Digital Reconstructions of Fossil Morphologies, Nama Group, Namibia

by

Wesley Andrés Watters

Submitted to the Department of Earth, Atmospheric, and Planetary Sciences in partial fulfillment of the requirements for the degree of

Master of Science in Earth & Planetary Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

~~January 2000~~
[February 2000]

Author ....
Department of Earth, Atmospheric, and Planetary Sciences
January 18, 2000

Certified by ....
Prof. John P. Grotzinger
Professor of Geology
Thesis Supervisor

Accepted by ....
Prof. Ronald Prinn
Chairman, Department of EAPS

Digital Reconstructions of Fossil Morphologies, Nama Group, Namibia

by

Wesley Andrés Watters

Submitted to the Department of Earth, Atmospheric, and Planetary Sciences
on January 18, 2000, in partial fulfillment of the
requirements for the degree of
Master of Science in Earth & Planetary Science

Abstract

Previously undescribed fossils of weakly calcified metazoans were recently discovered in the terminal Proterozoic Nama Group of central and southern Namibia (Grotzinger et al., 1995), in sediments that contain the terminal Proterozoic index fossil *Cloudina.* The new fossils are closely associated with thrombolites and stromatolites that form laterally continuous biostromes, isolated patch reefs, and isolated pinnacle reefs. Because these fossils are preserved as calcitic void-fill in a calcite matrix, individual specimens cannot be freed by conventional techniques. Rocks containing the fossils are ground and digitally photographed at thickness intervals of 25 μm. A battery of image processing techniques is used to obtain the contour outlines of the fossils in serial cross sections. A Delaunay triangulation method is then used to reconstruct the morphology from tetrahedral components which connect the contours in adjacent layers. It is found that most of the fossils resemble a single morphology with some well-defined characters that vary slightly among individual specimens. This fossil morphology is described in this thesis as *Namacalathus hermanastes.* A mathematical description of the morphology is used to obtain a database of randomly-oriented synthetic cross sections. This database reproduces the vast majority of cross sections observed in outcrop. In addition, the most common orientation, the mean size, and other population statistics are measured for *Namacalathus* fossils within an individual rock sample.

Thesis Supervisor: Prof. John P. Grotzinger
Title: Professor of Geology

Contents

Acknowledgements 4

Introduction 5

Chapter 1 Calcified Metazoans of the Nama Group 7

Chapter 2 The Serial Imaging Method 33

Chapter 3 A Mathematical Description of the Morphology 55

References 69

Appendix A Population Statistics 75

Appendix B Database of Synthetic Cross Sections 79

Appendix C Documentation and Source Code 87

Acknowledgements

I am profoundly grateful to John P. Grotzinger for his generosity, for his support and guidance, and most of all for his friendship. I have learned and I have grown in profound ways because of my interactions with John, because of my work on this project, and because of my visit to Namibia in 1997. I am also greatly indebted to my parents, Allan and Cecilia Farfán Watters, for their encouragement and support. Special thanks go to Rob and Marianne Field and Roy Magson for access to their farms in central Namibia, Zebra River and Donkergange, and for their hospitality. I wish to thank Roger Swart and NAMCOR for providing the Landsat TM image in Figure 1-3A, and the Geological Survey of Namibia for providing a field vehicle and logistical support. This research is supported by NSF grant EAR-9628257 and the NASA Astrobiology Institute.

Introduction

Siliciclastic successions of terminal Proterozoic age contain moderately diverse, commonly problematic fossils of soft-bodied organisms, collectively known as the Ediacaran biota. Many rocks that host Ediacaran assemblages also contain a limited diversity of metazoan trace fossils. In contrast—and in strong contrast to Cambrian and younger successions—terminal Proterozoic limestones have been thought to yield only limited evidence of animal life. A major exception to this pattern is provided by *Cloudina*, a calcified fossil known almost exclusively from uppermost Proterozoic carbonate rocks. Until now it has been unclear whether *Cloudina* should be regarded as the exception that proves the rule or as a hint that more diverse animals inhabited carbonate platforms before the dawn of the Cambrian.

This thesis presents evidence for a paleoecologically distinctive assemblage of calcified metazoans in thrombolite-stromatolite reefs and associated facies of the terminal Proterozoic Nama Group, Namibia, where *Cloudina* was originally discovered. Particularly abundant are cm-scale goblet-shaped fossils described in this thesis as *Namacalathus hermanastes* gen. et sp. nov. and interpreted as lightly mineralized, attached benthos comparable to simple cnidarians in body plan. These fossils are characterized by a slender stem open at both ends, attached to a broadly spheroidal cup marked by a circular opening with a downturned lip and six (or seven) side holes interpreted as diagenetic reflections of underlying biological structure. *Namacalathus* lived atop the rough topography created by ecologically complex microbial-algal carpets; they appear to have been sessile benthos attached either to the biohermal substrate or to seaweeds that grew on the reef surface. The phylogenetic affinities of *Namacalathus* are uncertain, although preserved morphology is consistent with a cnidarian-like body plan. In general aspect, these fossils resemble some of the unmineralized taxa found in contemporaneous sandstones and shales, but do not appear to be closely related to the well-skeletonized bilaterian animals that radiated in younger oceans.

The three-dimensional morphology of the reef-associated fossils was reconstructed by computer. Analog methods for reconstructing the morphology of stromatolites and fossil invertebrates are widely used (e.g., Vanyo and Awramik 1982), while digital reconstruction techniques are still uncommon. For one example, Schmidtling (1995) presented a digital reconstruction of the internal morphology of blastoids. The reconstruction techniques described in this study were developed independently of these methods.

The reconstructions in the present study are based upon digital images of sections taken at 25 μm intervals through numerous specimens. This process involves a number of image processing techniques, used for obtaining the contour outlines of fossils in cross-sectional images. The resulting "tomographic" models are then used to construct a mathematical description of the morphology, which is specified using a pair of two-dimensional vector-valued functions of a vertical position parameter. Several characters that are observed to vary between individual tomographic specimens are assigned parameters in the mathematical model. In this way it is possible to construct "idealizations" of individual tomographic reconstructions. Two mathematical models are used to generate a database of synthetic cross sections that is nearly comprehensive. This database contains the vast majority of cross sections observed in outcrop, and can be used to identify assemblages of *Namacalathus* in the field. Moreover, this database is used to identify *Namacalathus* specimens in the stacks of cross-sectional images, in order to collect statistics about the percentage of fossils by type, and also regarding the orientation, size, and shape of *Namacalathus* specimens from a single rock sample.

This thesis is organized as follows. Chapter 1 provides a description of the geological and paleoecological context in which the Nama specimens are found. This chapter also presents the tomographic and mathematical models of the *Namacalathus* morphology, as well a systematic description. Chapter 2 presents the methods used to obtain tomographic reconstructions of individual fossils, and Chapter 3 contains a detailed discussion of how the mathematical model was developed. Appendix A contains a discussion of statistics regarding the types of fossils found in typical rocks, as well as their distribution in size, shape, and orientation. Appendix B contains the aforementioned database of synthetic cross sections. Finally, Appendix C contains the documentation and source code for programs developed and used to obtain both kinds of models.

Chapter 1

Calcified Metazoans of the Nama Group

Geological Setting of Nama Reefs[1]

The geology of southern Namibia, including the distribution of the Nama Group, is shown in Figure 1-1. The Nama Group has been interpreted as a foreland basin fill (Germs 1983; Gresse and Germs 1993) related to convergence along the western and northern margins of the Kalahari craton and overthrusting in the Gariep and Damara orogens (Miller 1983). The general stratigraphy of the Nama Group (Figure 1-2) was outlined by Martin (1965) and developed in a series of papers by Germs (1972, 1974, 1983). Regional isopachs and facies distributions define two subbasins; the Witputs subbasin, located in southern Namibia, thickens toward the Gariep orogenic belt, while the more northerly Zaris subbasin thickens northward toward the Damara orogenic belt. The Osis Arch represents a site of depositional thinning of all Nama units that separates the two subbasins (Figure 1-1).

In general, the Nama Group consists of a number of marine shelf siliciclastic and carbonate sequences (Kuibis and Schwarzrand Subgroups) overlain by alluvial to shallow marine molasse (Fish River Subgroup) that documents unroofing of the Damara/Gariep hinterlands. Near the subbasin axes, thicknesses are on the order of 2-3 km, thinning to less than 1 km farther onto the craton toward the Osis Arch (Germs 1974; Germs 1983). Thrombolite-stromatolite reefs are well developed in the Kuibis Subgroup of the northern, Zaris subbasin, and in the Huns platform of the southern, Witputs subbasin (Figure 1-2).

Geochronologic constraints (Grotzinger et al. 1995) are provided by U-Pb zircon ages on several

[1]This chapter is adapted from Grotzinger, J.P., Watters, W.A., and Knoll, A.H., Calcified Metazoans in Thrombolite-Stomatolite Reefs of the Terminal Proterozoic Nama Group, Namibia. Submitted to *Paleobiology*, Dec 1999.

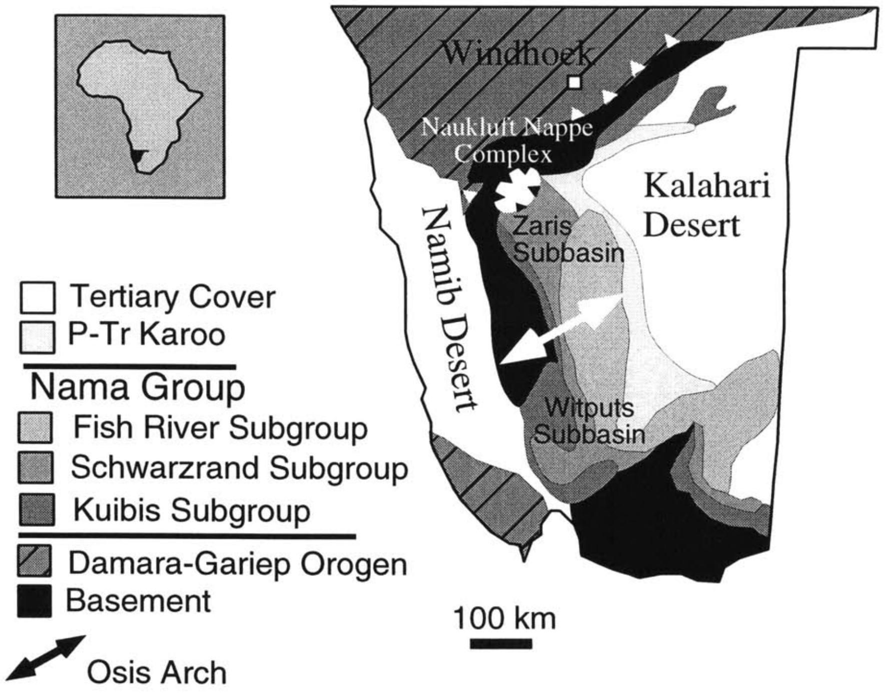

Figure 1-1: Geological map of southern Namibia, showing the distribution of major sedimentary and tectonic elements, including the Nama Group, discussed in the text.

units within the Nama Group (Figure 1-2). An ash bed within the northern Nama basin yields an age of 548.8+/-1 Ma for the middle Kuibis Subgroup (Grotzinger et al. 1995). In southern exposures of the Nama, the overlying Schwarzrand Subgroup contains ash beds that yield, in ascending order, ages of 545.1 Ma, 543.3 Ma and 539.4 Ma (all +/- 1 Ma; Grotzinger et al. 1995). The Precambrian-Cambrian boundary in Namibia is bracketed by the 543.3 Ma and 539.4 Ma ages, although this also includes a significant unconformity; on a global basis the Precambrian-Cambrian boundary is currently regarded to be on the order of 543 Ma.

Zaris Subbasin – Kuibis Platform. The basal unit of the Nama Group, the Kuibis Subgroup is regionally widespread, consisting of a thin, basal, transgressive sandstone that grades upward into a carbonate platform 150 to 500 meters thick (Figures 1-2, 1-3A). In the Zaris subbasin, Kuibis rocks form a well-developed carbonate ramp that thickens from the Osis arch northward to the Naukluft Mountains (Germs 1983; Saylor et al. 1998; Smith 1998). This change in thickness is accompanied by a gradation from shallow water facies in the south to deeper-water, basinal facies near the Naukluft Mountains (Germs 1983). Germs (1972b, 1974, 1983) subdivided the Kuibis platform into two members (Omkyk and Hoogland; Figure 1-2) that correspond approximately to stratigraphic sequences (Grotzinger et al. 1995; Saylor et al. 1995, 1998; Smith 1998). The surfaces defining the member boundaries, however, represent flooding events rather than sequence boundaries

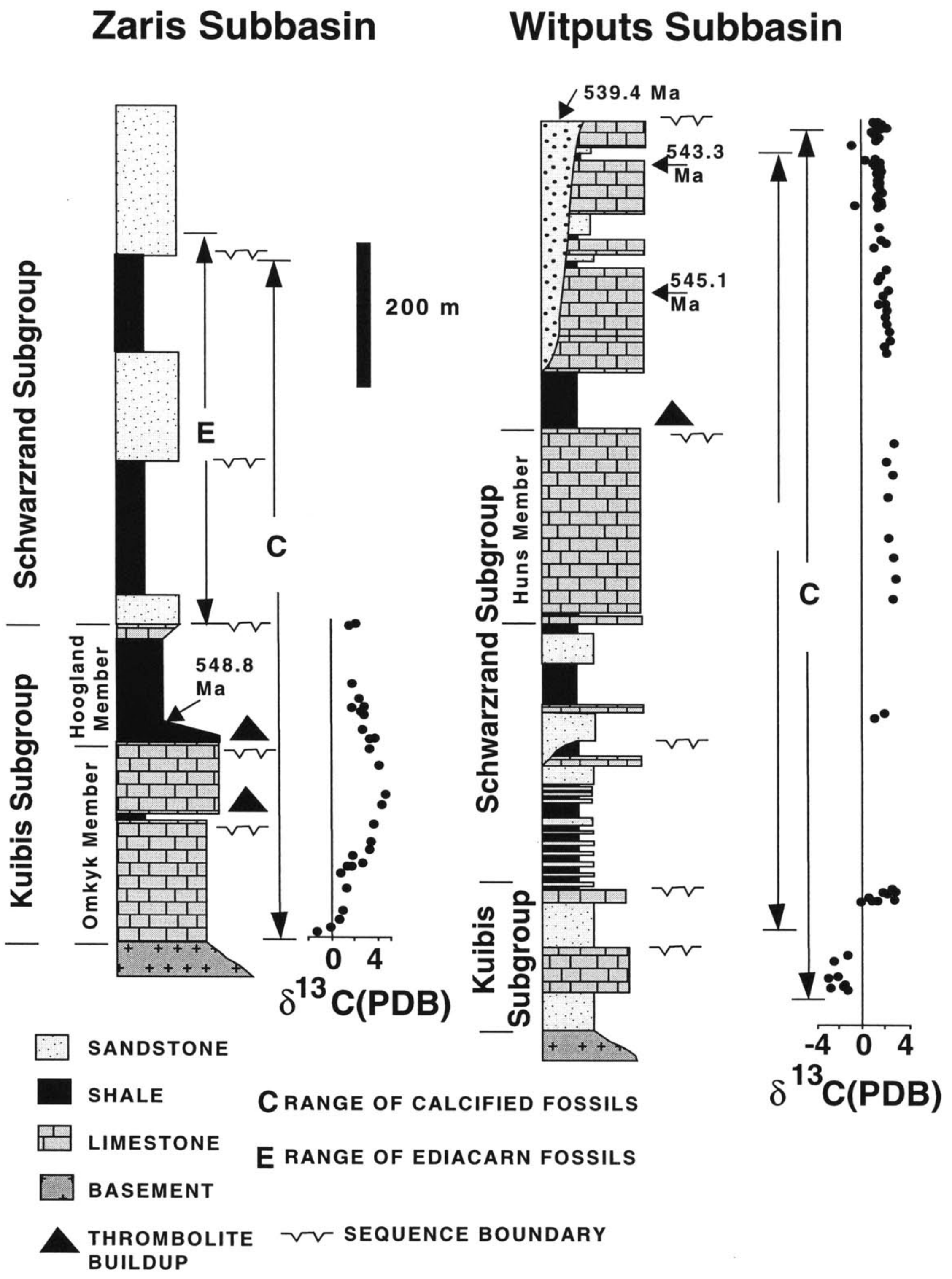

Figure 1-2: Generalized stratigraphy of the Nama Group for the Zaris (north) and Witputs (south) subbasins. Showing major lithostratigraphic, chemostratigraphic, biostratigraphic, and sequence stratigraphic attributes. Note the positions of thrombolite reef complexes above sequence boundaries.

(Smith 1998). Detailed mapping of these sequences (Smith 1998) shows that the reefs described here, including the Driedoornvlagte bioherm identified by Germs (1972a, 1974) all nucleated within the Transgressive Systems to early Highstand Systems Tract (TST to HST) at the base of their respective sequences (Figure 1-2). This implies that the most favorable conditions for reef growth occurred during times of increased accommodation and lowered sediment flux on the platform.

In updip sections, biostromes form continuous sheets (within the Zebra River farm) which break up into patch-reef bioherms toward and within the Donkergange farm (Figures 1-3B, C). Further downdip, large pinnacle reefs are developed at Driedoornvlagte in a position of maximum accommodation within the depositional profile (Figure 1-3A).

Witputs Subbasin - Huns Platform. The Huns Member consists of a thick section (up to 500 meters) of platform carbonates in the middle of the Nama Group, within the Witputs subbasin of southern Namibia (Figure 1-2). Germs (1972b, 1974, 1983) first recognized pinnacle reefs on the Swartkloofberg farm; subsequent work (Grotzinger et al. 1995; Saylor and Grotzinger 1996; Saylor et al. 1998) has shown that these reefs are associated with drowning of the platform. Fifty to 70 meters wide at their base and up to 50 meters high (Figure 1-4), Huns reefs are blanketed by shales deposited at or below wave base.

Facies and Diagenesis

The thrombolite biostrome facies consists of massive thrombolites, stratiform thrombolites and stromatolitic thrombolite-forming columns, branching columns, heads and domes of decimeter-scale dimensions and relief (Figure 1-5A). In general, the cores of domes and columns are characterized by a thrombolitic texture, whereas the margins become progressively more stromatolitic in nature, in the sense that they exhibit crude lamination (Figure 1-5A; equivalent to the stromatolitic thrombolites of Kennard and James, 1986). In the Kuibis reefs, columns are consistently elongated with an azimuth of $270^{\circ} - 310^{\circ}$. Calcified macrofossils occur within thrombolite domes and columns, and the intrachannel fill between domes and columns consists of trough cross-bedded, fossiliferous packstone and grainstone containing simple tubes, more complex cups and goblets, and their bioclastic detritus (Figure 1-5A). Figure 1-6 shows the close relationship between fossil content and thrombolitic texture.

The thrombolite biostrome facies characteristically developed as broad, laterally continuous reef complexes that became discontinuous down depositional dip to form isolated bioherms and pinnacle reefs. Energy conditions were high, as demonstrated by the trough cross-bedded intrachannel grainstone, the elongation of the stromatolitic thrombolite columns, and the association of this facies with the grainstone facies. The consistent orientation of the thrombolite elongation is comparable to that of other platforms where wave action is inferred to have been responsible for the elongation

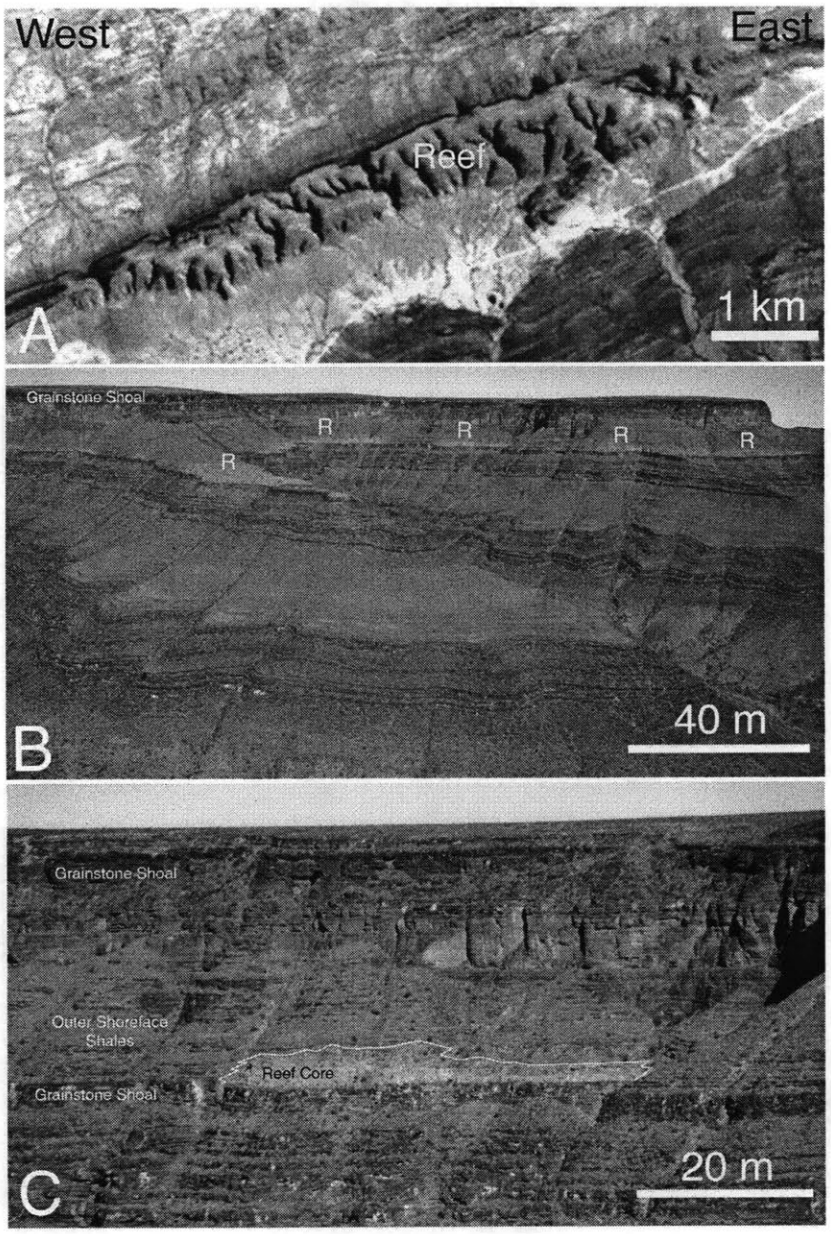

Figure 1-3: Reefs of the Zaris subbasin. (A) Landsat TM image showing pinnacle reef (Driedoornvlagte bioherm) on the Driedoornvlagte farm, Zaris subbasin. The reef is developed on platform carbonates (dark blue) and is overlain by deep-water shales (orange-red). The platform carbonates unconformably overlie much older quartzites (mauve-gray colors at top of photograph). Color in the TM image indicates composition—the reef has a gray-lavender color that reflects high dolomite content, whereas the dark blue color of shallow-water platform carbonates reflects dominance of limestone. Rocks have a structural dip of about 40° toward the south. (B) Laterally continuous biostrome at the top of the Omkyk Member within Donkergange passes laterally into discrete patch reefs, each indicated by an "R." Reefs are overlain by deepwater shales, which in turn pass upward into a grainstone shoal complex. (C) Close-up of a representative patch reef showing stratigraphic position immediately above a sequence boundary, which separates reef facies from underlying thick-bedded grainstone facies. The reef passes laterally into relatively thin-bedded shales and fine calcarenites. Bedding thickness decreases immediately above the sequence boundary, indicating a regime of increasing accommodation.

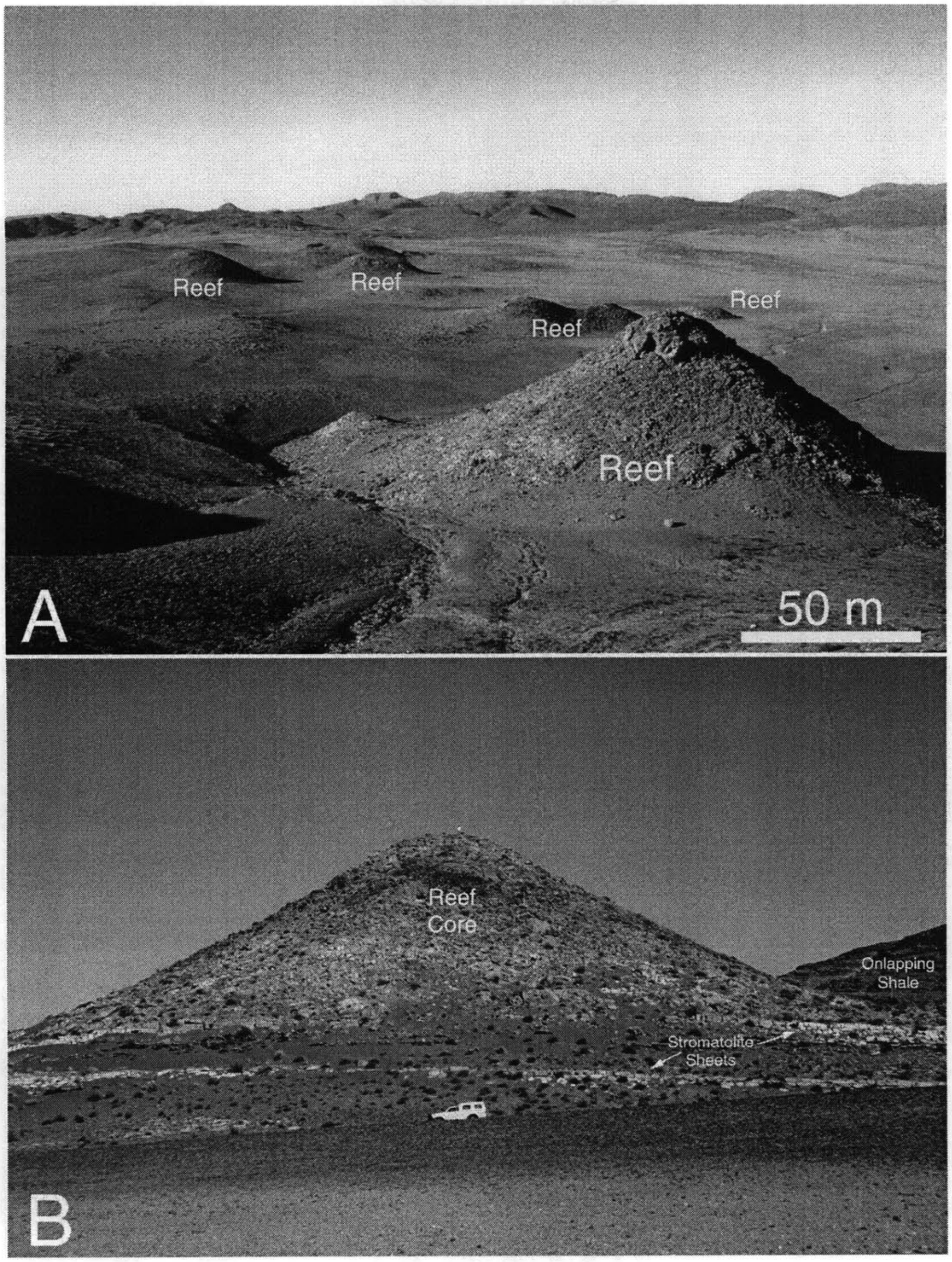

Figure 1-4: Thrombolite-stromatolite pinnacle reefs at top of Huns platform, Witputs subbasin, southern Namibia. Reefs are formed of a core of coalesced thrombolite mounds, overgrown by a stromatolitic mantle. (A) Overview of multiple pinnacle reefs formed at top of Huns platform. Reefs have been exhumed by erosion of overlying shales. (B) Cross section of representative pinnacle reef showing its foundation of stromatolite sheet facies and remnants of overlying shales which have been eroded to expose reefs. A sequence boundary separates the stromatolite sheet- and pinnacle reef facies; thus, the pinnacle reefs formed during a time of increasing accommodation space within a Transgressive System Tract.

