not in proceedings, 1985.

[AMP84] Y. S. Abu-Mostafa and D. Psaltis, "Recognitive aspects of moment invariants," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 698–706, November 1984.

[AMP85] Y. S. Abu-Mostafa and D. Psaltis, "Image normalization by complex moments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-7, pp. 46–55, January 1985.

[Att54] F. Attneave, "Some informational aspects of visual perception," *Psychology Review*, vol. 61, pp. 183–193, 1954.

[ATW88] A. A. Amini, S. Tehrani, and T. E. Weymouth, "Using dynamic programming for minimizing the energy of active contours in the presence of hard constraints," in *Proceedings of the International Conference on Computer Vision*, pp. 95–98, IEEE, 1988.

[Bal81] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111–122, 1981.

[BAM86] J. Ben-Arie and Z. A. Mieri, "3-d objects recognition by state space search: Optimal geometric matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recgnition*, pp. 456–461, 1986.

[BBR83] A. H. Bond, R. S. Brown, and C. R. Rowbury, "The effect of evironmental variation upon the performance of a second generation industrial vision system," in *Proceedings of the Cambridge Symposium on Intelligent Robots and Computer Vision*, SPIE, November 1983.

[BC82] R. C. Bolles and R. A. Cain, "Recognizing and locating partially visible objects: The local-feature-focus method," *International Journal of Robotics Research*, vol. 1, pp. 57–82, Fall 1982.

[Ben75] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[BF86] B. Bamieh and R. J. P. D. Figueiredo, "A general moment-invariants/attributed-graph method for three-dimensional object recognition from a single image," *IEEE Journal of Robotics and Automation*, vol. RA-2, pp. 31–41, March 1986.

[BGB79] R. A. Brooks, R. Greiner, and T. O. Binford, "The ACRONYM model-based vision system," in *Proceedings of the International Joint Conference on AI*, (Tokyo, Japan), pp. 105–113, 1979.

[BH86] R. C. Bolles and P. Horaud, "3DPO: A three-dimensional part orientation system," *International Journal of Robotics Research*, vol. 5, pp. 3–26, Fall 1986.

[BHH83] R. C. Bolles, P. Horaud, and M. J. Hannah, "3DPO: A three-dimensional part orientation system," in *Proceedings of the 8th IJCAI*, pp. 355–359, 1983.

[Bro81] R. A. Brooks, "Symbolic reasoning among 3-d models and 2-d images," *Artificial Intelligence*, vol. 17, pp. 285–349, 1981.

[Bro83] R. A. Brooks, "Model-based three-dimensional interpretations of two-dimensional images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, pp. 105–113, March 1983.

[BS85] D. H. Ballard and D. Sabbah, "Viewer independent shape recognition," in *Conference on Pattern Recognition and Image Processing*, pp. 67–71, IEEE, 1985.

[BU88] R. Basri and S. Ullman, "The alignment of objects with smooth surfaces," in *Proceedings of the International Conference on Computer Vision*, pp. 482–488, IEEE, 1988.

[CA87] C. H. Chien and J. K. Aggarwal, "Shape recognition from single silhouettes," in *IEEE First International Conference on Computer Vision*, pp. 481–490, IEEE, 1987.

[Cas88a] T. A. Cass, "Parallel computation in model-based recogntion," Master's thesis, MIT, May 1988.

[Cas88b] T. A. Cass, "A robust parallel implementation of 2d model-based recognition," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 879–884, IEEE, 1988.

[CCL84] C. K. Cowan, D. M. Chelberg, and H. S. Lim, "ACRONYM model-based vision in the intelligent task automation project," in *First IEEE Conference on AI Applications*, pp. 105–113, IEEE, 1984.

[Chi89] C.-H. Chien, "Model reconstruction and shape recognition from occluding contours," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 372–389, April 1989.

[CMST88] C. I. Connolly, J. L. Mundy, J. R. Stenstrom, and D. W. Thompson, "Matching from 3-d range models into 2-d intensity scenes," in *Proceedings of the International Conference on Computer Vision*, pp. 65–72, IEEE, 1988.

[DBM77] S. A. Dudani, K. J. Breeding, and R. B. McGhee, "Aircraft identification by moment invariants," *IEEE Tranactions on Computers*, vol. 26, pp. 39–46, January 1977.

[DKZ79] J. D. Desseimoz, M. Kunt, and J. M. Zurcher, "Recognition and handling of overlapping industrial parts," in *Proceeding of the Ninth International Symposium on Industrial Robots*, pp. 357–366, March 1979.

[Dud76] S. A. Dudani, "Distance weighted k-nearest neighbor rule," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 6, pp. 325–327, April 1976.

[Ett88] G. J. Ettinger, "Large hierarchical object recognition using libraries of parameterized model sub-parts," in *Conference on Computer Vision and Pattern Recognition*, pp. 32–41, IEEE, 1988.

[FH83] O. Faugeraus and M. Hebert, "A 3d recognition and positioning algorithm using geometrical matching between surface primitives," in *Proceeding of the 7th International Joint Conference on Artificial Intelligence*, 1983.

[FHK+82] T. J. Fang, Z. H. Huang, L. N. Kanal, B. Lambird, D. Lavine, G. Stockman, and F. L. Xiong, "Three dimensional object recognition using a transformation clustering technique," pp. 678–681, IEEE, 1982.

[Fis83] R. B. Fisher, "Using surfaces and object models to recognize partially obscured objects," in *International Joint Conference on AI*, pp. 989–995, 1983.

[Fre77] H. Freeman, "Shape description via the use of critical points," in *Conference on Pattern Recognition and Image Processing*, pp. 168–174, IEEE, June 1977.

[Ger88] G. Gerig, "Linking image-space and accumulator-space: A new approach for object-recognition," in *Proceedings of the International Conference on Computer Vision*, pp. 112–117, IEEE, 1988.

[GL87] R. R. Goldberg and D. G. Lowe, "Verication of 3-d parametric models in 2-d image data," in *Proceedings of the Workshop on Computer Vision*, pp. 255–257, IEEE, 1987.

[GLP84] W. E. L. Grimson and T. Lozano-Perez, "Model-based recognition and localization from sparse range or tactile data," *International Journal of Robotics Research*, vol. 3, pp. 3–35, Fall 1984.

[GLP85] W. E. Grimson and T. Lozano-Perez, "Recognition and localization of overlapping parts from sparse data in two and three dimensions," in ?, pp. 61–66, IEEE, 1985.

[GLP87] W. E. L. Grimson and T. Lozano-Perez, "Localizing overlapping parts by searching the interpretation tree," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, pp. 469–482, July 1987.

[Goa83] C. Goad, "Special purpose automatic programming for 3d model-based vision," in *Proceedings of the DARPA Image Understanding Workshop*, (Arlington, VA), pp. 94–104, 1983.

