1992

Prototype automated custom fit system for jet pilot oxygen masks

Christopher Myles Muir
Lehigh University

AUTHOR:

Muir, Christopher Myles

TITLE:

Prototype Automated Custom Fit System For Jet Pilot Oxygen Masks.

DATE: May 31, 1992

PROTOTYPE AUTOMATED CUSTOM FIT SYSTEM

FOR JET PILOT OXYGEN MASKS

by

Christopher Myles Muir

A Thesis
Presented to the Graduate Committee
of Lehigh University
in Candidacy for the degree of
Master of Science
in
Mechanical Engineering

Lehigh University
December 1991

This thesis is accepted and approved in partial fulfillment of the requirements of degree of Master of Science.

1/2/92
(date)

Professor in Charge
Professor John B. Ochs

Chairman of Department
Professor Robert Wei

Acknowledgments

I wish to express sincere gratitude to my advisor, Professor John B. Ochs for his guidance and thoughtful input during this project. Mr. Fred Wehden also deserves special thanks and gratitude for his help.

Recognition for educational software licences is made for McDonnel Douglas Corporation. Thanks to the United States Air Force and Mr. Greg Zehner for providing datasets and other technical information.

I would especially like to thank my parents Dorothy and William Muir for providing the opportunities and understanding that put me here. Finally, I would like to thank my wife Rose for her unbelievable patience and understanding throughout this and many other projects.

Table of Contents

Table of Figures

List of Symbols, Acronyms, and Terms

CAD/CAM	- Computer Aided Design and Computer Aided Manufacture
NC	- Numerically Controlled
DTIC	- Defense Logistics Agency's Defense Technical Information Center
UGII	- McDonnel Douglas' Unigraphics II design software product
IGES	- Initial Graphics Exchange Standard
VSP	- Very Small Patch
UNIX	- Flexible computer operating system popular at many universities
B-Spline	- A specific type of geometric curve
X-Windows	- A portable, network transparent window system
Starbase	- Hewlett-Packard's graphics library routines
GRIP	- UGII's GRaphics Interactive Programming language
IO	- Input/Output
RAM	- Random Access Memory
$\vec{ev}$	- Edge vector (used for defining control polyhedron)
$\vec{nv}$	- Normal vector (relative to surface or data)
$\vec{s}$	- Auxiliary vector needed to define $\vec{ev}$
P_i	- Four corner control polyhedron points
p_i	- Next and previous point in a specified direction
$B_{i,j}$	- Control polyhedron points
u	- parametric surface variable (u direction)
w	- parametric surface variable (v direction)
$N_i(u)$	- Bezier blending functions in the u parametric direction
$M_j(w)$	- Bezier blending functions in the w parametric direction
[B]	- matrix containing defining control polyhedron points
[C]	- matrix containing blending function combinations
[D]	- matrix of datapoints to be approximated
NURB	- NonUniform Rational B-Spline
Bezier	- A specific type of geometric curve
Z-Buffer	- a technique for removing hidden surfaces
Q(u,v)	- Generic B-Spline surface
R(u,v)	- Generic B-Spline surface
CRT	- Cathode Ray Tube
CNC	- Computer Numerically Controlled
$\vec{v}$	- Edge vectors for interior node of newly subdivided patch

Abstract

The oxygen mask for a jet fighter pilots must fit and seal exceptionally well due to the conditions under which the pilot is required to fly. Due to the intrinsic differences present from one face to the next, the face mask sizes provided do not always fit adequately. To solve this problem custom made masks are manually created for pilots with fitting problems. The existing custom fitting process contains many problems; it is time consuming, inaccurate in many ways, and can be inconsistent.

Computer Aided Software tools can be used to eliminate or, at least minimize many of the problems inherent in the existing process. A prototype custom fitting system has been developed to take digitized data of the facial area, convert this data to geometric surface entities. This data conversion reduces the amount of information necessary to define the facial area as well as reliably represents that data within a user specified tolerance. These geometric surfaces representing the face can then be interactively manipulated in real time with a geometric model of a face mask using advanced graphics techniques. Information can then be calculated from satisfactory face-mask configurations and used in an automated manner to manufacture that mask in the defined configuration.

1.0 Introduction

1.1 Problem Statement

Jet powered aircraft, military and otherwise, are ultimately only as effective as the pilots who fly them. The effectiveness of a pilot is dependent on a vast array of different factors, from the pilot's innate abilities to the interface between pilot and aircraft. This research's development focuses on a critical concern of the pilots, the fit of their face/oxygen mask.

The fit of a flight mask for a jet pilot is especially important because of the extreme conditions to which they may be exposed during the course of a flight. Discomfort and fatigue, the number one cause of pilot error, can result from an improper fit. During maneuvers that subject the pilot to high accelerations, the mask is tightened automatically and the air pressure in the flight mask is increased to force oxygen into the pilots lungs. If the seal between the face and the mask is poor, air will be allowed to escape. This is called blowby. If the blowby occurs in the area on the nose near the eyes, the air tends to dry out the pilots eyes as it passes over them, further compounding fatigue problems. In extreme cases the eye can dry out to the point where the eyelid "freezes" to the eyeball. While all pilots must have two eyes, a nose, and a mouth, the arrangement and size of these features will vary from one pilot to the next, in some cases drastically. In many cases, they will not conform exactly to small, medium, or large. These current mask sizes are based on data taken and research done on the male population during World War II.

Computer Aided Design techniques can provide the flexibility to incorporate slightly different facial features into the design of a custom fit mask. Using Computer Aided

Manufacturing methods these differences can be taken into account in the automated production of a custom-fit mask trimmed from a generic mask of the correct basic shape.

The research and development effort involves:

1) The investigation of current custom fit technologies.

2) The design and specification of the components of a system to automate the custom fit of candidate masks.

3) The development of a prototype system and demonstrate the concepts for a particular pilot and candidate mask.

4) The development of the software required to automate these tasks.

1.2 Current Process

Currently the Custom Mask Shop of the Physiological Training Directorate at the Wright-Patterson Air Force Base assumes the task of fitting flight masks for pilots whose faces do not conform to the usual small, medium or large size designations. The work load consists of approximately 30 custom-fit flight masks per month. Currently, the Air Force mask type MBU-5/p is used for the custom fitting operation[1].

A brief description of the custom fit mask procedure as it exists at Wright-Patterson is described below. This process defines the background for the existing process and also shows the rationale behind the system described in this thesis.

1) The subject's face is first cast in plaster. This is done by placing tubes in the subject's nose for breathing then dipping the person's face in a plaster bath and allowing the plaster to dry.

2) The mask maker then sketches an approximate curve on the plaster face cast where he

thinks the mask-face intersection curve will lie on the casting.

3) Next he selects a blank mask shell that appears to be the correct size relative to the previously sketched face-mask intersection curve and positions it over the casting in the correct orientation.

4) Clay is then used to surround the drawn intersection curve and the hard shell in a "valley".

5) This valley is then filled with quick drying dental stone; the excess dental is removed and is allowed to dry slightly.

6) The face-form mold with its dental stone extension is then removed from the cast of the face. It is then baked, sanded and coated.

7) Next, the prepared mold is fitted with a rod and dipped twice in a latex bath and placed in a cold water leach tank for 4-hours per each dip.

8) After the latex has cured onto the dental stone in the correct shape it is cut from the dental stone and removed from the rod.

9) The blank mask shell is then hand trimmed 1/4" following the contour of the latex face form insert. The rough cut edge is then sanded and fitted with the latex extension.

The above process involves a number of time consuming steps and contains the intrinsic inaccuracies of a manual process. Portions of the procedure as well as being inaccurate are physically uncomfortable for subjects (ex. the plaster casting of the face). The average lifetime of a mask is 2-3 years [1]; therefore, each time a new mask is needed for a pilot, the process must start from the point of the existing plaster cast. Since each new mask is created by stepping through the entire process again, the consistency from

one to the next is not assured, and cost for each successive mask will be at least equivalent to the first. Also, the quality and fit of the mask is largely dependent on the skill and experience of the person performing the manual work. The consistency and quality of personnel in the military is difficult to ensure.

1.3 System Objectives

This thesis details a system that will eliminate many of the problems of the current system. The general objectives are as follows:

- Reduce the error involved in the fitting processes to achieve an accurate fit.
- Increase the flexibility of the current custom-mask fitting process i.e. allow for simple creation and evaluation of multiple face-mask configurations with the same data set.
- Improve the efficiency and turnaround time of the process. Move from a manual process possibly requiring days of effort to a rapid process of hours for initial prototypes.
- An extended range of applications. Possible application of the same or similar technologies to other body areas such as the hand or buttocks.

1.4 General Approach

The general approach to this problem is to create a system that will automate the process of creating a custom fit face mask. This project has grown from an initial feasibility study conducted by Lehigh Valley Computer Services (LVCS), Bath PA, and the United States Air Force [2]. The approach and the objectives stated here reflect those set forth in that study.

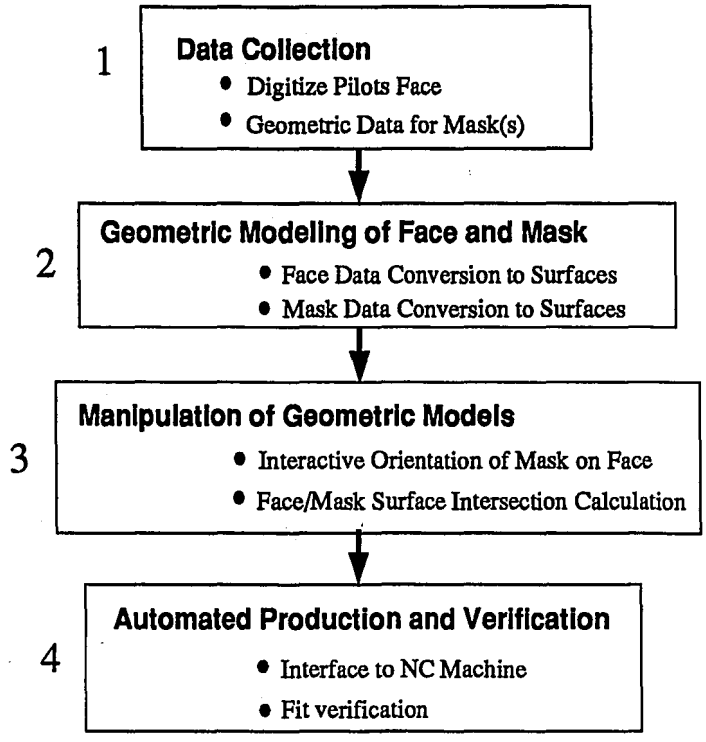


Figure 1.1 General Approach

The approach to the problem involves four steps. These steps are shown schematically in Figure 1.1.

1) Data collection.

As in the manual process above the face topography must first be defined. Digitized point data will be used to create a surface that will describe the area of interest. The distribution of points must be even and the density must be such that the most seriously convoluted portions of the face are sufficiently defined.

Information to completely define the mask must also be collected. This information will be in the from of either engineering drawings, physical models, or, possibly, a digital databases.

2) Geometric modeling of the face and mask.

Modeling in this case refers to defining of a geometric database consisting of points, curves, and surfaces. The B-Spline surfaces type has been chosen for its flexibility and the fact that it can be viewed and manipulated in a straightforward manner using computer graphics techniques.

The modeling of the face will involve the approximation of the digitized data with quiltwork of B-Spline surface patches. Since different regions of the face can be most easily and efficiently defined with different size surfaces, large patches will be used in regions of relative smoothness and smaller patches in more complicated areas.

Wherever the data necessary to model the mask comes from, here that data must be used to create a geometric surfaced model of a mask. It is assumed that the mask will be generic in form, i.e. the model defined will be the mask in its state before customizing. A database consisting of several different sizes and configurations of mask blanks will be created. The modeling of the mask will be done using a standard CAD package.

3) Manipulation of the geometric models.

The model of the mask and the face will then be able to be interactively manipulated relative to one another. Advanced graphics techniques will be used to realistically simulate the positioning of a mask on a face. Once a satisfactory face-mask configuration is found the actual face-mask intersection can then be calculated.

4) Automated Production.

The intersection curve will be postprocessed into a machine-readable form and used to trim a blank of the actual mask using a numerically controlled multi-axis milling

machine or robot. This is just one alternatives for the intersection data, other types of automated production will also be explored. As part of the system described here a verification of the fit and manipulation routines will also be provided.

The technology behind most of the elements of the system exist in one form or another. The integration of these technologies, as well as their adaptation for these specific purposes has not been done. The system developed here will concern itself primarily with the elements of modeling the face and manipulating a geometric model of a mask to achieve a user-controlled "exact" fit.

1.5 Specific System Software and Hardware Objectives

The starting point for this system is the digitized facial/head area of interest. The data must be arranged in a regular rectangular manner, i.e. not scatter data. This type of data collection can be performed in a variety of different ways [Appendix A describes several current technologies available for data collection]. The main steps in the process include:

- Automatically convert (within a given tolerance) raw digitized data into a usable representation, i.e. geometric surfaces. If possible, reduce the amount of information necessary to represent the given facial topology.
- Using advanced graphics techniques, interactively manipulate the surfaced face representation and a given specific mask geometry in a real time manner to produce visually acceptable face-mask orientation(s).
- Calculate the intersection curves and other information necessary for manufacturing face masks described by the desired configurations.
- Use Computer Aided Manufacturing techniques to produce a custom fit mask for a given face and mask data corresponding to any or all of the selected orientations.

In order to reduce the turnaround time and increase the flexibility and accuracy of the design, the system will require the creation of a geometric database. Some of the additional benefits of this database are listed below.

- Since the ideal, or exact fit is an objective criterion, an existing face database can be used over and over to tweak a particular mask configuration for optimum fit and seal as desired by the pilot.
- The face database is upwardly compatible with future mask types, i.e. rescanning of

the face will only be necessary if the pilots face changes, for example; if he gains or loses weight, or breaks his nose.

- Once a satisfactory face-mask configuration is stored it can be reused to produce identical copies of the original at only the manufacturing cost, reducing the overall cost of reproductions.
- The human aspect of design and its associated inconsistency can be reduced.

The system developed here was written using the C programming language. Hewlett-Packard Apollo, 9000 Series, Models 300/400/700 workstations were used. HP-UX 7.03 (Hewlett-Packard's version of UNIX) and the X window system provide the programming environment. Graphics operations were performed using Starbase, Hewlett-Packard's own callable graphics subroutine libraries.

1.6 Literature Search

This thesis has grown out of an initial feasibility study conducted by LVCS for US Air force. At the time of that study the following literature search was preformed. Technical report summaries of unclassified sources were obtained from the Defense Logistics Agency's Defense Technical Information Center (DTIC). Keyword searches were performed on the following terms: Digitization, Protective masks, Gas masks, Custom-fit, Ergonomics, Anthropomet, Human Factors, Diameters, length, and size measurements, CAD/CAM, Moire Patterns. None of the 34 matching articles found dealt with the integration of the technologies discussed above into a single system for custom fitting[1].

Many commercial packages are available for applications such as geometric

modeling and NC machining. For the feasibility study conducted, McDonnell Douglas' Unigraphics II design package (UGII) was used to demonstrate some of the surfacing, intersecting and machining techniques. Because of the specific nature of this project, a general application CAD package does not contain the specific functions necessary, such as the type of surface fitting required, and at the same time carries with it much unnecessary overhead, such as drafting.

Although a system that integrates all of the desired functions is not known to exist, studies relating to the components of such a system have been studied.

Much work has been done with geometric surface fitting. Rogers[3] and Adams describe a general method for interpolating datapoints with a B-Spline surface patch. Because of the volume of surface data, a method of reducing and approximating the data is necessary. This is an area addressed by Schmitt[4], Barsky, and Du. They describe a method of adaptive subdivision. The approach begins with a rough approximation and progressively refines it in successive steps in poorly approximated regions. An algorithm roughly based on this method is implemented in this system.

Barnhill[5], Farin, Jordan, and Piper describe a a very general and robust intersection algorithm for general surface types. The use of a very general method is important since the surfaces that are being intersected can come in almost any form. This algorithm is implemented here with an enhanced method of tracing intersection curves.

1.7 Organization of thesis

Chapter II contains a basic overview of the fitting process as described by this system, the prerequisites, basic methodology, and the options and output of each portion. Chapter

III contains the details of the implementation, algorithms used, the rationale and calculations behind each. Chapter IV contains a complete example from digitization to fitting demonstrating the capabilities of the package. Chapter V reviews the conclusions and recommendations for future study.

2.0 System Overview

2.1 Data Requirements

Figure 2.1 shows a schematic of the system developed in this thesis. As stated previously, for this system, the starting point is raw digitized data of the subjects facial area. The data must be arranged in a regular fashion, i.e. not scatter data. It is assumed that the data is relatively evenly distributed across the facial area, with the density of points in areas where the curvature changes rapidly, such as around the nose eyes and lips, the same as the density in areas that are smooth and regular, such as the cheeks and forehead. Since the positioning of the mask involves visual evaluation of the mask relative to the face, the data must contain the landmarks that are necessary to determine satisfactory location and fit, such as the forehead, eyes, and lips. Thus, the data set should

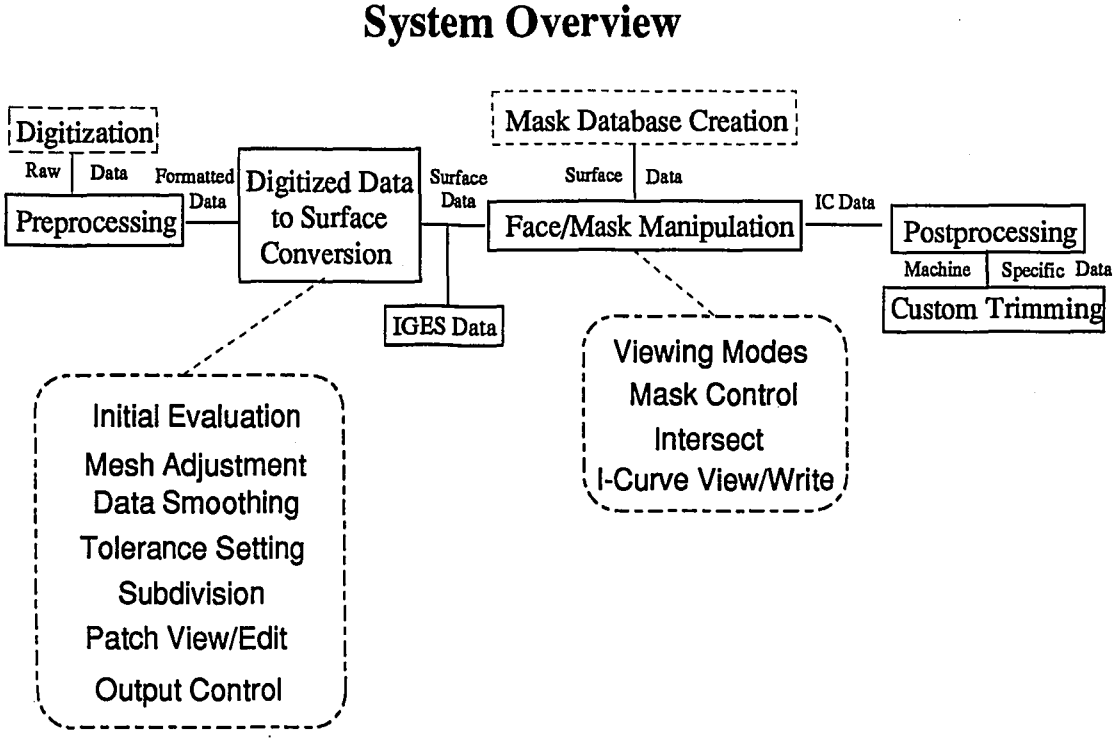


Figure 2.1 System Overview

contain information spanning the entire facial area, not just the areas necessary for the calculation of the face-mask intersection. Figure 2.2 shows a raw digitized data set of the area critical to the custom fit process.(See Figure 3.1 (a) for the complete dataset) This is

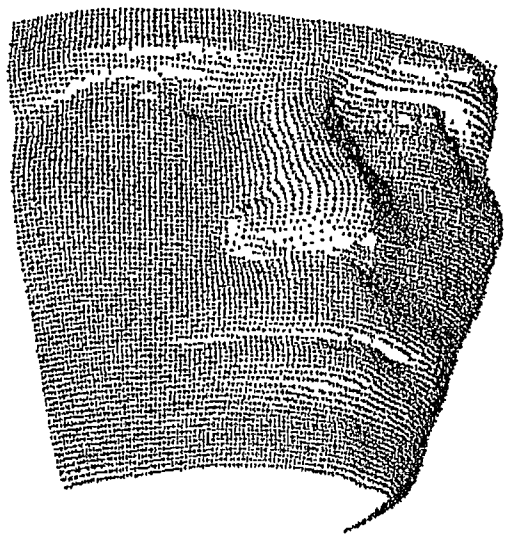


Figure 2.2 Raw Dataset

a portion of a dataset generated by the Cyberware laser scanning system, and has been provided by the United States Air Force located at Wright-Patterson AFB.