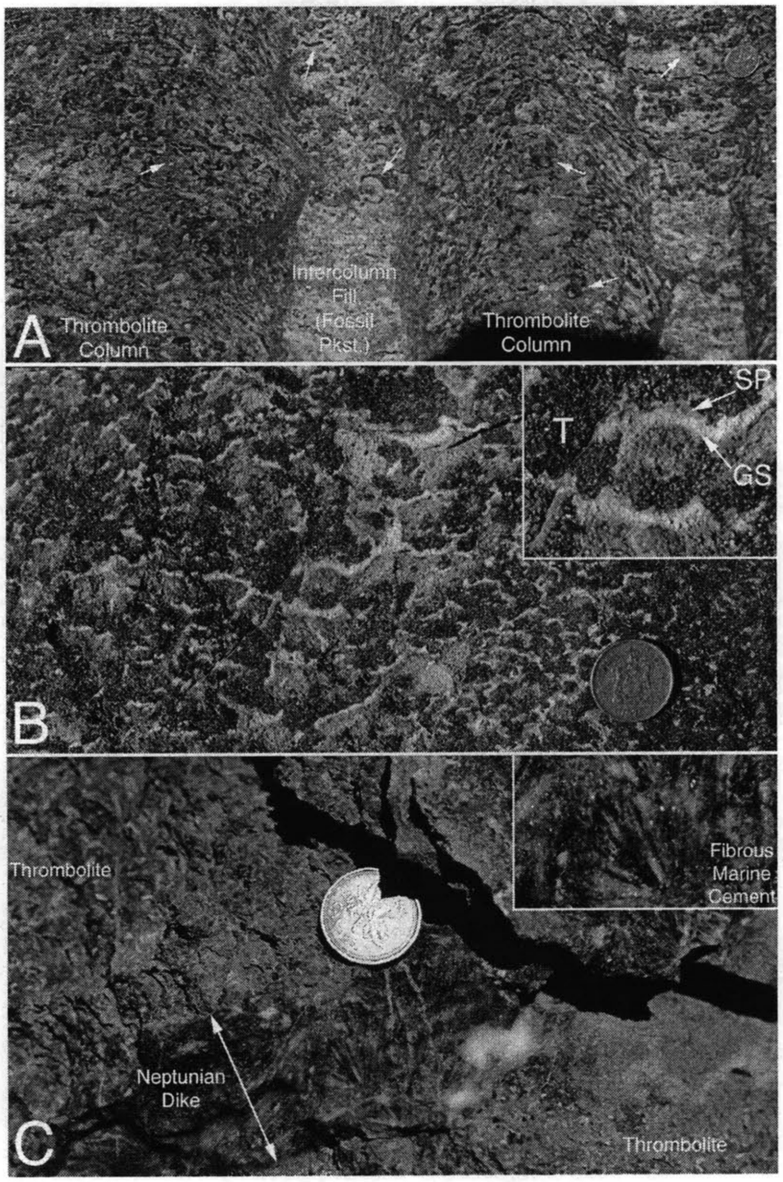

Figure 1-5: Representative photographs of microbialites and associated textures. (A) Stromatolitic thrombolite columns forming a laterally continuous biostrome at the top of the Omkyk Member, Zaris subbasin. The core of each column has a predominantly thrombolitic texture, whereas margins have better-laminated stromatolitic thrombolite texture. Columns are strongly elongate normal to the trend of the Omkyk carbonate ramp, indicating persistent high-velocity, wave-generated flows. Arrows point to goblet- and tube-shaped fossils that are abundant within the intercolumn fill sequences, but also occur within the thrombolitic texture. (B) Well-preserved thrombolite texture in a pinnacle reef at the top of the Huns platform, Witputs subbasin. Reefal framework is created by dark-colored mesoclots; these create shelter pores that are partially filled by light-colored geopetal sediment and are ultimately filled by fibrous marine and later blocky burial cements. Inset shows fabric details; T, thrombolite mesoclot; GS, geopetal sediment; SP, shelter porosity filled with cement. Coin is approximately 1.2 cm in diameter. Inset shows enlargement of region near center of photograph. (C) Neptunian dike transecting thrombolite mound, Driedoornvlagte bioherm, Zaris subbasin. Dike is infilled with botryoidal fibrous calcite cement, interpreted to have replaced primary aragonite. Cements fill fractures within the reef, providing evidence for early lithification of the reef. Coin is approximately 1 cm in diameter. Inset shows enlargement of region below coin.

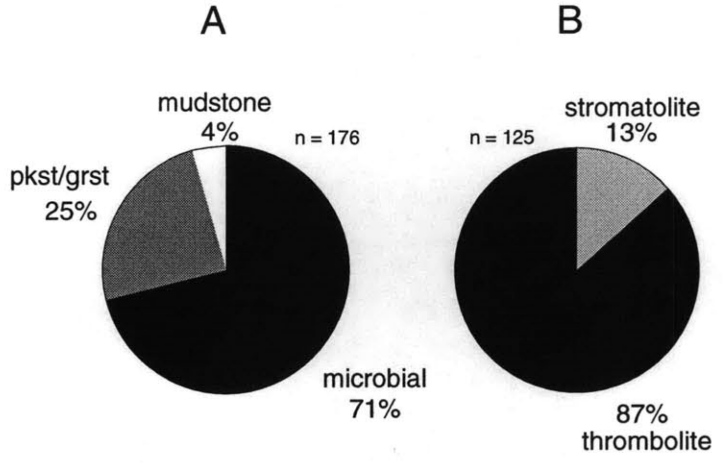

Figure 1-6: Proportional distribution of calcified fossils within broadly-defined lithologic facies of the Nama Group (A) and within microbial facies of the Nama Group (B). Results show that fossils are very strongly correlated with microbial facies in general, and with thrombolitic facies in particular. Fossils are less abundant in packstones and grainstones, which are composed principally of intraclasts and peloids, and are rare in mudstones. The simplest interpretation of this distribution is that the fossil organisms were benthic, with a strong preference for microbially colonized substrates. In addition, the preference for thrombolitic over stromatolitic substrates suggests further paleoecologic control on their distribution. See text for discussion.

of stromatolites (Hoffman 1969; Cecile and Campbell 1978; Grotzinger 1986; Bartley et al. 1998), analogous to the development of "spur and groove" structure in modern coral reefs. This interpretation supports other evidence that the Kuibis ramp was a wave- and storm-dominated system, where the dominant currents were induced by strong wave-generated flows.

The internal texture of the thrombolite reefs is characterized by mesoclots that range from a few millimeters to a few centimeters in diameter (Figures 1-5A and B, 1-7). Nama mesoclots are restricted to simple ovoid, globose, and colliform morphologies and do not include the complex digitate or dendriform fabrics found in younger thrombolites (e.g. Kennard and James 1986; Riding et al. 1995). Nama mesoclots formed an open framework that resulted in development of abundant cavities (Figures 1-5B, 1-7). Mesoclots typically were overgrown by a synsedimentary crust of fibrous marine cement that now consists of calcite but is interpreted to have replaced original aragonite (Figure 1-7). Marine cement crusts were subsequently covered by geopetal sediment, which partially filled framework pores (Figures 1-5B, 1-7). Remaining porosity was occluded by a dolomite rim cement, followed by infilling with blocky calcite spar (Figure 1-7). Recrystallization commonly makes interpretation of growth processes difficult, but sporadically distributed patches with relatively good fabric preservation show evidence that coccoidal microorganisms, expressed as poorly preserved spheroids in mesoclot cores, contributed to reef accretion (see Figure 1-7).

Early lithification of thrombolites and stromatolites in Nama pinnacle reefs resulted in the development of penetrative networks of neptunian dikes (Figure 1-5C). These are best expressed in the

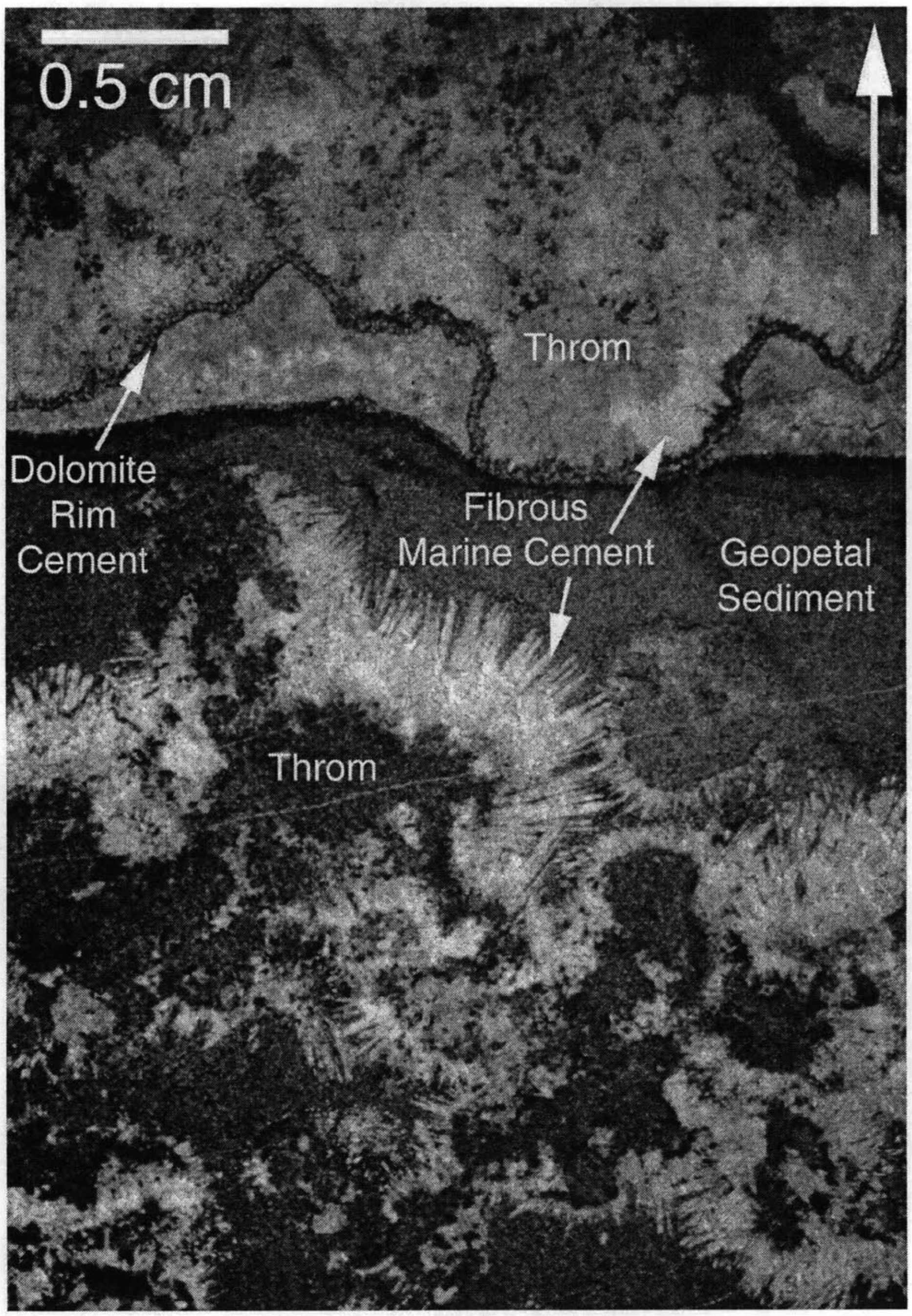

Figure 1-7: Thin section photomicrograph of thrombolite fabrics within a pinnacle reef at top of Huns carbonate platform, Witputs subbasin. Thrombolite mesoclots (Throm) are encrusted with a fringe of fibrous calcite (Fibrous Marine Cement) interpreted to have replaced original aragonite (note square crystal terminations). This was followed by deposition of a layer of crystal silt (Geopetal Sediment). Occlusion of remaining shelter porosity began with precipitation of an isopachous fringe of Dolomite Rim Cement, followed by a final infill of blocky calcite spar. Note that mesoclots are composed of distinct aggregates of smaller clots. These smallest fabric elements most likely are formed by the early cementation of coccoid microorganisms. Note the accumulation of Geopetal Sediment (crystal silt) which smoothes out the rugged relief created by clots and encrusting Fibrous Marine Cements.

Huns reefs and Driedoornvlagte bioherm, where fractures that cut sharply across reefs are filled with fibrous marine cements, interpreted as calcite-replaced aragonite (Figure 1-5C). Neptunian dikes are formed only in the larger reefs, where gravitationally-induced failure of the reef was likely. A related effect includes failure of the sides of the pinnacle reefs, which resulted in down slope transport of slide breccias. These breccias are composed of angular blocks up to 1 meter in diameter with thrombolite and stromatolite textures. Voids between breccia blocks are filled with massive lime or dolomite mudstones, or fibrous marine cements.

Calcified Metazoans of the Nama Group

Within thombolitic facies, fossils occur abundantly both in the domal and columnar structures that make up individual reefs and in trough cross-bedded intrachannel fill. The fossils—millimeter to centimeter-scale calcareous cups, goblets, and tubes—are particularly abundant in thrombolites and thrombolitic stromatolites, evidently preferring the firm substrate and bathymetric elevation that these structures offered (Figure 1-6)). Intrachannel fill consists of skeletal packstone and grainstone containing goblets, cups, cylindrical tubes, *Cloudina*, and their bioclastic detritus.

Morphology. Figure 1-8 shows a number of calcified goblet-shaped fossils. Two dimensional slices provided by outcrops, polished slabs, and thin sections reveal a moderate degree of assemblage variability, ranging from clearly defined goblet-shaped forms (Figure 1-8E), originally illustrated by Grotzinger et al. (1995), to rounded cups without "stems," to bulbous structures showing apparent hexagonal symmetry in cross section. Exposed specimens reflect some combination of intraspecific, interspecific, and taphonomic variation, but the relative contributions of these factors cannot be evaluated in the absence of information about the three-dimensional morphology of constituent fossils. Because the fossils are preserved as calcitic void-fill in a calcite matrix, individual specimens cannot be freed by conventional techniques—posing a challenge for morphological and, hence, systematic interpretation. Our response has been to develop a computer-based, "tomographic" reconstruction technique to determine their true three-dimensional geometry. This technique is described in detail in Chapter 2. In brief, appropriate samples were selected, the position of each sample and its contained fossils calibrated with respect to an external reference system, 25 microns ground from the polished sample surface, and the ground surface photographed using a digital camera. In the present study, the last two steps were iterated several hundred times for samples taken from reefs at Vioolsdrif (Witputs subbasin) and Donkergange (Zaris subbasin). Contours of the fossils in each cross-sectional image were obtained using a battery of image processing techniques, including a binary thresholding procedure. Ultimately, the three-dimensional surface of our models is determined by the shape of these contours, placed adjacent to one another in the third dimension. In the interpolation process, segments in a given contour layer are connected to the vertices in an adjacent layer, ultimately allowing the complete surface to be described by a set of tetrahedrons. Multiple views of one complete and eight partial tomographic reconstructions are shown in Figure 1-9.

All complete or nearly complete reconstructed specimens have a stem and an outward flaring cup. The cup, a few millimeters to 2.5 cm in maximum dimension, has a broad circular opening at the top with a lip that curves inward. The cup is perforated by six or seven holes of similar size and shape, and has a matching number of regularly arranged side walls. Cups taper to a shallow base from which a hollow cylindrical stem extends. Some specimens contain an additional hole near the cup base. Stems are commonly longer than maximum cup dimension and are open at both ends.

In constructing a model morphology from reconstructed individuals, we took note of variation in the following parameters: (1) the radial profile of the cup, (2) the aspect ratio of the cup (i.e., maximum diameter to total height), (3) the size and shape of the cup's inner lip, (4) the relative size of the cup's circular opening, (5) the curvature of cup walls (wall curvature varies with height, in some cases flattening midway up the cup wall and in others remaining rounded), (6) the size, shape, and position of holes (e.g., also called "windows," which are normally oval or circular, located at the top, middle or base of the cup, and with or without an inward-turning lip), (7) the thickness of walls, (8) the number of sides and holes (i.e., six or seven), (9) the length of the stem, (10) the size of the cup, and (11) the trajectory and radial profile of the stem.

In developing a mathematical model for the morphology of these fossils, only the features listed above were assumed. Individual tomographic reconstructions were used to obtain values for all 11 parameters, enabling us to generate individual models. Figures 1-10A and 1-10C show sample cross sections of tomographic reconstructions that were used to measure profiles of the cup's radius and inward lip, the cup's aspect ratio, and the number of holes and side wall sections. The corresponding mathematical models are shown in Figures 1-10B and 1-10D. The specimen shown in Figures 1-10A and 1-10B was collected from a locality in the Zaris subbasin; the specimen in 1-10C and 1-10D is from the Witputs subbasin. The two models are primarily distinguished by (1) the number of holes (6 in the case of 1-10A-B, 7 in the case of 1-10C-D), (2) the radial profile of the cup (reaching maximum diameter near the top of the cup for 1-10C-D and near the middle in the case of 1-10A-B), (3) the wall curvature (nearly flat at the cup's mid-height in 1-10C-D while still slightly round in 1-10A-B), and (4) the size of the inward-turning lip that lines the cup's large circular opening. A complete description of the mathematical model is supplied in Chapter 3.

A database of randomly-oriented synthetic cross sections was obtained from the models depicted in Figure 1-10, and is shown in Appendix B. These synthetic cross sections reproduce the vast majority of shapes observed in rocks, a subset of which is depicted in Figure 1-11. Because we can account for most observed variation by a combination of random sections through the morphologies reconstructed in Figure 1-10 and simple taphonomic alterations (e.g., physical distortion) observed in our samples, we interpret most of the variation seen in outcrop as representing a single species

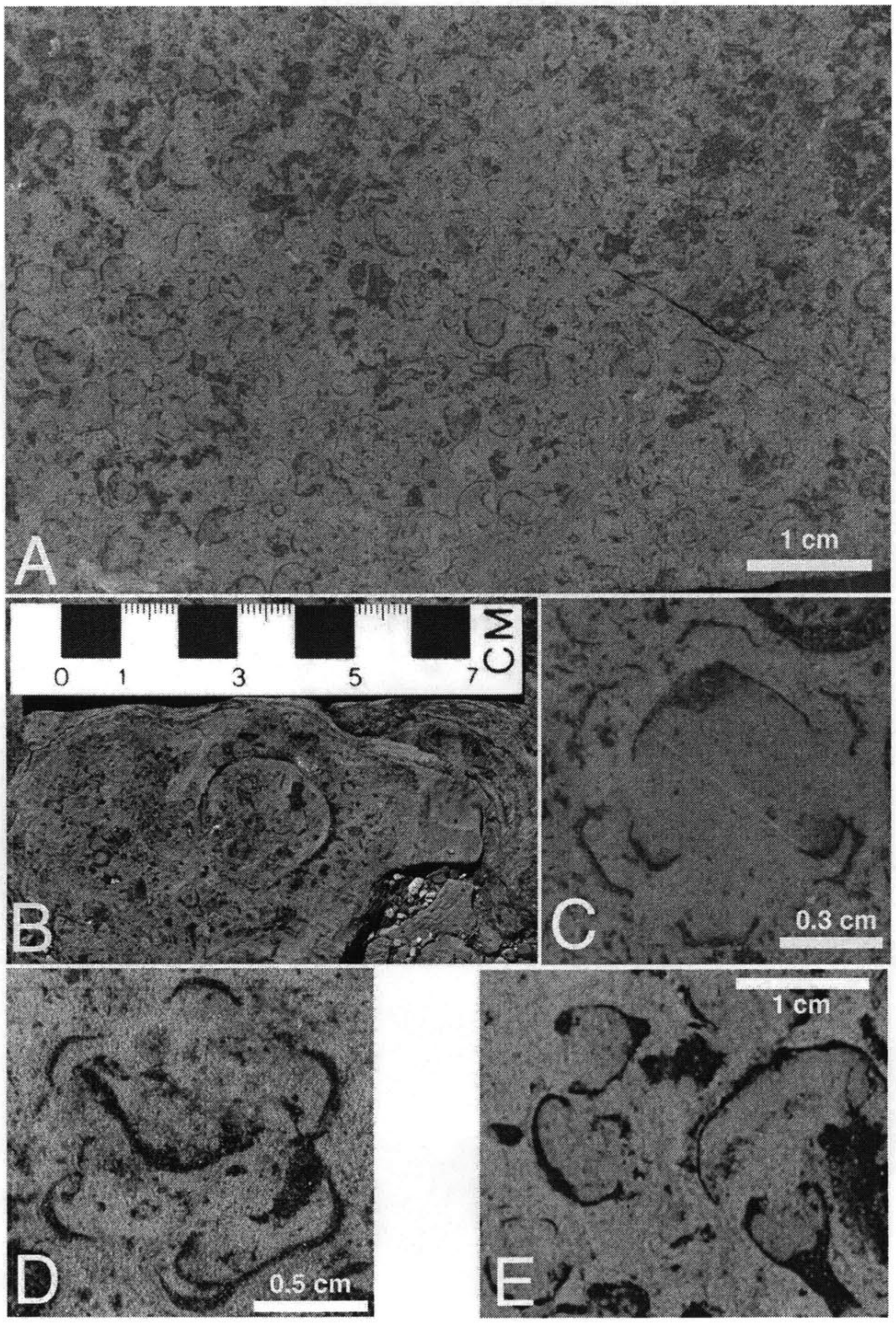

Figure 1-8: Assemblage of calcified *Namacalathus* fossils within reefal and related thrombolitic horizons within the Nama Group. (A) *Namacalathus* packstone illustrating a range of cross sectional morphologies. (B) Single large *Namacalathus*, over 2 cm in diameter. (C) Cross sections of two *Namacalathus* fossils. The first defines a well-developed hexagon, while the second, which came to rest inside the first, is composed of three segments. (D) Another specimen showing hexagonal geometry. (E) Original goblet-shaped fossil described by Grotzinger et al. (1995). Note adjacent cup-shaped fossils, which also represent cross sections of *Namacalathus*. All specimens in A-E can be explained as random sections through a single three-dimensional morphology (see text).

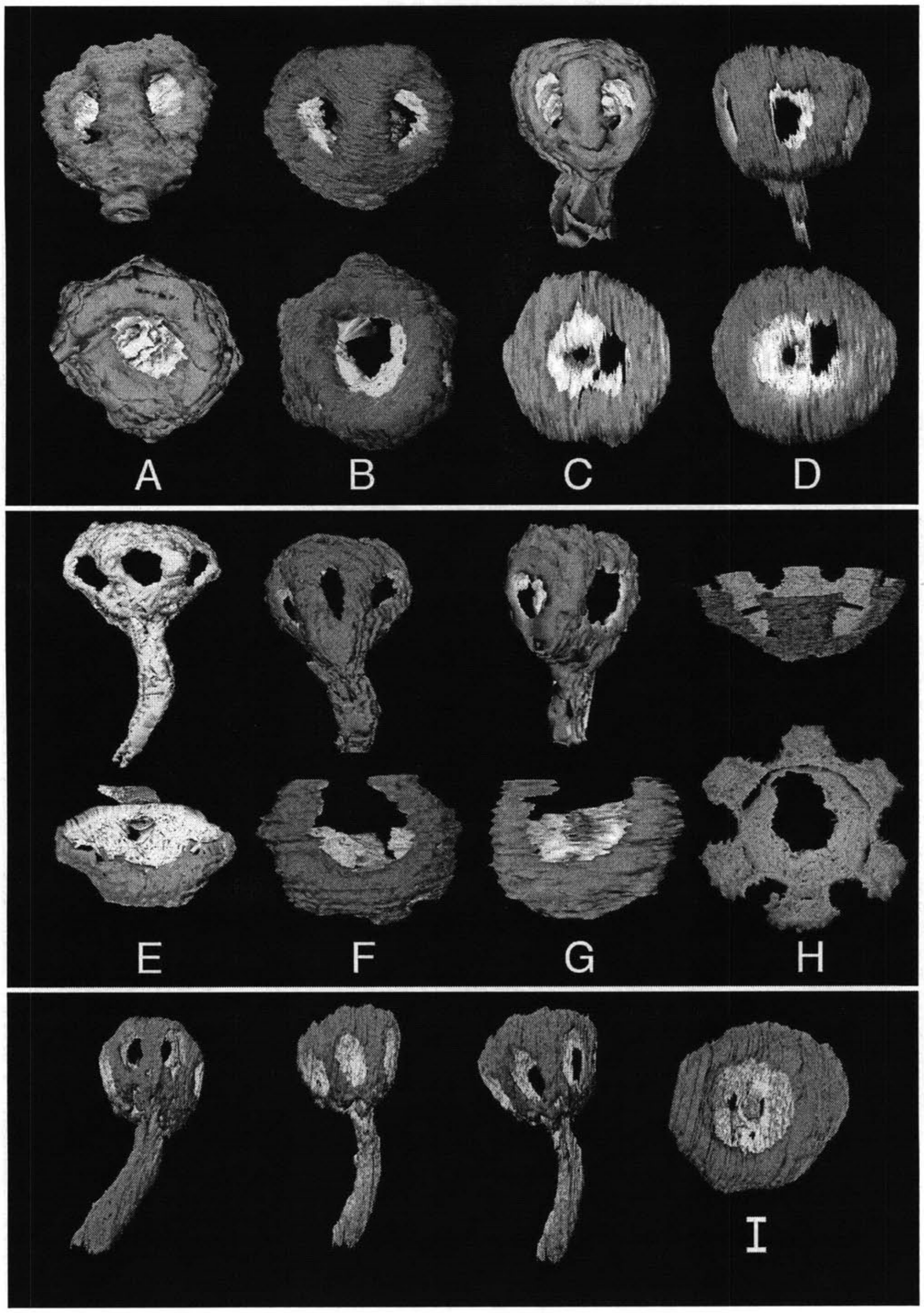

Figure 1-9: Tomographic reconstructions of the calcified fossil *Namacalathus hermanastes*. First described as having a "goblet" shape [Grotzinger, 1995 #1028], the fossil has a well-defined stem which is attached to an outward flaring cup perforated by 6 or 7 incurved holes, with an upper circular opening lined by an inward-turning lip. Partial specimens A through H are shown in inferred growth position (top) and also viewed from above (bottom). Note basal stem (or what is left of it) attached to a cup (A, C, D, E, F, G, I). The cup has 6 or 7 holes, and these are apparent with a regular symmetry in the majority of cases. Note also the single, circular opening (or part of it) in the inferred tops of each specimen. Complete specimen I is shown from several perspectives.