[Gra72] G. H. Granlund, "Fourier preprocessing for hand print character recognition," *IEEE Transactions on Computers*, vol. 21, pp. 195–201, February 1972.

[Gre86] K. Grebner, "Model based analysis of industrial scenes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 28–31, 1986.

[Gri88a] W. E. Grimson, "The combinatorics of object recognition in cluttered environments using constrained search," in *Proceedings of the International Conference on Computer Vision*, pp. 218–227, IEEE, 1988.

[Gri88b] W. E. Grimson, "On the recognition of parameterized 2-d objects," *International Journal of Computer Vision*, vol. 3, pp. 353–372, 1988.

[Gri88c] W. E. L. Grimson, "On the recognition of curved objects," in *IEEE Conference on Robotics and Automation*, pp. 1414–1420, 1988.

[Gri89] W. E. Grimson, "On recognition of curved objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 632–643, June 1989.

[GTM87] P. G. Gottschalk, J. L. Turney, and T. N. Mudge, "Two-dimensional partially visible object recognition using efficient multidimensional range queries," in *Proceedings of the IEEE Conference on Robotics and Automation*, pp. 1582–1589, 1987.

[GTM89] P. G. Gottschalk, J. L. Turney, and T. Mudge, "Efficient recognition of partially visible objects using a logarithmic complexity matching technique," *International Journal of Robotics Research*, December 1989.

[Guz69] A. Guzman, "Decomposition of a visual scene into three-dimensional bodies," in *Automatic Interpretation and Classification of Images* (A. Grasseli, ed.), New York: Academic Press, 1969.

[Hea86] D. J. Healy, "A skeletonizing algorithm with improved isotropy," in *Proceedings of the Conference on Visual Communications and Image Processing*, pp. 138–145, SPIE Vol. 707, 1986.

[HH87] C. Hansen and T. Henderson, "CAGD-based computer vision," in *Proceedings of the Workshop on Computer Vision*, pp. 100–105, IEEE, 1987.

[HH88] C. Hanson and T. Henderson, "Towards the automatic generation of recognition strategies," in *Proceedings of the International Conference on Computer Vision*, pp. 275–279, IEEE, 1988.

[Hil87] D. W. Hillis, "The connection machine," *Scientific American*, vol. 256, pp. 108–115, June 1987.

[Hov87] P. V. Hove, "Model-based silhouette recognition," in *Proceedings of the Workshop on Computer Vision*, pp. 88–93, IEEE, 1987.

[HT85] J. J. Hopfield and D. W. Tank, "?," *Biological Cybernetics*, vol. 52, pp. 141–152, 1985.

[Hu62] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE Transactions on Information Theory*, vol. 8, pp. 179–187, February 1962.

[HU88] D. P. Huttenlocher and S. Ullman, "Object recognition using alignment," in *International Conference on Computer Vision*, pp. 102–111, IEEE, 1988.

[Huf71] D. A. Huffman, "Impossible objects as nonsense sentences," in *Machine Intelligence 6* (B. Meltzer and D. Mitchie, eds.), Edinburgh: Edinburgh University Press, 1971.

[Hut88] D. P. Huttenlocher, *Three-Dimensional Recognition of Solid Objects from a Two-Dimensional Image*. PhD thesis, MIT, April 1988.

[HW75] H. Hemami and F. C. Weimer, "Identification of three-dimensional objects by sequential image matching," in *Proceedings of the Conference on Computer Graphics*, pp. 273–278, IEEE, 1975.

[Hwa87] V. S. S. Hwang, "Recognition of two dimensional objects using hypothesis integration technique," in *Proceedings of the Workshop on Computer Vision*, pp. 106–111, IEEE, 1987.

[Jac87] D. W. Jacobs, "GROPER: A grouping based recognition system for two dimensional objects," in *Proceedings of the Workshop on Computer Vision*, pp. 164–169, IEEE, 1987.

[JW84] L. Jacobson and H. Wechsler, "A theory for invariant reconition in the frontoparallel plane," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 325–331, May 1984.

[Kan78] T. Kanade, "A theory of Origami world," Tech. Rep. CMU-CS-78-144, Computer Science Department, Caregie Mellon University, 1978.

[KJ86] T. F. Knoll and R. C. Jain, "Recognizing partially visible objects using feature indexed hypotheses," *IEEE Journal of Robotics and Automation*, vol. 2, pp. 3–13, March 1986.

[KJ87] T. F. Knoll and R. C. Jain, "Learning to recognize objects using feature indexed hypotheses," in *Proceedings of the International Conference on Computer Vision*, pp. 552–556, IEEE, 1987.

[KK85] M. W. Koch and R. L. Kashyap, "A vision system to identify occluded industrial parts," in *Conference on Pattern Recognition and Image Processing*, pp. 55–60, IEEE, 1985.

[KK87] M. W. Koch and R. L. Kashyap, "Using polygons to recongize and locate partially occluded objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9; pp. 483–494, July 1987.

[KSSS86] A. Kalvin, E. Shonberg, J. T. Schwartz, and M. Sharir, "Two-dimensional, model-based boundary matching using footprints," *International Journal of Robotics Research*, vol. 5, pp. 38–55, Winter 1986.

[Kuh84] F. P. Kuhl, "Global shape recognition of 3-d objects using a differential library storage," *Computer Vision, Graphics and Image Processing*, vol. 27, pp. 97–114, 1984.

[LHD88] S. Linnainmaa, D. Harwood, and L. S. Davis, "Pose determination of a three-dimensional object using triangle pairs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 634–647, September 1988.

[Low85] D. G. Lowe, *Perceptual Organization and Visual Recognition.* Kluwer Academic Publishers, 1985.

[Low87a] D. G. Lowe, "Three-dimensional object recognition from single two-dimensional images," *Artificial Intelligence*, vol. 31, pp. 355–395, 1987.

[Low87b] D. G. Lowe, "The viewpoint consistency constraint," *International Journal of Computer Vision*, vol. 1, no. 1, pp. 57–72, 1987.

[LSW88a] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson, "Object recognition by affine invariant matching," in *Conference on Computer Vision and Pattern Recognition*, p. 335, IEEE, 1988.

[LSW88b] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson, "On recognition of 3-d objects from 2-d images," in *IEEE Conference on Robotics and Automation*, pp. 1407–1413, 1988.

[LW88a] Y. Lamdan and H. Wolfson, "Geometric hashing: A general and efficient model-based recognition scheme," in *Proceedings of the International Conference on Computer Vision*, pp. 238–248, IEEE, 1988.

[LW88b] Y. Lamdan and H. J. Wolfson, "Geometric hashing," Tech. Rep. Robotics Report No. 152, New York University Dept. of Computer Science Robotics Research Laboratory, May 1988.

[MC88] D. W. Murray and D. B. Cook, "Using the orientation of fragmentary 3-d edge segments for polyhedral object recognition," *International Journal of Computer Vision*, vol. 2, pp. 153–169, 1988.