2.2 Preprocessing

Preprocessing involves manipulation of the digitized data from its raw form into the complete, defined rectangular region of interest described above. The data is also tagged for future reference; this identification will be used to keep track of the data through all steps of the system. This preprocessing consists largely of conversion from one file format to another, but may also entail interpolating missing data and removal of extraneous regions.

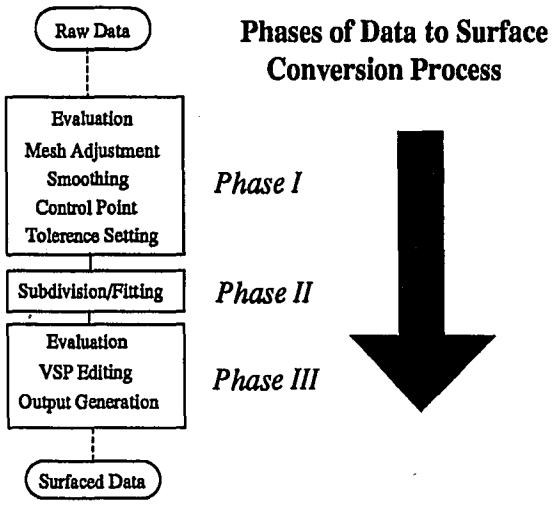


Figure 2.3 Data to Surface Conversion Process

2.3 Digitized Data to Geometric Surface Conversion

The manner in which this system converts the digitized point data to B-Spline surfaces is accomplished in three separate phases described below. Figure 2.3 schematically shows this process.

Phase I

First the face data is read into the system. The user can then view the data in a variety of different visual formats and from multiple viewpoints. At this time the overall quality of the data set can be evaluated; also the initial program options pertaining to the subsequent surface fitting can be adjusted.

For most point data collection methods the data distribution across the face is fairly even, with the density of the digitization set to account for the regions of highest curvature. Therefore, there will be areas of large radius of curvature (almost flat) that are grossly overdigitized such as the cheeks and forehead. The task for the data fitter is to

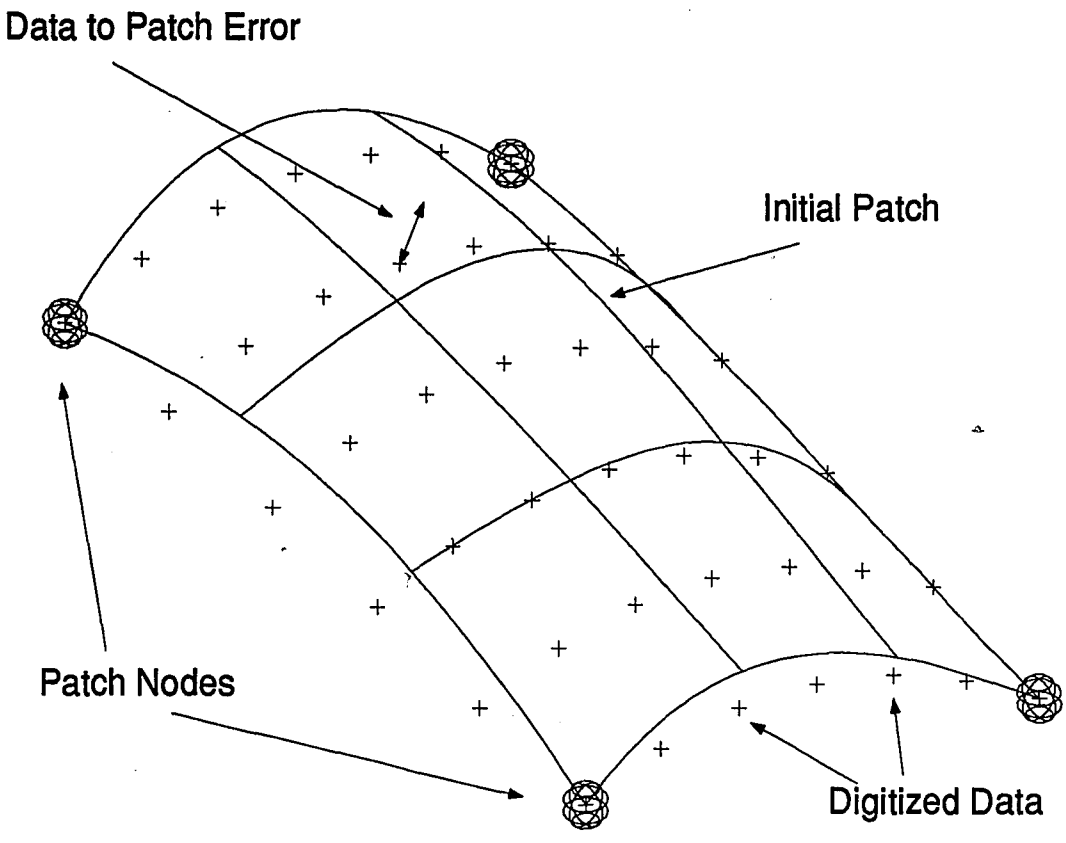


Figure 2.4 Patch/ Data Parameters

reduce the amount of data needed to describe the overall topography as well as approximate the critical topography within the given tolerance. In this initial phase, the system subdivides the initial data into roughly square subregions of data with a user defined density of regions across the face. Figure 2.4 shows a typical interpolating patch and associated parameters. The point at each corner of each subregion is called a node. Every node is associated with a point in the dataset. The positions of the four nodes along with information about the data surrounding each node define the corners of a geometric surface patch. All of the points in the subregion under the patch (in the area enclosed by the four nodes of the subregion) are considered the domain of the patch, and the minimum distance between each point in the subregion and the surface patch approximating that subregion is considered to be the error of that point. A maximum value for this error is

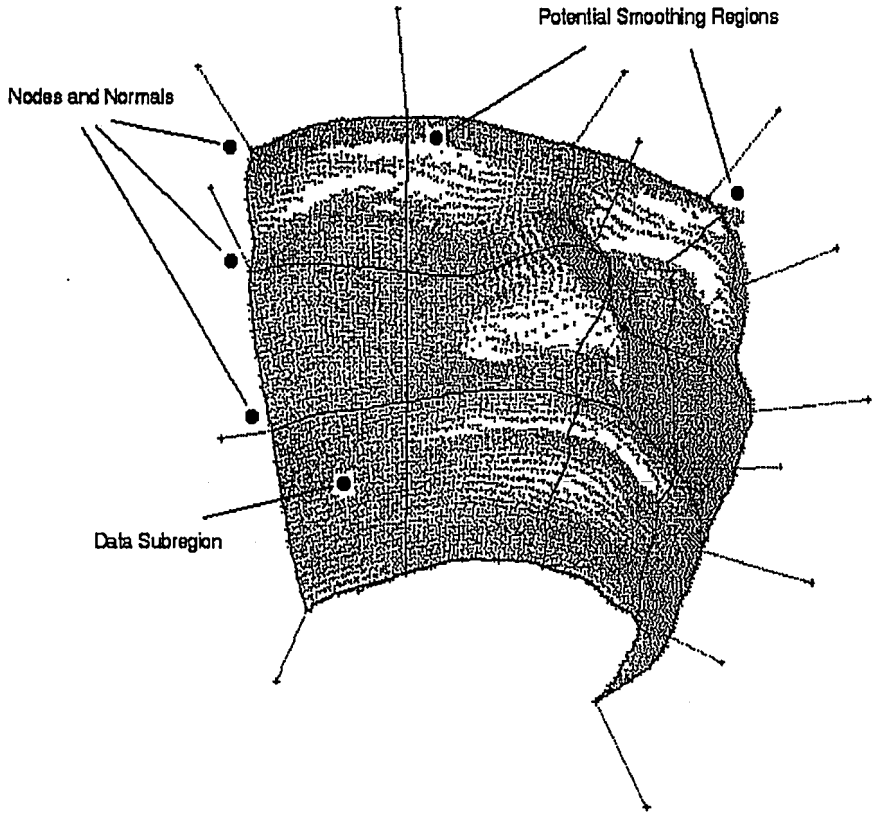


Figure 2.5 Initial Arrangement

set during Phase I and the system will attempt to create a quiltwork of patches that will satisfy this maximum error by recursively subdividing the initial patches into smaller and smaller subpatches that will converge to the limit of the actual point data, that is each corner node corresponds to a point in the data and there is no further subdivision possible.

Figure 2.5 shows a the initial arrangement of the raw data after it is read into the system. The squareness of each subregion ensures that as these patches are subsequently subdivided, their child patches will also be square. This initial arrangement can be

overridden if needed, for example, if large regions of smooth data can be made to be covered by a single large patch. Data can also be "smoothed" in non critical areas such as the eyebrows and hairline. This smoothing will increase the efficiency of the data approximation in these regions while retaining basic landmark information.

The control point of the data is also specified in this initial phase. The control point is the approximate position midway between the eyes on the nose in the data where the mask is most likely to intersect the face. This information will define the initial starting point for subsequent mask-face manipulations.

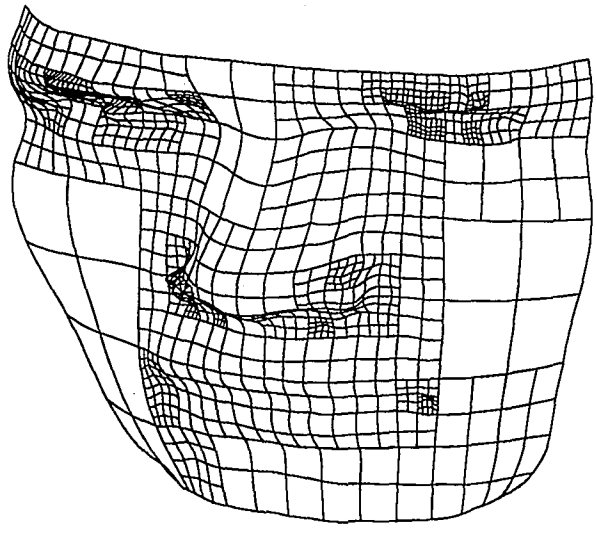


Figure 2.6 Quiltwork of Patches

Phase II

During this phase, the run time options pertaining to the subdivision are set. The subdivision can be run in an interrupted or an intermittent mode; and the progress of the subdivision can be monitored as the patches converge to the data.

Figure 2.6 shows the completed quiltwork of patches that will result from the subdivision process.

The subdivision of the patches will continue until the error tolerance is met by all of the patches and their respective domains, or the domain of a patch contains so few datapoints that any further subdivision would result in an increase in the amount of data necessary to describe a subregion, i.e. data reduction would be negative. If this is the case the patch is automatically accepted and is labeled as a Very Small Patch (VSP). The maximum and average error of all the VSP's is recorded and saved for later reference. A VSP may be the result of one bad data point, depending upon the data collection method used, possibly caused by a hair, or , a point on the eyeball.

Phase III

At the conclusion of the subdivision, the patches and the data can be viewed in the same manner as before the subdivision started. If there are any VSP's they can be viewed and evaluated individually or as a group. If a VSP falls in a critical area, a subdivision can be forced to improve the surface's approximation of a region while reducing the factor of data reduction.

Once the user is satisfied with the mesh of patches the patch data can be saved in either of two formats. A native format file of the patches can be written; this type of file will retain the surface/node associativity that is needed for the next phase of the fitting process. An Initial Graphics Exchange Specification (IGES) format file of the surface patches can also be written. This IGES file will allow the surface data to be transferred across different software and hardware platforms.

2.4 Face-Mask Manipulation/Intersection

Once a satisfactory surface dataset has been generated it can be read into the surface manipulation/intersection program. Along with the surfaced face, a surface file is also read in of a mask blank ready to be customized; this file can be any type, size, or style of mask generated by any one of a variety of CAD packages; it is read in as a standard format IGES file.

After the face and mask have been read, the user has several different types of surface display types and view possibilities. Figures 2.7 (a) and (b) show an example of different mask-face configurations. Figure 2.7 (a) shows a visually poor face/mask configuration, in configuration, although the mask appears to be located in the correct position, the intersection curve is not symmetric. In Figure 2.7 (b) the mask position is modified to achieve a more symmetric visual intersection curve. The shading and view manipulation are all done in real time, and with the Hewlett-Packard workstations used, manipulation

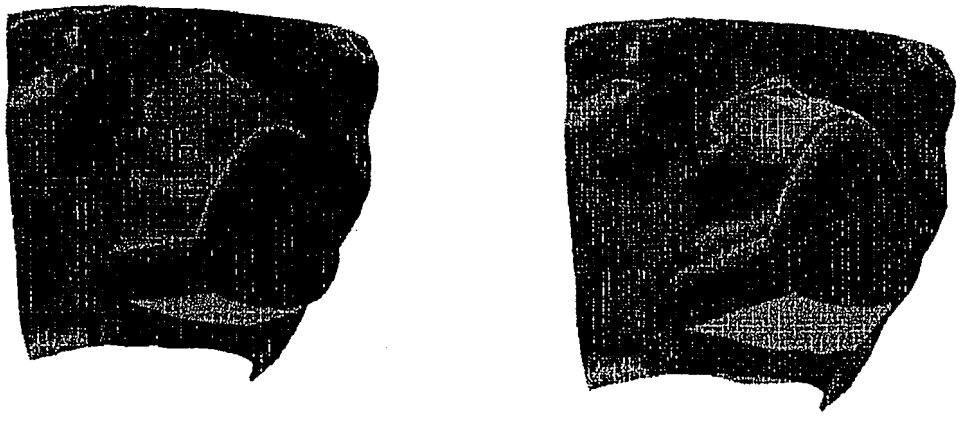


Figure 2.7 Different Visual Mask Configurations a,b (from left)

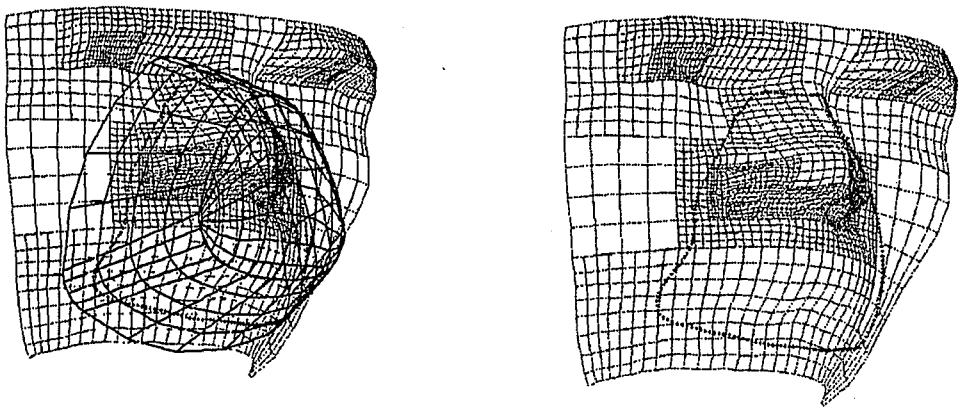


Figure 2.8 Calculated Curve

is very smooth.

Figure 2.8 shows the resulting calculated intersection curve from configuration (b), with, and without the mask surface displayed.

During orientation, the mask can be translated or rotated about its origin relative to the face to achieve desired configurations. The program uses Z-Buffering and several different surface shading techniques to create a visual effect of the intersection, this curve that can be used to evaluate the quality of the positioning of the mask on the face.

Once a satisfactory configuration has been reached the user can then calculate the actual intersection curve, in either on-line, or batch mode, as shown in Figure 2.8. Once calculated, the intersection curves can be evaluated and stored on disk. As the intersection curve is created all of the information necessary for the manufacture of the custom mask is created and stored, this allows for postprocessing into a number of different forms and custom fitting alternatives.

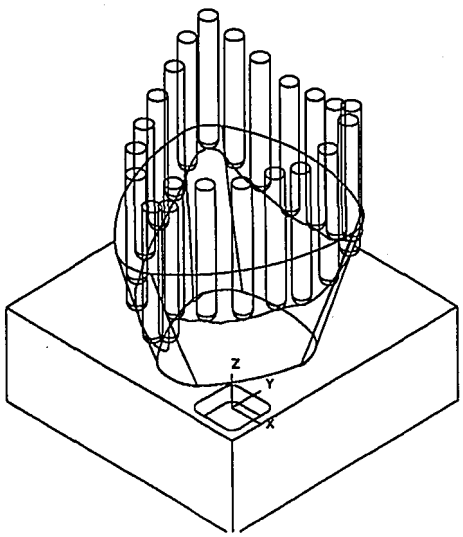


Figure 2.9 NC Tool Path

2.5 Use of intersection Curve Data (Postprocessing)

The intersection curves generated are stored to disk with their origin and orientation corresponding to that of the mask input file; together these two are used to create a custom fit mask. The intersection curve data can be used in either the production of the mold, possibly to define an insert piece to an injection mold, or, as in the example outlined in Chapter 4 of this report, to drive an NC trimming operation on an existing mask. Figure 2.9 shows a 1/4" ball end mill driven along the mask intersection curve. For this figure the mask shell was displayed inverted and fixtured for trimming. This display was generated using UGII's machining module, with the raw intersection data postprocessed manually. The raw intersection data corresponds to the above Figure 2.8.

2.6 Summary of System Requirements

2.6.1 Raw Data to Surface Conversion

This module will convert digitized data from point representation to a geometric surface representation. The objectives here are two fold. First to represent the point data

accurately to a user specified tolerance. Second, to incur a factor of data reduction, i.e. the amount of information necessary to represent the surfaces should be less than the amount to represent the raw datapoints. This process can be particularly painful depending on the type of digitization employed. The human face topography is not regular, facial features such as hair, lips, and eyes present a problem for digitization and these problems are carried forward into this process as scattered or missing data. To a certain extent, these problems can be dealt with in the digitization process and in preprocessing the data, but they cannot be eliminated and therefore, must be dealt with here.