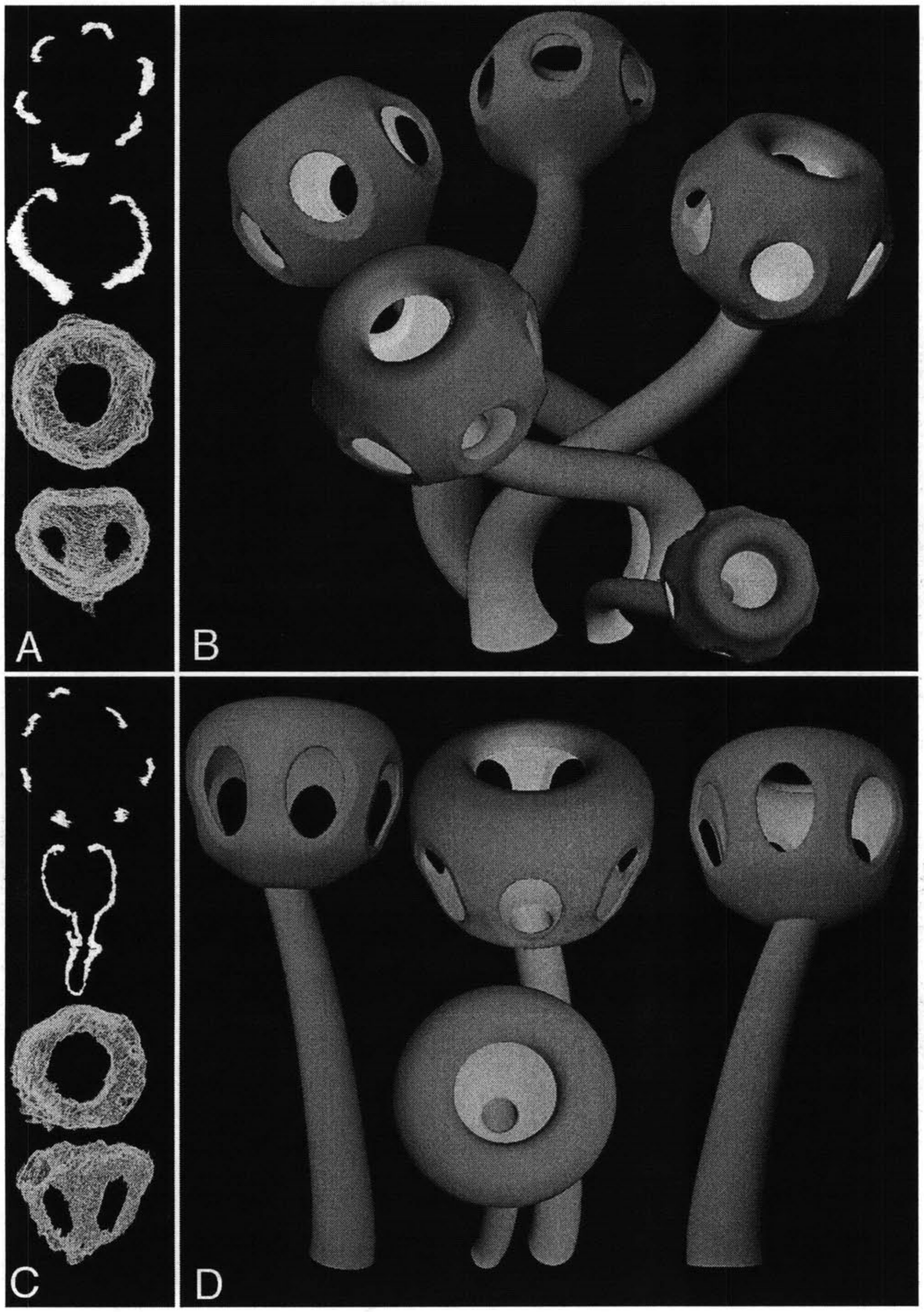

Figure 1-10: Mathematical models, based on 11 geometric parameters that capture the essential morphology of *Namacalathus hermanastes* (B and D) and cross sections of tomographic models upon which these are based and which were used to obtain parameter values (A and C). (A) From model B in Figure 1-9: a cross section showing regular holes sliced horizontally (relative to growth position) through center of cup (top), vertically through growth axis (upper-middle), and wire-frame views of circular opening at inferred top (lower-middle) and of the cup in profile (bottom). (B) Mathematical model informed by the images in A. (C) The same cross sections and wire-frame diagrams for Model I in Figure 1-9. (D) Mathematical model based upon the images in C.

population. Figure 1-12 illustrates the variation in cup size (maximum diameter) and shape (aspect ratio = maximum diameter/height) for a sample population of reconstructed specimens from a single rock sample (see Appendix A for a detailed discussion of these results).

Mineralization. *Cloudina* was only lightly mineralized, apparently by the precipitation of calcite in an organic template (Grant 1990). The same is true of the new taxon, *Namacalathus hermanastes*, reported here. Like *Cloudina*, *Namacalathus* had thin walls (little more than 100 μm thick in best preserved specimens) that deformed flexibly during compaction (Figure 1-13). Petrographically, most fossils are preserved as casts of void-filling calcite, with rare evidence of thin, organic-rich wall structures (Figure 1-13D-F). Such fossils might be interpreted as casts and molds of unmineralized organisms, were it not for fragmental remains that were almost certainly mineralized before fracture and dispersal. Original carbonate mineralogy is difficult to determine because of extensive dissolution or replacement, but common overgrowth of shells by euhedral calcite crystals supports the interpretation of a calcite precursor for *Namacalathus*. Overgrowths show strong preference for the outer walls of these fossils; inner walls tend to be smooth (Figure 1-13G). Overgrowths are not observed on the simple tubes or on *Cloudina*.

Evidence of calcite mineralization is intriguing because field and petrographic studies show that aragonite was favored as the common marine cement in Nama environments, accreting ooids, filling neptunian dikes that cross-cut pinnacle reefs, growing among thrombolitic mesoclots in reefs, and lining primary void space in grainstones (Grant 1990; JPG unpublished data). This implies that Nama animals had already evolved the physiological capacity to regulate mineral precipitation.

Paleoecology. *Namacalathus* populations are closely associated with thrombolitic textures, but they do not appear to have participated in reefal frame construction or sediment baffling. The "stem and cup" morphology of the Nama fossils suggests that they were benthic organisms tethered to rather than nestling on or within their substrate; however, most specimens are oriented horizontally and not in life position (Figure 1-12A).

Namacalathus may have been attached directly to reef surfaces or anchored by an unmineralized basal holdfast to locally accumulating sediments. It is also possible, however, that *Namacalathus* individuals were attached to seaweeds that colonized accreting microbialites—much as living stauromedusae of the scyphozoan *Halyclystus auricula* attach to algae and sea grasses by means of discs at the ends of their exumbrellar stalks (Ruppert and Barnes 1994; AHK personal observation). Invertebrate epibionts of seagrasses are well known from Cretaceous and Tertiary rocks, although the plants, themselves, are rarely preserved (Brasier, 1975). More to the point, diverse and abundant macroscopic algae occur in terminal Proterozoic successions of South China (Chen et al., 1994; Steiner 1994; Xiao et al. 1998), and thallose algae grow profusely on subtidal microbialite surfaces today in places like the Bahama Banks (Feldmann and McKenzie 1998) and Shark Bay, Australia

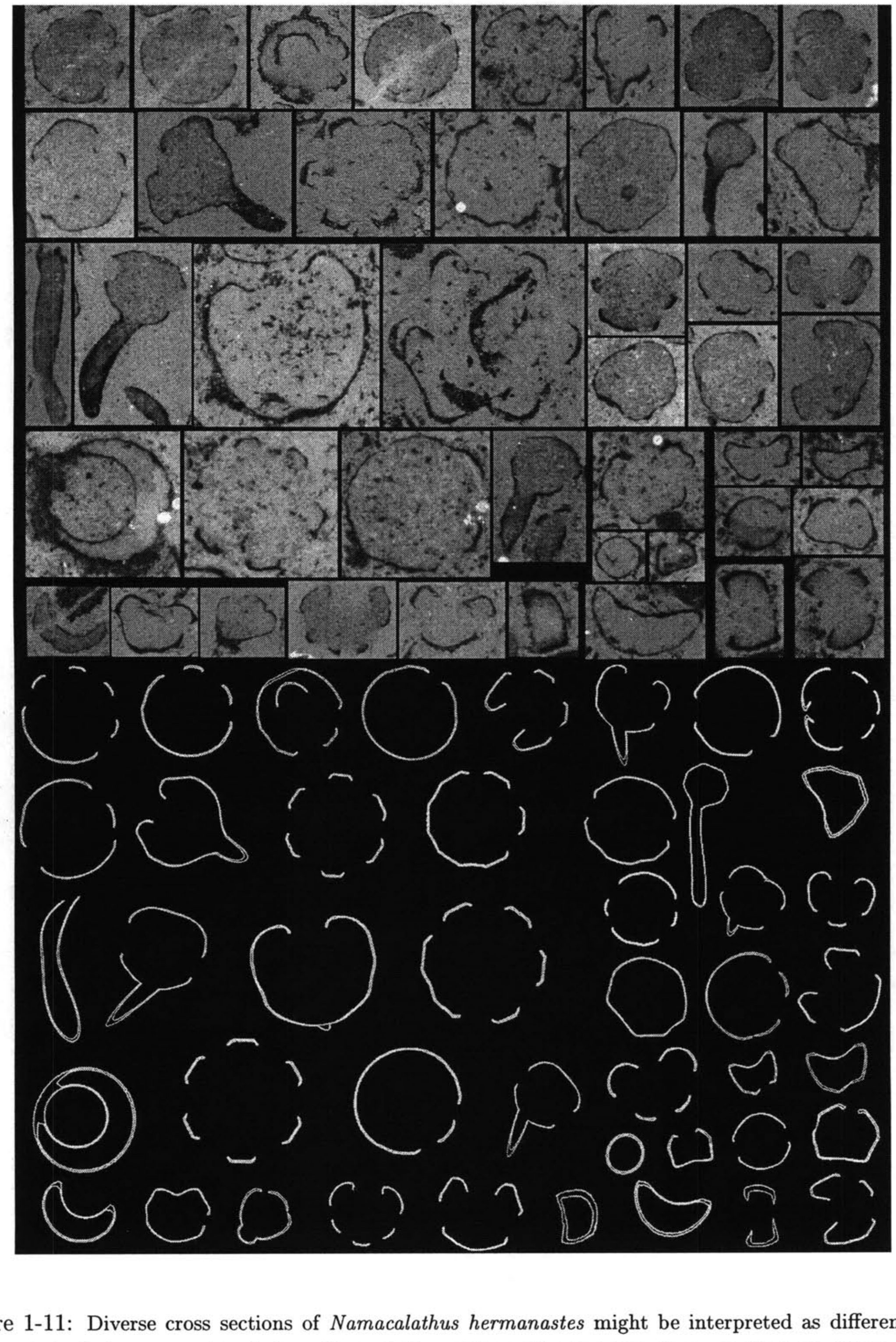

Figure 1-11: Diverse cross sections of *Namacalathus hermanastes* might be interpreted as different taxa (upper photos); however, virtual slices through the mathematical models shown in Figure 1-10 produce a diverse array of synthetic cross sections (lower images) that can be used to estimate species richness. Most of the diverse cross-sections of calcified fossils in Nama rocks can be explained in terms of a single morphology with some minor variations. The cross sections shown in the upper photos were obtained from 40 specimens, only two of which were used to generate tomographic reconstructions (Figure 1-9E and F).

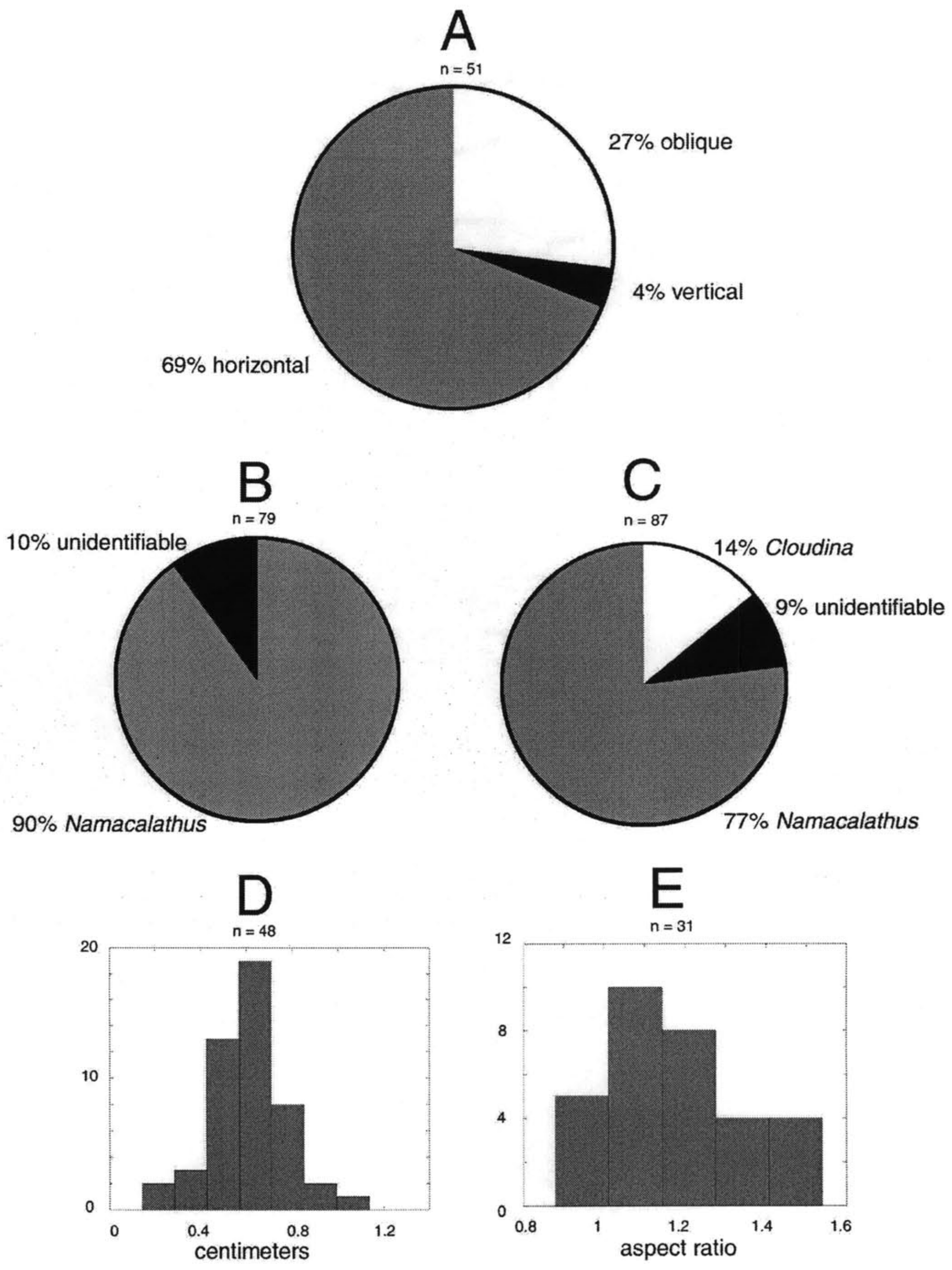

Figure 1-12: Statistics on the orientation, diversity, and geometry of calcified fossils in the Nama Group, from two samples. (A) Percentage of *Namacalathus* fossils in three categories of orientation. (B) Percentage of fossils by type. (C) Percentage of fossils by type found in a sample from the Witputs subbasin. (D) Size distribution of *Namacalathus* cups. (E) Distribution in aspect ratio (maximum cup diameter to cup height). See Appendix A for a detailed discussion of these results.

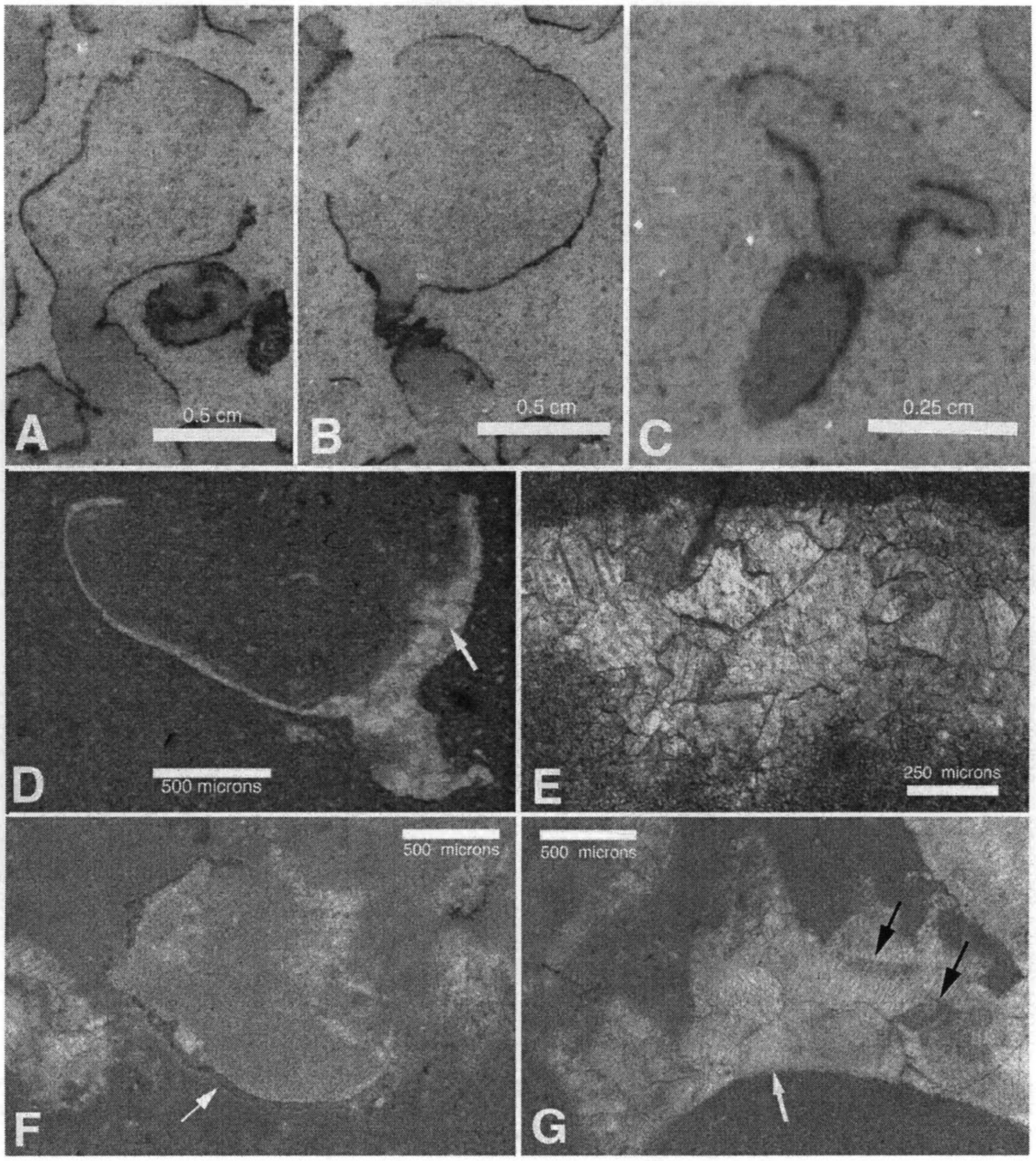

Figure 1-13: Taphonomic aspects of *Namacalathus*. Most of the photographs indicate the relatively flexible nature of the wall, suggesting only light mineralization at time of death. (A)-(C) Deformational features include: folds and wrinkles in walls and stems; squashed and flattened cups; cups without stems, presumably because of post-mortem breakage; wall ruptures, holes, and impressions caused by adjacent objects; and disintegration of walls. (D) Cross section through a goblet-shaped fossil, illustrating replacement by void-filling calcite. Arrow points to section enlarged in Figure 1-13E. (E) Void-filling calcite indicates decay/dissolution of primary wall. Note crystal size enlargement toward center of fossil wall. (F) Rare preservation of primary shell structure (arrow) showing calcite-impregnated, dark organic matrix. In most specimens, this material has decayed/dissolved, with secondary void space filled by void-filling calcite (as in Figure 1-13D, E). (G) Wall of cup-shaped fossil. Note that the inner part of wall is smooth (white arrow), while the outer part is overgrown by a relatively thick layer of replacive calcite. Note crystal terminations extending out into and replacing matrix. Black arrows point to replaced intraclasts in original matrix. This suggests that nucleation and orientation of mineral overgrowths were controlled by primary crystal orientation within the fossil wall.

(AHK and JPG, personal observations). Whatever their mode of attachment, *Namacalathus* far outnumbered *Cloudina* and other calcified metazoans in Nama reef communities (Figure 1-12B, C).

Paleobiological Interpretation. Among metazoans, goblet morphology might be expected to occur among the Porifera, the Cnidaria, or early stem branches of the Bilateria (before the evolution of strongly bilateral symmetry). *Namacalathus* could, thus, be viewed as the remains of simple asconid sponges with thin body walls and a relatively large spongocoel. Such an interpretation would be consistent with reports of Ediacaran impression (Gehling and Rigby 1996), biomarker (McCaffrey et al. 1994) and spicule (Brasier et al. 1997) fossils that suggest a broader importance of this group in terminal Proterozoic communities, and, more generally, with the phylogenetic expectation that sponges should have appeared early (Nielsen et al. 1996). On the other hand, the walls of the Nama fossils contain no evidence of pores or spicules. The former could, of course, have been obliterated during diagenesis and the latter is not conclusive insofar as not all sponges synthesize mineralized spicules. The observations are made to underscore the point that Nama goblets contain no features beyond overall shape that suggest affinities with sponges. The symmetric distribution of side holes is not easily reconciled with poriferan functional biology.

Chalice- or goblet-like morphologies can be found among the Cnidaria—for example, in hydrozoan polyps, the asexual scyphopolyps of some scyphozoans, and sessile scyphozoan stauromedusae in which exumbrellar surfaces are elongated into stalks (Ruppert and Barnes 1994). The side holes ("windows") characteristic of *Namacalathus* fossils can be reconciled with a cnidarian body plan, especially if these holes are interpreted as diagenetic features whose persistent occurrence and regular arrangement reflect underlying biological structure. In a number of specimens, sediment within goblets is distinct in color or texture from externally encompassing sediments. The boundary between sediment types is commonly just as sharp across side holes as it is across preserved walls (see Figure 1-11), suggesting that wall material once existed where holes are found today. In contrast, mixing of internal and external sediments suggests that the gap at the top of the cups was open at time of burial.

Features potentially capable of governing the positions of lateral holes include (unmineralized) tentacles at some distance from the mouth (Brusca and Brusca 1990), planuloid buds arising from surfaces of goblet-shaped scyphopolyps (e.g., *Cassiopea*; Lesh-Laurie and Suchy 1991, Figure 59), or radially distributed biological structures such as septa and gonads whose decay contributed to localized dissolution of wall carbonate. Viewed in this way, *Namacalathus* can be interpreted in terms of cnidarian structure and function: sessile organisms with a mouth that opened into a large gastrovascular cavity.

Other interpretations are less attractive. Stem groups to the bilaterian (or ctenophore + bilaterian; Ax 1989; Eernisse et al. 1992; Nielsen et al. 1996) clade might have had goblet-like morphologies, but the large internal cavity characteristic of Nama specimens isn't easily incorporated into such a view.

Calcification occurs within the red, green, and, to a limited extent, brown algae, but no known living or fossil forms (Wray 1977; Fritsch 1965; Graham and Wilcox 2000) are similar in morphology to the Nama fossils. *Namacalathus* does not display the internal cellular structure that is ubiquitous among calcified rhodophytes; nor is the goblet morphology of this fossil approximated by known red algae. Calcifying green algae in the orders Codiales and Dasycladales precipitate aragonite intercellularly within thalli, so that preserved skeletal carbonates include hollow molds of the complex filaments that underpin thallus morphology; no such internal structure marks the Nama fossils. Some Paleozoic dasyclads had a bulbous morphology, but their external profile reflects interlocking elements that arise from a central stalk rather than a continuous sheet of cellular material. *Namacalathus* is unlikely to be an alga.

Still less attractive are comparisons with broadly vase-shaped protists such as tintinnids, allogromid foraminifera, and testate amoebae. Known living and fossil species have volumes as much as three orders of magnitude less than those of the Nama fossils (Lee et al. 1985).