[MF75] P. M. Merlin and D. J. Farber, "A parallel mechanism for detecting curver in pictures," *IEEE Transactions on Computers*, vol. 24, pp. 96–98, January 1975.

[MGA88] E. Mjolsness, G. Gindi, and P. Anandan, "Optimization in model matching and perceptual organization: A first look," Tech. Rep. RR-634, Yale Dept. of Computer Science, 1988.

[NB80] R. Nevatia and K. R. Babu, "Linear feature extraction and description," *Computer Vision, Graphics and Image Processing*, vol. 13, pp. 257–269, 1980.

[PD87] H. Plantinga and C. R. Dyer, "The Asp: A continuous viewer-centered representation for 3d object recognition," in *Proceedings of the International Conference on Computer Vision*, pp. 626–630, IEEE, 1987.

[Per77] W. A. Perkins, "Model-based vision system for scenes containing multiple parts," in *Fifth International Joint Conference on Artificial Intelligence*, pp. 678–684, 1977.

[Per78] W. A. Perkins, "A model-based vision system for industrial parts," *IEEE Transactions on Computers*, vol. 27, pp. 126–143, February 1978.

[PF77] E. Persoon and K.-S. Fu, "Shape discrimination using fourier descriptors," *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC-7, pp. 170–179, March 1977.

[PILSL88] G. Pasquariello, M. Iannotta, S. Losito, and G. Sylos-Labini, "A system for 3-d workpiece recognition," in *Proceedings of the International Conference on Computer Vision*, pp. 280–284, IEEE, 1988.

[Ree81] A. P. Reeves, "The general theory of moments for shape analysis and the parallel implementation of moment operations," Tech. Rep. TR-EE 81-37, Purdue University, October 1981.

[RFC88] T. C. Rearick, J. L. Frawley, and P. P. Cortopassi, "Using perceptual grouping to recognize and locate partially occluded objects," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 840–846, IEEE, 1988.

[Rob64] L. G. Roberts, "Machine perception of three-dimensional solids," in *Optical and Electro-Optical Information Processing*, pp. 159–197, MIT Press, 1964.

[RPT85] A. P. Reeves, R. J. Prokop, and R. W. Taylor, "Recognitive aspects of moment invariants," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (San Francisco, CA), pp. 452–457, June 1985.

[RT89] A. P. Reeves and R. P. Taylor, "Identification of three-dimensional objects using range information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 403–410, April 1989.

[Rut89] R. A. Rutenbar, "Simulated annealing algorithms: An overview," *IEEE Circuits and Devices Magzine*, pp. 19–26, January 1989.

[SD71] J. Sklansky and G. A. Davison, "Recognizing three-dimensional objects by their silhouettes," *SPIE Journal*, vol. 10, pp. 10–17, Oct/Nov/Dec 1971.

[SDH84] T. M. Silberberg, L. Davis, and D. Harwood, "An iterative Hough procedure for three-dimensional object recognition," *Pattern Recognition*, vol. 17, no. 6, pp. 621–629, 1984.

[SE85] G. Stockman and J. C. Esteva, "3d object pose from clustering with multiple views," *Pattern Recognition Letters*, vol. 3, pp. 279–286, July 1985.

[SEB88] L. Stark, D. Eggert, and K. Bowyer, "Aspect graphs and nonlinear optimization in 3-d object recognition," in *Proceedings of the International Conference on Computer Vision*, pp. 501–507, IEEE, 1988.

[Seg83] J. Segen, "Locating randomly oriented shapes from partial views," in *Proceedings of the SPIE Cambridge Syposium on Robot Vision and Sensory Controls*, pp. 676–684, SPIE, 1983.

[SHD84] T. M. Silberberg, D. Harwood, and L. S. Davis, "Object recognition using oriented model points," in *First IEEE Conference on AI Applications*, pp. 621–629; 1984.

[SKB82] G. Stockman, S. Kopstein, and S. Bennet, "Matching images to models for registration and object detection via clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, May 1982.

[STT87] Y. Sato, R. Tamano, and S. Tamura, "Connectionist approach to 3-d object recognition: Matching of parameterized models and monocular image," in *Proceedings of the Workshop on Computer Vision*, pp. 252–254, IEEE, 1987.

[SU88] D. Shoham and S. Ullman, "Aligning a model using minimal information," in *Proceedings of the International Conference on Computer Vision*, pp. 259–262, IEEE, 1988.

[TC88] C.-H. Teh and R. T. Chin, "On image analysis by the methods of moments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 496–513, July 1988.

[Tea80] M. R. Teague, "Image analysis via the general theory of moments," *Journal of the Optical Society of America*, vol. 70, pp. 920–930, August 1980.

[TFF88] L. W. Tucker, C. R. Feynman, and D. M. Fritzsche, "Object recognition using the connection machine," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 871–878, IEEE, 1988.

[TM87] D. W. Thompson and J. L. Mundy, "Three-dimensional model matching from and unconstrained viewpoint," in *Proceedings of the Conference on Robotics and Automation*, pp. 208–220, IEEE, 1987.

[TMV85] J. L. Turney, T. N. Mudge, and R. A. Volz, "Recognizing partially occluded parts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-7, pp. 410–421, July 1985.

[TR87] R. W. Taylor and A. P. Reeves, "Classification quality assessment for a generalized model-based object identification system," in *Proceedings of the IEEE Conference on Systems Man and Cybernetics*, pp. 412–417, IEEE, 1987.

[Tur74] K. J. Turner, *Computer Perception of Curved Objects Using a Television Camera*. PhD thesis, University of Edinburgh, 1974.

[Wal72] D. I. Waltz, *Generating Semantic Descriptions from Drawings of Scenes with Shadows*. PhD thesis, MIT AI Lab, 1972.

[Whi88] G. Whitten, "Vertex space and its application to model based object recognition," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 847–857, IEEE, 1988.

[Wie83] J. S. Wiejak, "Moment invariants in theory and practice," *Image and Vision Computing*, vol. 1, pp. 79–84, May 1983.

[WM80] T. P. Wallace and O. R. Mitchell, "Analysis of three-dimensional movement using fourier descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-2, pp. 583–588, November 1980.

[WMF81] T. P. Wallace, O. R. Mitchell, and K. Fukunaga, "Three-dimensional shape analysis using local shape descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-3, pp. 310–323, May 1981.

[WS82] L. T. Watson and L. G. Shapiro, "Identification of space curves from two-dimensional perspective views," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-4, pp. 469–475, September 1982.

[WW80] T. P. Wallace and P. A. Wintz, "An efficient three-dimensional aircraft recognition algorithm using normalized fourier descriptors," *Computer Vision, Graphics and Image Processing*, vol. 13, pp. 99–126, 1980.

[YMA80] K. R. Yam, W. N. Martin, and J. K. Aggarwal, "Analysis of scenes containing several occluding curvilinear objects," Tech. Rep. TR-135, The University of Texas at Austin Department of Computer Science, Austin, TX 78712, February 1980.