2.6.2 Face/Mask Manipulation

This module will manipulate the approximated surface information from the above module together with a geometric model of a face mask to attempt to find an orientation between the two that "fits". Information about that orientation will then be calculated in an attempt to create or modify a mask to fit that particular face. Some of the problems faced here include the following: Development of intelligent intersection routines that can handle the irregularities of a face surface defined by multiple geometric patches, as well as the use of visualization methods that will show the intersection of the mask with the face surface without the actual numerical calculation.

With both of the modules described above problems such as identification, storage, and manipulation of the information must be handled in an intelligent fashion.

3.0 Software Design

3.1 Programming Style

The modules of the system will be described in some detail here. Figure 2.1 schematically shows these components and how they fit into the overall picture.

The system described below has been designed and programmed in a very deliberate fashion. The subroutines used are very general in form, and are written to deal with a wide range of input parameters. In some cases the elegance of a method has given way to functionality; the reason being the openendedness of the system. The digitized data may come from a number of different sources and the manufacturing method used to create the mask will vary; so it is important not to take advantage of a specifics of either process, but to remain as general as possible. In addition the arrangement of modules follows the logical progression of the design process; allowing for refinement of the design at each step.

3.2 Program Language and User Interface

The C programming language was used extensively because of its intrinsic compatibility with UNIX the operating system of the workstations used, Starbase (the graphics libraries used), and X-Windows (the user interface used); since these were written in C. C's ability to dynamically allocate memory and use data structures allowed for the efficient manipulation of both large and small datasets.

Starbase is Hewlett-Packard's own full functioned graphics libraries. As well as advanced graphical techniques such as shading with multiple light sources and Z-Buffer surface removal, Starbase also allows for control of peripherals such as the mouse pointer

and knob boxes; these were used in the implementation of the system. Hewlett-Packard's version of the X11 windowing system is written with low level Starbase calls which further ensure compatibility.

3.3 Preprocessing Routine

The task of the preprocessor is to identify and tag the dataset and to convert the data from its raw digitized form, into a complete, conceptually rectangular, array of points. If necessary the preprocessor should remove extraneous data and reduce the dataset to contain only the area of interest. Also, since the dataset must be complete , the preprocessor should fill any holes in the data that might have occurred during digitization.

The preprocessor described here contains all the elements necessary to reduce a set of raw digitized data. Since this system is independent of the method of digitization, and the preprocessor can tailored for different types of input data formats, this discussion will be limited to defining the input format for data to surface conversion and to outline some of the possible processing concerns with any specific raw data format.

The example preprocessor will convert a file of raw datapoints taken with the Cyberware laser scanning system (see Appendix A) to a formatted file that can be read by the data to surfaces conversion module. Unigraphics II v8.0 design software (UGII) with their Graphics Interactive Programming language (GRIP) were used for this task. GRIP is UGII's FORTRAN like interface language which allows the user to perform most of the operations in UGII, such as picking, view manipulation, and also extended features, such as file input and output (IO) as part of a customized program.

3.3.1 Raw Data Collection

The digitized data is collected in the following way. The system uses a laser scanning technique to digitize a pilots head in a cylindrical manner, with the z-axis orientated up thorough the center of the head. A radius value is generated for each of a set of datapoints in a grid containing 512 longitudes and 256 latitudes.

In the areas such as the hair line, neck and eyes, bad data may be collected, or data may be lost. Reflection off the eyeball may cause data loss around the eyes, and the hair or clothing may diffuse or scatter the laser beam causing problems. If the scanned dataset were complete there would be 131,072 points defined in space, contained in a 12x12x12 inch cube; however, much of this data, as mentioned above will be lost or scattered and will not be usable.

3.3.2 Raw Data Reduction and Conversion Process

Since the Cyberware system will create a dataset containing the entire head, with the format and problems described above, this preprocessor will allow the user to perform three basic functions:

1) Extract a subset of data from the complete dataset.

2) Fill in the holes in the dataset using a simple linear interpolation scheme.

3) Write out a correctly formatted file for the data to surface routines.

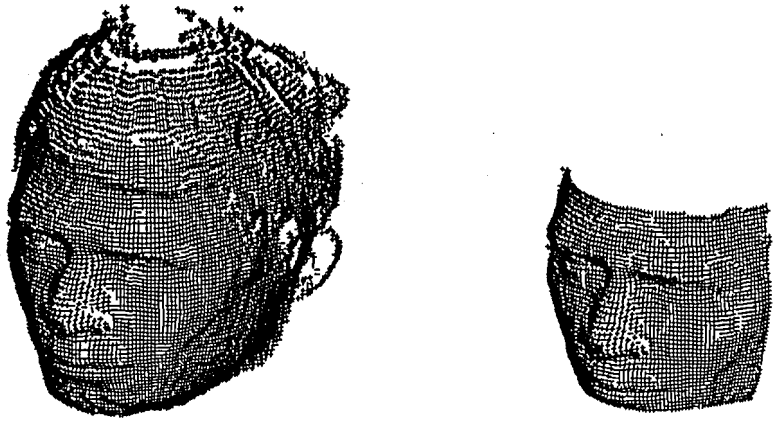
3.3.2.1 Programming Considerations

The size of the dataset presents some programming problems. Display and viewing transformations require a some time to complete and the density of the data causes difficulty in interpreting the display; the solution used here is to visually work with only

the even latitudes and longitudes, i.e. only one fourth of the data is actually displayed, while the entire dataset is internally manipulated. Also, since GRIP does not allow for dynamic allocation of arrays, rather than waste the additional memory by reading the entire dataset into a large three dimensional worst case array, IO is done directly from disk files, rather than Random Access Memory (RAM), thus reducing the amount RAM to disk memory swapping needed to manipulate the data.

3.3.2.2 Subregion Extraction

As the dataset is read into UGII each point is converted to its Cartesian representation in English units and assigned geometric attributes that define its latitude and longitude



(a) (b)

Figure 3.1 Raw Dataset (a) Complete / (b) Reduced

relative to the dataset. Figure 3.1 (a) shows the complete raw dataset. Once this dataset has been read in it can be viewed and rotated using all the functions of UGII.

The user is prompted to select data points that define two opposing corners of the reduced region of interest. The points contain attribute information describing their

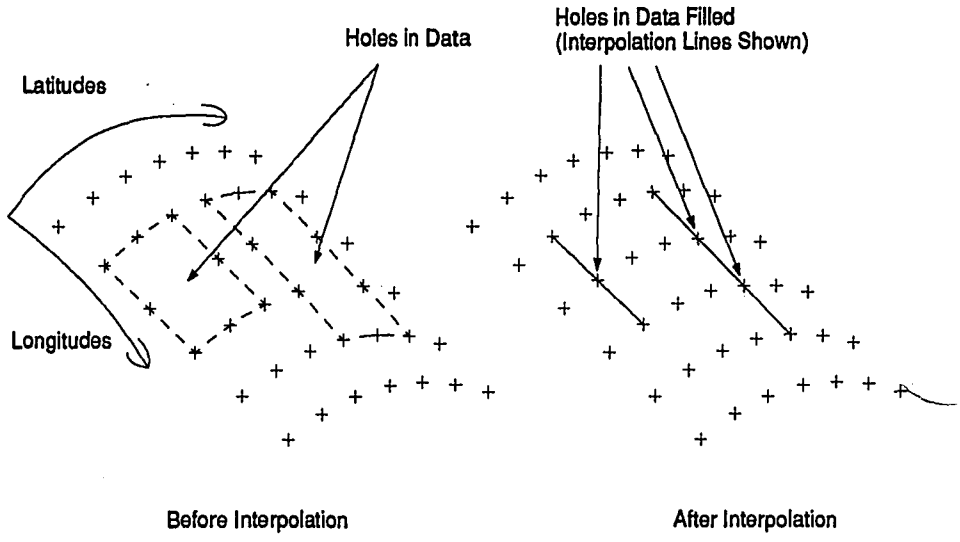


Figure 3.2 Data Interpolation Scheme

location in the dataset, this allows the region defined by these corner points to be extracted from the complete set. Figure 3.1 (b) shows the reduced dataset region with voids in the data shown. This region can either further reduced, if necessary.

3.3.2.3 Filling Voids in Data

Voids in the subset of data are filled with a simple linear interpolation scheme in the longitudinal direction. Figure 3.2 shows a small region of data before and after the data interpolation. Simply, the holes in the data are filled with points that lie on the line through adjacent longitude points, and are spaced appropriately to correspond with the data in the latitudinal direction. Other, more sophisticated methods of data interpolation can be employed, or, even better, the digitization method can be improved to avoid losing datapoints. This interpolation method is presented as one simple alternative.

3.3.2.4 Output File Format

The last step in preprocessing involves writing the data file with the information arranged in the proper format. Once the selected region of data is satisfactory, it is written

8,0,0	J. Prine SS#100-98-0110	** ID String
0,0,1.21	85 153	** Data Dimensions
0,1,2.42	-2.1928 .2813 -2.9358	** Data
0,2,3.63	-2.2107 .2813 -2.8802	
0,3,4.83	-2.2148 .2813 -2.8338	
0,4,6.04	-2.2054 .2813 -2.7972	
0,5,7.25	-2.2080 .2813 -2.7502	
0,6,8.46	-2.1857 .2813 -2.7232	
0,7,9.67	-2.1630 .2813 -2.6965	
•	•	
•	•	
-1,-1,-1	•	
(a)	(b)	

Figure 3.3 Raw (a) and Formatted File (b) Formats

out as a formatted file, the user is prompted for the name of the file and an identification string for the data. This identification will be carried through the remainder of the processes. Since, in this case, the raw input data is in column, or longitude major format the preprocessor first performs a transposition of the data, then writes the file. Figure 3.3, (a) and (b) show the input and output data files respectively. For the original format, the data is arranged as follows:

Latitude Value, Longitude Value, Radius (in mm)

The first line of the formatted output file contains an identification string. The second line contains the dimensions of the data (in rows and columns). The remainder of the file consists of the coordinate data points in Cartesian form, i.e. X Y Z.

This formatted data is now ready to be read into the fitting program and surfaced.

3.4 Data to Surfaces Conversion Routine

The task of this routine is to retrieve the formatted digitized data and to create geometric surfaces that will represent the topography of the face. The surfaces defined will approximate the data within a user specified tolerance. This is done using an adjustable recursive subdivision process. This process and its variables are described in this section.

3.4.1 Setup and Basic Patch Definition

Initially the formatted digitized data is partitioned into relatively square subregions as shown in Figure 3.4. There are several parameters that can be set, before the actual surface fitting takes place. The use of these parameters will be more clear after the fitting process has been explained.

As a first approximation each subregion is approximated with a single third degree Bézier surface patch. Simple 3rd degree Bézier patch representations are used to approximate all of the regions. This surface type was selected because of its flexibility as

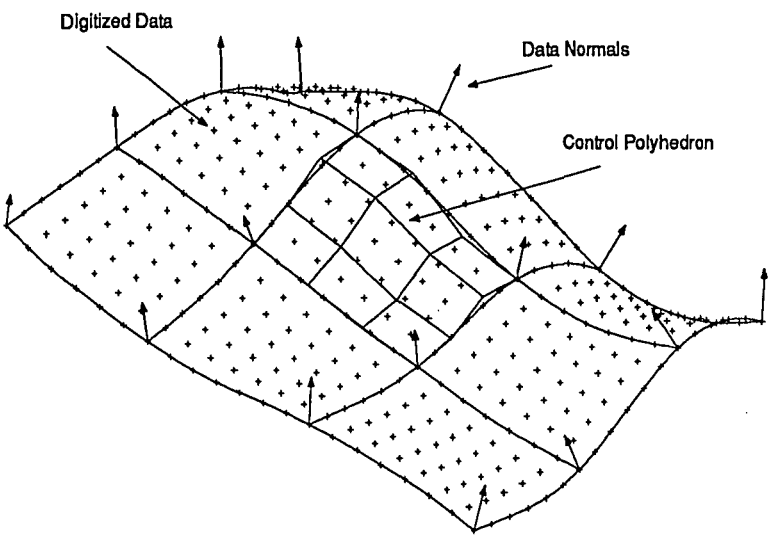


Figure 3.4 Initial Patch Arrangement

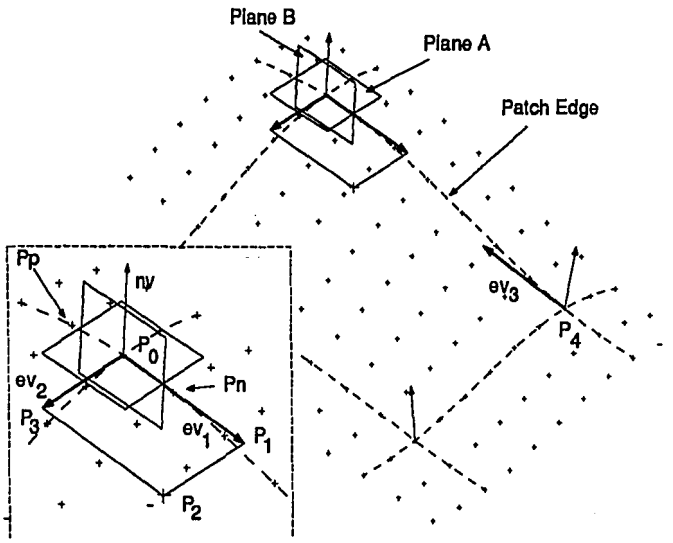


Figure 3.5 Definition of Bezier Control Polyhedron

well as its ease of manipulation and simple subdivision (see Appendix B for a discussion on B-Spline and Bézier patch representations). Figure 3.4 shows a simple partitioned dataset; data normals are shown constructed at the four corners of each patch. The point at the base of each normal is called a node. Nodes and their associated normals are shared between adjacent patches. The direction of each normal found by averaging the 23 point set of data surrounding the associated node. Figure 3.4 also shows a control polyhedron for one of the subregions, this mesh controls the shape of the Bézier patch that approximates the data in the subregion. The location of the nodes and the direction of the data normals completely define this control polyhedron.

Figure 3.5 shows the center patch in Figure 3.4, this patch will be used to explain the construction of the control polyhedron. The definition of the four points $P_0..P_3$ in one corner of a typical control polyhedron follows (the three remaining corners are defined in a similar fashion). The vector $e\vec{v}_1$ should lie on plane A perpendicular to the normal vector $n\vec{v}$ and passing through the the associated node; and at the same time, should be

orientated in the direction of the datapoints along the edge of the subregion close to that node. This is done by defining an auxiliary vector $\vec{s}$, (not shown in the figure). The direction of vector $\vec{s}$ is defined as follows:

$$\vec{s} = p_n - p_p \qquad \text{Eq \#1}$$

The vector $\vec{s}$ is moved so that it's tail corresponds to the associated node. The plane B is defined as containing these two vectors. Finally, the direction of $e\vec{v}_1$ is defined by the intersection of these two planes, A and B. The length of vector $e\vec{v}_1$ is defined as one third the linear distance between the two edge nodes P_0 and P_4. The vector $e\vec{v}_2$ is defined in a similar fashion from the adjacent edge information.

Figure 3.6 shows the surface patch defined by the control polygon created this way. The sense of these vectors $e\vec{v}_1$, and $e\vec{v}_2$ correspond to the direction of the tangent vectors of the surface they define at that corner point. By aligning the vectors in this fashion, i.e. using points on both sides of the node, this will ensure that the direction of the tangent vectors will correspond for all the surface patches associated with that node.

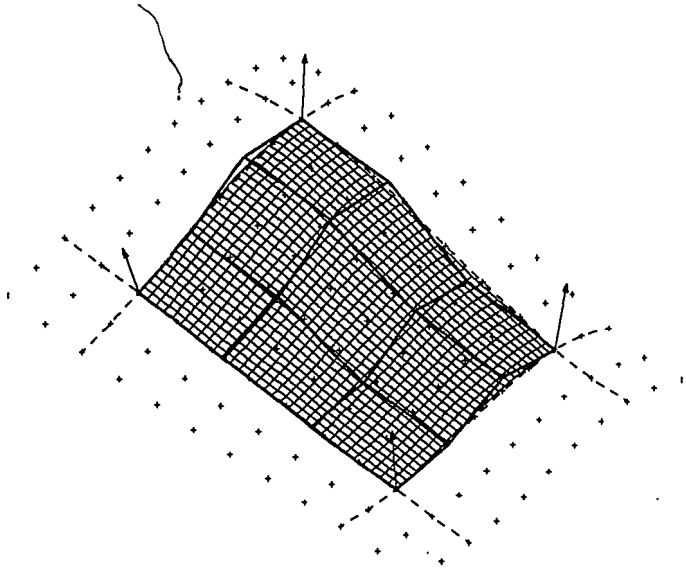


Figure 3.6 Constructed Bezier Patch

The opposite vector $e\vec{v}_3$ is defined in a similar fashion. This method is applied to each of the four edges, leaving only the internal points to be defined. The internal points are also simply defined, in the case of the patch corner shown in Figure 3.5:

$$P_2 = P_0 + e\vec{v}_1 + e\vec{v}_2 \quad \text{Eq \#2}$$

From the definition of the Bézier surface it can be seen that since the point data associated with a surface edge define the control polygon for that edge, the edges of adjacent patches will be identical, ensuring C_0 continuity. The manner in which the interior control polyhedron points are defined will ensure C_1 continuity from patch to patch, since, for each node, the set of four corner points for each associated patch will lie in the same plane. The manner in which the edge vectors are defined will further ensure coincident tangent vectors for adjacent patches.

3.4.2 Internal Storage of Information

During the subdivision process the system maintains continuity in adjacent patches through the use of the common patch nodes and normals. The program keeps track of this patch/node/normal associativity in the following way. The patch/node/normal database consists of two linked lists. A patch list and a node list.

The patch list contains data for each patch, this information includes the patches control polyhedron, the patch status, i.e. whether or not it has met the patch to data tolerance (1=accepted/0=not accepted), and for each corner of the patch there is a pointer to the corresponding node in the node list.

The node list contains data about each of the nodes, this information includes its location in relation to the overall dataset, its geometric location and normal, its status, i.e.

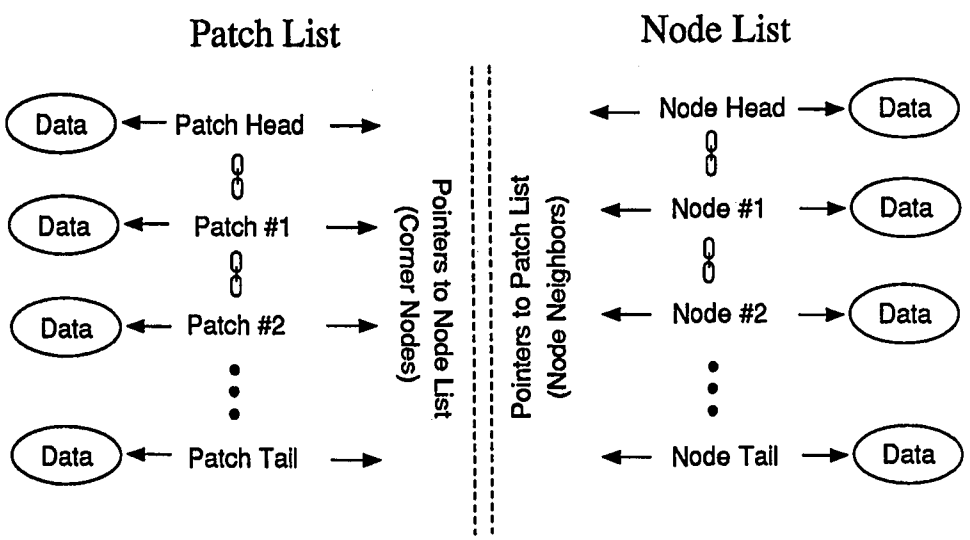


Figure 3.7 Double Linked Database

whether or not it lies on a point in the dataset or is a floating node (1=anchored/0=not anchored), and also contains pointers back to the patch list for each of the two, or four neighbors that each node will have. This is shown schematically in Figure 3.7. Through these lists the program can easily traverse the quiltwork of patches stepping across common nodes.