In addition to comparisons with living organisms, one can ask how *Namacalathus* compares with other fossils. The cup-like structure of *N. hermanastes* is at least broadly reminiscent of the small, radially symmetrical impressions found abundantly in some Ediacaran assemblages (e.g., Narbonne and Hofmann 1987; Sokolov 1997). It calls to mind, as well, specimens of the problematic calcareous fossil *Khatsaktia* and rare corallimorphs from Lower Cambrian carbonates in Siberia and elsewhere (Debrenne et al. 1990). The hexagonal symmetry of many *Namacalathus* specimens further recalls the triradiate symmetry characteristic of some Ediacaran organisms (e.g., *Tribrachidium* and *Albumares*; Fedonkin 1990) and the calcareous anabaritids, tubular fossils found in uppermost Proterozoic and Lower Cambrian successions (Bengtson et al. 1990; Qian and Bengtson 1989). The weakly calcified Nama fossils do not, in general, appear to be phylogenetic precursors of the diverse, skeletonized bilaterian animals found in Cambrian and younger rocks.

Evidence of Further Diversity. Germs (1972a) described two species of *Cloudina*, *C. reimkeae* and *C. hartmannae*. *Namacalathus hermanastes* adds to the diversity of calcified Nama animals, but does not exhaust it. Approximately 10% of the forms observed on thrombolitic surfaces cannot be ascribed to *Namacalathus* or *Cloudina*, and we believe that these represent one or more additional, if poorly characterized, taxa. Calcified Nama fossils certainly include additional populations of simple tubes. These fossils, which commonly show a gentle curvature, are typically 1-2 millimeters wide. Maximum lengths are harder to estimate because the fossils are almost always broken, but segments up to 3 cm long have been observed. Shell walls are very thin and appear to reflect carbonate impregnation of an organic matrix. At least some of the tubes are closed at the base (Figure 1-

14D), differentiating them from *Namacalathus* stems. Smooth walls, as seen in petrographic thin section, further differentiate these tubes from *Cloudina* (see discussion, below). Monospecific tube assemblages occur in micritic carbonates that accumulated on level bottoms on the Nama platform.

Like other Nama fossils, the simple tubes are problematic. Functional considerations suggest that their makers were benthic organisms that lived erect on the seafloor, perhaps filter feeders analogous to if not necessarily homologous with modern fan or feather-duster polychaete (annelid) worms (Brusca and Brusca 1990). Calcareous polychaete tubes are known from limestones as old as Ordovician (Steele-Petrovich and Bolton 1998), and unmineralized polychaetes occur in the Burgess Shale and its Lower Cambrian counterparts (Conway-Morris 1998).

Exceptionally preserved populations allow fresh observations of *C. riemkeae*, the smaller and lesser known of the two *Cloudina* species in Nama rocks. The highest concentrations of this taxon occur in association with a flooding surface developed atop the Driedoornvlagte bioherm. Like the larger *C. hartmannae*, *C. riemkeae* has been reconstructed as a series of stacked cones (see Grant 1990; Bengtson and Zhao 1992); however, primary marine cements that line a continuous inner surface emphasize that the "cones" are funnel-like flanges formed episodically during the growth of a cylindrical tube. The outer part of each flange diverges from the tube wall, whereas the inner part merges with the bases of earlier formed flanges (Figure 1-14A-C). Several workers (Conway Morris et al. 1990; Germs 1972a; Glaessner 1976) have noted points of structural similarity between *Cloudina* and tube-forming annelids such as the serpulid *Merceriella* (Fauvel 1923). The morphologies illustrated in Figure 1-14 also resemble the (unmineralized) tubes of some modern pogonophorans (Ivanov 1963), as well as diagenetically mineralized vestimentiferan worm tubes recovered from Cretaceous vent deposits in California (Little et al. 1999). (Pogonophorans and vestimentiferans are considered to nest phylogenetically within the Annelida; McHugh 1997.) Alternatively, *C. riemkeae* might be compared to the episodically accreting (unmineralized) sheaths of budding scyphozoan polyps (Werner, 1967). Whatever its affinities to living organisms, *C. riemkeae* does resemble *Saarina*, organically preserved tubes from the Vendian of the Russian Platform that are comparably organized (Sokolov 1997). Some Early Cambrian anabaritids also have well developed flanges (Bengtson et al. 1990; Qian and Bengtson 1989).

The Nama Group, thus, contains at least five taxa of calcified metazoans: *C. hartmannae* and *C. riemkeae* (Germs, 1972a; Grant 1990), *N. hermanastes*, simple closed tubes, and the uncharacterized forms in reef assemblages. More taxa may be recognized in continuing work; for example, fragmental remains in skeletal packstones hint at additional diversity. In absolute terms, observed diversity is modest, although only about a dozen species of canonical Ediacaran fossils have been reported from Nama sandstones (Richter 1955; Pflug, 1970a, 1970b, 1972; Germs 1972b, 1972c; Runnegar 1992; Narbonne et al. 1997).

Globally, the Ediacaran biota contains fewer than 100 taxa (Fedonkin 1990; Runnegar 1992).

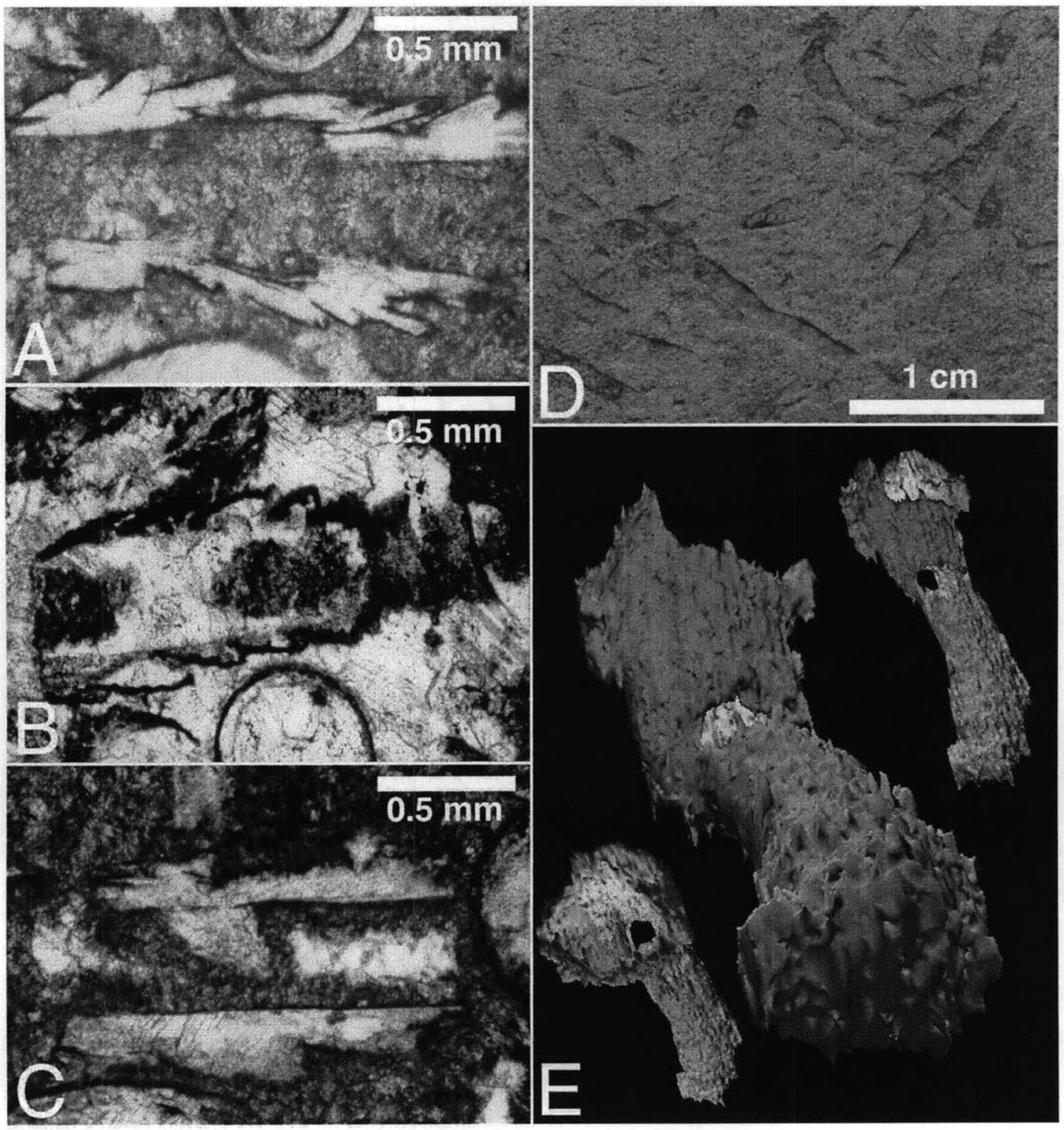

Figure 1-14: Other calcified fossils in the Nama Group. (A-C) Thin section photomicrographs of *Cloudina riemkeae* showing unusually good preservation of shell structure. In all three cases, note the straight inner wall, highlighted best in B and C, but also visible in A. A and B show external wall structure indicating en echelon stacking of flanges, which have been largely recrystallized in C. (D) Tube-shaped fossils comprising a distinct population of shapes, distinct from both *Cloudina* and *Namacalathus*. (E) Tomographic reconstruction of a tube-shaped fossil, showing its closed base and flared top.

Additional species of calcified metazoans have been described from terminal Proterozoic beds in Brazil (Hahn and Pflug 1985), and small mineralized fossils (foraminiferans?) have been claimed to occur in correlative units in Uruguay (Gaucher and Sprechmann 1999). Phosphatized tubes in the terminal Proterozoic Doushantuo Formatiuon, China, may or may not be of animal origin (Li et al. 1997). Regardless of uncertainties surrounding the Uruguayan and Chinese structures, calcified fossils comprise at least five percent of known Ediacaran diversity—not very different from the proportional representation of mineralized species in the Middle Cambrian Burgess Shale (Conway Morris 1998). The confirmation that a number of Proterozoic animals were able to precipitate mineralized skeletons lends support to hypotheses that relate the subsequent Cambrian diversification of well-skeletonized animals to increasing levels of predation rather than physiological innovation *per se* (Bengtson 1994).

Despite expanding a distinct taphonomic window on early animal evolution, calcified Nama fossils do not reveal Proterozoic roots for the skeletonized bilaterians that radiated across carbonate platforms in the Paleozoic Era. On the contrary, like the Ediacaran assemblages preserved in contemporaneous siliciclastic rocks, calcified Nama assemblages appear to document a distinctively Proterozoic fauna.

Systematic Paleontology

Genus NAMACALATHUS n. gen.

Type species. *Namacalathus hermanastes* n. sp.

Diagnosis. Centimeter-scale, chalice- or goblet-shaped fossils consisting of a calcareous wall less than 1 mm thick; a basal stem open at either end connects to a broadly spheroidal cup perforated by 6 or 7 holes with slightly incurved margins distributed regularly around the cup periphery and separated by lateral walls; the cup contains an upper circular opening lined by an incurved lip.

Etymology. From the Nama Group and the Greek *kalathos*, denoting a lily- or vase-shaped basket with a narrow base or, in latinized form, a wine goblet.

NAMACALATHUS HERMANASTES n. sp.

Diagnosis. species of *Namacalathus* distinguished by cups 2 to 25 mm in maximum dimension, with aspect ratio (maximum cup diameter/cup height) of 0.8 to 1.5.

Description. Goblet-shaped calcified fossils; walls flexible, ca. 100 microns thick (original wall dimensions commonly obscured by diagenetic cement growth); basal cylindrical stem, hollow and open at both ends, 1-2 mm wide and up to 30 mm long, attached to spheroidal cup; cup with maximum dimension 2-25 mm, broad circular opening at top with inward-curving lip, perforated by 6-7 slightly incurved holes of similar size and shape distributed regularly about cup periphery. Specimens preserved principally by void-filling calcite, with rare preservation of primary, organic-rich wall.

Etymology. From the Greek, *herma*, meaning sunken rock or reef, and *nastes*, meaning inhabitant.

Material. More than 1000 specimens from biohermal carbonates of the Kuibus and Schwarzrand subgroups, terminal Proterozoic Nama Group, Namibia.

Type specimen. Our understanding of *Namacalathus hermanastes* derives principally from virtual fossils modeled from serially ground surfaces. Systematic practice, however, requires that real fossils be designated as types. Accordingly, the specimen illustrated in the lower right corner of Figure 1-8E is designated as holotype for the species. The type specimen is to be reposited in the paleontological collections of the Namibian Geological Survey. Representative specimens are also housed in the Paleobotanical Collections of the Harvard University Herbaria (HUHPC #62989).

Type locality. Reefal biostrome developed at the top of the Omkyk Member, Zaris Formation, Kuibis Subgroup, exposed along the Zebra River near the boundary between Donkergange and Zebra River farms, south of Bullsport, Namibia.

Thrombolite-Stromatolite-Metazoan Reefs

Thrombolite/invertebrate associations characterize Early Cambrian and some younger Paleozoic reefs (Kruse et al., 1995; Riding and Zhuravlev, 1995; Soja, 1994). Nama bioherms show that the close ecological relationship between invertebrate and microbial populations was established before the end of the Proterozoic Eon, prompting questions about the origins of this distinctive reef association.

Microbial mat populations have contributed to stromatolitic reef accretion since the Archean (Hofmann et al., 1999; Grotzinger and Knoll 1999), but bioherms characterized by thrombolitic textures are rare before the latest Proterozoic Eon (Aitken 1967). Based in part on this stratigraphic distribution, Walter and Heys (1984) interpreted thrombolitic clots as ordinary stromatolitic laminae disrupted by burrowing. Kennard and James (1986), however, subsequently demonstrated that thombolitic textures are primary, confirming the original inference of Aitken (1967). Kennard and James argued specifically that the diagnostic clotted fabric of thrombolites is generated by in situ calcification of coccoid-dominated microbial communities. Coccoid cyanobacteria form rough, irregular mats that may, upon calcification, yield a clotted texture with only crude lamination (Hofmann, 1975), and recent work supports this interpretation for some Neoproterozoic thrombolites (Turner et al., 1997). Such an interpretation, however, leaves open an important question. Mats built by coccoidal cyanobacteria are common in Paleo- and Mesoproterozoic tidal flats (Golubic and Hofmann, 1976; Sergeev et al., 1995), but thrombolitic fabrics are essentially absent. How can this discrepancy be explained?

Feldmann and McKenzie (1998) have recently proposed that ancient thrombolites reflect the participation of higher algae in mat-building consortia. This inference is based on actualistic observations on the Bahama Banks, where cyanobacteria-dominated mat communities accrete well-developed laminae in intertidal environments, whereas eukaryotic communities produce poorly laminated, thrombolitic textures in open marine subtidal environments. Coccoid spores thought to be produced by dasycladalean green algae play a particularly important role in fabric generation.

The observations of Feldmann and McKenzie (1998) are consistent with paleoenvironmental research that locates Paleozoic thrombolites in subtidal environments (Aitken, 1967; Bova and Read, 1987; Pratt and James, 1986). They are also congruent with biostratigraphic data that bracket the expansion of thrombolites between the initial diversification of green algae and the appearance of well-skeletonized dasyclads (Wray 1977; Knoll 1992).

Observations of Nama thrombolites are also consistent with predictions of the Feldmann and McKenzie (1998) model. Nama thrombolites formed in open marine, wave-swept environments, often associated with grainstone shoals. Well-laminated stromatolites are rare in the Nama Group, but so are intertidal facies. The best developed Nama stromatolites form columns within a sheet-like biostromal unit at the top of the Omkyk Member (Figures 1-3B, C)—a unit interpreted to have formed in a very shallow subtidal environment. Significantly, thrombolitic textures are most abundant and best developed in pinnacle reefs at the top of the Huns platform and at Driedoornvlagte. These reefs are composed of large mounds, which are rarely elongate; where elongation is developed, it is defined by a low length-to-width ratio, consistent with low current velocities in deeper water subtidal settings. Thrombolite textures developed in this environment are similar to those found in Paleozoic rocks (compare Figure 1-5B with Figure 2B of Kennard and James 1986).

As noted above, *Namacalathus* and other Nama metazoans lived in specific association with thrombolite-forming communities, but they did not participate in any substantial way in reef building, either as frame-builders or sediment bafflers. With this in mind, it is straightforward to visualize a paleoecologic reconstruction in which the calcified organisms were loosely attached to the accreting thrombolitic substrate, much as calcisponges and green algae are in modern Bahamian thrombolites. Alternatively, *Namacalathus* may have been an epibiont attached to seaweeds that colonized the thrombolitic substrate. Upon death or dislocation of the algal components, calcified organisms would have collapsed to the sea floor as detrital particles to be swept into the depressions between thrombolite columns and mounds, or occasionally to be trapped in random positions within accreting mats. This interpretation also provides a mechanistic basis for understanding the locally great abundance of detached, horizontally oriented fossils in thin (a few cm) beds of thrombolitic laminites that form spatially extensive sheets within deeper subtidal facies. Before the radiation of metazoan macrograzers, broad expanses of the terminal Proterozoic seafloor may have been covered by microbial-algal carpets, with calcified metazoans tethered to a (tiered?) algal lawn. Animals assumed a constructional role in bioherm accretion only with the evolution of heavily skeletonized, attached epibenthos.

Conclusions

Nama bioherms document the emergence of a distinctive reef association. Animals that lived in the reef environment include some of the first organisms able to form skeletons by the enzymatic precipitation of $CaCO_3$ on an organic template. In important ecological and physiological respects, then, Nama bioherms and the fossils they contain foreshadow Paleozoic biology. At the same time, however, the distinctive and problematic morphologies of *Namacalathus hermanastes* and other calcified Nama fossils underscore the phylogenetic differences between terminal Proterozoic and Paleozoic skeleton formers. At present the significance of these similarities and differences is not completely known, but continuing study of the vast and beautifully exposed carbonate accumulations of the Nama Group promises to clarify our understanding of terminal Proterozoic biology and geobiology.

Chapter 2

The Serial Imaging Method

Three methods were tried for obtaining the morphology of the Nama fossil specimens. First, an unsuccessful attempt was made to dissolve the calcite matrix using an acid solution. Second, a Nuclear Magnetic Resonance (NMR) scanning device was used to image the distribution of hydrogen atoms in a water-saturated sample. There was, however, only a slight difference between the amount of water absorbed by the fossil and matrix materials. Ultimately, a serial imaging method was adopted, whereby the morphology is reconstructed from cross sections of the fossils, imaged with a digital camera at 25.4 μm intervals. This process is described in detail in the present chapter.

Obtaining Serial Images

In overview, the process whereby serial images are obtained occurs as follows. First, a small square slab of sufficient thickness containing suitable fossils is cut out from a rock. The surface of this sample is then photographed by a digital camera to obtain an image. Then, a thin layer measuring 25.4 μm in thickness is removed from the surface using a surface-grinding apparatus. The photographing and grinding is repeated in this way until the sample is ground away, producing a set of several hundred images obtained at 25.4 μm intervals.

What follows is a discussion of how the fossils are prepared for processing. First, rocks are chosen according to their suitability for analysis. The fossils reside in layered limestone slabs that are normally less than two inches thick and at least five inches in diameter. Suitable rocks have a relatively high concentration of fossils that are well-preserved. In particular, the fossils are 5 mm apart on the average, and not usually in contact with each other. Therefore, these were not deformed from contact with neighboring fossils, as occurs frequently in rocks where the fossils are densely packed. Shapes with a clear radial or bilateral symmetry are assumed to represent the original, undeformed morphologies, and therefore these are sought for study. Fossils are selected with the finest structure preserved—such as with thin walls and with minimal evidence of recrystallization. A suitable rock contains a sedimentary layer that is thick enough to include several complete specimens. That is, the thickness of this layer exceeds the average diameter of fossils observed on the surface by at least two diameters, in order to ensure that entire fossils will be obtained by the process.

Having obtained a suitable rock, a sample measuring 7.5 cm on a side is cut out from the most promising section. The sample is then installed in a square steel mount measuring 10 cm on a side (see Figure 2-1). The mount is made of a durable metal so that the sample can be temporarily fastened to the magnetic platform of the surface grinding apparatus, and so that its shape and the sample's orientation will not change with repeated grinding. The mount has a square depression measuring 7.5 cm on a side and 1 cm deep, into which the sample fits snugly. In particular, the mount is made from a square steel slab with four rectangular steel bars fastened around the perimeter to form a square depression. The square depression is filled with a durable epoxy adhesive, and the sample is placed inside so that most of its total thickness is exposed above the surface of the mount. Then the surface of the sample is ground using the surface grinding apparatus, so that it is made parallel with the base of the mount, and therefore also parallel to the platform of the surface grinder.

A Kent Industries Model K08-250AHD surface grinder[1] with a changeable 1 cm diamond wheel is used for this work. A magnetic platform slides back and forth in a direction perpendicular to the axle of the wheel and at a regular interval which can be adjusted (see Figure 2-2). As the platform slides back and forth, it also advances in a direction parallel to the wheel's axle, and also at a rate which can be adjusted. The magnetic field of the platform is engaged using a lever once the sample is installed.

After the surface of the sample is ground parallel to the base of the mount, six 1 mm holes are drilled through the whole sample, perpendicular to the surface and with regular spacing. These "registration holes" will later serve as marks for correlating the positions of images at each layer of the sample. That is, the positions of these holes in each image are used to determine the absolute position of the image data in a plane parallel to the sample surface. Once these holes are drilled, the sample is ready for the process of repeated scanning and grinding at regular intervals.

A Kodak DCS420 digital camera with a Nikon N90 camera body and 55 mm Nikon macro lens is used for taking images of the sample. The limiting color resolution is 24 bits per pixel, and the CCD array has a total spatial resolution of 1524 x 1012 pixels. The significance of the color resolution will be explained in the next section. The camera is mounted on the carriage of a copy stand whose height above the sample can be adjusted (see Figure 2-3). The camera's height above the sample is adjusted so that the whole surface can be captured in two images, and then the camera is rigidly fastened in this position. It is important that the sample be placed under the camera in the same position for every scan in order to minimize later corrections to the relative positions of image data.

[1]Located in the Rock Mechanics Laboratory in the Department of Earth, Atmospheric, and Planetary Sciences at MIT.

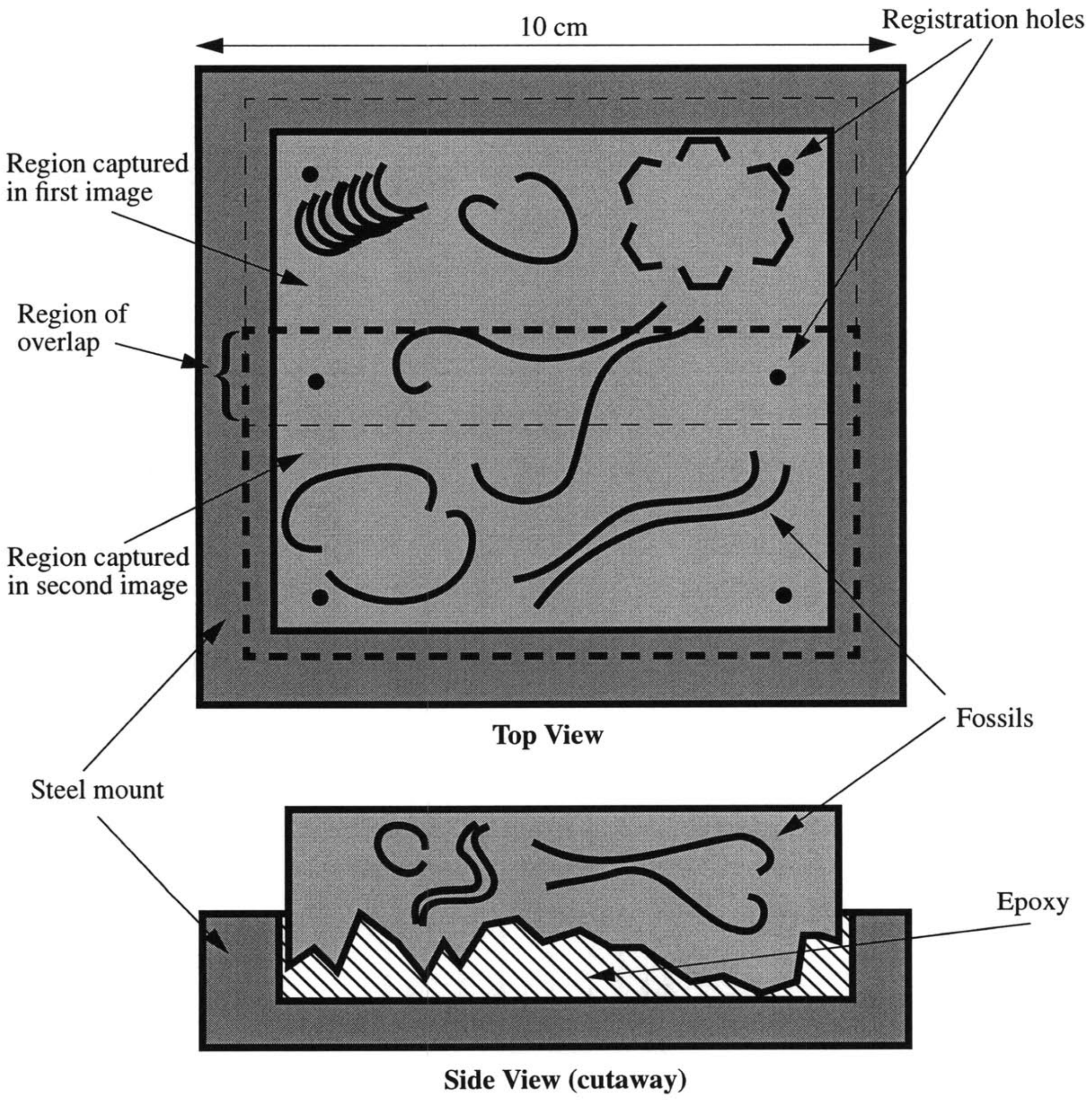

Figure 2-1: Sample and mount

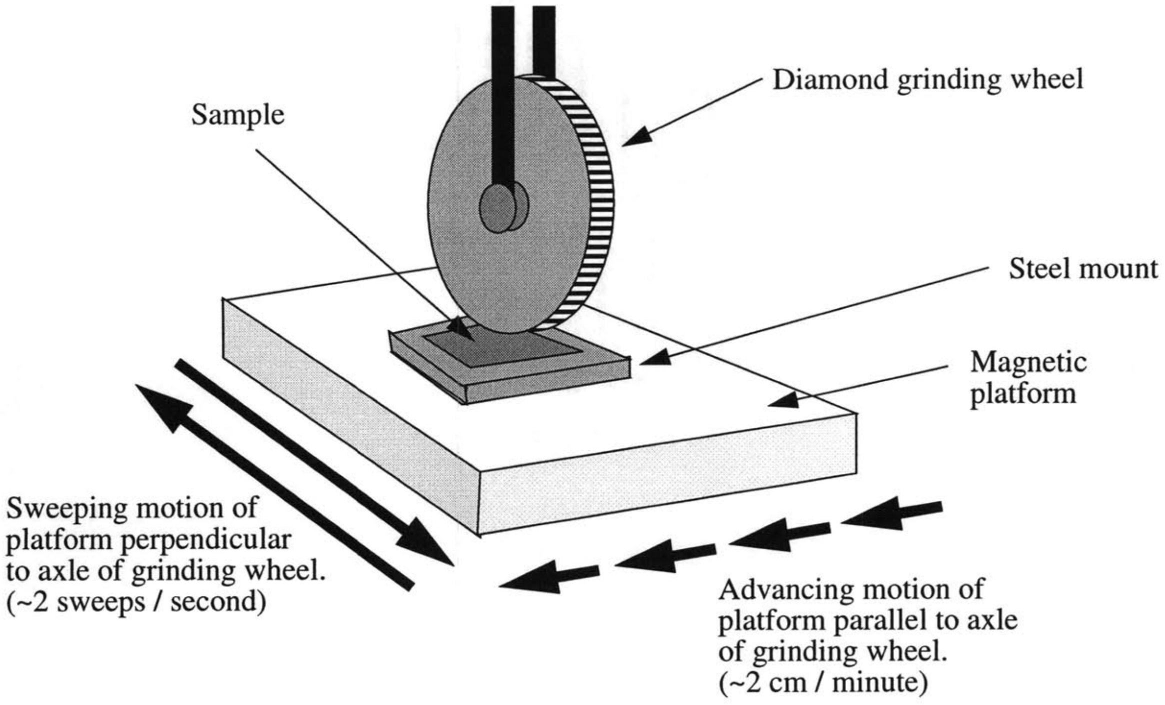
Sample
Diamond grinding wheel
Steel mount
Magnetic platform
Sweeping motion of platform perpendicular to axle of grinding wheel. (~2 sweeps / second)
Advancing motion of platform parallel to axle of grinding wheel. (~2 cm / minute)

Figure 2-2: Surface grinding apparatus

For this reason, a simple metal brace is constructed from two rectangular steel bars fastened at right angles on the platform of the copy stand. The sample is pushed against the brace in order to justify its position for the first in each pair of scans. For the second scan, the sample is displaced from the brace and a rectangular steel bar is inserted between the sample and brace. The sample is pushed firmly against the bar and brace in order to justify its position. The position of registration holes are chosen so that four holes appeared in each image. The region of overlap between the two images is comparable to the size of one fossil diameter, and it contains two registration holes (see Figure 2-1). In such a way, fossils at the middle of the sample are not split between the two images. From now on, each *pair* of images for a given layer of the sample will be referred to as a *single* "frame". The frames are stored on the camera's DCS 420 removable PCMCIA drive, and once its capacity (50 frames, or 100 images) is exceeded, these are downloaded to a computer.