Appendix B

Review of 3D Model Acquisition Techniques

The modeling of three-dimensional objects using multiple camera views is an active research area in computer vision and is poorly defined at present. The goals of this research have ranged from rendering realistic environments for simulation, modeling of the world for intelligent robot navigation, acquiring three dimensional models for object recognition, to supplying model data for CAD and biomedical applications. Approaches for automatic model acquisition can be characterized along two basic data fusion methods 1) volume intersection of object silhouettes projected on the image planes, and 2) fusion of object surface patches reconstructed from multiple viewpoints. This appendix describe the various approaches that have been attempted to achieve these goals, and provide an outline of the respective research efforts classified along these two lines.

The different research efforts can also be described by the type of input data and the object representation scheme used in fusing the data. Intensity and range images are the two most commonly methods for inputting data. Intensity data is typically captured with video cameras, while range data is collected using either a laser-range camera, structure lighting, or stereopsis. The sensor used dictates the type of information that can be recovered. Structure lighting yields sparse but more easily registered range data points than dense range maps. Dense range maps are constructed slowly with a laser rangefinder that scans an object scan-line by scan-line. Conversely, intensity data can be instantaneously captured by inexpensive sensors. However, such images do not yield depth information directly. In addition, techniques such as stereopsis must be applied to recover range data.

To build up a complete model of an object, the data from different images must be first registered and then combined. Data has generally been registered using a) a fixture to hold an object at well-defined view angles, b) a fixed array of lights and cameras to capture the image data, or c) registration points on the surface of an object, where a sufficient number of points are shared between views to register the views. Multiple views of an object must be combined in order to capture the shape of a 3D object. Viewpoints can be constrained to 3 (orthogonal) or 13 positions to yield an efficient fusion algorithm. Fusion of data can be performed efficiently if the objects to be modeled are of some specific type, eg. polyhedral and cylindrical objects. Similarly, fusion can be simplified if object silhouettes can be obtained.

Finally, different representations have been used to model objects and for their reconstruction. The three basic categories have been volumetric, boundary, and wire frame

models. Object models can be further organized by decomposing an object into modeling primitives. For example, volumetric models have been used that have volume elements, i.e. voxels, as their basic modelling primitives. Surface patches, on the other hand, are used to construct surfaces in boundary models. Lastly, coarse models can be derived by composing an object from generic primitives, such as cylinders and ellipsoids. In order to capture accurate surface detail, the VGD system will employ boundary models of surface patches for the 3D objects to be modeled.

1 Volume Intersection

A very popular data structure used for volume intersection are octrees. Octrees are 3D counterparts of quadtrees [Sam85], a hierarchical data structure for image regions. An octree is a true volumetric model representing a 3D object space, defined by a right-handed object coordinate system, to contain a $2^n \times 2^n \times 2^n$ array of volume elements called voxels. A voxel, the 3D equivalent of a pixel (picture element), is assumed to contain a homogeneous volume of material and can be either occupied or vacant. It also defines the resolution of the modeling system. The usual shape of a voxel is a cube although a rectangular parallelepiped could be used in a more general system.

The volume of the array of voxels is called a 3D voxel universe and it is aligned with the object coordinate system. It is assumed that all objects of interest are within this universe and remain there during all operations. The space outside the universe is assumed to be void of all objects. An octree is an 8-ary tree structure generated by a recursive subdivision of the modeling universe into octants until homogeneous blocks of voxels are reached. If an octant does not consist entirely of the same type of voxels, then it is further subdivided until homogeneous cubes, possibly single voxels, are obtained. The root of an octree is at level 0 and the voxels are at level n.

The generation of octree models of 3D objects consists of three steps. The first step generates conic octree volumes with the silhouettes of objects projecting from the image plane into the octree universe. The second step combines a sequence of such conic volumes into a model of the objects using simple volume intersection. The last step enhances an octree model by labeling individual objects, adding surface-normal vectors, and mapping intensity textures and other intrinsic properties on the surfaces of these objects.

After extracting the contour information and constructing the view cones, the volume intersection step can be done either in 3-space, the octree universe, or in the individual image planes. Other variations include the permitted number and flexibility of viewpoints during model construction. Restricting the possible viewpoints decreases the flexibility and resolution

of the construction process but enables one to derive simple and fast algorithms for handling (simple) common cases. A variety of data structures for encoding an octree have also been derived for specific performance advantages in processing speed and storage overhead. These differences and their corresponding systems are highlighted below.

1.1 Review of Volume Intersection Systems

[Pot87] does the volume intersection by back-projecting the octree universe into the image plane. The silhouette of a perspective projection of a cube into an image plane depends on the relative position of the center of projection of the image and the coordinates of the six faces of the cube. The faces partition the 3D space of the cube into 27 half-spaces: 26 outside and 1 inside. The index number of a partition that contains the center of projection, relative to the current octree node, is used to find, in a lookup table, the number of silhouette vertices and their coordinates. The coordinates of the cube vertex are projected into an image plane using 12 lookup tables for the object-to-image space transformations. The polygonal silhouettes of the objects, defined by the vertex coordinates, are then approximated by their bounding rectangles encoded as quadtrees. Each polygon is finally raster-scan converted for volume intersection, ie. the pixels inside the polygon are determined in scan line order and are individually compared with the contents of the (binary) image.

This is one of the more flexible octree schemes but the overhead in arbitrary back-projection of the cube and the individual raster-scan conversions is high. Moreover, it requires good camera calibration and viewpoint registration techniques to compute accurate object-to-image space transformation for each image.

Instead of performing the costly perspective backprojection, view cones can be projected into the octree universe and intersected directly in 3-space. [Nob88a,Nob88b] constructs polyhedral view cones for each image and projects them into the cube for volume intersection. If a view cone is not convex, it is defined as the union of partitioning convex cones. The cones are then used to check and classify the eight subregions of the parent cube as inside, intersecting and outside each convex cone. Cone unification rules are next used to do the same classification for the eight subregions with respect to each non-convex cone. Lastly, cone intersection rules are used to integrate the information on the subregions from the multiple view cones into the common region. A DF (Depth-First) representation [Kaw80] for linear encoding of the initial eight subregions is generated and recursively applied during model construction.

This octree building algorithm does not build a complete octree for each cone, but instead builds only a part of the octree within the common region. It is thus able to process an arbitrarily selected region in 3D space independent of all other region. The procedure is fast

when restricted to polygonal silhouettes. More complex contours can first be approximated by linear segments but at the cost of processing a larger number of convex cones and introducing additional digitization inaccuracies. Accurate viewpoint positions are needed for the projections into the cube.