3.4.3 Patch Testing and Subdivision

Once all the initial patches are defined, the program then cycles through the patches testing the distance between the points in the subregion of the patch and the patch itself. If the distance found exceeds the maximum tolerance distance set, the patch is tagged as having failed and the testing continues until all the patches have been evaluated. All the patches that have failed will be subdivided. The subdivision occurs in three passes through the database of surfaces and data as follows.

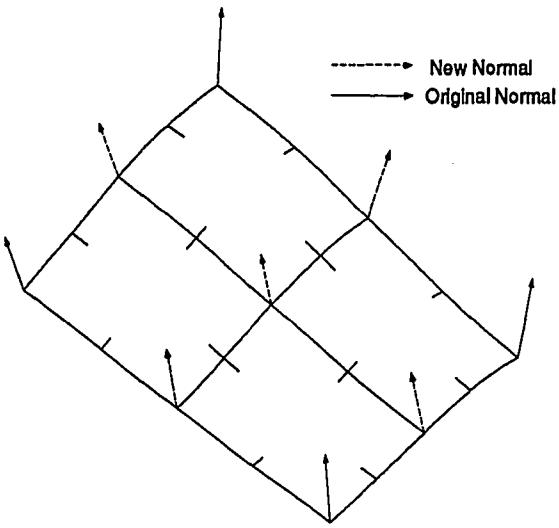


Figure 3.8 Subdivided Patch

Pass I

During the first pass each failed patch is subdivided into four parametrically equal patches that exactly represent the previous single patch, this arrangement is shown in Figure 3.8 The original patch is replaced in the patch list with these four patches and each of these patches is initially labeled as not accepted. This subdivision creates 5 new nodes and corresponding normals for each subdivided patch, these new nodes do not necessarily correspond to a point in the dataset and are added to the node list as floating nodes.

Pass II

If two adjacent patches have been subdivided a common node/normal will be produced. This pass first traverses the node list combining any common nodes and updating their patch dependencies. The updated node list is again traversed and any node that is not anchored and is associated with four unaccepted patches is attached to the center point of the associated subregion in the dataset and its status is changed to anchored.

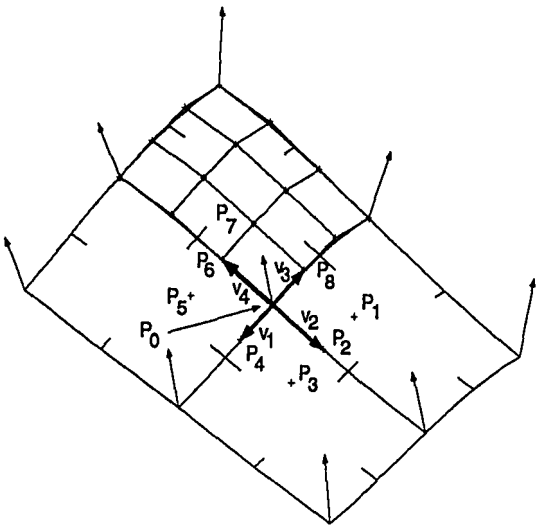


Figure 3.9 Center Node Readjustment

Pass III

In the third pass any anchored node whose list of patches contains only unaccepted patches is tagged and the surrounding patches are adjusted. For the center node of each subdivided patch this will always be the case.

Figure 3.9 shows a subdivided patch with the center node, normal and associated control polyhedron points shown. As an example of this process, a description of the patch adjustment for this center node is as follows. The center node corresponding to point P_0 is attached to the center point of the subregion of data and the corresponding normal to the data is defined. The four vectors $v_1..v_4$ are defined in the same manner as the initial the initial patch definition, as well as their internal points. Since only patch control points $P_0..P_8$ are moved, this adjustment only effects the associated corners of the four patches involved. The remaining control polyhedron points for these four patches are left untouched, maintaining the existing continuity with the patches surrounding these four. The net result is a readjustment of the center of what was the original patch closer

to the data. Figure 3.9 shows the entire control polyhedron for one of the new patches. This subdivision and adjustment of patches increases the amount of information necessary to define the region, while improving the overall surface approximation to the data.

The process starts over and continues until all the surface patches are within the tolerance defined or until the number of data points in either parametric direction of the subregion is less than or equal to 5 (the number of points necessary to define this type of Bézier patch is 16, or 4 points in either direction), any further subdivision will not yield a decrease in the amount of data necessary to define that region. Since data reduction is also a criterion for this process, a patch that has 5 or less points in either parametric direction in its subregion is labeled a Very Small Patch (VSP) and is automatically accepted, this patch type can be modified later if necessary.

3.4.4 Parameters That Enhance Subdivision

There are a number of parameters that are available for adjustment before the subdivision takes place. These parameters can improve the surface approximation and data reduction, and also reduce the time required to converge. With the above subdivision process in mind, the use and reasoning behind these can be better understood.

3.4.4.1 Mesh adjustment

The first method available is to adjust the initial distribution of patches. If it is possible to change the distribution of, or to arrange the patches such that single patch will cover a large area of relatively smooth data, the subdivision will produce fewer patches in that area, thus the amount of data reduction will be greater.

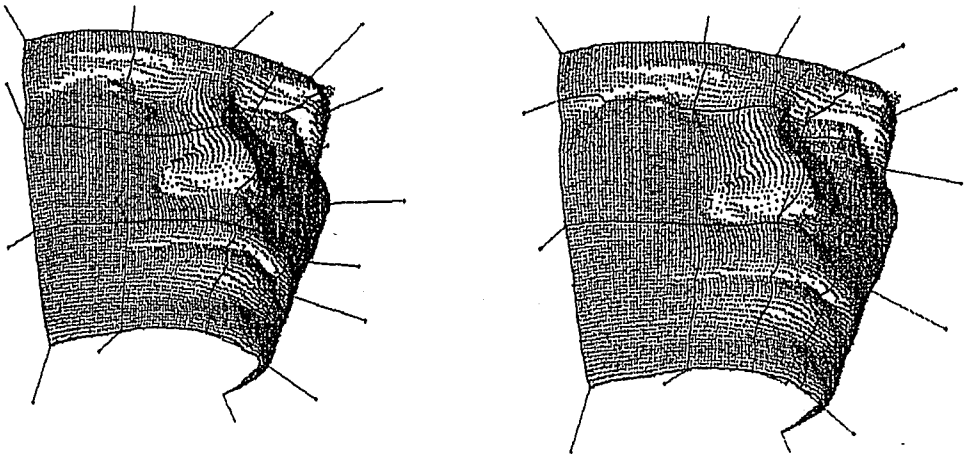


Figure 3.10 Initial Mesh Adjustment

Figure 3.10 shows an initial mesh and an adjusted mesh, from left to right respectively. This Figure shows the regions defined by the new mesh covering much larger smooth areas, which will yield a greater amount of data reduction. For this type of adjustment the initial data region boundaries are displayed and the user is prompted to select boundary line to be moved. This boundary is then attached to another user defined point in the data. The system automatically calculates the new data normals and updates all the necessary pointers. This will also allow the user to ensure that the initial quiltwork of patches is symmetric about the face, even if the dataset is not symmetric.

3.4.4.2 Data Smoothing

A second means available to improve the efficiency of the subdivision is to smooth the data in the noncritical regions of the face. In this case, noncritical areas are defined as areas that will not be directly involved in the intersection of a mask. Noncritical areas that may require smoothing may include the eyebrow hairline, and eye areas. The value of smoothing is to remove any nonessential topography that might cause the subdivision to

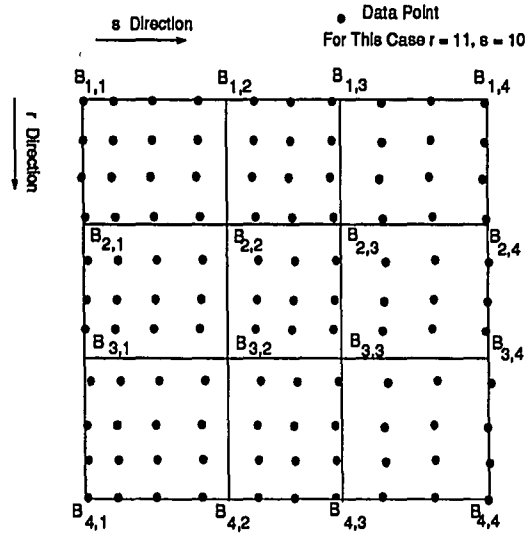


Figure 3.11 Data Array

continue to converge to VSP's in a region that is only necessary as a landmark. There are two methods implemented for data smoothing. For both methods, the user interactively selects datapoints at the corners of the region of data that is to be smoothed, and the program will attempt to fit a smooth fourth order Bézier surface that spans all the points in the region. The actual points in the dataset will then be replaced with corresponding points on the smoothed surface.

A forth order Bézier surface will interpolate exactly a dataset that contains four or less points in each parametric direction, i.e. a maximum mesh of 16 points; when more than 16 defining datapoints are used in the surface calculation, a median or smooth approximating surface results. So, the larger the region of data selected, the greater the smoothing effect. Figure 3.11 shows an array of datapoints with the vertices of an interpolating control polyhedron shown, smoothing will certainly occur in a region of this size.

Creating Bézier curves and surfaces for this type of data interpolation/approximation

problem has been studied by Rogers[3] and Adams and is briefly described here.

There are two different types of smoothing surfaces created. The first method uses the entire region of data to calculate the surface. Each data point is considered to correspond to a point D(u,w) on the interpolating surface. The data is conceptually arranged in a rxs topologically rectangular array. This type of data arrangement is shown in Figure 3.11. $N_i(u)$ and $M_j(w)$ (i=1..4,j=1..4) are the fourth order blending functions for each datapoint and its corresponding $u_{k,l}$ and $w_{k,l}$ value, (k=1..r, l=1..s). For each datapoint the equation for a Bézier surface (see Appendix B) provides a linear equation in the unknown control polygon vertices $B_{i,j}$.

The equation for a single datapoint is as follows:

$$D(u_{1,1},w_{1,1}) =$$

$$N_1(u_{1,1})[M_1(w_{1,1})B_{1,1} + M_2(w_{1,1})B_{1,2} + M_3(w_{1,1})B_{1,3} + M_4(w_{1,1})B_{1,4}]$$

$$\vdots$$

$$N_4(u_{1,1})[M_1(w_{1,1})B_{4,1} + M_2(w_{1,1})B_{4,2} + M_3(w_{1,1})B_{4,3} + M_4(w_{1,1})B_{4,4}] \quad \text{Eq \#3}$$

An equation of this form is written for each data point yielding a system of simultaneous equations, in matrix form:

$$[D] = [C][B] \qquad \text{Eq \#4}$$

where D is the r*s by 3 matrix of datapoints, C is the r*s by 16 matrix containing the N*M combinations of elements shown above, and B is the 16 by 3 matrix containing the corresponding vertices of the Bézier control polyhedron. If there are 16 data points, matrix C will be square, and B can be found by a simple inversion of matrix C:

$$[B] = [C]^{-1}[D] \qquad \text{Eq \#5}$$

But, this would be worthless for our case of data smoothing since this surface would pass through every datapoint, and the smoothed points on the surface would correspond to the unsmoothed points. If there are more than 16 datapoints then the matrix C is not square and the problem is overspecified and a solution can only be obtained in some mean sense. This is given by:

$$[B] = [[C]^T[C]]^{-1}[C]^T[D] \qquad \text{Eq \#6}$$

The above equation produces an approximating surface to the datapoints that follows its general shape and is everywhere up to C_3 continuous.

The values u,v for each datapoint are calculated using a chord length approximation in the following way:

For r data points in the u direction, the parameter value for the l^{th} point in a particular row is:

$$u_1 = 0,\ u_l/u_{max} = [\Sigma\, dis(p_g,p_{g-1})\ g=2..l]/[\Sigma dis(p_g,p_{g-1})\ g=2..r] \qquad \text{Eq \#7}$$

Similarly, for s data points in the v direction, the parameter value for the l^{th} point in a particular column is:

$$w_1 = 0,\ w_l/w_{max} = [\Sigma\, dis(p_g,p_{g-1})\ g=2..l]/[\Sigma dis(p_g,p_{g-1})\ g=2..s] \qquad \text{Eq \#8}$$

The above method is a useful smoothing technique when it is necessary to retain the overall shape of the data to be smoothed.

The second smoothing method uses only the datapoints on the perimeter of the selected region. This method fits fourth order Bézier curves through each set of edge points in a one dimensional version of the above process, then uses these four curves as edge curves to define a Bézier surface. This will cause the complete removal of any

Figure 3.12 Data Smoothing

features contained within in the area to be smoothed. Figure 3.12 shows a region around the eyebrow in the dataset. The original shape is shown in the center, the effect of method A is shown on the left, and the effect of method B is shown on the right.

3.4.5 Final Preparation for Subdivision

One other parameter that can be adjusted before the subdivision begins is the control point. This is a user defined point that will act as the origin for the mask model for subsequent operations. For our purposes, the control point corresponds to a point on the nose between the eyes, but it can be lie anywhere as long as its location is reflected in the mask database.

Once these parameters have been adjusted to the satisfaction of the user, the subdivision is allowed to run. At the completion of the subdivision, statistics pertaining to the quality of the approximation are tabulated. The system will calculate the number of patches used, the amount of data reduction, and the number, if any, VSP's produced along with their average and maximum error.

3.3.6 VSP Editing

Figure 3.13 shows a typical VSP, and its associated datapoints. As the final quiltwork of patches is viewed, the VSP's are shown highlighted in a different color. All of the geometric patches can be viewed and inspected individually, as well as just the VSP's. If a VSP does not fall in a critical area for the calculation of the face mask intersection curve it is not necessary to edit. Each VSP can be viewed by itself and its statistics such as number of points in its subregion, maximum and average error are displayed. Often a VSP results from one badly taken digitized datapoint that causes the system to futilely continue subdividing trying to converge to that point. If the patch does lie in a critical area, subdivisions can be forced to improve the surface approximation, while reducing the amount of data reduction.

3.3.7 Storing of Surface Data

Once the dataset has been satisfactorily approximated, the surfaces can be saved to a disk file in either of two formats. Format I is referred to as native format; here the system

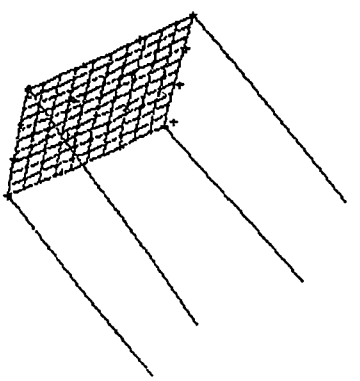


Figure 3.13 VSP and Associated Data

dumps to disk the node and patch information as it is stored in memory. This retains the identification and the patch/node associativity that were necessary for the patch subdivisions and will be necessary for the face/mask manipulation. Format II writes a standard format IGES file of the surfaces. This IGES file can be used across many software and hardware platforms, and allows this fitting algorithm to be used as a stand-alone data to surface utility program, possibly as a front end for another application.

3.5 Face Mask Manipulation/Intersection Routine

A surfaced representation of the face is now available for manipulation and intersection. The task for this routine is to read the geometric databases for both the face and a perspective mask, then to orient the mask on the face in a position that satisfies visual functional requirements. This positioning will be done with the aid of advanced display techniques that allow the face mask intersection to be displayed without actual numerical calculation. Once the user interactively locates the mask relative to the face, the actual intersection curve can then be processed for each desired face/mask orientation. The steps involved in this process are described below.

3.5.1 Conversion to NURB Representation

Where the data to surface routines manipulated the surfaces as simple fourth order Bézier surfaces, for the face/mask intersection algorithms the surfaces are converted to their equivalent NonUniform Rational B-Spline (NURB) surface representations (this conversion is described in Appendix B). The reason for this is as follows.

Any type of B-Spline or Bézier, as well as other parametric geometric surface types currently used, can be converted to an equivalent representation as a NURB surface. The conversion from Bézier or B-Spline is developed in Appendix B, and public domain conversion software for other parametric surfaces is available from the IGES office of the National Bureau of Standards[6]. This makes NURB entities the logical choice for a common entity type for portability to and from other CAD/CAM computer graphics systems.

Rather than constrain the mask creation process to only using fourth order Bézier

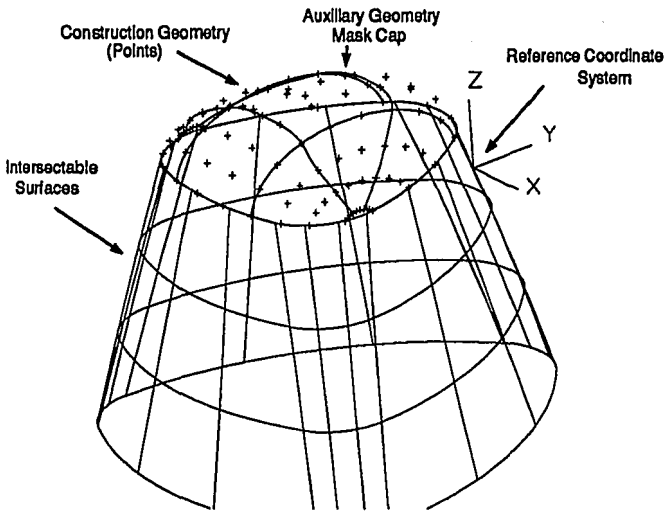


Figure 3.14 Typical Mask

representations, the mask maker can use any type of surface desired, then convert these to their NURB formats for later processing with this program. This ensures the flexibility of the mask design process as well as allowing the mask/face manipulation routines to deal with the face and mask surface patches in the same manner; increasing their efficiency and simplifying the internal database.

3.5.2 Variable Mask Creation Parameters

Although the mask can be of any shape, size, or number of surfaces, there are a few parameters used by the intersection routines that are adjustable during the mask database creation process. Figure 3.14 shows a typical mask with the different parameters discussed here annotated.

Many modeling packages support a layer or level function. Layers or levels can be thought of as a stack of slides, each slide can contain geometry and can be made visible or transparent. The first parameter for mask creation is that only the surface entities that are present on level or layer 1 will be considered for intersection with the face surfaces.

This allows for auxiliary reference surfaces to be created on layers other that 1 without causing unnecessary processing time during intersection calculation. These surfaces are displayed in a different color than the intersectable surfaces.