In the present case, just two images are used to scan the entire surface because the resolution of each scan is deemed adequate for resolving the smallest structures observed. Smaller fossils or fossils with finer structure would otherwise require a greater spatial resolution, and therefore more images would be taken at higher magnification in order to capture the entire surface. Larger fossils or fossils whose smallest features are comparatively large might otherwise be captured using a single scan of the surface. Ultimately the choice of resolution is based upon a qualitative judgment about whether the smallest structures are clearly visible in the scanned images. In order to ensure that the

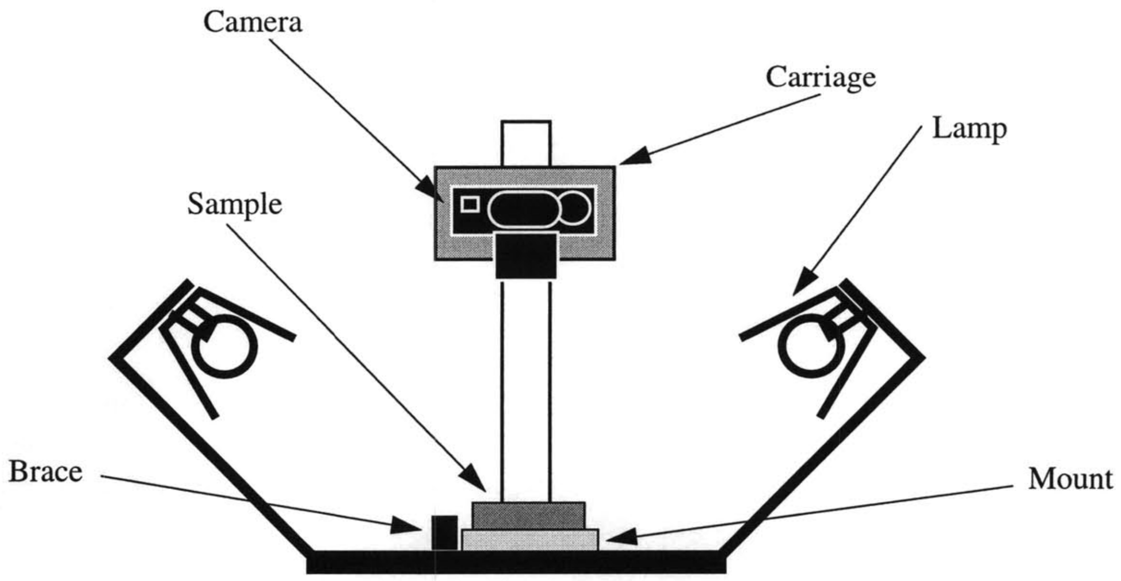

Figure 2-3: Configuration of the camera, sample, and copy stand.

highest possible spatial resolution is obtained for a given configuration of the camera and sample, the camera lens is refocused for each frame (i.e., for each fresh layer of the sample). For this purpose it is best to use a lens focused by means of aligning straight edges across a pair of adjacent semicircles at the center of the field-of-view (a so-called "split-screen" lens).

Several measures are taken to ensure that lighting is uniform across the surface of the sample so that color contrasts are well-resolved and constant throughout a given frame. First, before each image is taken, the steel metal frame of the mount is covered with black construction paper in order that bright reflected light from the copy stand lamps will not reach the camera. This stray light will otherwise diminish the sensitivity of the camera's CCD for measuring subtle variations in color. That is, the apparent and measured contrast between fossil and matrix material can be greatly diminished otherwise. Ceiling lights located directly above the copy stand are extinguished to prevent anomalous bright regions from appearing in the images. In this way, the programs used later to process the image data can be used repeatedly across frames without having to alter parameter values.

Additional measures are taken to improve color resolution by optimizing optical properties of the surface being photographed. When grinding, the surface grinding apparatus constantly supplies a lubricating milky oil-and-water solution on the sample. This oil can leave a faint white residue that will diminish the color contrast unless it is carefully cleaned away. Therefore, before each frame is taken, the surface is cleaned using a jet of compressed air and using a moist cloth towel. Moreover, it is possible to adjust the rate at which the platform sweeps under the grinding wheel (in the direction perpendicular to its axle) and also the rate at which the wheel advances across the sample (in the direction parallel to the axle), and these rates can have a significant effect upon the apparent contrast between fossils and matrix material. In particular, if the sweep rate and the rate of advance are too fast, then microscopic surface roughness will refract light into all visible wavelengths, so that the surface turns rapidly white. Moreover, a rough surface traps particles of ground rock and the aforementioned milky oil-and-water solution more easily, so that it becomes more difficult to cleanse away. Alternatively, if the sweeping rate and the rate of advance are slow, then the resulting surface is very smooth: it does not turn white and it is easily cleaned. The disadvantage of very slow rates is that more time is needed to grind and process a single layer. Ultimately, the aim is to find an optimal compromise between the amount of time needed for grinding and for cleaning, in order to produce a sharp apparent contrast between fossils and matrix material. Also, it is found that applying a thin layer of mineral oil will reduce the refractive index of visible wavelengths at the surface and produce an even sharper apparent contrast.[2]

It is found that a sweep rate of 2 sweeps per second and a rate-of-advance of about 2 centimeters per minute is approximately ideal. Summing the amount of time needed for cleaning, focusing, justifying, and taking an image, it is possible to process six or seven frames per hour. Since the layers are being shaved at 25.4 μm (1/1000 inch) intervals, a sample that is one inch in diameter will produce 2000 images in all, and require $1000/6.5 \approx 150$ man hours to process completely. From this one can see clearly the main disadvantage of this approach. Many hours are required to process a single sample, and one sample may contain only several complete fossils that are well-preserved. The next sections contain a discussion of techniques used to analyze the images in order to obtain two-dimensional outlines of the fossils (contours) so that these can be joined to make a three-dimensional reconstruction of an entire fossil morphology.

Preparing the Images for Processing

The following three sections provide a description of the process by which the outlines or "contours" of fossil cross sections are obtained from each frame using a battery of image processing techniques. In overview, the frames are first examined to select an interesting fossil for reconstruction. The volume that contains the fossil is then cut out from the collection or "stack" of frames. This cropped stack of frames is then studied in order to decide which structures belong to the fossil in order to exclude those which do not. Then, each image in the cropped stack is enhanced in order to exaggerate the color contrast between fossil and matrix material. Each frame is then converted to a binary (black-and-white) image that shows the fossil in black, and the matrix in white. The pixels at the perimeter of the black regions are then tabulated and stored as "contours".

Image processing techniques have been applied to a wide range of problems in the geological

[2]i.e., a lower refractive index enables the wavelengths in visible light to maintain their coherence, and hence to better preserve color contrasts in the image being reflected.

sciences over the past 20 years. The majority of these are problems of a statistical nature inside the purview of "mathematical geology," a relatively young discipline (Vistelius, 1969). Examples include the automatic characterization of textures in thin section by measuring the size, orientation, and color of crystals (Haralick 1979), and the spatial correlation of stratigraphic units in geologic maps (Agterberg and Fabbri, 1978). In the present study, image processing techniques are not used to make statistical measurements, but are instead used to isolate recognizable shapes for an additional purpose—namely, for the reconstruction of fossil morphologies from serial cross sections. As mentioned in Chapter 1, however, digital images of serial cross sections can be used to make simple measurements of fossil population statistics—such as the distribution in size, shape, and orientation. A detailed description of these statistics and how they are obtained can be found in Appendix A.

It should be noted that all of the computer operations described in this study are carried out using software designed for a Silicon Graphics Indigo 2 computer and the IRIX 6.2 operating system. All images are downloaded from the camera's removable PCMCIA hard drives in the form of compressed TIFF image files using the "GBA Camera" software package (GBA inc.). The GBA Camera software is used to decompress each image file into one 24-bit color bitmap (`.bmp`) file and one 24-bit jpeg (`.jpg`) file for subsequent image processing. All of the image processing—and all of what follows in the remainder of this section—is carried out using scripts written by the author for use with the Image Processing Toolbox within Matlab version 5.2 (MathWorks Co.), except where otherwise noted. Many of the scripts mentioned in this Chapter are described in Appendix C.

What follows is a description of how the raw image data is studied and prepared for image processing. A script is designed for the purpose of easily perusing all of the frames in order to identify fossils for reconstruction. Jpeg image files are used for this purpose, because of their smaller size. The fossil cross sections in the middle portion of the stack of frames are examined first, and checked to see that an entire fossil is contained in the stack. That is, a script is used to select a rectangular region of interest (ROI) in a frame near the center of the stack, and then to crop every image in the stack using the same ROI to produce a cropped stack of frames, also called a "movie". The movie is viewed to confirm that the fossil is entirely contained within the volume. Also, these movies are stored as a convenient and rough representation of the data, where time represents the third spatial dimension. Because they are meant for this purpose, generally only every 5th frame is stored in a jpeg movie so that it can be very quickly replayed. The highly-compressed jpeg image files consume less memory, and are therefore especially well suited to this purpose: i.e., they can be quickly displayed, and they consume relatively small amounts of disk space. Through the process of making jpeg movies, it is possible to find and collect all of the interesting fossils contained within the stack of images.

Before any fossil is processed for making a reconstruction, its movie is carefully studied in order to decide what features in the images belong to the fossil. For especially complex shapes, notes are taken about what features to exclude from the reconstruction, and in which frames they reside. Such notes, and the spatial coordinates of the ROIs for each fossil, and also the range of frame-numbers containing each fossil are recorded in an ASCII text file, along with a numeric label and short description. Scripts are designed for the easy recall and display of movies listed in this file. Later, once the reconstructions are obtained, scripts are used for the easy recall and display of reconstructed models and still-images of reconstructed models. In effect, this system of scripts, notes, descriptions, movies, images, and models is a database for several kinds of data in different stages of analysis.

As already mentioned, many precautions are taken to ensure that the sample is always scanned in the same position relative to the camera. Inevitably, slight differences will occur in the scale, rotation, and position of the sample within images from different frames. These differences can be very slight, or quite large if, for example, the camera is accidentally moved. For this reason, once a collection of fossils has been selected for reconstruction and before each one can be processed individually, the absolute positions of each image in a three-dimensional system of coordinates are determined. The first step in this process is to tabulate the (x, y) positions of the four registration holes in each image. This is accomplished with a script that displays each image in turn, and prompts the user to point to each registration hole in clockwise order, using a mouse. The positions of registration holes in each frame are then compared to the positions recorded for the first frame. This comparison is carried out in the manner described below, and tabulated for each frame as a relative adjustment in position, angle, and scale. Later, these adjustments are applied to the contours finally obtained from each frame, in a process that is described in a later section.

What follows is a description of the procedure used to calculate the adjustments in position, scale, and rotation. For a given image, let the coordinates of the four registration holes be denoted by (x_i, y_i) for $i = 1, 2, 3, 4$, where (x_1, y_1) is the position of the registration hole in the upper-left corner, proceeding clockwise until (x_4, y_4) in the lower-left corner. For an image in the first frame, these values are denoted using the subscript "A": i.e., (x_{Ai}, y_{Ai}) for $i = 1, 2, 3, 4$. The translations Δx and Δy, the rotation ϕ, and the scale-factor s are calculated as follows, where s is the average change in scale[3] for the distances measured from (x_1, y_1) to (x_2, y_2), (x_3, y_3), and (x_4, y_4).

$$\begin{aligned}
\Delta x &= (x_{A1} - x_1) && (2.1)\\
\Delta y &= (y_{A1} - y_1) && (2.2)\\
d((x,y),(x',y')) &\equiv \sqrt{(x-x')^2 + (y-y')^2} && (2.3)\\
s &= \frac{1}{3}\sum_{j=2}^{4}\left(\frac{d((x_{A1},y_{A1}),(x_{Aj},y_{Aj}))}{d((x_1,y_1),(x_j,y_j))}\right) && (2.4)
\end{aligned}$$

[3] An average is taken in order to diminish the effect of slight errors in the measurement of the positions of registration holes. These errors are discussed in more detail in a later section.

$$a \equiv d((x_{A1}, y_{A1}), (x_{A2}, y_{A2})) \tag{2.5}$$

$$b \equiv s(d((x_1, y_1), (x_2, y_2))) \tag{2.6}$$

$$c \equiv d((s(x_2 + \Delta x - x_{A1}) + x_{A1}, s(y_2 + \Delta y - y_{A1}) + y_{A1}), (x_{A2}, y_{A2})) \tag{2.7}$$

$$\phi = -\frac{\Delta y}{|\Delta y|} \arccos\left(\frac{a^2 + b^2 - c^2}{2ab}\right) \tag{2.8}$$

Image Processing Techniques for Isolating Fossils

To begin the process of obtaining contours for an individual fossil, a cropped stack of frames using the ROI recorded in the database is generated first. Unlike before, these are cropped from the bitmap (*.bmp) image files. The bitmap image files contain the full color and spatial resolution of the original compressed TIFF files, and are hence preferred for the image processing that follows. The next step is to convert each color image into a so-called "grey-scale" image (also called an "intensity" image). In a grey-scale image, the color of each pixel is represented by a single number, normally in the range from zero (black) to 1 (white) (e.g., 0.1 represents a very dark grey). As mentioned, every image in the stack has 24-bit color resolution. That is, every color is represented using 24 bits of information. In the case of bitmap files, the 24 bits are sub-divided into three sets of eight bits, or, equivalently, three numbers on a scale from 1 to $2^8 = 256$. It is well-known that the majority of colors can be derived from a combination of three primary color intensities, such as red, green, and blue. Accordingly, all of the colors in a 24-bit image are composed of three color components, which are coded by an intensity value from 1 to 256. A bitmap image measuring $m \times n$ pixels is, in effect, a $m \times n \times 3$ matrix with elements that have a value from 1 to 256. In effect, the camera obtains three images of the sample, in all the intensities of red, green, and blue light, and these are combined to produce a single 24-bit color image (Gonzalez 1992).

There are several methods for converting a 24-bit color bitmap image into an intensity image. Our approach is chosen according to a qualitative assessment of the final result; that is, the method is chosen which produces the sharpest color contrast after an intensity histogram adjustment, which is described later. First, the three color components are examined separately. These are plotted in Figure 2-4 using a grey-scale color map.[4] Because all the colors of the rock sample are combinations of red and green, the blue component contains virtually no information and therefore it is ignored. The color values of the red and green components are then combined using a simple product and rescaling, in order that each color is represented by an intensity in the range from 0 to 1. That is, given a red color value r where r is an integer from 1 to 256, and given a green color value g in the

[4] i.e., in the case of the red component, where normally a pixel value of 1 would represent black, and a value of 256 would represent light red, in these print-outs the number 256 represents white (very light grey), and smaller numbers represent shades of grey.

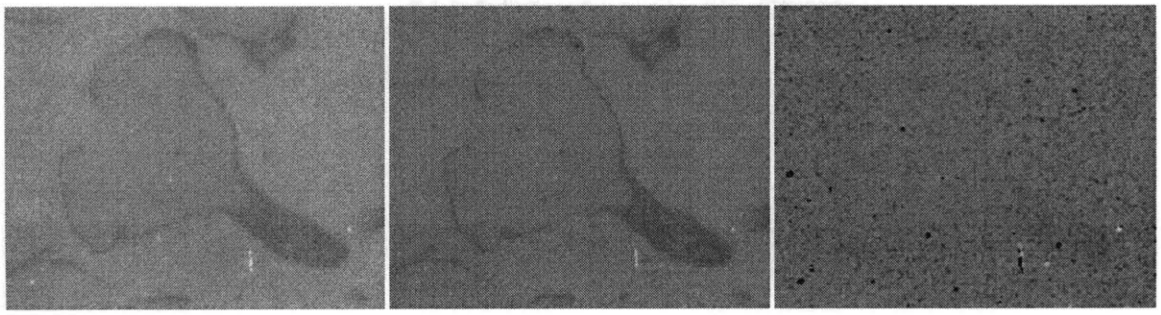

Figure 2-4: From left to right: red, green and blue components of the 24-bit image

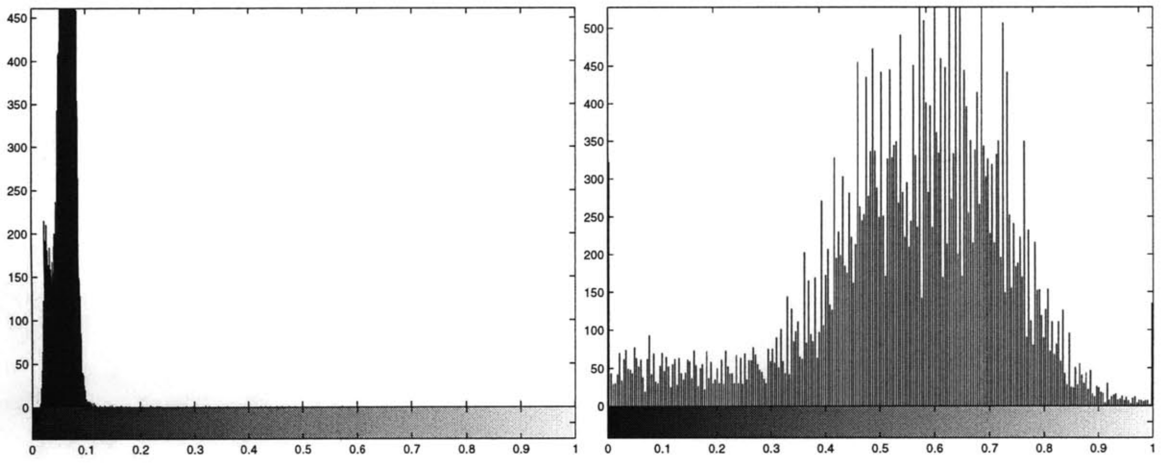

Figure 2-5: Left: Original color histogram; Right: Adjusted histogram.

same range, the combined color intensity $I(r, g)$ is given by:

$$I(r,g) = \frac{rg}{256^2} \tag{2.9}$$

In total, $256^2 = 65,536$ intensities are possible in the range between $I(0,0) = 0$ and $I(256,256) = 1$. In this way, pixels that are very luminous in the green and red components (i.e., with large color values) are accordingly luminous in the intensity image, and inversely for pixels that are dark in both components. An intensity histogram for the "product image" is shown on the left-hand-side in Figure 2-5. An intensity histogram is a plot of the number of pixels in the image that have intensity values in each of many small intensity intervals. For example, the y-axis coordinate that corresponds to the x-axis value x = 0.2 is the number of pixels that have an intensity of about 0.2. Because the original color components contain mostly dark pixels, their product image is similarly dark, as can be seen from Figure 2-5.

In order to enhance the contrast between fossil and matrix material, a simple intensity histogram adjustment is performed, also known as "contrast stretching" (Gonzalez 1992). It happens in this case that the majority of colors in the image lie within a narrow range of intensity values. In effect, the aim is to "stretch" these distinct intensity values over the whole spectrum from black to white, in order to enhance the intensity contrast. In mathematical language, the histogram adjustment is a simple bijective mapping from a subset $[a, b]$ of the interval $[0, 1]$ to the whole interval $[0, 1]$. That is, given $a, b \in [0, 1]$ such that $a < b$ and some intensity value x in the product image such that $x \in [a, b]$, then the adjusted intensity mapping $I'(x)$ is given simply by:

$$I'(x) = \frac{x - a}{b - a} \tag{2.10}$$

A script was designed to automatically select appropriate values of a and b. This is accomplished by scanning the original histogram values from $x = 0$ with increasing x until an intensity value is found exceeding some value y_{min1}. The script continues along the histogram with increasing x until y drops below a value y_{min2}. The script takes as a parameter the value of y_{min1}, which in effect determines the amount of discrimination between darker intensities. The value of y_{min1} is chosen according to a qualitative assessment of intensity contrast in the final result. Because this resulting contrast depends partly upon the original color contrast between fossil and matrix material, the optimal value of y_{min1} is sample-dependent; it is generally chosen once per sample and is not adjusted for every image. In the present case a relatively small value of y_{min1} is selected. The adjusted intensity histogram is shown on the right-hand-side in Figure 2-5, and the resulting image is shown in Figure 2-6. Notice in the right-hand-side of Figure 2-5 that that there is one large peak in the middle range of intensities, and a constant level of "counts" in the darkest end of the intensity spectrum. As we shall see, the intensity peak at the right corresponds mainly to colors in the matrix material, while the darker intensities correspond to fossil material. In effect, the intensity histogram confirms that the images are mostly comprised of matrix material.

Occasionally it happens that the color of matrix material is not constant throughout the image. In particular, at times there is a marked difference in the color contrast between fossil and matrix material between two or three parts of the image that have recognizable boundaries. In that case, the color histogram adjustment that is needed for different parts of the image will vary. For this reason, a script was designed which enables the user to isolate different parts of the image and to perform a histogram adjustment for each part separately. Each part of the image is then processed separately in the manner described below, and the result is summed to produce a single image from which the contours are obtained.

A "binary" image is obtained from the intensity image, using a simple thresholding procedure. A binary image contains two colors: the number 1 represents white, and 0 represents black. The binary image is generated simply by choosing a threshold value T so that all pixels with an intensity

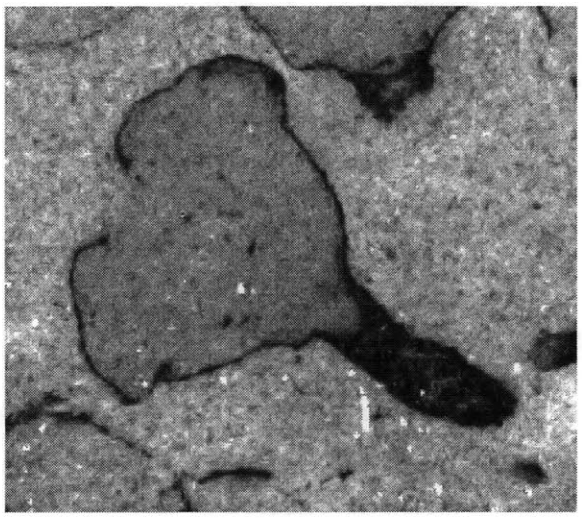

Figure 2-6: High-contrast image

value exceeding T are turned to white, and all pixel values less than or equal to T are turned to black. In mathematical language, for a pixel with an intensity value x in the histogram-adjusted product image, the binary intensity I'' for that pixel is obtained using a surjective map $I''(x)$ from the interval $[0, 1]$ to the set $\{0, 1\}$:

$$I''(x) = \begin{cases} 0 & \text{if } x \leq T, \\ 1 & \text{if } x > T. \end{cases} \tag{2.11}$$

The value of T is adjusted until the fossil is clearly distinguishable in black against a white background and amid other fossil fragments in black. Because care is taken to maintain a constant apparent color contrast between images taken of the sample (e.g., constant lighting conditions), the value of T requires little adjustment between frames. That is, T is approximately constant for a given sample, unless the matrix changes color between adjacent layers because of changes in the quality of illumination. Binary images for three values of T are shown in Figure 2-7. For the present example, the optimal value is $T = 0.32$. Notice that this value corresponds approximately to the boundary suggested earlier between the fossil and matrix intensities, visible in the adjusted histogram of Figure 2-5. Occasionally an image is very poorly resolved for one of several reasons (e.g., the focus is poor, or the person taking the images forgot to install the black construction-paper mask over the mount, etc.). If the apparent color contrast of the original image is poor, then the thresholding procedure may not produce a binary image in which the fossil can be clearly distinguished from the surrounding material. For these cases, a separate script was designed which obtains the coordinates of a contour by simply prompting the user to select many points along the boundary of the fossil with a mouse, using the histogram-adjusted image (Figure 2-6). (Alternatively, if the color contrast

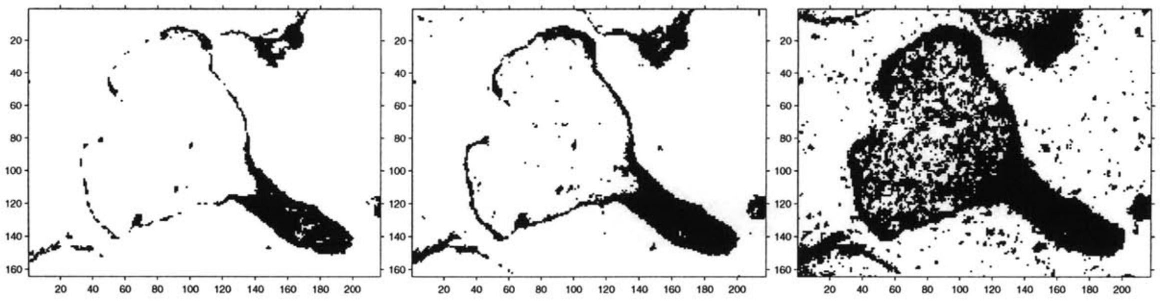

Figure 2-7: From left to right: binary images for $T = 0.25, 0.32$, and 0.55

is very good, then the fossil can be clearly distinguished for a range of values of T.)