The above approaches are flexible but computationally expensive. An alternative is in restricting the possible viewpoints to yield simple and fast algorithms for handling specific cases. [Chi86a,Chi86b,Chi89] constructs the volume/surface octree from silhouettes obtained at three orthogonal views. Each occluding contour, encoded into the "principal" quadtree of the associated silhouette, is first fitted with a tension-spline [Sch66] to allow better approximations of the contour normals and hence more accurate surface information. With scaled orthographic projection along the viewing directions, each of the three "principal" quadtree sweeps out an oblique cylinder into the cube. Since sub-cylinders inside this volume may be identical, its exact octree can be replaced with a pseudo-octree that contains no identical subtrees. These orthogonal pseudo-octree are then intersected using simple tree traversal techniques to obtain the resultant model octree [Jac80].

This approach may be useful for simple objects that are symmetric along its three principle axes. The major disadvantage, on the other hand, is the inability to guarrantee that all significant features will be captured by the constructed model.

[Ahu89] considers a less restrictive case in using silhouette images obtained from any subset of 13 orthographic viewing directions. By restricting to these 3 "face" views, "6 "edge" views and 4 "corner" views, a simple relationship between pixels in the image and the octant labels in the octree is derived. The detection of intersections between the octree and the objects is thus replaced by a simple table lookup operation between a pair of views.

This approach provides more accurate information with the higher degree of reconstruction accurarcy but it still has problems. Discontinuities cannot be modelled, as evidenced by the approximation of object edges by rectangular steps. The proposed set of 13 viewpoints is not sufficient to build accurate models. The number of viewing samples needed for building an accurate model is arbitrarily large. The number of viewpoints necessary for reconstruction to within a desired accuracy depends on the viewing direction and the target object.

- [Car85] proposes a solution for modelling discontinuities with the polytree model. It attempts to include more surface information with three extra classes of voxel structure: vertex cell, edge cell and surface cell. The result shows only negligible improvement. A forerunner of the octree is the "volume-segment" representation proposed by [Mar83]. It borrows scan-line techniques from computer graphics to construct a 3D object model from the bounding volumes

carved out by the occluding contours. A wire frame with planar polygonal facets is then derived from the volume segments. This approach is slow, inaccurate and suffers from the usual problems associated with silhouettes. [Cap87] extends it to handle real images of parts and compares the constructed models with renderings of the original CAD designs.

Other less attractive volume intersection techniques have been proposed. [Che88] constructs a 3D model from a set of 3-view (orthogonal) type line drawings. Objects are restricted to polyhedral, cylindrical and composites of the two that are not rotationally symmetric objects. The object must be decomposable into subparts for individual reconstruction and then merged to form a Constructive Solid Geometric (CSG) [Man88] representation of the object. [Ide86] has an even more restricted system that reconstructs models of only polyhedral objects from a set of 3-view type drawings. [Sha86] uses simulated 2.5D sketches containing only depth information for their model construction but their volumetric representation and algorithms show no advantages over octrees. Similarly [Jiy86] describes an iterative 3D algebraic reconstruction technique with results for two synthetic test models under a variety of projections. The technique is, however, primitive compared to octrees.

1.2 Disadvantages of Octrees and Volume Intersection

Octrees are an elegant data structure which enables a simple implementation of volume intersection for model construction. This advantage is, unfortunately, offsetted by numerous problems.

One difficulty is that octrees can only provide coarse models of 3D objects. Astronomical storage and processing overhead are required to capture surface details accurately. Furthermore, the models are unstable and not invariant to rotation and translation. Thus, completely different octrees have to be constructed for slight changes in voxel resolution or object position.

A second problem with volume intersection, in general, stems from the dependence on silhouette information. First, using silhouettes requires reliable contour extraction and smoothing techniques. Second, silhouettes do not generally capture surface concavities and object self-occlusion effectively. Third, digitization effects during volume intersection often result in false boundaries in the final model.

There are possible remedies to some of these problems. For example, fine details can be generated with surface interpolation over the voxels with more (post)processing. The accuracy of this process depends, however, on the initial octree approximation. An alternative would be

to use long sequences of frames to refine an octree. This, however, requires that the best views be selected, a difficult problem for unknown objects. Typically, arbitrary decisions are made on the trade-off between accuracy and computation, and the trade-off between coarse and fine resolution.

Not surprisingly, all experimentation with octrees to date has only been done with synthetic data.

2 Surface Patch Fusion

In general, surfaces can be expressed in an implicit or explicit form [Bol89]. An explicit form of a surface is the graph of a function of two variables. Let $f : U \subset R^n \rightarrow R$. The graph of f is defined to be the subset of R^{n+1} consisting of the points $(x_1,...,x_n,f(x_1,...,x_n))$ for $(x_1,...,x_n) \in U$, where U is an open subset of R^n. The implicit form of a surface in R^3 space is expressed as a set function $f : R^3 \rightarrow R \mid f(x,y,z) = \text{constant}$, where (x,y,z) are Cartesian coordinates of the surface points. The explicit form is a special case of the implicit equation and is sometimes referred to as a Monge patch. Another useful representation of a surface is the parameterized form. A parameterized n-surface in R^{n+k} ($k \geq 0$) is a smooth map $\phi : U \rightarrow R^{n+k}$, where U is a connected open set in R^n, such that $\partial \phi_p$ is non-singular (has rank n) for each $p \in U$ (which is called the regularity condition). A parameterized n-surface in R^n is simply a regular smooth map from one open set U in R^n onto another.

These surface patches are integrated together in an adjacency graph for the final representation of a 3D object. An adjacency graph is basically a bidirectional graph with each node representing a region and the links between nodes the region connectivity relationships. Such a graph is not, however, restricted to describing surfaces but can directly incorporate information and relations on edges, vertices (corners) or even generic primitives – all of which are derived from surface patches. Compared with volume intersection, the approach based on surface patches yields surface models of higher accuracy and detail in adition to allowing incorporation of surface reflectance properties during model reconstruction.

The standard model construction process has four major steps. The first step of segmenting the image into coherent regions can be driven initially by edge detection or region growing [Bal82,Hor86]. In the former method, the areas bounded by the extracted edge points are connected and tested for coherence as segmented regions. In the latter method, image pixels are grouped together on the basis of region coherence using intrinsic surface properties derived from differential geometry [Bes86]. In the second step, the surface patches derived using a least-squares surface fit are either labelled with an unique symbol, or described in terms of parametric equations. The patches may be recursively combined to form larger entities as long

as some coherence predicate is satisfied. The third step relates the various surface primitives in an adjacency graph. Information on the different properties of a surface, such as reflectance and texture, may also be included at each node. Lastly, integration of multiple views is done via the transformation of the graph constructed from a new view to the global coordinate system, the matching of corresponding nodes in the two view graphs, and the modification and/or insertion of surface primitives to refine the relational object description.