The second parameter is the origin of the mask surfaces. When the mask is read into the face/mask manipulation routines, the origin and orientation of the mask will correspond in location and orientation to the face control point specified in the data to surface conversion routine. This allows for an initial guessed mask location to be controlled. Figure 3.14 shows this origin displayed as a Cartesian coordinate system.

Figure 3.14 also shows some of the construction geometry used to create the mask. As this IGES file is read, only the NURB surface data is processed. This allows the file to contain auxiliary/construction geometry along with the mask surfaces without adding additional processing time.

Since the mask file will be read as a standard IGES file, the mask database creation process can be accomplished by any system that supports an IGES conversion. The first record of the Start Section of this IGES file will be read and used as the mask identification string. This should be used identify the mask's model and size, and will be carried through the rest of the fitting process.

3.5.3 Face and Mask Storage and Display

As the face file is read the surfaces are converted to their NURB representations, and loaded into a double linked list data structure that is similar to the one used in the data to surface conversion routine. As the mask file is read the surface patch information is stored in the same manner in a separate linked list, although there is no associativity from one patch to the next through a node list, as in face surface list.

The face and mask surfaces can be displayed in four different formats from any user defined orientation or view. Figure 3.15 (a)..(d) shows these representations, a simple "face" surface was used for explanation purposes. These formats allow the user to

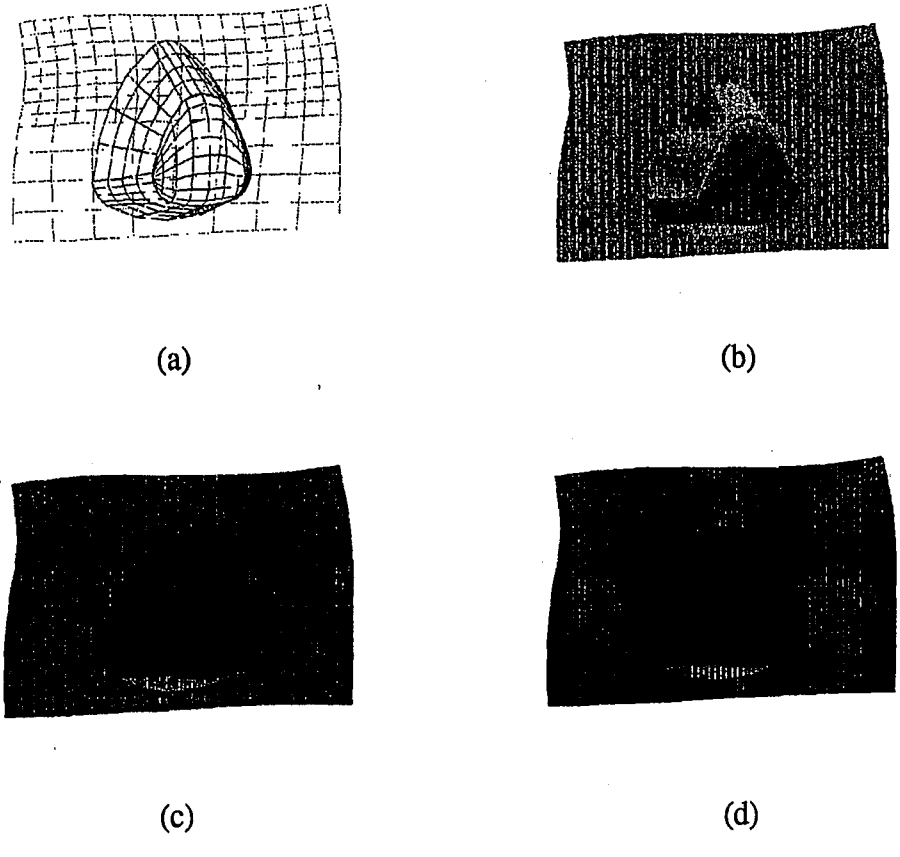


Figure 3.15 Surface Display Types

visually evaluate the potential face mask intersection curve without numerically calculating it.

Figure 3.15 (a) shows a wireframe representation. For this type of display each surface is drawn as a three by three array of isoparametric curves; this type has the fastest display time and is useful for finding an initial orientation from which the face mask manipulations can be done.

Figure 3.15 (b) shows the mask and face with Gouraud shading and multiple light sources. Smooth surfaces are modeled as meshes of polygons. Color and reflective surface coefficients for each polygon are calculated based on the angle between the polygon normal and the position of the light source in space. The more polygons used, the better the approximation to the actual surface and the smoother the shading appears, in exchange for greater the computation required for display. Gouraud shading produces a smooth surface display with fewer polygons by linearly interpolating the normal along the edge of a polygon from the normals to the surface calculated at each polygon vertex. This produces a smooth transition from one polygon to the next, rather than the sharp change that results from a single normal for each polygon.

Figure 3.15 (c) and (d) show the face and mask displayed using smooth shading, with either the mask (c) or face (d) translucent. Gouraud shading is used here with ambient type light, rather than point sources of light. This allows the user to see the amount of room between the mask and face; and also aids in judging the symmetry of the displayed intersection curve.

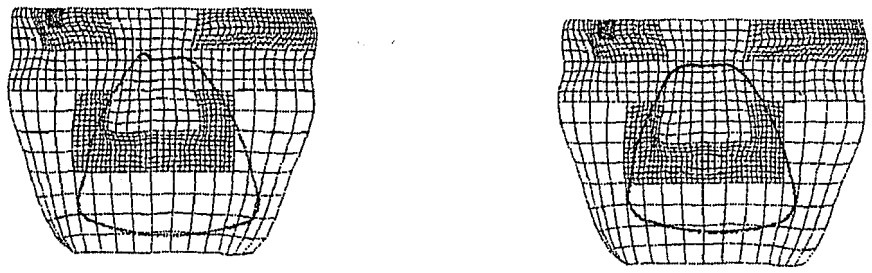
For the shaded images the hidden surfaces are automatically removed, and the visual

intersection curve displayed, this is done through the use of a Z-Buffer algorithm. The Z-Buffer algorithm uses a variation of the frame buffer concept. Where the frame buffer contains information about each pixel in the display, for Z-Buffering, a depth buffer is defined that contains information about the depth of each pixel in the display. This is usually the Z value in coordinate space, thus Z-Buffer. Hidden surface removal is done by comparing the Z value of each new pixel with the one currently displayed. If the new pixel is behind the old it is discarded, if it is in front it is displayed. Both Z-Buffering and Gouraud shading are functions are implemented through the Starbase graphics libraries.

3.5.4 Mask Orientation

The mask can be interactively manipulated independently of the face using the peripheral control knob box. Each knob is associated with a particular type of geometric transformation, translation and rotation are available about each Cartesian axis. The mask is transformed about its own origin, and relative to its own axis. All of these transformations are accomplished in real time, the hidden surface removal and visual intersection are recalculated automatically. As the mask is manipulated, each new orientation can be viewed in any of the view formats described above, from virtually any viewpoint.

Any orientation of the mask can be selected for intersection calculation. This process is relatively time consuming since it involves the numerical calculation of the intersection of multiple surfaces. The intersection calculation can either be done on line mode, processing each orientation in turn, or several different orientations can be stored and processed as a background process or batch job after the program is terminated. Once



(a) (b)

Figure 3.16 Calculated Intersection Curves

calculated, these intersection curves can be retrieved and visually evaluated.

Figure 3.16 (a) and (b) shows intersection curves generated from two different mask orientations. Although the mask positions vary only slightly, this Figures show the difference in symmetry produced, Figure 3.16 (b) would be the preferred curve.

3.5.5 Intersection Calculation

Since there are multiple face surfaces and also multiple mask surfaces, finding the face mask intersection curve involves more than just a basic surface to surface intersection algorithm. The intersection routine must have enough intelligence to hunt for intersection curves across adjacent surface patches, as well as within each path.

Since the B-Spline surface representation is not analytic, the intersection curve formed by two surfaces cannot be found in closed form. An intersection curve of this type is found as a collection of points in a iterative manner. Simply stated, finding the intersection curve is reduced to a minimization problem formulated in such a way as to be satisfied when each new point lies within a tolerance of the actual intersection curve.

That is, for two surfaces Q(u,v) and R(u,v), if a point P corresponds to a position on surface $Q(u_1,v_1)$, and also corresponds to a position on surface $R(u_2,v_2)$ and

$$| Q(u_1,v_1) - R(u_2,v_2) | < \text{Tolerance}, \qquad \text{Eq \#9}$$

then it is on the intersection curve. Once an initial point is found, the direction of the intersection curve can be found from the surrounding surface topography. The curve is traced across the two patches with each new point on the curve satisfying the above relation. For this formulation an existing point on the curve is required to start from.

Since this intersection routine will be run without any user intervention the system must be able to automatically find an initial intersection curve starting point for each of the mask/face surface combinations that will contain an intersection curve. This task is made even more difficult by the fact that both the mask and face databases can contain any size, shape, or number of surface patches.

3.5.6 Hunting for Intersection Start Points

The database for the surfaces is contained in two linked lists of surface patches, one for the face and one for the mask. The control polyhedron of a Bézier or B-Spline surface follows the general shape of the surface and its location and orientation data is stored for each surface in the database of linked lists. This information will be exploited in searching for intersection start points.

For two intersecting surfaces, the intersection start points are found reducing the problem to the intersection of a line segment and a triangle. For the case here, this reduces to finding any intersections between the triangulated control polyhedron of one patch with the line segments of the other.

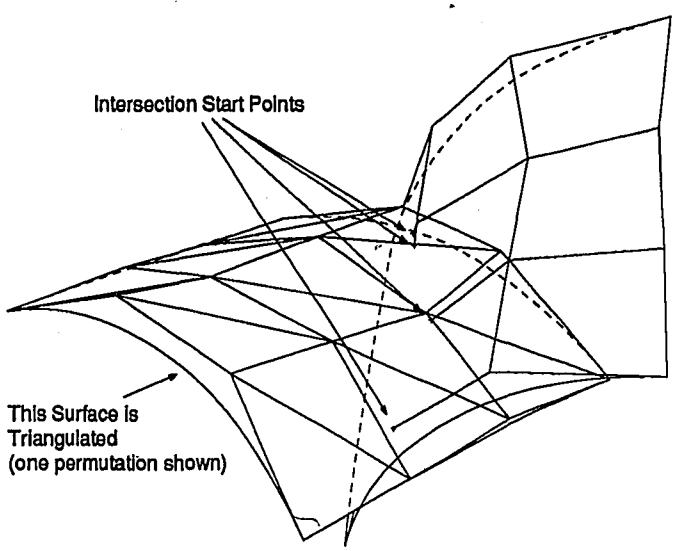


Figure 3.17 Initial Start Points

Figure 3.17 show this process for two intersecting patches. When the control polyhedron is triangulated, there are two possible triangle permutations for each set of four points; Figure 3.17 shows one of these permutations and the resulting intersection start points. This process is done by traversing the face patch list, triangulating each control polyhedron, then finding any and all of the intersections between this set of triangles and the line segments of the entire list of mask polyhedrons. The two lists are switched and the process is repeated, i.e. the mask polyhedrons are triangulated and tested against the face's. The traversal of the lists is very rapid, and since the face and mask lists store the control polygons in the same manner the switching of lists for the second pass is simple. This produces a list of potential intersection points. Stored along with each start point are pointers to both patches involved.

This method does not insure that each one of these points will lie on an actual intersection curve, or that all of the intersecting patch combinations have been found. The intersection algorithm has provisions to fill in any potential gaps that may have been

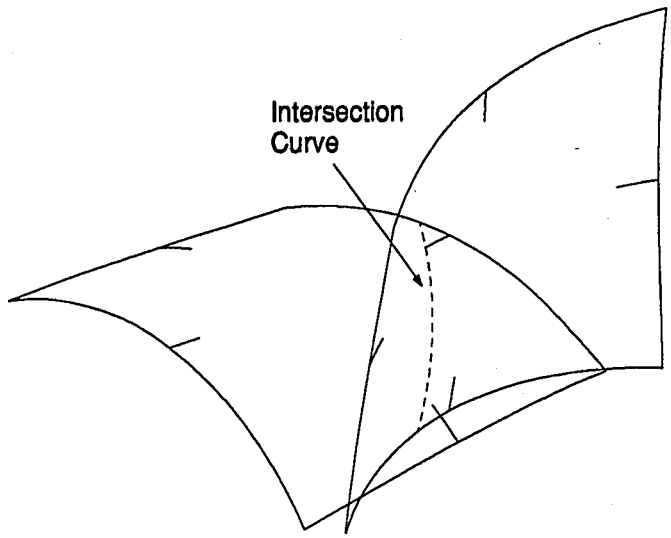


Figure 3.18 Patches with Traced Intersection Curve

missed here.

3.5.7 Tracing Intersection Curves

The list of potential intersection points is then passed to the intersection tracing routines. Here tracing involves first refining each potential intersection point into the intersection valley, then hunting along the base of the valley to the extents of the intersection curve contained by the two patches. Figure 3.18 shows a completed traced intersection curve for the two patches shown. The intersection curve for these two patches is bounded by the edges of the two patches, each patch limits one end of the curve.

Since there are multiple face surfaces and also multiple mask surfaces, finding the face mask intersection curve involves more than just a basic surface to surface intersection algorithm. The intersection routine must have enough intelligence to hunt for intersection curves across adjacent surface patches, as well as within each path.

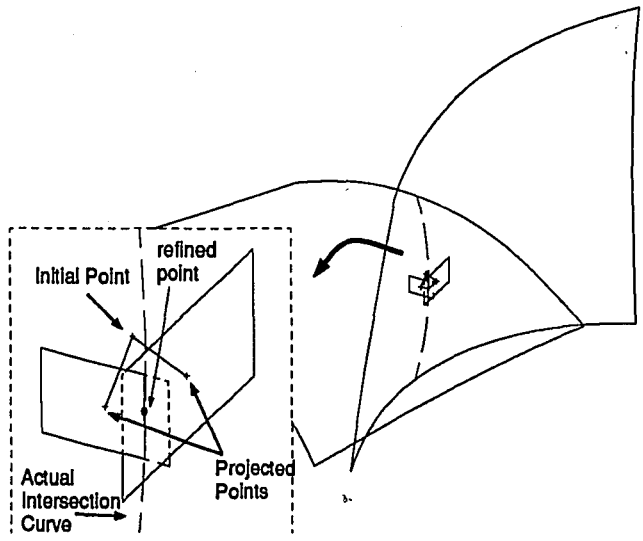


Figure 3.19 Point Refinement Arrangement

3.5.8 Point Refinement

The point refinement procedure is critical in this process. This routine takes a candidate intersection point and "refines" its position into the intersection valley. Since the candidate intersection start points, for example, may fall far away from the actual intersection curve and must be refined onto the curve.

The procedure of point refinement is shown schematically in Figure 3.19. The initial point to be refined is projected to each of the two surfaces along their respective surface normals. Two planes are constructed, one passing through each surface point, tangent to that surface. The intersection of these two planes forms a line. The next refined surface point lies on this line . The new refined point is at the intersection of this line and the plane that is formed by the initial point to be refined and the two projected points. This refinement process continues until the minimum distance from the new point to either of the surfaces is less than the tolerance defined, or a maximum number of iterations is reached. If the iteration limit is reached, the point is discarded

3.5.9 Curve Tracing

If a potential intersection point can be refined into the intersection valley it is then traced along the corresponding curve. If this is the first point in the curve, two points are created on either side of the first point by projecting a distance along the intersection line of the two tangent planes shown in Figure 3.19. The projected distance is defined as the Curve Refinement Tolerance or CRT. Each one of these points is itself refined and location on either surface examined. If either point is off, or on the edge of either surface, it is marked as a curve end point, and the tracing continues in the opposite direction. If these three points lie somewhere in the center of the intersection curve, tracing continues in an arbitrary direction until an endpoint is reached i.e. a point that is on the edge of either surface, then the list is reversed and the tracing continues until a second endpoint is found, thus completing the curve. The intersection points are kept track of by means of a double linked list structure. This allows the points to be stored in the order they are produced, and the list can easily be traversed in either direction for hunting toward both edges.

The completed list of intersection points is added to an overall list of intersection curves. This list of intersection curves also contains other information pertaining to the curves, such as pointers to the involved patches.

3.5.10 Next Point Prediction

Since the intersection points for each curve are stored in the order they are produced, it is possible to use their arrangement, as well as the surface topography to predict the next probable intersection point during the curve tracing process. Since the surfaces

defined by the face and mask surfaces will, in general, not be planer, whenever possible the next guessed intersection point is calculated using a circular prediction technique. Figure 3.20 shows the methods used. For the circular prediction method, the positions of the previous three intersection points are employed to define the arc, the new point lies on that arc, a chord length CRT distance from the last point. When three points are not available, or the previous three points a close to colinear, the linear prediction method is used. This technique projects a distance CRT along the line of the two intersecting surface tangent planes away from the previous two points.

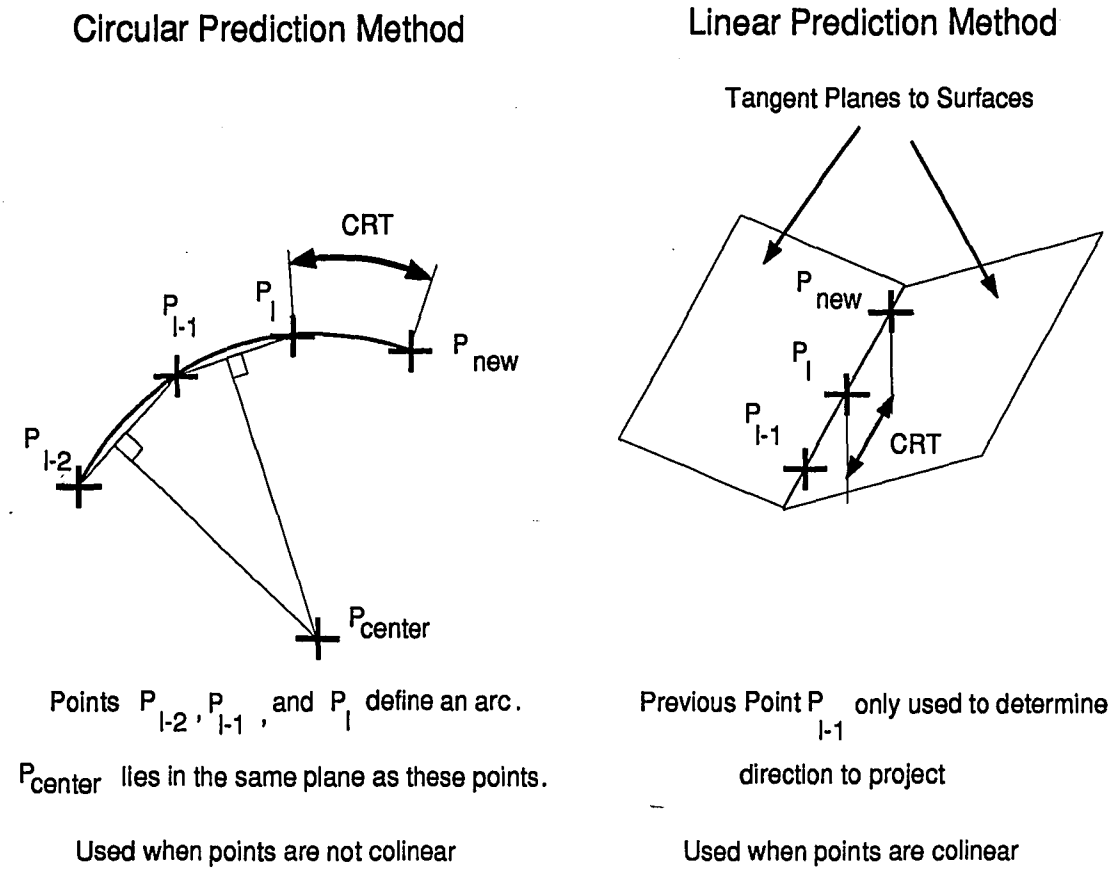


Figure 3.20 Linear and Circular Point Prediction

3.5.11 Tracing Across Patches

It is possible that the intersection start point routine might fail to find start points for all of the actual intersecting surfaces, therefore, it is necessary to hunt across patch edges for new curves. The face surfaces retain their associativity through the node list, so for each face patch, all of the surrounding patches are known.