Having obtained a binary image A with an appropriate threshold, it is possible to isolate the fossil from other features in the image. Recall that at this stage all fossil fragments are displayed in black (zeros) and the matrix in white (ones). A script was written which prompts the user to select desired features in the image by choosing at least one point within a contiguous black region. In practice, at least one point P_i is chosen from every feature that is to be isolated. Let us call this the set of points $\mathcal{P} = \{P_1, P_2, ..., P_n\}$. The script then generates a separate binary image B which is equivalent to A except that all the points in $\mathcal{P}$ and all points connected to points in $\mathcal{P}$ have been changed from the value 0 (black) to 1 (white). (Two points P and Q are *connected* if there is a set of points of the same color as P and Q such that each point is the immediate neighbor of at least two others in the set, and such that one point is a neighbor of P and one point is a neighbor of Q. One pixel is the "neighbor" of another pixel if it occupies one of eight adjacent positions.) In practice, the [Matlab] procedure may begin at a point $P \in \mathcal{P}$ and turn to white all the black points neighboring P. This is repeated for each of the points adjacent to P, and so on. In effect, the script performs a "flood fill" which converts pixels in the selected regions in A from black to white, and stores this in the binary image B (i.e., all the regions that are black in A are also black in B, except for those which have been flood-filled).[5] Finally, a script generates the binary image $C = B - A$, which contains just the flood-filled region in white, against a black background. For the present example, the binary image C containing all of the selected features is shown on the left in Figure 2-8.

There are a variety of so-called "morphological" operations which can be used to manipulate a binary image. Many of these operations were tried, and just two were chosen which have the effect of reducing the image "noise": i.e., the points which contribute to the roughness of boundaries or which are probably artifactual. The first operation is called a "closing," and this involves two simpler operations, called "dilation" and "erosion." Recall that at this stage the selected binary objects are

[5]See page 532 of Gonzalez (1992) for an alternative explanation of the flood-filling procedure.

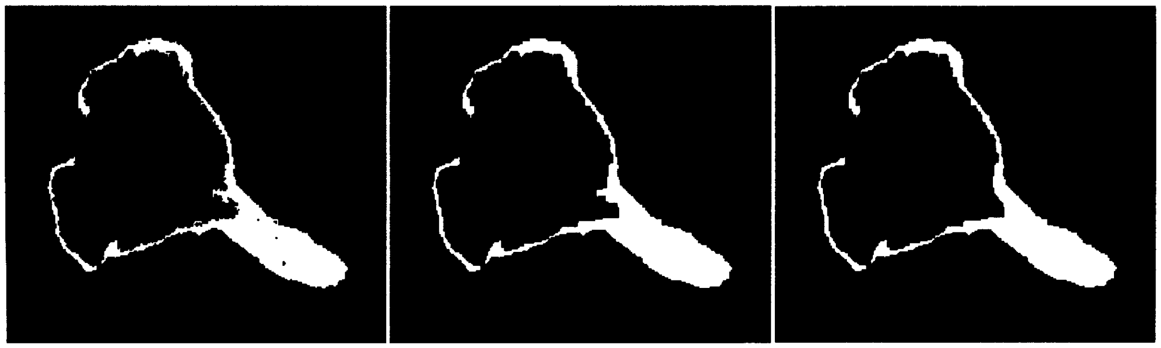

Figure 2-8: Left: Selected objects; Center: Closing of the image; Right: Cleaned image

shown in white (color value 1) against a black background (color value 0). The "dilation" is performed first, by printing a small binary image, called a "structural element," over every white pixel in the original image. In the present case, the structural element is a 5×5 matrix of 1s. For example, the dilation of a binary image containing just two white pixels which are separated by a distance greater than five pixels is a pair of white squares, measuring 5 pixels on a side, and centered on the positions of the original white pixels. The "erosion" occurs by eliminating all white pixels in the dilated image in which the same structural element, when printed over each white pixel, fits entirely within a region of white pixels. The main effect of a closing is to connect objects that are located nearby to one another, and also to fill gaps and holes (Gonzalez 1992). Figure 2-8 contains an example of the original image with selected objects, and the closing of this image. After the closing is applied, the enhanced image is compared to the high-contrast, histogram-adjusted product image (Figure 2-6) to decide if any features remain which do not belong to the fossil. Another script was designed for use as an erasing tool, for eliminating unwanted features and artifacts in order to produce a "cleaned" binary image (shown on the right in Figure 2-8). This script prompts the user to select a polygonal region from the binary image closing, using a mouse, which is then filled with 0s (black). Such kinds of manual editing devices are frequently a part of image processing projects, since minute and artificial features inevitably arise (e.g., see Chapter 6 of Fabbri, 1984).

The second morphological operation that is sometimes used is the thinning of binary objects in order to obtain "skeletons" and skeleton-refined objects. The skeleton of a region R with a border B is defined as follows. For each point p in R, its closest neighbor in B is found. If p has more than one such neighbor, it is said to belong to the skeleton of R (Blum 1967). While this is a useful definition, it is computationally prohibitive to implement. A more efficient means of obtaining skeletons is outlined in Gonzalez (1992), and a simple program for this purpose is included with the MATLAB Image Processing Toolbox. In the present implementation, skeletons which best represent the underlying shape of a fossil wall are selected with a mouse. Remaining skeleton lines are erased

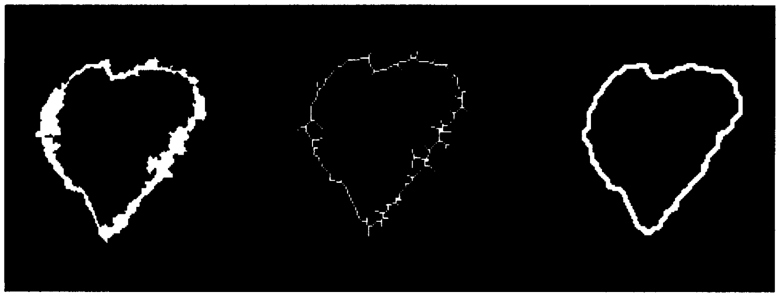

Figure 2-9: Left: Original object; Center: Skeletons; Right: Dilated and with spur pixels repeatedly removed until none remain. A different binary object from that in Figure 2-8 is shown here in order to supply a more dramatic demonstration of the effect, and because obtaining skeleton-refined contours is not recommended in that case (see text for discussion).

by repeated elimination of so-called "spur pixels" until none remain. (Spur pixels are pixels with one neighbor that has the same color.) Then, repeated dilations are performed in order to thicken the skeletons that have been saved. The whole process is illustrated in Figure 2-9, and described in more detail in Appendix C under the subheading "`skeletons.m`" (i.e., the title of the script used for this purpose). Since this process obtains a dramatic idealization of the original image data, it is performed after all of the image contours have been obtained. That is, this technique should not be applied until all of the other procedures discussed in this section have been carried out. Moreover, skeleton-refinements can be performed only in the case of specimens that have a constant, slender wall thickness in every frame, and not for those, like that shown in Figure 2-8, which contain large interior spaces that were filled by calcite. Finally, since this process must be applied to each contour in turn, obtaining skeleton-refined contours can nearly double the time needed to obtain a reconstruction, and hence normally this is not undertaken at all.

Obtaining Contours

The last step in the process is to extract contours from the cleaned binary image. A contour is a list of adjacent coordinate values which, when plotted in order, trace the boundary of a feature in the image in a consistent direction (e.g., clockwise). In the present case, the contours are obtained using a routine in Matlab which finds and tabulates points at the boundary between black and white regions in a binary image. If this routine were not preinstalled, some others are possible and these are easy to construct. In the simplest approach, the first step is to find a white pixel that has at least one adjacent black pixel. In his case "adjacent" means immediately above, below, or to the left or right (and not with a diagonal connection—e.g., displaced by one pixel and "above and to the left"). Then the program checks the 8 neighbors of the first pixel to find a second white pixel with at least one adjacent black pixel. The process records the coordinates of each white pixel in turn, and prohibits the selection of any pixel already in the list. The whole shape has been traced when the first coordinate in the list is reached again. The process is then terminated and the contour is stored. Since the interior is known (i.e., colored in white), it is easy to decide whether the object has been traced in a clockwise or counter-clockwise direction, and the order of the coordinate list may be reversed accordingly if needed.

A simple filtering scheme is used to eliminate contours shorter than a minimum length, which either result from inadequate cleaning of the binary images, or from small enclosed regions within the fossil image which seem not to belong to the fossil itself (e.g., large particles of dust, a registration hole, or other objects photographed on the surface of the sample can sometimes survive in the data until this stage). Then, all that remains is to apply the rotation, position, and scale corrections recorded earlier for the current frame. A contour of length n (i.e., containing n points and describing n connected segments) is stored in a $2 \times n$ matrix, designated D. Let Δ_1 be a $2 \times n$ matrix with the quantity Δx for all the elements in the first row, and Δy for all the elements in the second row, where Δx and Δy are defined as before in equations (2.1) and (2.2). Moreover, let Δ_2 be a $2 \times n$ matrix with the quantity x_{A1} for all the elements in the first row, and y_{A1} for all the elements in the second row, where x_{A1} and y_{A1} are the coordinates of the first registration hole in the first image of the stack, as before. Let R be the rotation matrix for a rotation of ϕ around the point (x_{A1}, y_{A1}). Moreover, let s be the scale factor, defined as before in equation (2.4). The contour D'', which results from the rotation, rescaling, and position adjustment, is obtained as follows:

$$\Delta_1 \equiv \begin{bmatrix} \Delta x & \Delta x & \dots & \Delta x \\ \Delta y & \Delta y & \dots & \Delta y \end{bmatrix} \quad (2 \times n \text{ matrix}) \tag{2.12}$$

$$\Delta_2 \equiv \begin{bmatrix} x_{A1} & x_{A1} & \cdots & x_{A1} \\ y_{A1} & y_{A1} & \cdots & y_{A1} \end{bmatrix} \quad (2 \times n \text{ matrix}) \tag{2.13}$$

$$R \equiv \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \quad (2 \times 2 \text{ matrix}) \tag{2.14}$$

$$D = \text{Matrix containing coordinates of the contour} \quad (2 \times n \text{ matrix}) \tag{2.15}$$

$$D' \equiv D + \Delta_1 - \Delta_2 \tag{2.16}$$

$$D'' = sRD' + \Delta_2 \tag{2.17}$$

The contours that result from the whole process are displayed in Figure 2-10. Notice that there are three contours in all. The entire process is repeated for every frame in the cropped stack, or as many as the user decides are needed. The frames can be processed and their contours obtained in either one of two orders. The frames can be processed in the order in which they appear in the stack, or instead in an order which obtains the entire reconstruction at each stage with increasing vertical resolution—in a piecemeal fashion. This order begins with the first frame in the stack, and

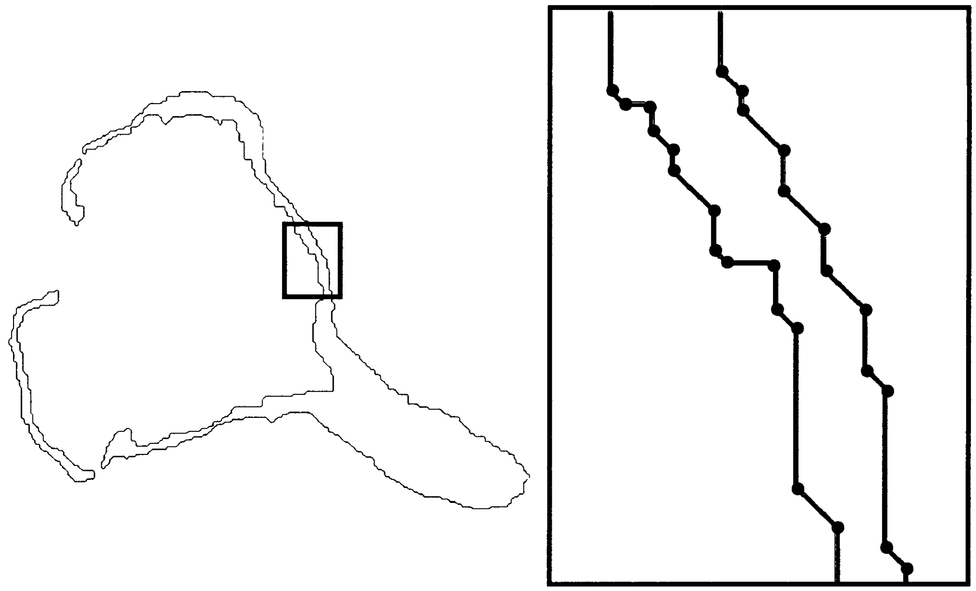

Figure 2-10: Final contours (left) and exploded view (right).

then proceeds directly to the last frame, and then to the middle frame. The next frame in the order is the one closest to 1/3rd of the stack's total height, and this is followed by the frames at 2/3rds, 1/4th, 3/4ths, 1/5th, 2/5ths, 3/5ths, 4/5ths, 1/6th, and so on. In this way, the user can view the entire reconstruction at any time (in the manner described below) and decide to stop processing frames when it is clear that the essential morphology has been obtained (i.e., so that additional resolution is not needed). This kind of judgment is not possible, however, until adequate numbers of specimens have been reconstructed using the full resolution, and once it is clear that they do not possess features that are smaller than the "stopping resolution." The majority of specimens shown in Figure 1-9 were obtained using the full 25 μm resolution.

Once a final set of contours for the entire specimen has been obtained, other scripts are used to display and compare contours from adjacent frames (see the subheadings "`pview.m`" and "`pmover.m`" in Appendix C). If all of the aforementioned corrections to orientation, scale, and position produced a contour that is badly misaligned with respect to adjacent ones, these scripts are used for changing the position incrementally until the alignment is adequate. This will have the effect of reducing slightly the surface roughness of the final reconstruction. These techniques are used sparingly, and only to correct obvious mistakes in alignment.

Slight errors in the final position and orientation of contours result from errors in measuring the positions of registration holes for a given frame. If the contour is badly misaligned, then the registration hole positions for the frame are checked and sometimes measured again. In the case of one sample, our registration holes measured approximately 13 pixels in diameter (total spatial resolution of 7.7×10^{-5} m / pixel). Because the position of each registration hole is carefully measured using a mouse, an error of at most 3 pixels in reckoning the center is expected. (The center of registration holes can be reckoned to within at most three pixels because the image is magnified 6 times in order to view each hole clearly when its position is measured.) This introduces an error of about 0.2 mm in the absolute positions of each frame. (This amounts to about 3.3% of the average size of the Nama specimens (0.6 cm).)

As an additional source of error, differences in the color contrast of the original images can result in subtle dilations in the diameter of contours from different frames. For example, if lighting conditions are not held constant between scans of the sample, then possibly a different threshold value T will be needed to obtain an optimal binary image. As a result, an additional set of intensity values are included or excluded from the optimal binary image of the fossil. Hence, the thickness of features in the binary image, and therefore also the diameter of the final contours, will be larger or smaller with respect to contours in other frames. In such a way, the contour diameters dilate subtly throughout the model, contributing to surface roughness in the final reconstruction.

All contour data is formatted for use with the public domain software *Nuages* and is stored in a so-called "contour file." Nuages is used to obtain a three-dimensional reconstruction from serial contour data using tetrahedrons, and was designed by Bernhard Geiger of INRIA in France.[6] The method used by Nuages is described briefly in the next section. The formatted text file that contains the contour data also contains the elevation of each contour in the model, and therefore also the separation between contours in the third spatial dimension. To determine the appropriate size of this separation one needs the ratio of height to width of the reconstructed surface model. Therefore, this discussion also is postponed until later.

The Geiger-Delaunay Reconstruction Method

This section contains a brief outline of the process used by the program Nuages (designed by Bernhard Geiger) to obtain an overall reconstruction from contour outlines of serial cross sections. Geiger (1993) confirmed the accuracy of his method by comparing the reconstructed model of a geometrical object with its mathematical description. In what follows, this is assumed to be adequate justification for using Geiger's method, and therefore individual steps in the following explanation of his process are not motivated separately. In his discussion, Geiger reduces the problem of finding an overall reconstruction to the problem of computing a solid slice between adjacent cross sections P_1 and P_2 (i.e., frames containing one or multiple contours).

The first step is to calculate the *Voronoi diagram* and *Delaunay triangulation* for points in each contour of P_1 and P_2. Given a set S of N points ($S = \{p_i \in \mathbf{R}^2 \mid i = (1...N)\}$) such that no four

[6]Institut National de recherche en informatique et en automatique

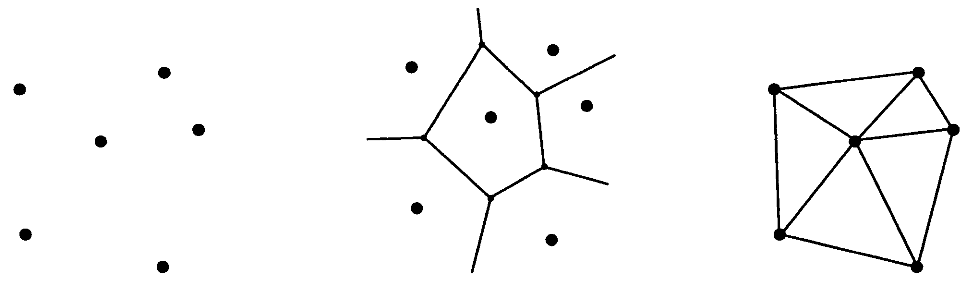

Figure 2-11: Left: A sample set of points S; Center: The Voronoi diagram $V(S)$ for the points S; Right: The Delaunay triangulation of the points S.

points lie on a common circle, let $V(i)$ be the set of points closer to the point p_i than to any other point in S. $V(i)$ is called the *Voronoi cell* associated with p_i. The union $V(S)$ over all $V(i)$ is called the *Voronoi diagram* of S. The Voronoi diagram is a subdivision of the plane into convex polygonal regions, which are the Voronoi cells. The boundaries of these regions are called *Voronoi edges*, and the points at which (always three) Voronoi edges connect are called *Voronoi vertices.* Each Voronoi edge borders just two Voronoi cells. The Voronoi diagram for a set of sample points is shown in Figure 2-11. The *Delaunay triangulation* of S is the set of points along straight lines which connect any two points in S whose Voronoi cells share an edge. These straight lines are called *Delaunay edges.* The Delaunay triangulation for a sample set of points is shown on the right in Figure 2-11.

The Voronoi diagram and Delaunay triangulation is calculated for the set of points in the contours of P_1 and P_2. In the next step, the Voronoi diagrams and Delaunay triangulations are updated after the addition of points to contours and to the interiors of contours in both cross sections. These points are added for several reasons. First, points are added and the triangulation is updated until every segment of every contour is a Delaunay edge. Second, points are added until all obtuse angles are eliminated from the triangulation. Third, portions of the Voronoi diagram that are exterior to the contours in the cross sections adjacent to P_1 and P_2 are projected orthogonally onto P_1 and P_2. Points are added to the contours in P_1 and P_2 along these projected lines, and the triangulation is updated once again. The final Delaunay triangulation for two example contours C_1 and C_2 in P_1 and P_2, respectively, are shown in Figures 2-13 and 2-12, where these contours have been greatly simplified for the purposes of illustration (i.e., these have many fewer points than a typical contour, such as that shown in Figure 2-10).

The reconstruction is assembled in three dimensions from a set of tetrahedral components. That is, tetrahedrons are used to connect points, triangles, and line segments in the final Delaunay triangulations of contours in the adjacent cross-sectional planes P_1 and P_2. For each triangle $t \in P_1$, the point $p \in P_2$ is found which is closest to the center of t. The triangle t is connected with p to form a tetrahedron of type t_1. The process is repeated for triangles in P_2 and points in P_1,

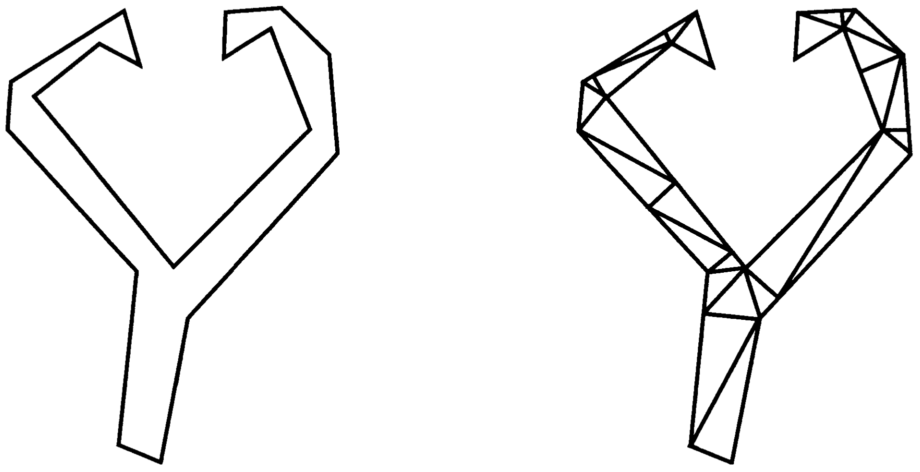

Figure 2-12: Left: A contour C_1 in the plane P_1; Right: Final Delaunay triangulation of the original and added points in the contour C_1.

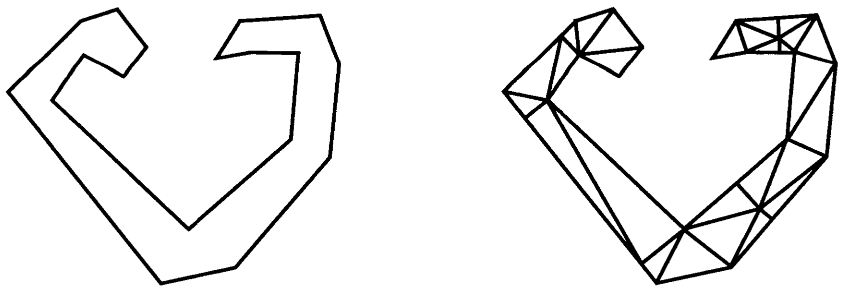

Figure 2-13: Left: A contour C_2 in the plane P_2; Right: Final Delaunay triangulation of the original and added points in the contour C_2.

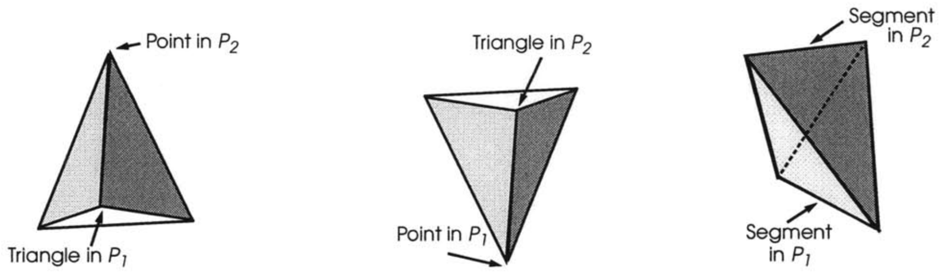

Figure 2-14: Left: t_1 tetrahedron; Center: t_2 tetrahedron; Right: t_{12} tetrahedron.

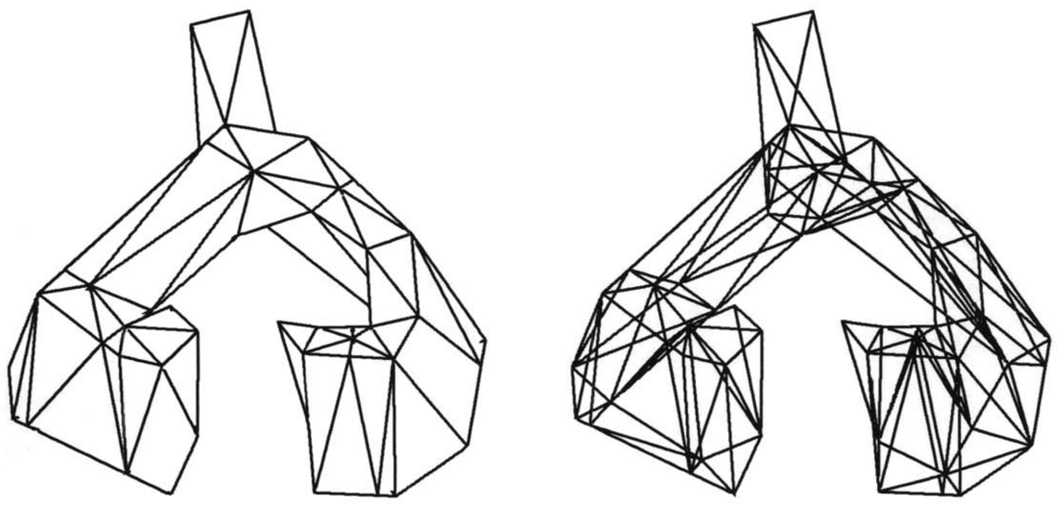

Figure 2-15: Left: Reconstruction based upon contour C_1 in plane P_1 (lower) and contour C_2 in plane P_2 (top), where all unobstructed connecting lines (i.e., sides of tetrahedrons) on the exterior are shown. Right: The same reconstruction except where all connecting lines on the exterior are shown.

forming tetrahedrons of type t_2. A third kind of tetrahedron with just one edge in P_1 and one edge in P_2 is drawn wherever an intersection occurs in the union of Voronoi diagrams for the two planes (i.e., the projection of the Voronoi diagram in P_2 onto that for P_1). That is, wherever two Voronoi edges intersect, the corresponding Delaunay edges are joined by a type t_{12} tetrahedron. All three types of tetrahedrons are shown in Figure 2-14. Some of the "connecting lines"—the sides of the tetrahedrons used to connect the contours C_1 and C_2—are shown in Figure 2-15. The fully-rendered reconstruction with opaque triangular panels is shown in Figure 2-16.