2.1 Review of Surface Patch Fusion Systems

Few past systems that follow the above general strategy based on surface patches meet with complete success. Although [Und75] may have proposed the first system that learns models of objects from multiple views, only results for a synthetic image of a single convex polyhedral object are demonstrated. To simplify the learning process, objects are restricted to be planar and convex and useful viewpoints that are known in advance. [Dan82] constructs a model from four image views for a single object. Partial results for nine sets of synthetic data are produced by an incomplete system. [Osh79,Osh83] consider multiple 3D objects in a single range image. Region growing at multiple, successively more abstract resolutions is used to generate an incomplete relational model.

[Asa87] segments the input images into spherical, cylindrical, or planar surfaces using shading analysis. A synthetic image of a single object consisting of cylindrical and planar Lambertian surface with constant albedo is projected orthographically. [Dou81] uses depth cues and object models to construct a representation of outdoor scenes. After coarse segmentation of the input image, a highly abstract semantic net relating the regions and object models is built for the environment. [Xie86a,Xie86b] outlines an expert system that constructs a relational model of a scene using artificial 2.5D sketches. No result/output is shown.

After extracting edges and lines from intensity images taken at different vantage points, [Sha77,Sha84] constructs an object model by using the concept of vertex cycles to match junctions and line segments between images. This approach stems from the various early works of Clowes, Falk, Guzman, Huffman and Waltz [Bar81]. [Yam88] considers only objects composed of hinges, slides and solids. The modeler learns the number of these features and the relationships between lines and vertices in image. Results are shown for a pair of compasses, essentially a 2D object, lying on an uncluttered tabletop. [Bak77] describes a scheme used in building models of 3D objects through binocular and motion parallax analyses. Curvature irregularities in the the region boundaries are then correlated to construct the 3D object model. [Yac75] analyzes images of objects taken from a nearly vertical direction by a TV camera. After simple thresholding and contour tracing of the object outline, the object model is constructed as a list of propeties.

[Vem86,Vem87] scans objects resting on a base plane with a laser. The range image is segmented into regions that are a collection of surface patches homogeneous in certain intrinsic surface properties. A straight line on the plane is captured in intensity images. It is used as a calibration mark for calculating interframe transformations. Simple averaging is then used to merge two overlapping views. [Lau87] constructs simple and incomplete relational models of (convex or non-convex) polyhedral objects. Jump boundaries are extracted from a 3D range image by gradient-based edge detection. The edge points are used to generate occluding and interior contours which segment the object from the image background and into coherent regions. The hemispheric histogram, a specialization of the Extended Gaussian Image, is used to extract surface orientation information in detecting corresponding regions in an image of multiple objects. Regions, however, can not contain holes and undersampling may occur with increased obliqueness of a facet.

[Her84a,Her84b,Her86] incrementally constructs a 3D model of complex scenes from multiple images. Stereo and monocular analyses are performed for image data collected at different viewpoints. Linear structures representing building boundaries are extracted from the images and combined with hypothesized new vertices, edges and faces, using task-specific knowledge on block-shaped objects in an urban scene. The edges and vertices from such analyses are then used to construct the 3D wire frames. The MOSAIC systems handles very complex scenes but in order to achieve success, a lot of assumptions and constraints have to be used: task-specific knowledge on urban scenery, trihedral polyhedra scene primitives, and limited linear 3D scene features of edges and corners. Due to the task complexity, the matching of junctions in the scene is slow and the scene interpretation is incomplete. Surface detail is provided as simulation by registering image regions with object surfaces.

[Ste86,Con86,Con87] constructs 3D models of polyhedra objects. The objects are placed on a turntable and scanned by a laser stripe. Wire frames are built from connected chains of curvature and step edges approximated by straight line segments. They are fleshed and adjusted before obtaining a least-mean planar fit of the surfaces bounded by viewplane cycles (closed set of edges and vertices). A refined model is produced by intersecting the view polyhedra from multiple views using a boolean intersection algorithm implemented in the GEO-CALC system [Bar86]. Surfaces are rendered with a Lambertian scattering model. This system is restricted by the need of placing the polyhedral objects on a turntable. No surface detail such as texture is modeled by the wire frame representation.

Instead of just surface patches, [Fer85,Fer88] take the surface patch representation a step further. A 3D object model from multiple views is derived in terms of volumetric primitives such as ellipsoids and cylinders. This is done by taking the intrinsic surface feature points, derived on the basis of surface principle curvatures, and grouping them together to form

extremal and parabolic contours for parsing a surface into patches. 3D surface point correspondences for multiple views is done by normalized cross-correlation of the feature fields in the surface graphs. In matching candidate feature neighborhoods between views, the interframe transformations are recovered to construct a composite surface graph for the object. This composite surface graph, realized as a set of dynamically intrinsic images (DII), is used in the geometric inference process to identify the volumetric primitive associated with a patch (or set of patches) and to obtain the primitive's parameters. Mulitple primitive instantiations for the same subset of image data is resolved via shape similarity functions. Results are presented for a symmetric synthetic image and a range image of a statuette. The final models based on inferred volumetric primitives are coarse and imprecise. Not only very little surface detail is captured, the modeling accuracy of the selected set of primitives is not obvious.

[Hu89] recovers 3D surface points for a scene containing multiple objects using structure lighting with a uniform grid. Either a striped image or a gray-scale image can be taken by merely switching the lights of the projector and the global coordinate system is fixed on the worktable. A calibration procedure computes the tranformation matrices M_C (worktable-camera) and M_P (projector-worktable) for computing the position of a surface point. After extracting the network of light stripes, geometric and topological constraints are used to hypothesize and test matches for solving the line labelling problem. The geometric and topological constraints are based on the uniqueness constraint (C_1) and the continuity constraint (C_2) of [Mar76]. Based on using these two sets of constraints for disambiguating surface solutions, [Hu89] presented five algorithms for 2D network extraction, single 3D point solution, single 3D network solution, network boundary extraction, and scene solutions. This approach is guided by three basic assumptions. First, objects in the scene are solid, static and opaque. Second, object surfaces are smooth and are of low order in the sense that the spatial frequency of surface undulation is less than the stripe frequency. Third, surfaces are much larger than stripe spacing so that a surface patch is covered by a multistriped network.

[LeM85]'s approach is based on the observation that the type of range data to be collected will differ with regard to sampling frequencies (in space and time) and resolution of range texture. A grid of horizontal and vertical lines including several dots is projected onto the scene. The dots are used as landmarks for initiating the line labelling process and "covers" the entire image. The camera and projector are separated by a vertical baseline so that range information is apparent in the distortions and discontinuities of the horizontal lines. The vertical lines are used to guide the extraction of the horizontal lines and to normalize the albedo variations. Grids can be designed with patterns of different thickness to be used for a multi-resolution range sensor. Associated with the finite thickness of the lines is an inherent "smoothing" of range texture. Higher frequency of range discontinuities cause the thinner projected lines to break up, whereas the thicker lines display very little distortion. A simple formula relates this smoothing of range texture to the thickness of the grid lines. In particular,

the finite thickness of the lines imposes a maximum on the detectable range. After filtering the image with the Laplacian of Gaussian operator, a shrink and expand procedure is applied to extract the vertical lines. Row and column accumulators are used to locate the vertical bars and the grid intersections in the image. The albedos in the orginal gray level image is normalized by computing a local threshold based on a square neighborhood at each point. Labelling of the intersections is initiated with the location of the dots and completed by combining and interpolating among the initial labels. Disparity values are then finally assigned to the extracted and labelled intersections.