Figure 3.21 shows a completely traced intersection curve and the adjacent patches indicated. At the completion of curve tracing for a single face-mask surface pair, possible adjacent intersecting curves are accounted for in the following way. If the intersection curve has an endpoint on an edge of a face patch, that point is added to the end of the intersection start point list along with the face patch adjacent to that edge, and the current mask surface patch. This will ensure that, as long as one intersection start point is found for each intersecting mask patch, all of the intersection curves associated with that patch on the face will be traced. In the far majority of cases, each mask patch will have multiple associated face patches due to their respective creation processes. Note that if a curve

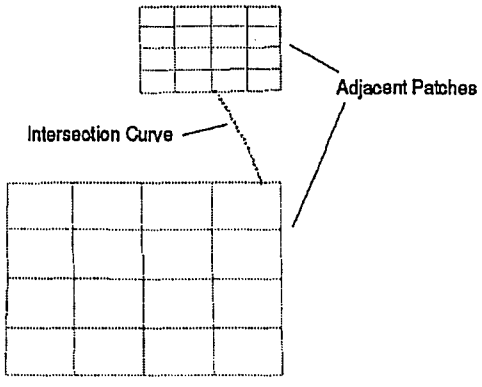


Figure 3.21 Intersection Continuation Across Patches

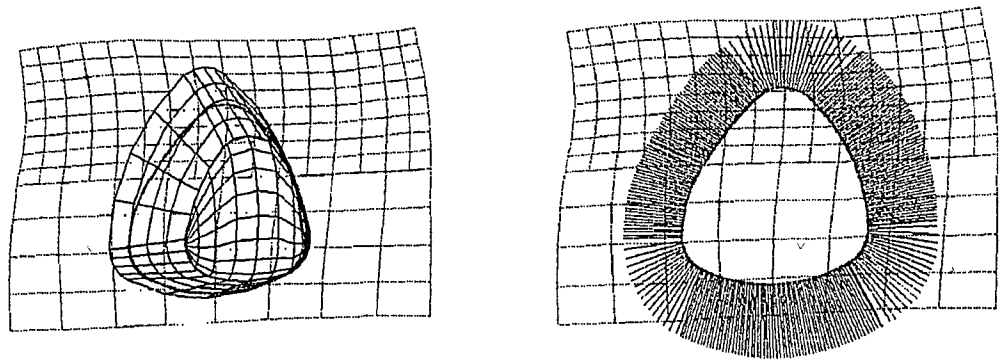
endpoint falls close to a node, all of the patches associated with that node are processed.

While the list of potential intersection points is exhausted, the list of intersection curves grows, and each new potential intersection start point is checked against this list. If the new point is a start point is associated with two surfaces for which a curve has already been found the distance between the new point and any of the points in the existing curve is checked against the CRT. If it is within the CRT it is discarded. This allows for multiple separate intersection curves between two surfaces without generating coincident curves.

3.5.12 Curve Segment Sewing

Once all the start points have been exhausted, the list of intersection curves segments is sewn together in the correct order. Coincident points are eliminated (these will occur from the end of one curve to the start of another).

Figure 3.22 (a) shows a completed sewn intersection curve. As each intersection point was generated, the normal from that point to the mask surface was also generated, these



(a) (b)

Figure 3.22 (a) Complete Intersection Curve / (b) with Normals

normals are also ordered and are all adjusted to point in the same direction, i.e. in or out of the mask surface. Figure 3.22 (b) shows the intersection curve with associated normals.

3.5.13 Curve Storage

This complete intersection curve can now be written to disk, where it is stored in a simple format. The identification string for both the face and the mask along with the transformation matrix applied to the mask for this particular intersection curve are contained in the header of the file. A sequential list of each intersection point with its associated unit normal follows. As the file is written the location and orientation of the points are converted to that of the original mask file. These points, in conjunction with the original mask file are now ready for the next step, postprocessing.

3.6 Intersection Curve Postprocessing

The task of the postprocessor is to convert the mask intersection file into a machine readable program that will be used to produce an actual mask using an automated manufacturing technique. The stored intersection curve contains information to support several different types of manufacturing methods. The information can be used to design a specific mask mold, insert to a mask mold, or possibly to trim a mask blank.

Here, a simple postprocessor was written for the latter. This postprocessor will use a 3-axis CNC milling machine to trim an oversized mask blank. As in the case of the preprocessor this system does not specifically concern itself with the type of manufacturing method used. The postprocessor presented here is used as an example showing some factors that must be addressed in postprocessing.

The setup will be as follows: An oversized mask blank will be made through a thermoforming process. The mask blank will be inverted and fixtured as shown in Figure 3.23, an example intersection curve is also shown. A ball end tool will then be driven

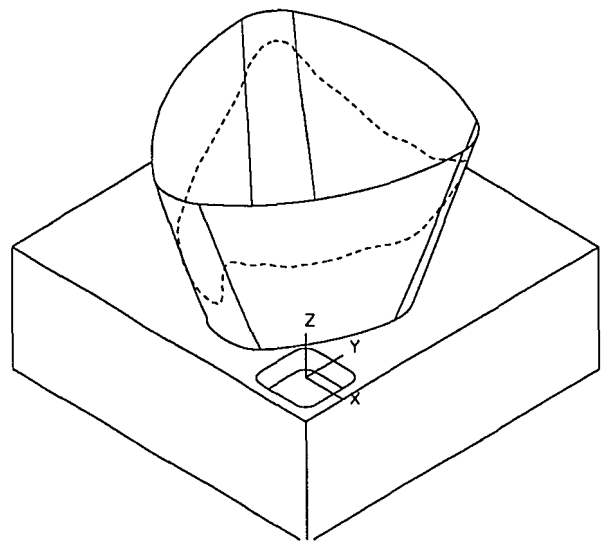


Figure 3.23 Fixtured Mask with Intersection Curve

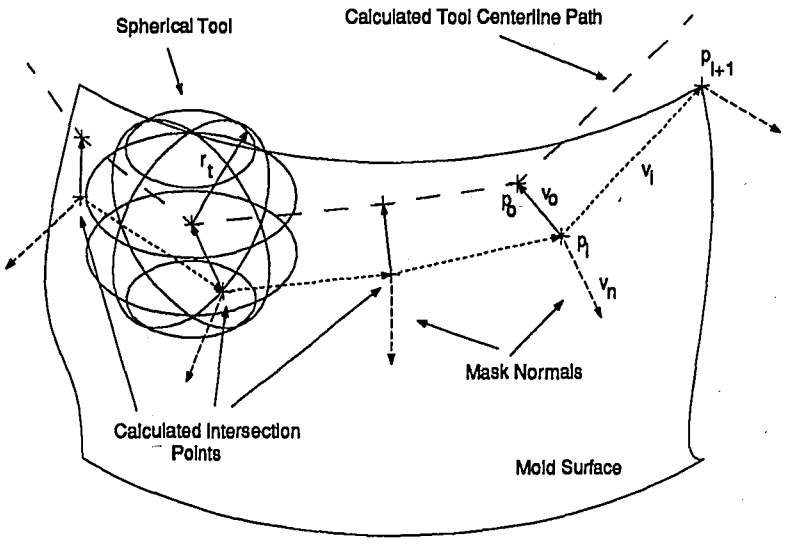


Figure 3.24 Curve Offset Calculation

along the intersection curve, with the proper offsetting to cause the tool to be tangent to the intersection curve.

The postprocessor must reorient the intersection curve data to correspond to the origin of the fixture as shown, then calculate the proper offset so that a ball end tool will move tangent to the calculated intersection curve.

One method to accomplish this tool compensation is shown in Figure 3.24. This Figure shows the calculated intersection curve points and also those points offset the proper length and direction for the tool centerline path.

For this example p_i is the point for which the offset to the centerline of the tool will be calculated. A vector $\vec{v}_i$ is constructed from the point of interest p_i to the next point in the intersection curve p_{i+1}. The cross product of this vector with the mask normal vector at p_i, $\vec{v}_n$, define a vector $\vec{v}_o$. The center point of the tool, p_o is found by offsetting the radius of the cutting tool r_t in the direction of the vector $\vec{v}_o$ from the point p_i.

The calculation $\vec{v}_o$ and subsequently p_o is as follows:

$$\vec{v}_o = | \vec{v}_n \times \vec{v}_i | \qquad \text{Eq \#10}$$
$$p_o = p_i + r_t * \vec{v}_o \qquad \text{Eq \#11}$$

For this postprocessor, it is not critical that the normals all point into or out of the mask shell, just that as a group they all point into or out of the mask shell. The newly calculated offset points are checked to ensure an offset in the proper direction. Figure 3.22 also shows the properly offsetted path for the center of the tool. For the actual machining process, since the tool length offset is generally calculated from the end of the tool, not the center, the tool center points must be translated the radius of the tool, in the negative Z direction, relative to the coordinate system shown in Figure 3.23. This set of points defines the path that the tip of the tool will travel through. This path is then converted directly into machine code. In this case, G codes compatible with a Bridgeport BOSS 15 type control are used. Figure 3.25 shows a segment of this code.

Manufacturing factors such as engage and cutting feeds and speeds as well as cooling considerations need to be addressed. For this particular method these factors are

```
'J. Prine   SS#100-98-0110<->Murmask_001'
N0000G70G90T01M06
N0001G00X0.0283Y-0.1297
N0002G01X0.0283Y-0.1297Z0.1869F5.
N0003G01X-0.0537Y-0.0412Z-0.0320F10.
N0004X-0.1037Y-0.0508Z-0.0171
N0005X-0.1520Y-0.0653Z-0.0038
N0006X-0.2002Y-0.0853Z0.0081
N0007X-0.2447Y-0.1053Z0.0195
                    •
                    •
```

Figure 3.25 Actual G-Coded Intersection Curve

dependent on the materials selected for the thermoforming process and the type of cutters used.

A complete example of this method is shown in the next section where these factors are defined for specific tools and materials.

4.0 System Example and Verification

4.1 Overview

This chapter will describe the system through two examples. Both examples will include data to surface conversion. Then, using the surface information, the face mask manipulation routines will be utilized to fit an actual mask to the face surfaces. The resulting intersection curve will be calculated.

The first example will show all of the features of the system and will follow the sequence of an actual fitting case. This requires a very close and therefore, fairly complicated mesh of patches to approximate the digitized data, and subsequent face/mask intersection curve.

Since the subject is no longer available to confirm the "exactness of fit", a second example is presented as a means to verify the integrity of the system. In this example, several physical models of the face and a model of a sample mask will be created and their various "fits" compared.

For the examples given here issues such as file and program management will not be addressed. Appendix C contains a brief user guide for the software system. This guide concerns itself with these issues as well as setting up the proper user environment, and estimating calculation times and specifying disk space requirements.

4.2 System Example

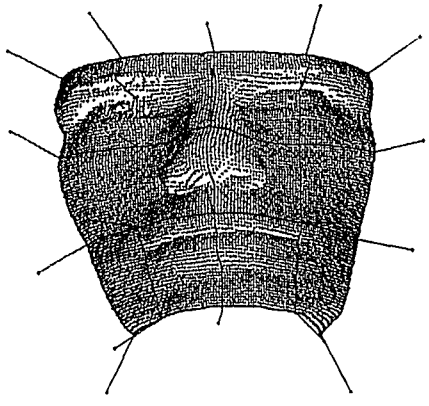
4.2.1 Preprocessing the Data

Since the raw dataset used here is the same as that used to describe the function of the preprocessor, preprocessing of the raw data will not be addressed here and the

example will begin with converting the raw formatted data to surfaces. The preprocessed file is "ex1.dat".

4.2.2 Converting Raw Data to Surfaces

In order to convert the raw data to the proper surfaces, the data to surface routine, "fit" is run. The preprocessed data files have been previously placed in the data directory. All of the system file manipulation routines read from, and write to, this same data directory. The system scans this directory and presents a list of potential preprocessed data files. The desired file, "ex1.dat", is selected using the mouse pointer and the system reads the file, displaying some initial dataset identification. Figure 4.1 (a) shows the condition of the initial dataset as it is read into the surfacing routine. Figure 4.1 (b) shows the available menu items. The datapoints are initially partitioned into relatively square subregions and the data is displayed with the normals shown.



Display Control
View Control
Set Face C-Point
Tolerances
Edit Initial Mesh
Smooth Data Area
Continue
Quit

(a) (b)

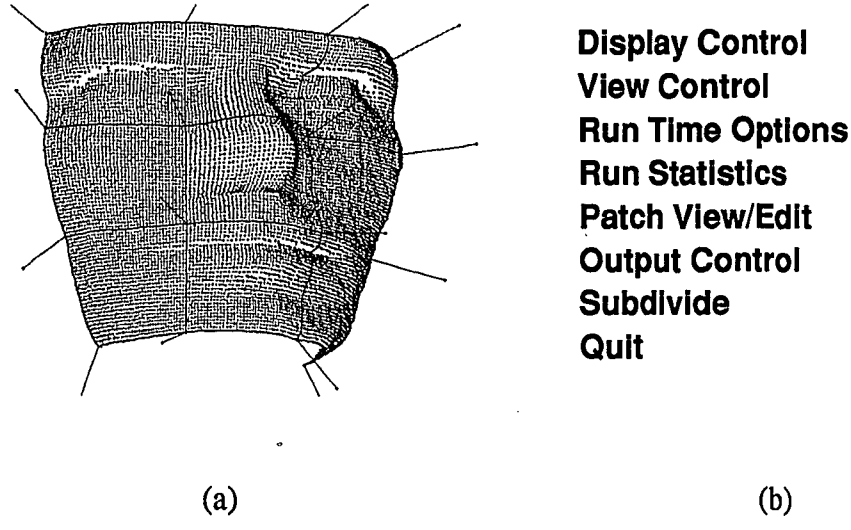
Figure 4.1 (a) Initial Dataset / (b) Menu Options

4.2.2.1 Initial Parameters

First, **Set Face C-Point** is selected from the menu and this prompts the user to select a point in the dataset. The control point will correspond to the origin of the mask model and is selected to be on the bridge of the nose, roughly midway between the eyes. The next parameter set to is the surface to data tolerance, or allowable error for the surface approximation; this is also located under the **Tolerances** option, and is set to be 0.100.

4.2.2.2 Data smoothing and Mesh Adjustment

By using 3D real time animation of the dataset as seen in Figure 4.1 (a), a user can see that there are some areas of the dataset that could be improved by smoothing, also the initial mesh of patches can be adjusted to enhance the efficiency of the subdivision process. The result of these smoothing and adjustment operations is shown in Figure 4.2. (a), note particularly the areas above the eyes and nose, this editing is performed as follows.



(a) (b)

Figure 4.2 (a) Smoothed and Adjusted Dataset / (b) Subdivision Menu

To clean up the data in poorly digitized areas, the **Smooth Data Area** option is selected from the menu and the corner points of each of these regions is subsequently selected. Either of the two smoothing methods described in Chapter 3 can be applied to reduce the scatter in the selected region. Different methods are employed where appropriate for this dataset. Five regions in all have been modified in Figure 4.2, these include two eyebrows, two eyes and in the nostril area.

For the adjustment of the initial mesh arrangement of patches we want to contain large, relatively smooth regions of data, in single subregions. This can be done by first changing the number of patches in the mesh across the face in the horizontal direction to three, this function is found under the **Tolerances** option, **#PU** is set to 3 (#PU is the number of patches in the U parametric direction of the data). Next, **Edit Initial Mesh** is selected, and the user is prompted to select boarders of the initial data subregions and attach them to new points in the data.

This concludes the adjustment of all the pre-subdivision parameters, and the dataset is now ready for subdivision. The **Continue** option is selected, and Figure 4.2 (b) shows the new available menu options.

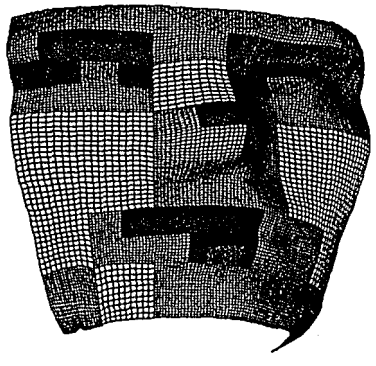
4.2.2.3 Subdivision and VSP Editing

To start the subdivision process, **Subdivide** is selected from the menu; by default this will run in an uninterrupted mode. The run mode can be changed to an intermittent mode where the system will pause between subdivision passes, this option is found under the **Run Time Options** menu. As each subdivision pass is made through the data, the system echoes the pass number and other related information. This information includes the

number of the patch being tested and the status of each patch test. The graphics display will also reflect the subdivision process; depending on the type of display chosen the system will update the display at the completion of each pass. The subdivision is allowed to run to completion and the system comes back with a fully updated display.

Figure 4.3 (a) shows the resulting quiltwork of patches with VSP's indicated by the darkest shade of gray. Evaluation of the overall surface to data fitting can now be done. The **Run Statistics** option provides information pertaining to the quality of the subdivision. The statistics for this example are shown in Figure 4.3 (b). This information includes the number of patches and nodes created as a result of the subdivision, the amount of data reduction, and also, data about VSP's that were created.

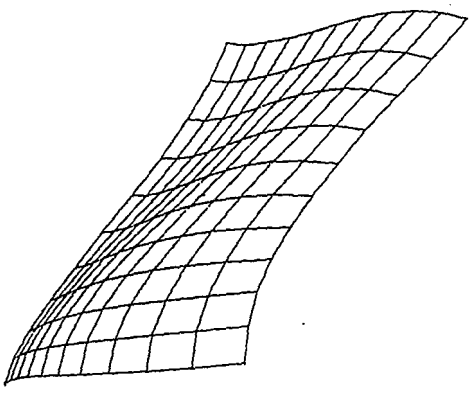
The overall collection of # VSP's produced show an average error of 0.045589, which is acceptable for our set tolerance, but a maximum error of 0.254396, which is unacceptable. Since there are a few VSP's that fall in critical areas, it would be



Number of Passes = 5
Number of Accepted Patches = 156
Number of Unaccepted Patches = 0
Number of Very Small Patches = 27
Number of Nodes = 242
Number of Data Points = 11011
Data Reduction = 26.6 %
VSP Error Data:
Maximum Error = 0.254396
Average Error = 0.045589

(a) (b)

Figure 4.3 (a) Patch Quiltwork / (b) Subdivision Statictics



Number of Points = 30
Average Error = 0.033422
Maximum Error = 0.111031
Minumum Error = 0.00000

(a) (b)

Figure 4.4 (a) VSP / (b) Associated Statistics

worthwhile to investigate them individually. This is done by selecting the **Patch View/Edit** option from the menu and each patch is examined individually.