As mentioned, a different vertical position is assigned to each cross section in the contour file before an accurate reconstruction can be assembled. The scale of this vertical position is obtained as follows. First, the user specifies a scale of 1 unit between adjacent contour layers in the contour file, and assembles the reconstruction accordingly. The user then measures the ratio of a known horizontal distance in the reconstructed model (known in terms of pixels and real physical units), to

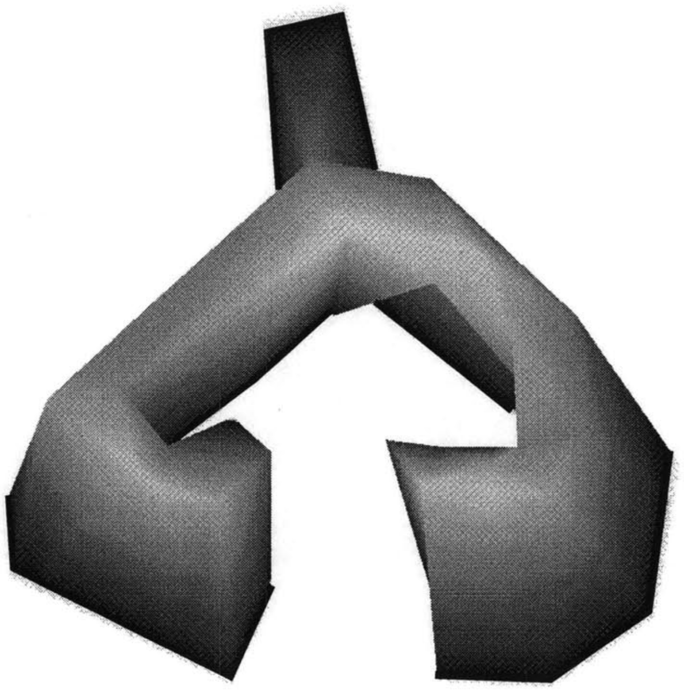

Figure 2-16: Fully-rendered reconstruction based upon contour C_1 in plane P_1 (lower) and contour C_2 in plane P_2 (top).

the total vertical height of the model. Since the vertical height of the specimen is known in terms of physical units (i.e., the number of frames that span the fossil, multiplied by 25.4 μm), the correct vertical scaling in pixels can be obtained.

Conclusions

Digital images of serial cross sections obtained at 25 μm thickness intervals are used to generate a three-dimensional "tomographic" reconstruction of the Nama fossil morphologies. The serial images are obtained by repeated grinding of a rock sample using a surface grinding wheel, and repeated imaging using a digital camera. The images are then manipulated using a battery of image processing techniques in order to obtain binary images of the fossil material in white, and the matrix material in black. Points along the boundary between these regions are tabulated and stored in a "contour file," for use with the software program Nuages. Nuages is used to articulate a three-dimensional reconstruction from tetrahedral components which connect the contours in adjacent cross sections. Examples of the reconstructions are shown in Figures 1-9 and 1-14E.

Chapter 3

A Mathematical Description of the Morphology

The Model and its Global Parameters

This chapter provides a mathematical description of the morphology of *Namacalathus hermanastes* obtained using the techniques described in the previous chapter. A mathematical model should reflect the features common to all of the tomographic reconstructions, with some parameters assigned to characteristics which vary among individual specimens—such as absolute dimension and proportion. The purpose of such a model is to obtain a database of cross sections through random orientations of the morphology (see Appendix B). This database can then be used to identify specimens in the field, which in turn makes possible the measurement of population statistics for fossils in a stack of serial images (see Appendix A). As seen already in Chapter 1, all of the complete or nearly complete reconstructed specimens have a stem and an outward flaring cup. The cup has a broad circular opening at the top with a lip that curves inward. The cup is perforated by six or seven holes (called "windows") of similar size and shape, which often have slight or pronounced inward-curving lips (called "window lips"), and a matching number of regular sides. Cups taper to a shallow base from which a hollow cylindrical stem extends. Stems are commonly longer than the maximum cup dimension and are open at both ends.

The model assumes these universal properties of the morphology, and the characters which are observed to vary among specimens are assigned parameters. The most important assumption is that the overall morphology in its original and undistorted form has a regular symmetry according to the number of windows in the cup. It is possible to construct a complete mathematical description which specifies the shape of any horizontal cross section (i.e., perpendicular to the cup's symmetry axis) at any vertical position long the model's height. The software program Nuages can then be

Table 3.1: Global dimension parameters

Dimension parameter	Description
N	Number of windows in the cup
T	Wall thickness
$r_{\rm mc}$	Maximum cup radius
$h_{\rm mc}$	Total cup height
$h_{\rm ms}$	Total stem height
$h_{\rm ml}$	Vertical span of lip

used to construct a properly rendered three-dimensional realization of the model from a set of cross sections chosen at regular intervals. This process has been described already in the previous chapter, and the final result is shown for two examples in Figure 1-10.

The model is composed of three geometrical objects: (a) the external wall of the cup, (b) the wall of the inward curving lip, and (c) the stem wall. These components are labeled in a cutaway diagram, shown in Figure 3-1. Each wall is made up of an interior and exterior boundary, whose diameters differ by the wall thickness T. The boundaries of each wall are described using just two kinds of two-dimensional vector-valued functions which are defined in the next section, called $\vec{\chi_1}(\psi, z)$ (used to describe the external cup wall and the lip wall) and $\vec{\chi_2}(\psi, z)$ (used to describe the stem wall). Each value of (ψ, z) yields a unique value of (x, y) in the case of each function, where the parameter ψ varies over the entire domain $[0, 2\pi]$. For example, the cross section of the exterior boundary of the cup wall, at an absolute vertical position z, is given by the set of (x, y) values $\vec{\chi_1}([0, 2\pi], z)$. The functions $\vec{\chi_1}$ and $\vec{\chi_2}$ can be used to completely specify an individual model, according to a set of global parameters that can be classified in two categories: (a) parameters which specify the ***absolute*** dimension of objects and features (called "global dimension parameters"), and (b) parameters which specify *relative* position quantities and which are functions of position (called "scale-invariant profiles"). The informal definitions of dimension parameters and profiles are listed in Tables 3.1 and 3.2, respectively. When these parameters are defined, the vector-valued functions $\vec{\chi_1}$ and $\vec{\chi_2}$ are completely determined and can be used to draw an entire model in three dimensions.

In the case of each profile in Table 3.2, a subscript c, l, or s has been assigned according to whether the profile is used to describe some part of the cup, lip, or stem. Accordingly, each profile is a function of the variable z_c, z_l or z_s, which indicates a vertical position along the height of each component. The quantities z_c, z_l, and z_s should be regarded as functional parameters whose significance is component-specific, and which are *not* absolute positions for the fully assembled model. In particular, each profile is defined only on the interval $[0, 1]$, and each bound corresponds to a different position in the fully assembled model. In the case of the inward-curving lip, $z_l = 0$ corresponds to its base, where it is joined to the top of the cup. Therefore, $z_l = 1$ corresponds

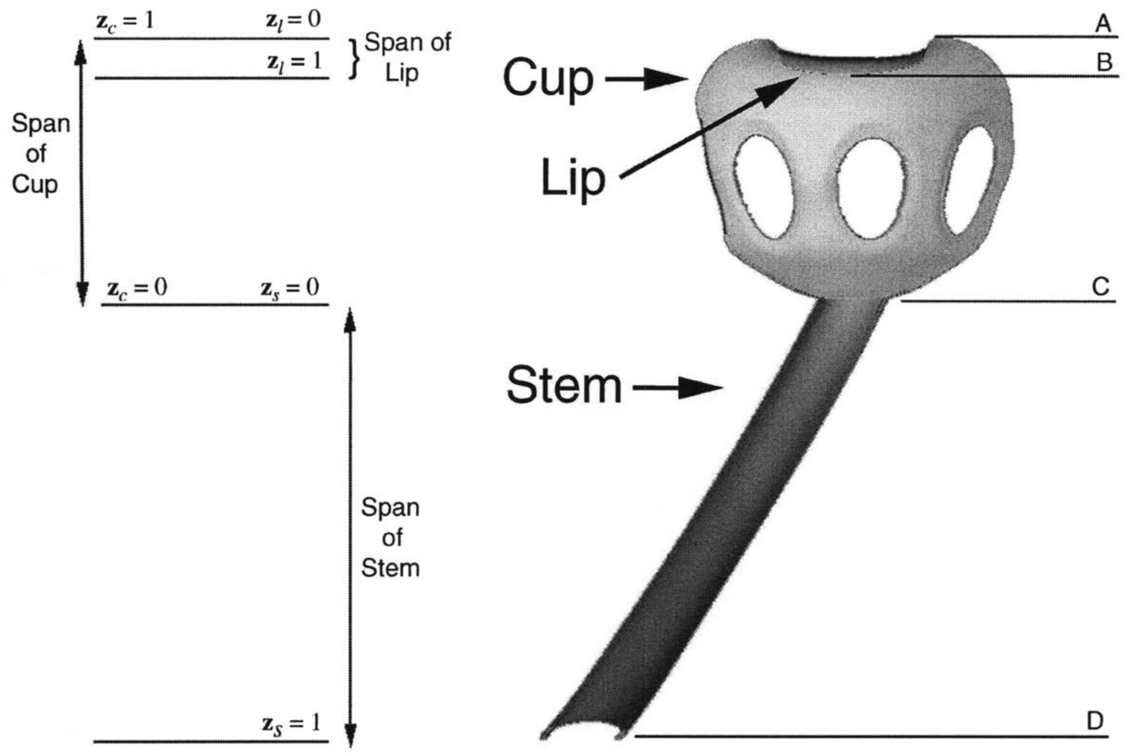

Figure 3-1: Cutaway diagram of the mathematical model, showing each of three components from which it is composed (the cup, lip, and stem). The line A corresponds to the vertical positions $z_c = 1$, $z_l = 0$, and absolute position $z = h_{\text{mc}}$. The line B corresponds to $z_l = 1$ and absolute position $z = h_{\text{mc}} - h_{\text{ml}}$. The line C corresponds to $z_c = 0$, $z_s = 0$, and absolute position $z = 0$. The line D corresponds to $z_s = 1$ and absolute position $z = -h_{\text{ms}}$.

Table 3.2: Global profile parameters

Profile parameter	Description	Range	α
$r_c(z_c)$	Radius of the external cup wall	$[0, 1]$	r_{mc}
$p_c(z_c)$	Window diameter as fraction of side width	$[0, 1]$	1
$l_c(z_c)$	Window lip position as fraction of side width	$[0, 1]$	1
$\gamma_c(z_c)$	Inclination of window lip	$[0, 1]$	1
$k_c(z_c)$	Curvature of the external cup wall	$[0, 1]$	1
$r_l(z_l)$	Radius of the inward-curving lip	$[0, 1]$	r_{mc}
$r_s(z_s)$	Radius of the stem	$[0, 1]$	r_{mc}
$\theta_s(z_s)$	Angle of stem trajectory as fraction of 2π	$[0, 1]$	1
$\phi_s(z_s)$	Angle of stem inclination as fraction of $\pi/2$	$[0, 1)$	1

Table 3.3: Positions corresponding to the bounds of z_c, z_l, and z_s

Position parameter (PP)	Component	PP has value 0	PP has value 1
z_c	External cup wall	Base of cup	Top of cup
z_l	Lip wall	Base of lip (top of cup)	Termination of lip
z_s	Stem wall	Top of stem (base of cup)	Base of stem

to the lip's termination. For the cup, $z_c = 0$ corresponds to the base, and $z_c = 1$ corresponds to the top. Finally, in the case of the stem, $z_s = 0$ corresponds to where it attaches to the cup, and $z_s = 1$ corresponds to its farthest extent from the cup (its base). These position relationships are summarized in Table 3.3 and Figure 3-1. The variables z_c and z_l (cup and lip, respectively) are a measure of position along the symmetry axis, whereas z_s is merely a measure of "altitude" in the case of the stem.

An individual model can be specified in terms of a set of scale-invariant profiles, and a list of absolute dimension parameters (see Tables 3.1 and 3.2). The magnitude of the scale-invariant profiles range from 0 to 1. For example, the cup's radial profile $r_c(z_c)$ is defined so that its maximum value is 1, which corresponds to its maximum radius. Other profile magnitudes can have maxima that are less than or equal to 1. Their minima are likewise greater than or equal to 0. The purpose of specifying profile magnitudes and the profile position parameters $\{z_c, z_l, z_s\}$ within a range from 0 to 1 (i.e., in a scale-invariant form) is so that many features of the model can be easily scaled according to the global dimension parameters given in Table 3.1. Moreover, scale-invariant profiles are very easy to express, visualize, and compare. In effect, each profile is expressed as a fraction of some quantity which depends upon the global dimension parameters (Table 3.1), especially the number of windows N, and the maximum cup radius r_{mc}. For example, the radial profile of the cup in its ***absolute*** dimensions is given by $r_{\text{mc}} r_c(z/h_{\text{mc}})$, where h_{mc} is the cup's height, and z is the absolute vertical position coordinate.

Whereas scale-invariant profiles and global dimension parameters are a convenient way of specifying individual models, to obtain the complete mathematical description in three absolute dimensions, an additional step is needed. In particular, the two-dimensional vector-valued functions $\vec{\chi_1}$ and $\vec{\chi_2}$ are fully determined and properly scaled in three absolute dimensions when all dimension parameters and profiles are defined, and when the profiles are properly scaled. That is, every scale-invariant profile $f_i(z_i)$ where $i \in \{c, s, l\}$ must be properly scaled to obtain the form $f_i^*(z)$, where z_i has undergone a simple change of variables. The proper scaling is given by:

$$f_i^*(z) = \alpha f_i(z), \tag{3.1}$$

$$\text{where} \quad z = z_c h_{\text{mc}} \quad \text{for cup profiles,} \tag{3.2}$$

$$z = h_{\mathrm{mc}} - z_l h_{\mathrm{ml}} \quad \text{for lip profiles,} \tag{3.3}$$

$$z = -z_s h_{\mathrm{ms}} \quad \text{for stem profiles,} \tag{3.4}$$

where the value α for each profile is supplied in Table 3.2, and where z is the absolute vertical position. In most cases, α is equal to r_{mc}, or simply 1 if the absolute magnitude depends upon a combination of other parameters. For example, the window diameter $p_c(z_c)$ is expressed as a fraction of the size of one of the cup's N sides, which in turn depends upon N and r_{mc}.

The cup, lip, and stem can be assembled seamlessly so long as the top of the stem has the same radius as the basal aperture of the cup, and so long as the base of the inward-curving lip is smaller than the radius at the top of the external cup wall. Namely, the following constraints must be respected:

$$r_c(0) = r_s(0) \tag{3.5}$$

$$r_l(0) \leq r_c(1) \tag{3.6}$$

In summary, the entire model can be constructed in a three-dimensional Cartesian space with absolute position variables x, y, and z. As mentioned, the interior and exterior boundaries of the three walls are described by the two-dimensional vector-valued functions $\vec{\chi_1}(\psi, z)$ (for the lip and external cup walls) and $\vec{\chi_2}(\psi, z)$ (for the stem walls), which determine a unique value of (x, y) for each vertical position z and each value of the parameter ψ in the domain $[0, 2\pi]$. According to the change-of-variables outlined in equations (3.2)-(3.4) and the component-specific position scheme supplied in Table 3.3, the cup's base and top of the stem correspond to $z = 0$. The top of the cup is at a position z $= h_{\mathrm{mc}}$, the lowest point of the lip is at $z = h_{\mathrm{mc}} - h_{\mathrm{ml}}$, and the base of the stem is located at $z = -h_{\mathrm{ms}}$ (see Figure 3-1). The following section provides the definitions of the vector-valued functions $\vec{\chi_1}(\psi, z)$ and $\vec{\chi_2}(\psi, z)$, and the last section provides the dimension parameters and scale-invariant profiles for an individual model, which is based upon the dimension and shape of a tomographic reconstruction.

Mathematical Description of the Fossil Walls

This section provides the definitions of the two-dimensional vector-valued functions $\vec{\chi_1}(\psi, z)$ and $\vec{\chi_2}(\psi, z)$. The first of these is used to describe the interior and exterior boundaries of the external cup wall and the inward-curving lip wall, whereas the second is used to describe the interior and exterior boundaries of the stem wall. Recall that the difference between the interior and exterior boundaries of a wall is that the radial profile of the interior boundary is shorter by a value $T/2$, where T is a global dimension parameter (see Table 3.1). Both functions describe an entire boundary cross section for a given value of z where the parameter ψ ranges over the whole interval $[0, 2\pi]$.

Both functions are completely determined when the entire compliment of dimension parameters and profiles are defined for a given component. In the following discussion, the subscripts are dropped from the labels of each profile, and profiles are assumed to be scaled in the manner specified by equations (3.1)-(3.4). The scale-invariant forms of profiles will be used to specify an individual model in the next section.

The function $\vec{\chi_1}(\psi, z)$ is constructed from several simpler functions. Two of these are defined in equations (3.7) and (3.8), where $\{u, v, w\}$ are dummy variables, and N is the number of sides and windows in the cup.

$$\begin{aligned} \zeta(u,v) &\equiv \pi\left(\frac{1}{2} + \frac{2u(1-v)}{N}\right), && (3.7)\\ f(u,v,w) &\equiv \left[\sin(\zeta(u,v)) + w\cos(\zeta(u,v))\right]^{-1}, && (3.8)\\ & && (3.9) \end{aligned}$$

The following function $g(\beta, z)$ represents the shape of a cup or lip wall boundary across a fraction $1/N$ of the total cup circumference, where β is a parameter that ranges from the left-hand-side ($\beta = 0$) to the right-hand-side ($\beta = 1$) of this "slice". Since a regular symmetry based upon N regular windows has been assumed, each of the N sides will have the same form.

$$g(\beta, z) \equiv \begin{cases} r\eta f(\beta, k, 0) & \text{for } 0 \leq \beta \leq \nu^- - l, \\ r f(\frac{1}{2} - \beta, 0, \gamma) + \delta & \text{for } \nu^- - l \leq \beta < \nu^-, \\ 0 & \text{for } \nu^- \leq \beta < \nu^+, \\ r f(\beta - \frac{1}{2}, 0, \gamma) + \delta & \text{for } \nu^+ \leq \beta < \nu^+ + l, \\ r\eta f(1 - \beta, k, 0) & \text{for } \nu^+ + l \leq \beta \leq 1, \end{cases} \qquad (3.10)$$

$$\begin{aligned} \text{where } \nu^{\pm} &\equiv \frac{1}{2}(1 \pm p), && (3.11)\\ \eta &\equiv \sin(\zeta(0,0)) && (3.12)\\ \delta &\equiv r[\eta f(\nu^- - l, k, 0) - f((1/2) - \nu^- + l, 0, \gamma)], && (3.13)\\ & && (3.14) \end{aligned}$$

and where γ, r, p, k, and l are profiles with the informal definitions given by Table 3.2. (Here, the subscripts and functional dependence upon z have been dropped, but recall that each profile is a function of z, so that $g(\beta, z)$ is also a function of z.) In practice, the function $g(\beta, z)$ is sampled at regular intervals in β, and then a spline curve is fitted to these points in order to smooth the corners. Figure 3-2 contains several spline-smoothed plots of $g(\beta, z)$ in Cartesian coordinates, where β is the x coordinate (representing an angle) and $g(\beta, z)$ the y coordinate (representing a radius)

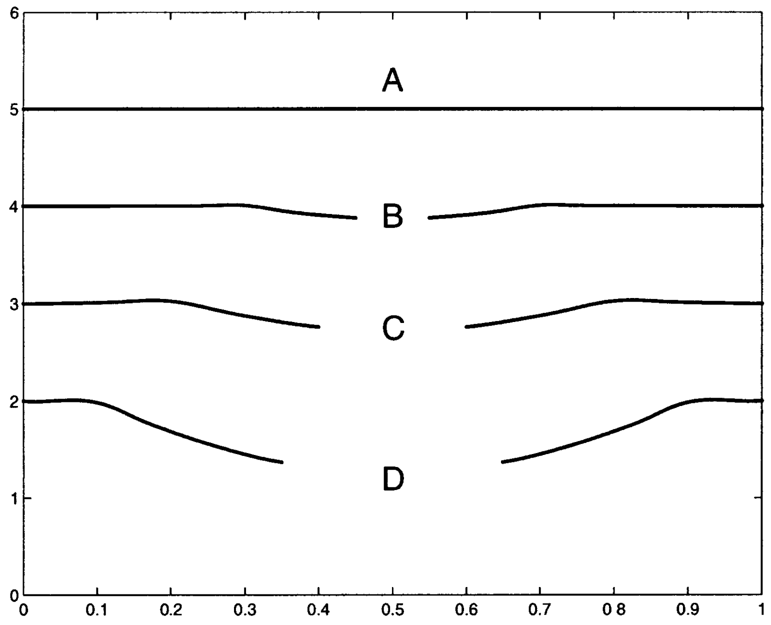

Figure 3-2: Spline-smoothed plots of $g(\beta, z)$ (equation (3.10)) in Cartesian coordinates, for constant z and several values of the parameters γ, r, l, p, and k. In particular, in order from lines A to D, these values are: $\gamma = \{0, 0.1, 0.25, 0.6\}$, $r = \{5, 4, 3, 2\}$, $l = \{0, 0.15, 0.20, 0.25\}$, $p = \{0, 0.1, 0.2, 0.3\}$, and $k = \{1.0, 0.8, 0.2, 0\}$.

for different values of the profile parameters. Several trends are readily apparent. In particular, the window lip inclination γ is flat for $\gamma = 0$ (the line A), and becomes relatively steep for $\gamma = 0.6$ (the lines D). Trends in window diameter, window lip position, and radius are also clearly visible in this figure.