[Pot79] uses a pair of images containing a single object for such model construction. The object is pattern-illuminated by photogrammetric techniques and imaged under perspective projection inside a calibration fixture. Calibration marks on the fixture are extracted using the Laplacian operator and the Hough transform [Bal82]. The marks are used in computing the camera transformation matrix. The grid patterns are extracted in the form of straight lines and cubic curves using scan-line-to-vector conversion. Points and curves are then matched between images on the basis of the topology of the projected grid network to generate the 3D model. An inverse mapping is used to reconstruct the matched cycle in 3D space using a least square-error technique. Nodes adjacent to this initial cycle are reconstructed and iteratively propagated in all directions using heuristic search methods. This process reaches quiescence when all the nodes are processed. This approach suffers from several problems. Surfaces are required to be photographed within a camera calibration fixture. The ten calibration marks have to be visible among the set of images. The surfaces cannot be transparent, highly reflective or totally black. Results are presented on the reconstruction of isolated surfaces using just polygonal shape approximations that contain no surface detail.

[Ver87] describes a method of obtaining 3D replica made of polyurethane foam from an arbitrary part of the human skin. An integrated system consisting of a photogrammetric stereo restitution system and a CAD system integrated with a NC 3D milling machine is used in constructing the replica. The depth values for a stereopicture of the frontal view of a human face is manually digitized. The surface is then fitted with B-spline functions which are used by the CAD system for the replica. [Duf88] scans the facial dimensions of a live human subject with a line of light from a low power laser under 5 seconds. The facial boundary is obtained by thresholding the range image and fitted with the longest possible line segments. Selected points in the depth map form vertices for the polygonal model of the face. Polygons are formed in such a way that where the surface details are complex, the polygons are small and where the surface is featureless, large polygons are generated. After hidden surface removal, texture mapping using Phong shading [Rog85] incorporates surface details onto the model facets.

[Sat86a] uses passive stereo techniques to measure the shape of statues on the Easter Islands from images taken at multiple viewpoints. For the solving the correspondence problem,

small mark seals are stck on the statutes during the day and structured light patterns are projected onto the statutes during the night. Partial sterreo mathcing is done by dynamic programming and the epipolar line search. The correspondence search is completed with manual pointing by a human operator to construct the final range map. Results for images at a single viewpoint are demonstrated. [Sat87] obtains range data using the Liquid Crystal Range Finder (LCRF) based on a nematic liquid crystal mask [Sat86b] and Gray coding [Ino84]. The 3D model of an object on a turnatable is then constructed from a set of such range images taken at multiple view. The global coordinates is calibrated with respect to the floor (z-plane) and the rotation angle is calibrated against a reference cube. The wraparound 3D data is derived from horizontally sliced contours. Experimental resutlts is shown for one statuette.

2.2 Advantages of Surface Patch Fusion

Compared with volume intersection, the surface patch fusion approach yields surface models of higher accuracy and detail. It allows direct incorporation of surface reflectance properties during model reconstruction. It minimizes necessary storage and processing overheads. And last, nut not least, the parametric form for representing surface patches is invariant to motion.

Currently, robust techniques exist for calibrating cameras, calculating interframe transformations, recovering range information via stereopsis and triangulation, computing and reconstructing 3D surface solutions, and fusing 3D surface data from multipe views. The best work in this area are exemplified by the systems of [Pot79,Hu89,Sat86a] as described above. Although they are incomplete and experimental in nature, they show considerable promise. Undoubtedly, a automatic model acquisition system can be realized in the near future.

References

[Ahu89] N. Ahuja, J. Veenstra, "Generating octrees from object silhouettes in orthographic views," IEEE Trans. on PAMI, 2:2, pp.137-149, Feb. 1989.

[Asa87] M. Asada, "Cylindrical shape from contour and shading without knowledge of lighting conditions or surface albedo," Proc. IEEE ICCV'87, London, England, pp.412-416, 1987.

[Bak77] H. Baker, "Three-dimensional modelling," Proc. IJCAI'77, pp.649-655, 1977.

[Bal82] D.H. Ballard, C.M. Brown, Computer Vision, Englewood Cliffs, NJ:Prentice-Hall, 1982.

[Bar81] A. Barr, E.A. Feigenbaum (Eds.), The Handbook of Artificial Intelligence, Vol. 1-3, Menlo Park, CA: W. Kaufman, 1981.

[Bar86] M.M. Barry, C.I. Connolly, G. Spradlin, J.R. Stenstrom, D.W. Thompson, GEO-CAIC System Methods Reference Manual Version 1.0, Internal Report of Computer Science Branch, General Electric R&D Center, 1986.

[Bes86] P.J. Besl, R.C. Ramesh, "Invariant surface characteristics for 3D object recognition in range images," CVGIP, 33, pp.33-80, 1986.

[Bol89] R.M. Bolle, B.C. Vemuri, "On 3D surface reconstruction methods," IBM Research Report, RC 14557, IBM Research Division, Apr. 1989.

[Cap87] V. Cappellini, R. Casini, M.T. Pareschi, C. Raspollini, "From multiple views to object recognition," IEEE Trans. on Cir. Sys., 34:11, pp.1344-1350, Nov. 1987.

[Car85] I. Carlbom, I. Chakravarty, "A hierarchical data structure for representing the spatial decomposition of 3D objects," CG&A, 5:4, pp.24-31, Apr. 1985.

[Che88] Z. Chen, D. Perng, "Automatic reconstruction of 3D solid objects from 2D orthographic views," Pattern Recognition, 21:5, pp.439-449, 1988.

[Chi89] C.H. Chien, J.K. Aggarwal, "Model construction and shape recognition from occluding contours," IEEE Trans. on PAMI, 2:4, pp.372-389, Apr. 1989.

[Chi86a] C.H. Chien, J.K. Aggarwal, "Computation of volume/surface octrees from contours and silhouettes of multiple views", Proc. IEEE CVPR'86, Miami Beach, Florida, pp.250-255, 1986.

[Chi86b] C.H. Chien, J.K. Aggarwal, "Volume/surface octrees for the representation of 3D objects," CVGIP, 36, pp.100-113, 1986.

[Con86] C.I. Connolly, J.R. Stenstrom, "Construction of polyhedral models from multiple range views," Proc. IEEE ICPR'86, Paris, France, pp.85-87, 1986.

[Con87] C.I. Connolly, J.L. Mundy, J.R. Stenstrom, D.W. Thompson, "Matching from 3D range models into 2D intensity scenes," Proc. IEEE ICCV'87, London, England, pp.65-72, 1987.