This feature allows the user to view each patch as a single entity and tabulate its statistics, since here the user is only interested in the VSPs, the non VSP patches are turned off and the user is permitted to cycle through only the VSPs.

Figure 4.4 shows a single VSP, it's associated data and statistics Because of the location of some of the VSPs, their subdivision is forced to achieve a slightly better approximation in those areas, this is done by choosing **Subdivide** option from the available patch edit options, while that particular patch is being edited.

Now that the overall result of the subdivision is satisfactory, therefore the quiltwork of patches can be written out in native format for the next phase of custom fitting, the face/mask manipulation and intersection routines. This is done by selecting the **Native Output** option under the **Output Control** menu and entering a file name, in this case

"ex1" (no file extension) is entered, and the system saves the file. The data-to-surface step is now complete and **Quit** is selected from the main manu to terminate the program.

4.2.3 Manipulation of the Face and Mask.

In order to manipulate the face and mask, it is necessary to have an existing geometric model of a mask. At the time of this project, the contact at Wright -Patterson was unable to provide the necessary engineering drawings of the MBU 5/p mask, it's material or manufacturing process. As a substitute, a simple mask has been created using UGII design software and is shown in Figure 4.5. This mask consists of 7 geometric surfaces, one of which (mask cap) will not intersect the face. This model is created and converted into an IGES file in the manner described in Chapter 3. Note that the origin and coordinate system of the the mask are shown, these will correspond to the control point selected by the user as previously described (see section 4.2.2.1).

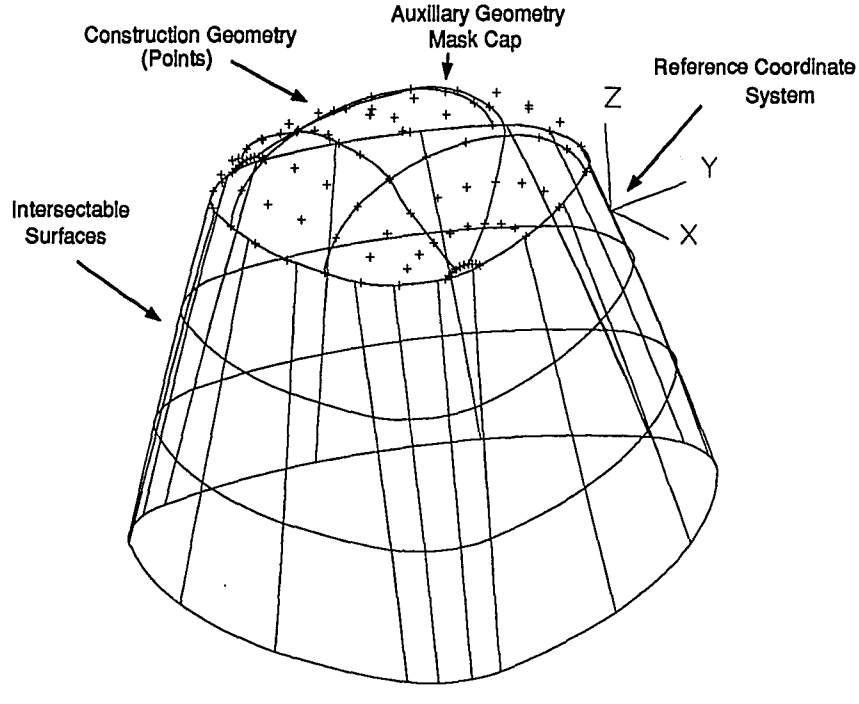
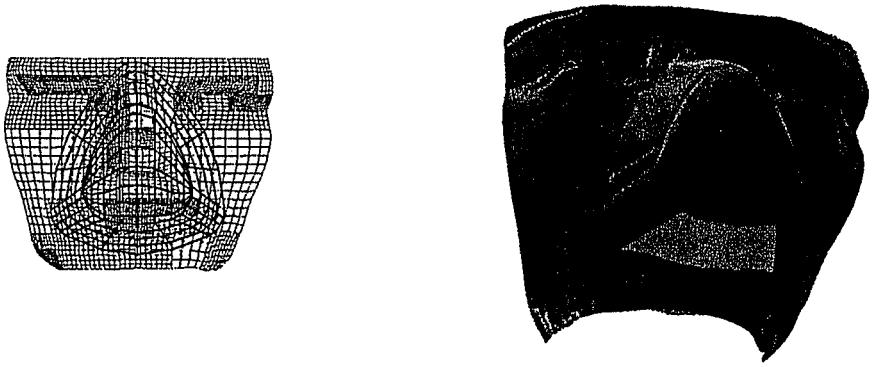


Figure 4.5 Simple Mask Created with UGII



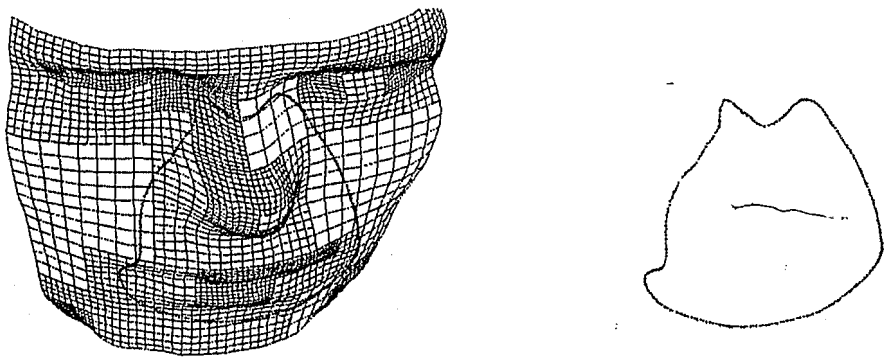
(a) (b)

Figure 4.6 (a) Initial Arrangement / (b) Adjusted Arrangement

The face/mask manipulation program, "sect", is run and the data directory is scanned for output files from the data to surface conversion program. The desired file, "ex1", is selected with the mouse pointer, and the file is read in and identified. The mask file to be used has been previously specified as part of the working environment and after the face file is read, the mask file is automatically loaded into memory also and displayed.

Figure 4.6 (a) shows the initial arrangement of the quiltwork of face patches and the mask. After these are read, viewing and display parameters are set for optimum visualization of their resulting intersection curve.

The next step here is to modify the position, or "fit" the mask. To accomplish this, **Mask Control** is selected from the main menu; this allows for translation and rotation of the mask about its own origin through the use of the peripheral knob box. The position and orientation of the mask are adjusted until the visual intersection curve defined appears to be symmetric, as well as located in the correct position relative to the face



(a) (b)

Figure 4.7 Intersection Curve (a) With Surfaces / (b) Without Surfaces

surface. Figure 4.6 (b) shows a visual intersection of this type. The different surface display options can be used to evaluate the intersection. These include: wireframe, Gouraud shading with multiple light sources, solid face with see through mask, solid mask with see through face.

Once the position of the mask is satisfactory, the actual numerical intersection curve must be calculated. The **Intersect** option is selected, and the user is given the choice of two calculation modes, either on line or batch processing. If batch mode processing is selected, the orientation is stored and the batch job will be submitted at the termination of the program, "sect" In this way several different orientations can be saved, then calculated later. For this example, **On line** is selected and the system displays the intersection points as they are generated.

Figure 4.7 (a) and (b) show a complete calculated intersection curve with and without the face and mask surfaces. Since every mask-face configuration is slightly different and

may include different numbers of intersecting patches, the time required to calculate a complete intersection curve will vary. The curve shown in Figure 4.7 required approximately 25 minutes to generate on a HP 400T.

After the complete set of curve points has been calculated it is stored in RAM and must be written to disk it it is satisfactory for later postprocessing. The **I-Curve View/Save** option is selected and the user can view any of the intersection curves currently in memory. This options allows for the curve to be displayed with or without mask normals. This menu, along with the regular display options allow the curve to be displayed in a number of different formats with or without the face and/or mask surfaces. The **Write IC File** option is chosen and the filename "ex1" is again entered and the curve is written to disk. Notice that the name for all the files written has been identical, the system allows for this by attaching a suffix that identifies the particular data type to the given name when it is written, in this way all the file can be written and stored in the same directory. The face mask intersection routine is terminated by selecting the **Quit** option from the main menu.

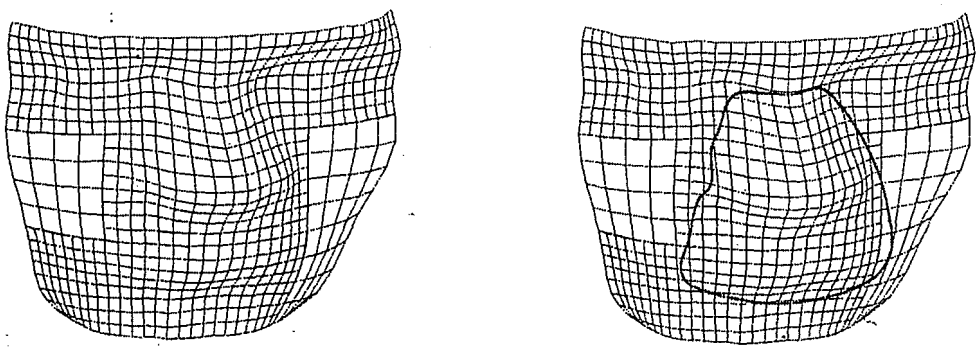
The intersection curve file contains information that identifies the face and mask datasets from which it was produced and also the particular transformation matrix applied to the mask to produce the intersection curve contained in the file. This concludes the fitting example. The system verification described in the next section takes the process one step further in postprocessing.

4.3 System Verification

For this verification process, the above steps will be repeated, but the surface tolerance specified during the data to surfaces portion will be much more relaxed, this will produce relatively simple quiltwork of patches to approximate the data. The reason for the creation of this additional surface approximation, and it's simplicity is two fold: First, since these surfaces will be used only to verify the integrity of the algorithms used, a close approximation to the digitized data is not critical. If the intersection curve matches a simple quiltwork of patches, it will also match a complicated one. Second, these surfaces will be used to create a physical model of the face using NC machining methods; due to tool path calculation and machine time considerations, a relatively simple quiltwork of patches is necessary. For the simple face made here generation and postprocessing of the tool paths required four hours, and the face itself required about 8 hours to rough and finish.

The geometric model of the face will be made by writing an IGES file of the quiltwork of surface patches after the data-to-surfaces conversion process is completed. This file of surfaces will be translated into UGII geometric model. UGII's manufacturing module will be used to generate machine code that will reproduce the surfaces using 3-axis machining techniques. This face surface will then be cut from wax.

The model of the mask will be made by using the existing geometric model created in UGII to produce a mold for thermoforming; and also a fixture for holding the inverted thermoformed mask for trimming. The intersection data generated by the face/mask manipulation routines will also be postprocessed it into machine readable code, and this

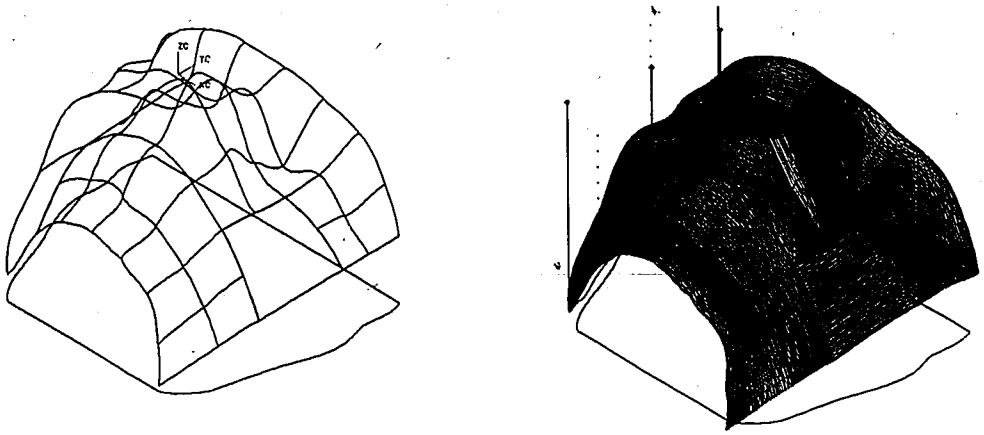


(a) (b)

Figure 4.8 (a) Patch Quiltwork / (b) Resulting Intersection Curve

code will be used to trim the thermoformed mask blank. The trimmed mask will then be checked against the face model to close the loop. This will demonstrate confidence in the surfacing routines as well as the intersection routines. The problem of determining a proper "fit" of the mask to the face is not explicitly addressed here and will be dealt with in the Chapter 5.

Figure 4.8 (a) and (b) show the quiltwork of patches and the subsequent intersection curve generated. The generation of these will not be discussed in detail, it is similar to the process described above, with the surface tolerance on the data to surface fitting routine set much larger. This example will deal primarily with the creation of the face and mask models, and their verification.



(a) (b)

Figure 4.9 (a) Surfaces modeled in UGII / (b) Generaed Tool Paths

4.3.1 Creation of Face Model

During the use of the data-to-surface routines, only one subdivision pass was needed to satisfy the tolerance requirements, this produced 39 surface patches. An IGES file as well as a Native file was written of the surfaces at the completion of the data to surfaces routine .

Figure 4.9 (a) shows the surfaces as modeled in UGII. The surface patches were read into UGII through its IGES conversion, these surfaces are shown in the orientation necessary for machining. Figure 4.9 (b) shows the tool path generated to create the face model. Because of the arrangement of the surfaces, the machine code generation must be done on separate portions of the face, then combined. The finished machined face is shown in Figure 4.10. These surfaces were machined with a 1/4" ball end tool, and the tool path to surface tolerance was held to within 0.005" of the exact surface definition. The surfaces are a rough approximation to the digitized face data, but even in this form

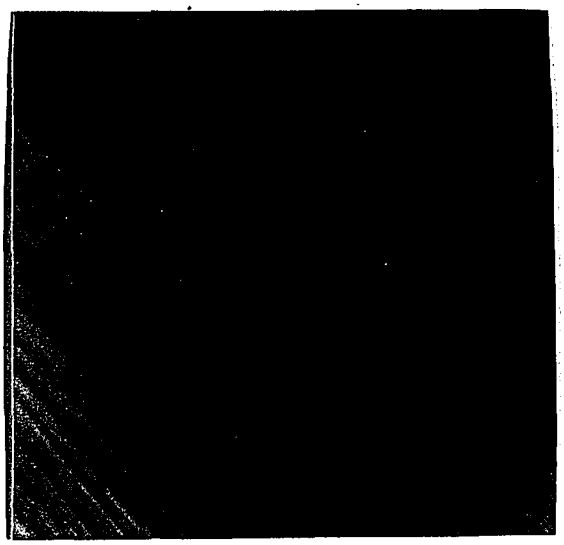


Figure 4.10 Machined Face Surfaces

examination of the machined wax replica shows that problems exist in the dataset around the eyes and nostril area.

4.3.2 Creation of the Trimmed Mask

Figure 4.8 (b) shows the intersection curve points generated from the face mask intersection routine. These points have been written to disk in a raw form, with their origin corresponding to the origin of the mask model. Before these points can be used to drive an NC machine they must be translated to the proper orientation for the fixture, offset to compensate for the diameter of the tool used, and reformatted into the machine code that is understandable by the target machine. For the example here, the machine is a Bridgeport Series I 3-axis vertical milling machine with a BOSS 15 control.

These specifications for a postprocessor correspond to those for the example postprocessor described in Chapter 3. The only user interaction necessary for this postprocessor is the name of the input file and the diameter of the tool to be used for the cutting procedure. Input and output files are read from the appropriate directories.

```
$ postp
Input File name (without .if): ex2
Data to be postprocessed...

Face Data -> J. Prine SS#100-98-0110
Mask Data -> Murmask_001

Transformation matrix used:
 1.000  0.000  0.000  0.000
 0.000  1.000  0.000  0.000
 0.000  0.000  1.000  0.000
 0.000  0.000  0.000  1.000

Tool Diameter: 0.125
      •
      •
      •
Postprocessing Statistics...

Program Limits:
  X  -2.84  -0.19
  Y   0.10   2.70
  Z   0.12   0.89

Approximate Machine Time:  0.92 minutes
Postprocessing Finished...
$
```

Figure 4.11 Postprocessor Run

Figure 4.11 shows a portion of the output for this run of the postprocessor. After the input file is entered, the system identifies the face and the mask sources, and also displays the transformation matrix applied to the mask, which, in this simple case, is the identity matrix. The tool diameter is then entered and the the properly offsetted output data is written. The system finishes with the X, Y, Z limits of the output program and the approximate machine time. The output is written to a file with the same input name but with the suffix of the three letter extension ".pun". This file is ready to be loaded and run on the specified machine.

Figure 4.12 shows the mask model in its fixtured position with the raw intersection curve points, and also the offset points calculated in the post The coordinate system for the raw curve is shown inverted corresponding to that of the original mask model. The coordinate system for the offset curve also corresponds to that of the fixture; this is the location of the origin that will be set on the NC machine when the mask is trimmed.

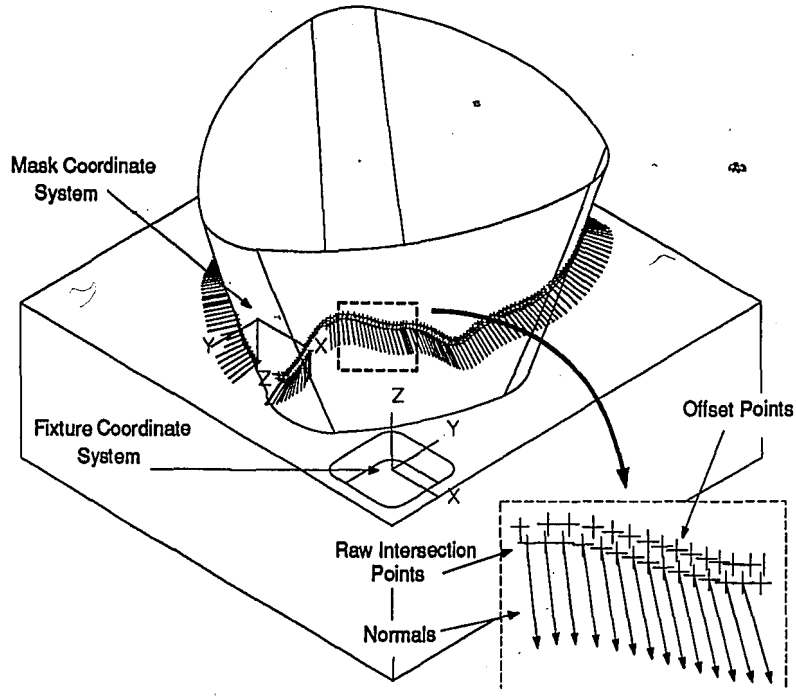


Figure 4.12 Mask Model with Raw and Postprocessed Intersection Curve

Figure 4.13 shows the machined face with its custom fit face mask. There is a close correspondence between the face surface and the mask shell trim. Some deviation has

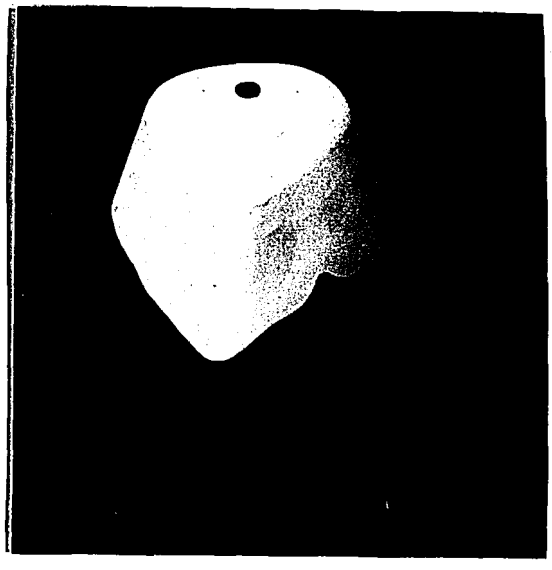


Figure 4.13 Machined Face with Mask

been produced by the mask shell itself flexing during the trimming process, also the mask shell deformed slightly while being thermoformed. Even with these factors, the mask trim is "close" to the face surface.

This correspondence indicates that the routines used in this system produce valid intersection information; and this information can be used to customize a face mask to follow the contours of a particular face.

This concludes the specific examples. Example one shows the necessary steps and the different parameters that are available for the custom fitting operation. Example two verifies the validity of the routines used, and, at the same time shows some of the extended options of the system in the form of the IGES conversion. A further discussion of the results and implications of these examples is contained in Chapter 5.

5.0 Conclusion and Recommendations

A prototype system for custom fitting of masks has been developed. A process that was previously completely manual has been transformed into a process that can be performed to a large degree in a computer aided environment. As well as providing a greater degree of flexibility and repeatability into custom fitting, the system also provides the ability to fine tune a design. The development of geometric models also expands the range of custom fit applications, and the existence of the geometric model can be utilized in other areas as well, as was seen with the physical creation of the face model.

The objectives set forth for the system were achieved, but during this study many new problems and questions surfaced. The issue of the quality of fit has not been specifically addressed. Although the mask trim may closely follow the contour of the digitized face, this does not ensure that the mask will function properly. Proper fit is not well defined and is subjective by nature.

Problems of scatter and poorly sampled areas in the raw digitized data sets effect the quality of the surface approximation, and this in turn will effect the fit of the mask. Tolerances defined, such as the data-to-surface tolerances, intersection curve, and machining tolerances are all variables that effect the fit of a mask on a particular face. The compliance of the face will surely effect the fit, this being a particularly complicated factor since the amount of compliance will vary in different areas where the mask is in contact with the face. It is doubtful that answers to these questions can be found in other than an empirical fashion. A population of subjects could be examined with these parameters studied, in an effort to define what constitutes a good mask fit. The geometric

databases created by this system of the face, and mask as well as the intersection curve would be very valuable in such a study. These models contain useful information about the face and mask in those regions. Presumably, in an effort to improve the fit, the mask trim could be adjusted in the normal direction of the mask or the face.

A method of implementation of this system might be to use the intersection curve as a starting point in the fitting process, then the curve can be adjusted in different regions by the designer to achieve a better fit. The intelligence behind which regions to modify and how much, must come from the type of study mentioned above, possibly integrated into the system using artificial intelligence techniques.

With regard to the to the routines contained in this system. The data-to-surfaces routine might be improved by developing ways to initially partition the data set more efficiently. Also, algorithms might be developed to locate the control polyhedron points to achieve a better surface approximation. For the face-mask manipulation and intersection routines as well as the data-to-surface routines, if more information is known about the origin of the raw data and destination of the intersection data, the routines might be modified to take advantage of the specifics of either process. The intersection routine, for example, might ,considering the type of manufacturing process used, output a specific density of points depending on a user specified machining tolerance.

The system that has been developed here is a good starting core for a complete automated custom fitting system. As many pertinent factors as possible have been addressed given the constraints of this research project, and for the majority of problems, workable solutions have been developed. Continued research, as mentioned above,

would be necessary to make the system viable in a Department Of Defense type setting.

Appendix A

Three different type of digitization will be briefly explained here. The first two are non-contact digitization methods and the third is a contact method.

Laser Scan Technology

The Cyberware Inc. system is used as an example of this type of digitization. With this method the entire head is scanned in a cylindrical manner. The grid consists of 256 latitudes and 512 longitudes, and each latitude and longitude pair has an associated radius. If the object being scanned were a perfect cylinder the system would collect an even grid of data, i.e. the 3D distance between each datapoint will be the same. If the object is not a perfect cylinder, areas where the cylinder has a smaller radius will produce a greater density of datapoints than areas of larger radius.

The digitized points are collected in the following manner. The system uses a low intensity laser to spot a point on the head then two mirrors are used to triangulate the associated radius distance from the center axis. The laser beam starts at the base of the neck and moves up to the top of the head defining a complete longitude, then increments about the head and continues. An entire scan can be completed in 15 seconds. This system type is installed and operational at the Human Engineering Division of the Armstrong Aerospace Medical Research Laboratory at Wright-Patterson Air Force Base. The dataset used throughout this thesis as examples was produced using this technology.

Shadow Moire Technology

This technology produces a topographical map of the surface, i.e. nonintersecting bands of constant depth. This type of pattern, in this case called a shadow moire pattern

is produced when a "master" ruling grid is placed close to the surface under examination, and is illuminated at an angle with a collimated light source. The opaque lines on the screen will project shadows on the surface and optical interference will occur between the shadow and the actual master ruling. This interference will produce the contour map of the surface.

The sensitivity of the method is determined by the pitch of the master ruling. Grids can range from 0.3 L/mm to 20 L/mm.

The contours or fringes can be evaluated using two different approaches, the full, and the fractional fringe techniques. The full fringe method assigns an integer value to the center of each of the either dark or white bands; this method requires many fringes to accurately evaluate the surface topography. The fractional fringe technique quantifies the area between the completely dark and light bands into shades of gray. The intensity of the gray can then be assigned a z value between the fringes.

This technology is being developed in the Fractional Fringe Laboratory at Lehigh University. A Charged-Coupled Device camera and a frame grabber digitize the field of view into a 512 x 512 pixel array with 8 bit resolution, or 256 shades of gray. In the optimum case, the field of view should contain only the area of interest.

For this method the grid will contain 512x512 datapoints, the resolution here depends on the actual size of the frame being examined. For more information on this technique refer to [1].

Contact Digitization

This method refers to the use of a Numerically Controlled (NC) type machine tool or

a Coordinate Measuring Machine (CMM) equipped with a contact probe to find the z depth of a rectangular array of points defined in a plane above the surface to be digitized. Since, for most types of machines it is impossible or impractical to fixture an actual face onto the table of a NC machine, a model of the face must be used. This model could be made by a plater casting technique, or deformable clay or plastic.

Here the model is attached to the machine table and the region of interest is defined by selecting the two opposing corners of a rectangle. The number of points, or the distance between points along each edge of the rectangle it then specified. The system projects this rectangle onto a plane that is clear of the surface, and moves the probe to each point in the data array. At each point the probe samples a depth z, then moves to the next data point. This method has the advantage that it can be done in such a way as not to require preprocessing. A disadvantage here is a amount of error is produced equal to the radius of the probe in a direction normal to the surface, this error must be accounted for.

Many machine tool venders produce machines with this capability. Bridgeport Machines' Series I 3-axis CNC machine with a BOSS 15+ control provides this functionality for example. The ME/MECH Department of Lehigh University has this type of machine and the software necessary, but as of the time of this thesis not the required probe hardware.

The grid resolution here is only limited to the locating tolerance of the machine. For the case of the machine described above, the machine is programmable to 0.0001" in the X,Y, and Z directions. The 3D distance between datapoints is then dependent on the slope of the facial topography.

Appendix B

A rational B-Spline surface in four-dimensional homogeneous coordinate space is given by:

$$Q(u,w) = \sum_{i=1}^{n+1} \sum_{j=1}^{m+1} B_{i,j}^{h} \; N_{i,k}(u) \; M_{j,l}(w) \qquad \text{Eq \# B-1}$$

where $B_{i,j}^{h}$ are the 4D homogeneous defining polygon vertices and $N_{i,k}(up)$, and $M_{j,l}(w)$ are the nonrational B-Spline basis functions:

$$N_{i,1} = \begin{cases} 1 \text{ if } x_i \le t < x_{i+1} \\ 0 \text{ otherwise} \end{cases}$$

and

$$N_{i,k}(t) = \frac{(t - x_i)\, N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t) N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}} \qquad \text{Eq \# B-2}$$

the x_i's correspond to knot vector values, these values define the vertex blending.

Expanding Eq #13 into three dimesional space:

$$Q(u,w) = \frac{\sum_{i=1}^{n+1} \sum_{j=1}^{m+1} h_{i,j} B_{i,j} \; N_{i,k}(u) \; M_{j,l}(w)}{\sum_{i=1}^{n+1} \sum_{j=1}^{m+1} h_{i,j} \; N_{i,k}(u) \; M_{j,l}(w)} \qquad \text{Eq \# B-3}$$

where $h_{i,j}$ is the homogeneous coordinate for each polyhedron vertex.

When $h_{i,j} = 1$ for all vertices, the rational B-Spline surface reduces to the nonrational formulation. If the number of defining polygon vertices is equal to the order in each parametric direction, and there are no duplicate interior knot values, the surface is a rational Bézier surface. The degree of a rational Bézier surface is equal to one less than the order.

The definition and influence of the knot vectors is not discussed in detail here. For a more detailed discussion of these, as well as other surface parameters and manipulation methods, refer to [3], [6].

The surfaces used in this thesis approximate datapoints. The order of a B-Spline surface can be increased, as well as the number of defining polyhedron points in order to force the surface to pass through the desired datapoints. The higher the degree of the surface the larger the amount of storage area required for the surface. More intensive calculations are also required to find surface information with higher order surfaces.

The simple rational Bézier formulation allow for very simple manipulation, subdivision and display, these can be found in the above mentioned references as well. Whenever possible, in this thesis, the rational Bézier formulation of degree 3, order 4 is used.

Appendix C

Environment

The system in its present form requires several UNIX environment options to be set before run time. The setup shown is for the HP Model 400T Personal VRX. Necessary environment variables include (values used are shown in parenthesis):

dad = directory for data files ($HOME/develop/common/data)
ind = directory for include files ($HOME/develop/common/include)
lid = directory for library files ($HOME/develop/common/lib)
obd = directory for object files ($HOME/develop/common/obj)
SB_OUTDRIVER = Starbase driver (hp98705)
SB_OUTDEV = Starbase ouput device file (/dev/screens/Grafix)
DRIVER = Starbase Driver for linking (98705)
SB_INDEV1..3 = Knob devices (/dev/hil3../dev/hil5)
SB_INDRIVER = Knob drivers (hp-hil)

The X11 window system must be running and an existing graphics window must be available for writing, the window name must correspond to that of the environment variable SB_OUTDEV. In the case here the command would be:

$ xwcreate -geometry 780x610+489+378 -depth 8 Grafix &

This command would be issued from an X11 window. The geometry switch defines the position and size of the window and the depth switch defines the number of graphics planes to use. The hardware contains sixteen planes and by specifying 8 the system will allow for double buffering.

Program execution

Now the environment is set for the data to surface and intersection program runs. For the data to surfaces routine, move to the proper directory, in this case $HOME/develop/research/fit and type:

$ fit

The program is menu driven and Figure C.1 shows the initial look of the system. An xclock and xload meter are also running for user information. The Figure shows the

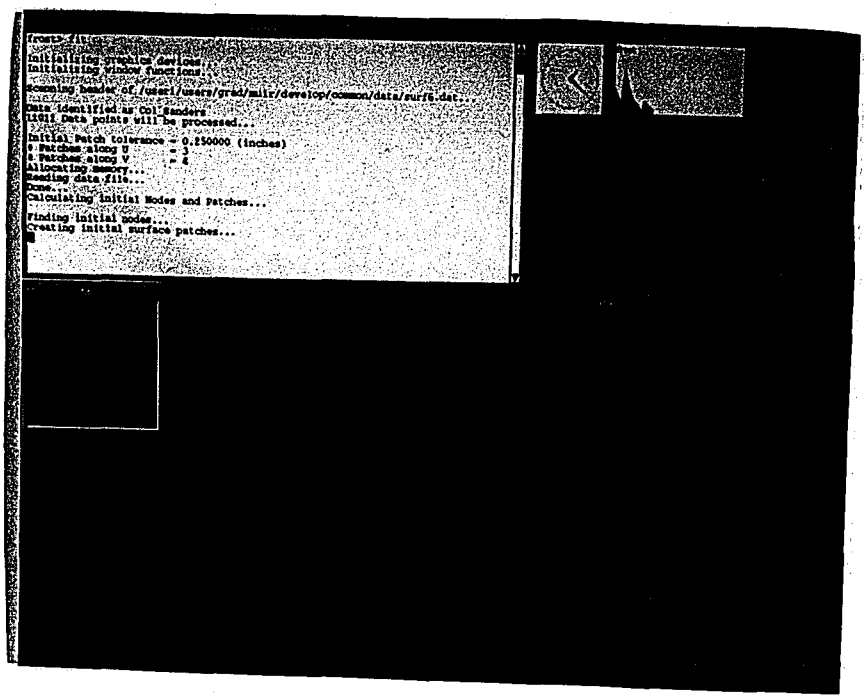


Figure C.1System Setup

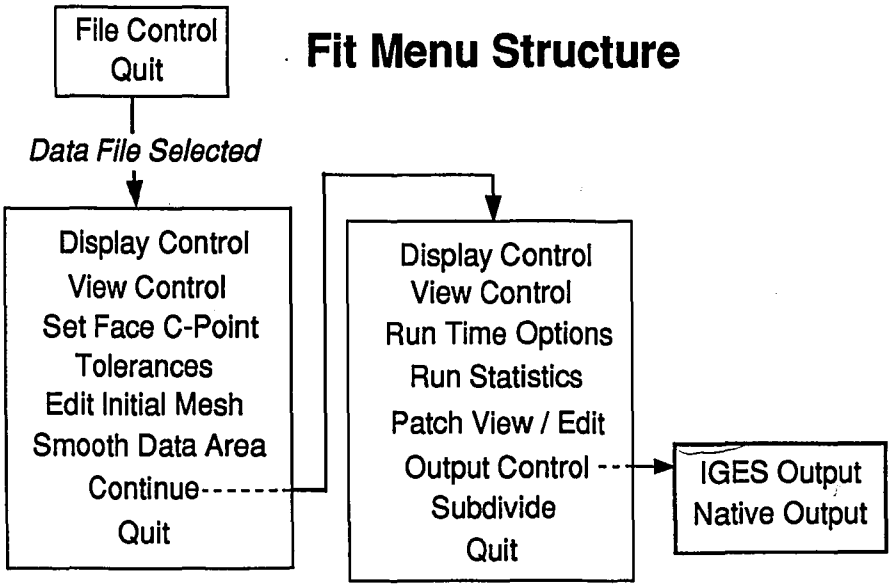


Figure C.2 Fit Menu Structure

graphics window, Grafix, the dialog window and the initial program identification banner. The program is mouse driven and Figure C.2 shows the complete menu structure. A detailed description of these functions has been shown in the Chapter 4 examples.

The intersection program is run by moving into the proper directory $HOME/develop/research/intersect and typing:

$ sect

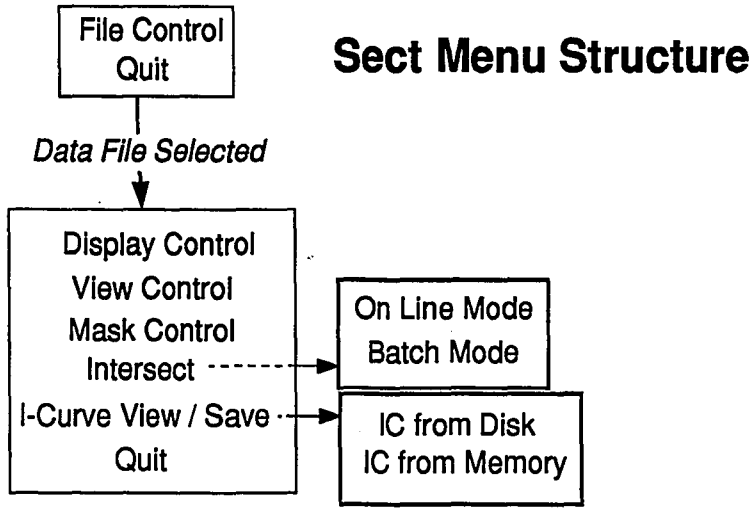


Figure C.3 Sect Menu Structure

As the program initializes itself it looks similar to the the data to surfaces program ,fit. This program is again completely mouse driven and the menu structure is shown in Figure C.3. As with the fit program the flow of operation has been addressed previously.

Preprocessing and Postprocessing

The preprocessing routine was written as a grip program and therefore must be executed while running Unigraphics. This must be done from the GRIP menu with an initialized part . The raw and formatted data files will be read and written to the default directory. The Cyberware preprocessor exists in the $HOME/develop/research/prep/cyber

directory and is named **prep**. The prep program is also mouse driven and self guiding.

The postprocessor is a stand-alone program with no graphics calls, therefore it can be run outside of windows. This program **postp** resides in the directory $HOME/develop/research/post/bridge. This program will read and write to the directory pointed to by the **dad** environment variable.

References

[1] LVCS Inc., "Automated Custom-Fit Production", Defense Small Buisness Innovation Research (SBIR) Proposal (Phase II), Topic Number AF89-076C, May (1990)

[2] LVCS Inc.,"Automated Custom-Fit Production", Defense Small Buisness Innovation Research (SBIR) Proposal (Phase I), Topic Number AF89-076C, January (1989)

[3] Rogers, David F., and Adams, Alan J., Mathematical Elements for Computer Graphics, New York: McGraw-Hill Publishing Company, 1990.

[4] Schmitt, Francis J. M., and Barsky, Brian A., and Du, Wen-Hui, "An Adaptive Subdivision Method for Surface-Fitting from Sampled Data", SIGGRAPH 86 Procedings, pp 179-188, 1986.

[5] Barnhill, R.E., and Farin, G., and Jordan, M., and Piper, B.R., "Surface / surface intersection", Computer Aided Geometric Design, January (1987) , pp 3-16

[6] Smith, Bradford, and Rinaudot, Gaylen R., and Reed, Kent A., and Wright, Thomas, "Initial Graphics Exchange Specification (IGES) Version 4.0", U.S. Department of Commerce, June (1988).

[7] Faux, I.D., and Pratt, M.,J., Computational Geometry for Design and Manufacture, New York: Ellis Horwood Publishing Company, 1985.

Vita

Christopher Myles Muir was born in Mt. Holly NJ, was graduated from Holy Cross High School, Delran NJ in 1988, and received a Bachelor of Science degree in Mechanical Engineering from Lehigh University in Bethlehem PA in January 1989. From January 1989 to January 1990 he was employed as an engineer at Lehigh Valley Computer Services (LVCS). At LVCS he worked on projects ranging from mold design for thermoforming and injection molding using CAD/CAM techniques, training, to strait FORTRAN, C and NC programming for specific applications. He has been enrolled in the ME/MECH graduate program at Lehigh since January of 1990.

END OF TITLE