[Dan82] C. Dane, R. Bajcsy, "An object-centered 3D model builder," Proc. ICPR'82, Munich, Germany, pp.348-350, 1982.

[Dou81] R.J. Douglass, "Interpreting 3D Scenes: A model building approach," CVGIP, 17:2, pp.91-113, Oct. 1981

[Duf88] N.D. Duffy, J.F.S. Yau, "Facial image reconstruction and manipulation from measurements obtained using a structured lighting technique," Pattern Recognition Letters, 7, pp.239-243, Apr. 1988.

[Fer85] F.P. Ferrie, M.D. Levine, "Piecing together the 3D shape of moving objects: An overview," Proc. IEEE CVPR'85, San Francisco, California, pp.574-584, 1985.

[Fer88] F.P. Ferrie, M.D. Levine, "Deriving coarse 3D models of objects," Proc. IEEE CVPR'88, Ann Arbor, Michigan, pp.345-352, 1988.

[Her84a] M. Herman, T. Kanade, "The 3D MOSAIC scene understanding system: Incremental reconstruction of 3D scenes from complex images," Proc. DARPA Image Understanding Workshop, New Orleans, Louisianna, pp.137-148, Oct. 1984.

[Her84b] M. Herman, T. Kanade, S. Kuroe, "Incremental acquisition of a 3D scene model fomr images," IEEE Trans. on PAMI, 6:3, pp.331-340, May 1984.

[Her86] M. Herman, T. Kanade, "Incremental reconstruction of 3D scenes from multiple, complex images," AI, 30, pp.289-341, 1986.

[Hor86] B.K.P. Horn, Robot Vision, Cambridge, MA:MIT Press, 1986.

[Hu89] G. Hu, G. Stockman, "3D surface solution using structured light and constraint propagation," IEEE Trans. on PAMI, 11:4, pp.390-402, Apr. 1989.

[Ide86] M. Idesawa, "3D model reconstruction and processing for CAE," Proc. IEEE ICPR'86, Paris, France, pp.220-225, 1986.

[Ino84] S. Inokuchi, K. Sato, F. Matsuda, "Range-imaging system for 3D object recognition," Proc. ICPR'84, Montreal, Canada, pp.???-???, 1984.

[Jac80] C.L. Jackins, S.L. Tanimoto, "Octrees and their use in representing 3D objects," CVGIP, 14, pp.249-270, 1980.

[Jiy86] X. Jiye, O.H. Kapp, "3D algebraic reconstruction from porjections of multiple grey level objects," Optik, 72:3, pp.87-94, 1986.

[Kaw80] E. Kawaguchi, T. Endo, "On a method of binary picture representation and its application to data compression," IEEE Trans. on PAMI, 2, pp.27-34, 1980.

[Kem88] K. Kemmotsu, Y. Sasano, K. Oshitani, "Model-based 3D vision system," SPIE: Expert Robots for Industrial Use, 1008, pp.40-47, 1988.

[Lau87] D. Laurandeau, D. Poussart, "Model building of 3D polyhedral objects using 3D edge information and hemispheric histogram," IEEE Journal Rob. Auto., 3:5, pp.459-470, Oct. 1987.

[LeM5] J. Le Moigne, A. M. Waxman, "Multi-resolution grid patterns for building range maps," Proc. VISION'85, Detroit, MI, pp.8.22-8.36, Mar. 1985.

[Man88] M. Mantyla, An Introduction to Solid Modeling, Rockville, MA:Computer Science, 1988.

[Mar76] D. Marr, T. Poggio, "Cooperative computation of stereo disparity," Science, 194, pp.283-287, 1976.

[Mar83] W.N. Martin, J.K. Aggarwal, "Volumetric descriptions of objects from multiple views," IEEE Trans. on PAMI, 5:2, pp.150-158, Mar. 1983.

[Nob88a] H. Noborio, S. Fukuda, S. Arimoto, "Construction of the octree approximating 3D objects by using multiple views," IEEE Trans. on PAMI, 10:6, pp.769-782, Nov. 1988.

[Nob88b] H. Noborio, S. Fukuda, S. Arimoto, "A fast algorithm for building the octree for a 3D object from its multiple images," Proc. IEEE ICPR'88, Rome, Italy, pp.860-862, 1988.

[Osh79] M. Oshima, Y. Shirai, "A scene description method using 3D information"," Pattern Recognition, 11, pp.9-17, 1979.

[Osh83] M. Oshima, Y. Shirai, "Object recognition using 3D information," IEEE Trans. on PAMI, 5:4, pp.353-361, Jul. 1983.

[Pot79] M. Potmesil, "Generation of 3D surface descriptions from images of pattern-illuminated objects," Proc. IEEE Conf. Pat. Reg. Image Proc., pp.553-559, 1979.

[Pot87] M. Potmesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images," CVGIP, 40, pp.1-29, 1987.

[Rog85] D.F. Rogers, Procedural Elements for Computer Graphics, New York, NY: McGraw-Hill, 1985.

[Sam85] H. Samet, "The quadtree and related hierarchical data," ACM Comput. Surveys, 16:2, pp.187-260, 1985.

[Sat86a] K. Sato, H. Yamamoto, S. Inokuchi, "3D shape measurement of megalithic statute - MOAI -," Proc. ICPR'86, Paris, France, pp.675-677, 1986.

[Sat86b] K. Sato, H. Yamamoto, S. Inokuchi, "Tuned range finder for high precision 3D data," Proc. ICPR'86, Paris, France, pp.675-677, 1986.

[Sat87] K. Sato, S. Inokuchi, "Range-imaging system utilizing nematic liquid crystal mask," Proc. ICCV'87, London, England, pp.657-661, 1987.

[Sch66] D.G. Schweikert, "An interpolation curve using a spline in tension," J. Math. Phy., 45, pp.312-317, 1966.

[Sha77] R. Shapira, H. Freeman, "Reconstruction of curved-surface bodies from a set of imperfect projections," Proc. IJCAI'77, pp.628-634, 1977.

[Sha84] R. Shapira, "The use of objects' faces in interpreting line drawings," IEEE Trans. on PAMI, 6:6, pp.789-794, 1984.

[Sha86] A. Sharma, S.A.R. Scrivener, "Constructing 3D object models using multiple simulated 2.5D sketches," Intl. J. Man-Machine Studies, 24, pp.633-644, 1986.

[Ste86] J.R. Stenstrom, C.I. Connolly, "Building wire frames from multiple range views," Proc. IEEE Conf. Rob. Auto, San Francisco, California, pp.615-620, Apr. 1986.

[Und75] S.A. Underwood, C.L. Coates, "Visual learning from multiple views," IEEE Trans. on Comp., 24:6, pp.651-661, Jun. 1975.

[Vem86] B.C. Vemuri, J.K. Aggarwal, "3D model construction from multiple views using range and intensity data," Proc. IEEE CVPR'86, Miami Beach, Florida, 1986.