The equation for $g(\psi, z)$ captures all of the features observed in horizontal cross sections of the tomographic reconstructions (i.e., cross sections that are perpendicular with respect to the symmetry axis). In particular, it is capable of modeling smoothly the transition between horizontal cross sections that have a circular shape at the top of the cup, and those with N lipped windows and sides that have a flat curvature. In the case of wall boundaries for the large inward-curving lip, all horizontal cross sections are circular. The complete boundary in horizontal cross section is described by the function $h(\psi, z)$ in polar coordinates, where h is the radial coordinate and ψ the angular coordinate:

$$h(\psi, z) \equiv g(N\psi/2\pi - n) \quad \text{where} \quad \frac{2\pi n}{N} \leq \psi < \frac{2\pi(n+1)}{N}, \tag{3.15}$$

and where $n = \{0, 1, ..., N-1\}$. Figure 3-3 contains several spline-smoothed plots of $h(\psi, z)$ for different values of the profile parameters. In these plots the trend in wall curvature k is more easily recognized. In particular, the wall curvature is circular for $k = 1$ (the line A), and becomes

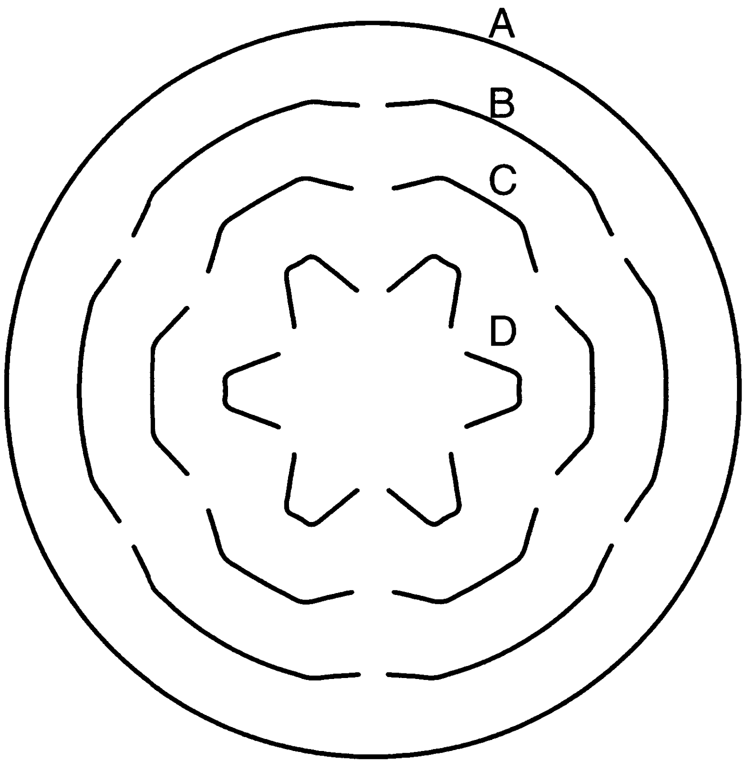

Figure 3-3: Spline-smoothed plots of equation (3.15) (cup- and lip cross sections) in polar coordinates, for constant z and the same values of the parameters γ, k, r, l, and p for the lines A, B, C, and D as shown in Figure 3-2.

approximately straight as k approaches 0 (the broken lines C and D). Equation (3.15) can be cast in terms of Cartesian (x, y) coordinates, in the form of a two-dimensional vector-valued function $\vec{\chi_1}(\psi, z)$ with the parameter ψ ranging over the whole domain $[0, 2\pi]$:

$$\vec{\chi_1}(\psi, z) \equiv \begin{pmatrix} h(\psi, z)\cos\psi \\ h(\psi, z)\sin\psi \end{pmatrix}. \tag{3.16}$$

By comparison, a description of the interior and exterior boundaries of the stem wall is more easily constructed. Recall from the systematic description in Chapter 1 that the stem is a curving circular cylinder. Therefore, all oblique cross sections of the stem, with the exception of those which intersect an opening, have the shape of an ellipse. If the stem were aligned with the cup's symmetry axis, its horizontal cross sections would be circular. In a realistic model, however, the stem has a more interesting trajectory in three dimensions, and therefore its "horizontal" cross sections (i.e., parallel to the cross sections described by $\vec{\chi_1}(\psi, z)$) can be described by ellipses. (The model forbids a trajectory that is bent so far that the circular basal opening of the stem is intersected by a horizontal cross section.) Every cross section is described by an ellipse with semi-major axis length $A(z)$ and semi-minor axis length $B(z)$. The x and y coordinates are thus given by the following set of equations, in terms of a parameter ψ:

$$\begin{aligned} x(\psi, z) &= A(z)\cos\psi, && (3.17) \\ y(\psi, z) &= B(z)\sin\psi. && (3.18) \end{aligned}$$

By inspection of the geometry, the values $A(z)$ and $B(z)$ bear the following relationship to z:

$$\begin{aligned} A(z) &\equiv \frac{r(z)}{\sin(\frac{\pi}{2}(1-\phi(z)))}, && (3.19) \\ B(z) &\equiv r(z), && (3.20) \end{aligned}$$

where $r(z)$ and $\phi(z)$ are the radial and inclination profile parameters, and where the subscripts have been dropped (recall from Table 3.2 that ϕ ranges over the interval $[0, 1)$, so that a value of $\pi/2$ in the argument to (3.19) is not allowed). The stem's trajectory also has an orientation in the (x, y) plane for each value of z, given by the profile parameter $\theta(z)$. This quantity represents an angle measured in a counter-clockwise direction with respect to the line $y = 0$. The value $\theta(z)$ ranges over the interval $[0, 1]$, which translates to $[0, 2\pi]$ in the argument to the rotation matrix $R(\theta(z))$:

$$R(\theta(z)) \equiv \begin{bmatrix} \cos(2\pi\theta(z)) & -\sin(2\pi\theta(z)) \\ \sin(2\pi\theta(z)) & \cos(2\pi\theta(z)) \end{bmatrix} \tag{3.21}$$

Now, let $\Delta x(z)$ be the displacement along the x-axis from the center of the (x, y) plane to the center of a stem cross section at vertical position z, defined as follows (where v is a dummy variable):

$$\Delta x(z) = \int_0^z \frac{\cos(2\pi\theta(v))dv}{\tan(\frac{\pi}{2}(1-\phi(v)))}. \tag{3.22}$$

(Equation (3.22) is obtained from inspection of the stem cross section geometry.) However, if $\cos(2\pi\theta(z))/\tan(\frac{\pi}{2}(1-\phi(v)))$ is not easily integrated, the following discrete form works also, where $z = n\Delta z$ for n horizontal cross sections separated by a distance Δz, between the vertical positions 0 and z:

$$\Delta x(n\Delta z) = \sum_{j=1}^{n} \frac{\cos(2\pi\theta(j\Delta z))\Delta z}{\tan(\frac{\pi}{2}(1-\phi(j\Delta z)))} \tag{3.23}$$

The displacement $\Delta y(z)$ in the y direction is also given by equations (3.22) and (3.23), except with a sine function in place of the cosine in the numerator.[1] Finally, the two-dimensional vector-valued function $\vec{\chi_2}(\psi, z)$ describes the entire stem boundary cross section, where ψ is a parameter that

[1]In the author's implementation, a simplified version of equation (3.23) was used.

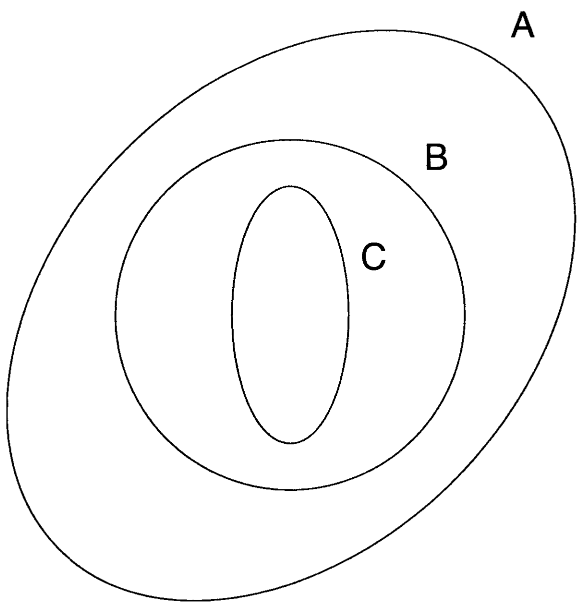

Figure 3-4: Plots of equation (3.24) (stem cross sections) for different profile parameter values. In particular, for the ellipse labeled A the values are: $(r, \phi, \theta) = (4, 0.5, 0.25)$. For the ellipse labeled B the values are: $(r, \phi, \theta) = (3, 1.0, 0.25)$, and in the case of C: $(r, \phi, \theta) = (1, 0.7, 0.5)$. Recall that r represents the stem radius, θ is the planar trajectory angle, and ϕ is the inclination.

ranges over the whole interval $[0, 2\pi]$.

$$\vec{\chi_2}(\psi, z) \equiv R(\theta(z)) \begin{pmatrix} x(\psi, z) \\ y(\psi, z) \end{pmatrix} + \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}. \tag{3.24}$$

Plots of equation (3.24) are shown in Figure (3-4) for three sets of values of (ϕ, θ, r).

In summary, the two-dimensional vector-valued function $\vec{\chi_1}(\psi, z)$ given by equation (3.16) provides a complete description of the interior and exterior boundaries of the external cup wall and the inward-curving lip wall, and is fully determined by the profile parameters $\gamma(z)$, $r(z)$, $p(z)$, $k(z)$, and $l(z)$. The function $\vec{\chi_2}(\psi, z)$ provides a complete description of the interior and exterior boundaries of the stem wall, and is fully determined by the profile parameters $\phi(z)$, $\theta(z)$, and $r(z)$. In the case of both functions, ψ is a parameter that ranges over the whole interval $[0, 2\pi]$.

Specification of an Individual Model

This section provides the global dimension parameters and scale-invariant profiles for an individual model, shown in Figure 1-10D, that is based upon the tomographic reconstruction shown in Figure 1-9I. Values of the global dimension parameters for this model (called "Model A") are listed in Table 3.4.

Table 3.4: Global dimension parameter values for Model A.

N	7
T	0.10
r_{mc}	6.00
h_{mc}	4.84
h_{ms}	10.20
h_{ml}	0.44

In what follows, the scale-invariant profiles for the external cup wall are constructed first. The cup's windows are approximately oval in shape, and therefore the window diameter profile $p_c(z_c)$ is given the following analytic form:

$$p_c(z_c) = \begin{cases} 0 & \text{for } 0 \le z_c < a_0 \text{ and } b_0 < z_c \le 1 \\ c_0[1-(1-2x)^2]^{1/2} & \text{otherwise,} \\ & \text{where } x \equiv (z_c - a_0)/(b_0 - a_0), \end{cases} \tag{3.25}$$

where a_0 and b_0 represent the lower and upper extent of the windows as a fraction of the cup's total height, and c_0 represents the largest diameter as a fraction of the width of one side. These values are $a_0 = 0.25$, $b_0 = 0.70$, and $c_0 = 0.6$ for the tomographic reconstruction shown in Figure 1-9I. (These and many of the other values mentioned in this section are easily measured from a projection of the surface of a tomographic model onto a nearby plane, as shown in Figures 1-10A and C.) A plot of equation (3.25) is shown in Figure 3-5. A spline curve is fitted to the radial profile of the external cup wall in a cross section of the tomographic model, shown in Figure 1-10C. This profile is then rescaled to fit inside the square determined by $[0,1] \times [0,1]$ to obtain $r_c(z_c)$. That is, $\max(r_c(z_c)) = 1$ as z_c varies over the interval $[0,1]$. This profile is shown in Figure 3-5. The window lip position profile $l_c(z_c)$ must mimic the shape of equation (3.25) if the lip size is to remain constant. This is in fact what is observed in tomographic reconstructions, and therefore the following functional form is adopted:

$$l_c(z_c) = \begin{cases} 0 & \text{for } 0 \le z_c < a_0 - \Delta l_v \text{ and } b_0 + \Delta l_v < z_c \le 1 \\ (c_0 + \Delta l_h)[1-(1-2x)^2]^{1/2} & \text{otherwise,} \\ & \text{where } x \equiv (z_c - a_0 + \Delta l_v)/(b_0 + 2\Delta l_v - a_0), \end{cases} \tag{3.26}$$

where the values $\Delta l_v = 0.07$ and $\Delta l_h = 0.2$ are typical for tomographic models. (Notice that Δl_v is the window lip size as a fraction of the cup's height, whereas Δl_h is the window lip size as a fraction of the total width of one side of the cup). The inclination of window lips $\gamma_c(z_c)$ is observed to reach a maximum near the window's middle height. Hence, the following analytic form is a decent

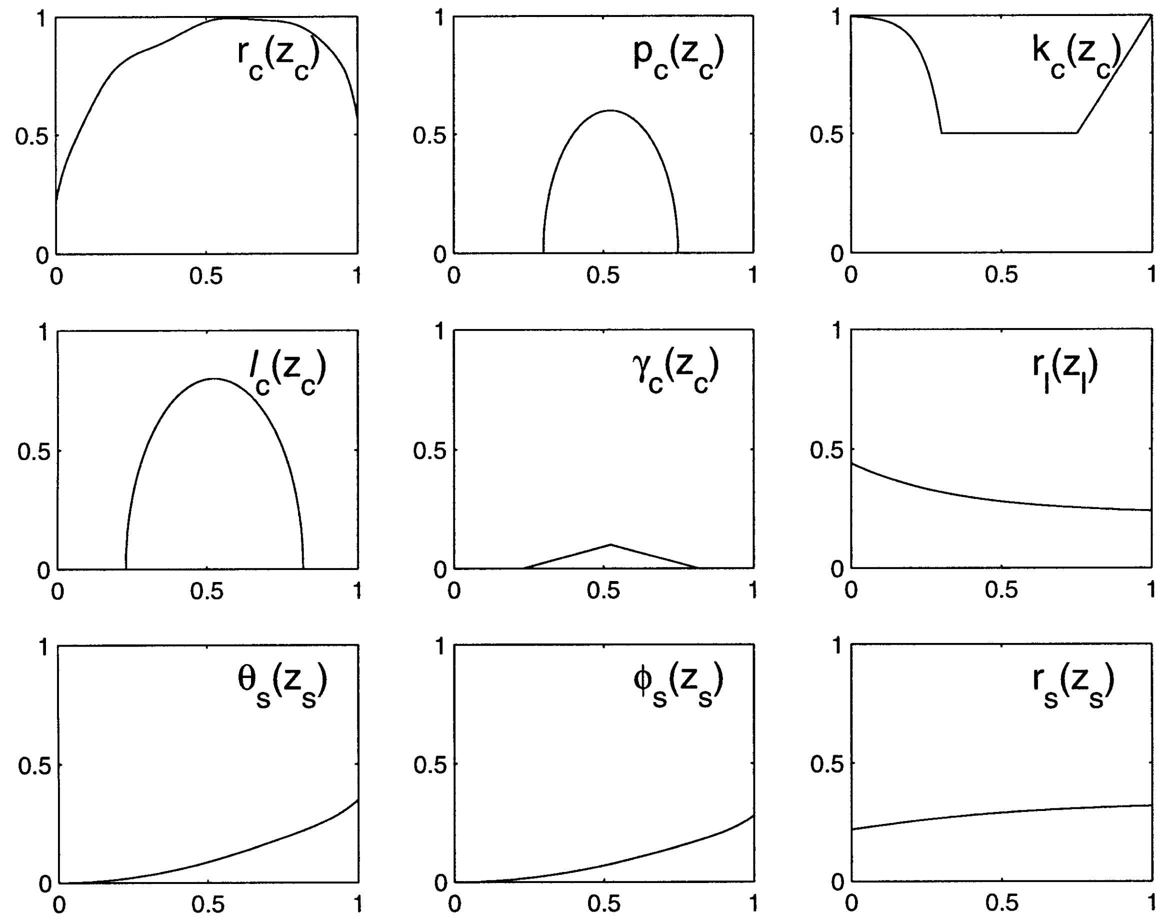

Figure 3-5: Plots of scale-invariant parameter-profiles $\gamma_c(z_c)$, $k_c(z_c)$, $r_c(z_c)$, $l_c(z_c)$, $p_c(z_c)$, $r_l(z_l)$, $\phi_s(z_s)$, $\theta_s(z_s)$ and $r_s(z_s)$ for the Model A.

approximation to what is observed in tomographic reconstructions (where x is defined as in (3.26)):

$$\gamma_c(z_c) = \begin{cases} 0 & \text{for } 0 \leq z_c < a_0 - \Delta l_v \text{ and } b_0 + \Delta l_v < z_c \leq 1 \\ 2\gamma_{\max} x & \text{for } a_0 - \Delta l_v \leq z_c < (b_0 - a_0)/2, \\ 2\gamma_{\max}(1-x) & \text{for } (b_0 - a_0)/2 < z_c \leq b_0 + \Delta l_v, \\ & \text{where } x \equiv (z_c - a_0 + \Delta l_v)/(b_0 + 2\Delta l_v - a_0), \end{cases} \tag{3.27}$$

A typical value for $\gamma_{\max}$ is 0.1. Equation (3.27) is plotted also in Figure 3-5. The curvature of the N sides, $k_c(z_c)$, is circular at the top of the cup ($k_c(1) = 1$), becomes *nearly* flat where the windows appear ($k_c(a_0)) = 0.5$), and gradually becomes circular again at the base ($k_c(0) = 1$). It should be noted that in other tomographic reconstructions, such as the one shown in Figure 1-9B, the curvature becomes completely flat at the cup's middle height ($k_c = 0$). Figure 1-8 shows cross sections through vertically-oriented fossils whose side walls are flat (C) and round (D). As mentioned, for the present model this minimum curvature $k_{\min}$ is set equal to 0.5. An analytic description for $k_c(z_c)$ which reflects this behavior is given by the following equation:

$$k_c(z_c) = \begin{cases} 1-(1-k_{\min})e^{5(z_c/a_2-1)} & \text{for } 0 \leq z_c < a_2 \\ k_{\min} & \text{for } a_2 \leq z_c < b_2, \\ \left(\frac{1-k_{\min}}{1-b_2}\right)(z_c-b_2)+k_{\min} & \text{for } b_2 \leq z \leq 1 \end{cases} \tag{3.28}$$

It is found that the values of a_2 and b_2 are approximately equivalent to a_0 and b_0, respectively (i.e., the minimum curvature is maintained throughout the region in z_c that is occupied by the windows).

A spline curve is fitted to the inward curving lip of the tomographic model in cross section (see Figure 1-10C) and rescaled to obtain the radial profile $r_l(z_l)$. It is easy to see from the plots of $r_l(z_l)$ and $r_c(z_c)$ in Figure 3-5 that the inequality (3.6) is respected. Finally, the scale-invariant stem profiles $r_s(z_s)$, $\phi_s(z_s)$, and $\theta_s(z_s)$ given below represent a rough approximation to the radial profile and trajectory of the stem in the tomographic model. These are given by equations (3.29)-(3.31) and are plotted in Figure 3-5. It is clear by inspection that the condition stated in equation (3.5) is satisfied.

$$r_s(z_s) = r_c(0)+\frac{\arctan(2z_s)}{\arctan(z_s)} \tag{3.29}$$

$$\phi_s(z_s) = \frac{2}{5}(z_s^{10}-z_s^9+\frac{7}{10}z_s^2) \tag{3.30}$$

$$\theta_s(z_s) = \frac{1}{2}(z_s^{10}-z_s^9+\frac{7}{10}z_s^2) \tag{3.31}$$

Conclusions

A mathematical description of the morphology of *Namacalathus hermanastes* is constructed in a three-dimensional Cartesian space with absolute position variables x, y, and z. The general model obtained by this description is composed of three objects: the external cup wall, the stem wall, and the wall of the inward-turning lip that lines the cup's largest circular opening. The interior and exterior boundaries of the fossil walls are described by the two-dimensional vector-valued functions $\vec{\chi_1}(\psi, z)$ (for the lip and external cup walls) and $\vec{\chi_2}(\psi, z)$ (for the stem walls), which determine a unique value of (x, y) for each vertical position z and each value of the parameter ψ in the domain $[0, 2\pi]$. The configuration of walls for an individual model is specified by a set of scale-invariant profile parameters and global dimension parameters. The profile parameters specify the change with vertical position of several morphological characters observed to vary among individual specimens of *Namacalathus*, while the global dimension parameters specify the absolute dimensions of the features of an individual model. Parameter values used to specify an individual model can be measured from an individual tomographic reconstruction. When all the parameters are defined, the functions $\vec{\chi_1}(\psi, z)$ and $\vec{\chi_2}(\psi, z)$ are completely determined and can be used to draw the horizontal cross section for any value of the vertical position z. Horizontal cross sections can then be obtained at regular intervals in z, and a three-dimensional reconstruction can be articulated using Nuages in the manner described in Chapter 2.

References

1. Aitken, J.D. 1967. Classification and environmental significance of cryptalgal limestones and dolomites, with illustrations from the Cambrian and Ordovician of southwestern

2. Alberta. Journal of Sedimentary Petrology 37: 1163-1178.

3. Agterberg, F. P., and Fabbri, A.G., 1978, Spatial Correlation of Stratigraphic Units Quantified from Geological Maps, *Comput. Geosci.* 4:285-294.

4. Ax, P. 1989. Basic phylogenetic systematization of the Metazoa, Pp. 229-245 in B.

5. Bartley, J. K., M. Pope, A. H. Knoll, M. A. Semikhatov, and P. Yu. Petrov. 1998. A Vendian-Cambrian boundary succession from the northwestern margin of the Siberian Platform: Stratigraphy, palaeontology, chemostratigraphy and correlation. Geological Magazine 135: 473-494.

6. Bengtson, S. 1994. The advent of animal skeletons. Pp. 412-425 in S. Bengtson, ed. Early Life on Earth. Columbia University Press, New York.

7. Bengtson, S., S. Conway-Morris, S., B. J. Cooper, P. A. Pell, and B. Runnegar, B. 1990. Early Cambrian fossils from South Australia. Memoirs of the Association of Australasian Palaeontologists 9: 1-364..

8. Bengtson, S. and Y. Zhao. 1992. Predatorial borings in late Precambrian mineralized exoskeletons. Science 257: 367-369.

9. Blum, H. 1967. "A Transformation for Extracting New Descriptors of Shape." in Models for the Perception of Speech and Visual Form, Wathen-Dunn, W., ed., MIT Press, Cambridge, Mass.

10. Bova, J. P. and J. F. Read. 1987. Incipiently drowned facies within a cyclic peritidal ramp sequence, Early Ordovician Chepultepec interval, Virginia Appalachians. Geological Society of America Bulletin 98: 714-727.

11. Brasier, M.D. 1975. An outline history of seagrass communities. Palaeontology 18: 681-702.

12. Brasier, M., O. Green, and G. Shields. 1997. Ediacaran sponge spicule clusters from southwestern Mongolia and the origins of the Cambrian fauna. Geology 25: 303-306. Brusca, R.C., and G. J. Brusca. 1990. Invertebrates. Sinauer Associates, Sunderland MA, 922 pp.

13. Chen Menge, Zongzheng Xiao, and Xunlai Yuan. 1994. A new assemblage of megafossils Miaohe biota from the upper Sinian Doushantuo Formation, Yangtze Gorges. Acta Palaeontologica Sinica 33: 391-403.

14. Cecile, M. P. and F. H. A. Campbell. 1978. Regressive stromatolite reefs and associated facies, Middle Goulburn group (Lower Proterozoic), in Kilohigok Basin, N.W.T.: An example of environmental control of stromatolite form. Bulletin of Canadian Petroleum Geology 26: 237-267.

15. Conway Morris, S., B. W. Mattes, and M. Chen. 1990. The early skeletal organism *Cloudina*: new occurrences from Oman and possibly China. American Journal of Science 290-A: 245-260.

16. Conway-Morris, S., 1998. The Crucible of Creation. The Burgess Shale and Rise of Animals. Oxford University Press, Oxford, 242 pp.

17. Crimes, P. T. and G. J. B. Germs. 1982. Trace fossils from the Nama Group (Precambrian-Cambrian) of Southwest Africa. Journal of Paleontology 56: 890-907.

18. Debrenne, F., J. Lafuste, and A. Zhuravlev. 1990. Coralomorphes et spongiomorphs l'aube du Cambrien. Bulletin, Museum National Histoire Naturelle, Paris, 4th Series, Section C, 12: 17-39.

19. Eernisse, D. J., J. S. Albert, and F. E. Anderson. 1992. Annelida, and arthropoda are not sister taxa: a phylogenetic analysis of spiralian metazoan morphology. Systematic Biology 41: 305-330.

20. Fabbri, A. G. 1984. "Image Processing of Geological Data," Van Nostrand Reinhold Company.

21. Fauvel, P., 1923. Bulletin, Societe Zoologique du France, 47: 424.

22. Fedonkin, M. A. 1990. Systematic description of the Vendian metazoa. Pp. 71-120 in B. S. Sokolov, and A. B. Iwanowski, eds. The Vendian System, Volume 1. Springer- Verlag, Berlin.

23. Feldmann, M. and J. McKenzie. 1998. Stromatolite-thrombolite associations in a modern environment, Lee Stocking Island, Bahamas. Palaios 13: 201-212.

24. Fernholm, K. Bremer, and H. Jornvall, eds. The Hierarchy of Life. Elsevier, Amsterdam.

25. Fritsch, F. E. 1965. The Structure and Reproduction of the Algae, Volume I. Cambridge University Press, Cambridge, 791 pp.

26. Gaucher, C. and P. Sprechmann. 1999. Upper Vendian skeletal fauna of the Arroyo del Solidado Group, Uruguay. Beringeria 23: 55-91.

27. Gehling, J. and J. K. Rigby. 1996. Long expected sponges from the Neoproterozoic Ediacaran fauna of South Australia. Journal of Paleontology 70: 185-195.

28. Geiger, B., 1993, Three-dimensional modeling of human organs and its application to diagnosis and surgical planning. PhD Thesis, Ecole des Mines.

29. Germs, G. J. B. 1972a. New shelly fossils from the Nama Group, South West Africa. American Journal of Science 272: 752-761.

30. Germs, G. J. B, 1972b. The stratigraphy and paleontology of the lower Nama Group, South West Africa. Bulletin, Precambrian Research Unit 12: 1-250.

31. Germs, G. J. B. 1972c. Trace fossils from the Nama Group, South West Africa. Journal of Paleontology 46: 864-870.

32. Germs, G. J. B. 1974. The Nama Group in South West Africa and its relationship to the Pan African Geosyncline. Journal of Geology 82: 301-317.

33. Germs, G. J. B. 1983. Implications of a sedimentary facies and depositional environmental analysis of the Nama Group in South West Africa/Namibia. Pp. 89-114 in R.M.

34. Miller, ed. Evolution of the Damara Orogen. Geological Society of South Africa, Special Publication 11.

35. Germs, G. J. B., A. H. Knoll, and G. Vidal. 1986. Latest Proterozoic microfossils from the Nama Group, Namibia (South West Africa). Precambrian Research 32: 45-62.

36. Glaessner, M. F. 1976. Early Phanerozoic annelid worms and their geological and biological significance. Journal of the Geological Society, London 132: 259-275.

37. Golubic, S. and H. J. Hofmann. 1976. Comparison of modern and mid-Precambrian Entophysalidaceae (Cyanophyta) in stromatolitic algal mats: cell division and degradation. Journal of Paleontology 50: 1074-1082.

38. Gonzales, R. C., 1992, Digital Image Processing, Addison-Wesley.

39. Graham, L.E. and L. W. Wilcox. 2000. Algae. Prentice Hall, Upper Saddle River NJ, 640 pp.

40. Grant, S.W.F. 1990. Shell structure and distribution of *Cloudina*, a potential index fossil for the terminal Proterozoic. American Journal of Science 290-A: 261-294.

41. Gresse, P. G. and G. J. B. Germs. 1993. The Nama foreland basin: sedimentation, major unconformity bounded sequences and multisided active margin advance. Precambrian Research 63: 247-272.

42. Grotzinger, J. P. 1986. Evolution of an early Proterozoic passive margin carbonate platform, Rocknest Formation, Wopmay Orogen, Northwest Territories, Canada. Journal of Sedimentary Petrology 56: 831-847.

43. Grotzinger, J. P. 1990. Geochemical model for Proterozoic stromatolite decline. American Journal of Science 290-A: 80-103.

44. Grotzinger, J. P., S. Bowring, B. Z., Saylor, and A. J. Kaufman, A.J. 1995. Biostratigraphic and geochronologic constraints on early animal evolution. Science 270: 598-604.

45. Grotzinger, J. P. and A. H. Knoll. 1999. Stromatolites in Precambrian carbonates: evolutionary mileposts or environmental dipsticks? Annual Review of Earth and Planetary Science 27: 313-358.

46. Hahn, G. and H. D. Pflug. 1985. Die Cloudinidae n. fam., Kalk-Rohren aus dem Vendium und Unter-Kambrium. Senckenbergiana lethaea 65: 413-451.

47. Haralick, R. M., 1979, Statistical and Structural Approaches to Texture, it IEEE Proc. 67:786-804

48. Hofmann, H. J. 1975. Stratiform Precambrian stromatolites, Belcher Islands, Canada: Relations between silicified microfossils and microstructure. American Journal of Science 275: 1121-1132.

49. Hofmann, H. J., K. Grey, A. H. Hickman, and R. I. Thorpe. 1999. Origin of 3.45 Ga coniform stromatolites in Warrawoona Group, Western Australia. Geological Society of America Bulletin 111: 1256-1262.

50. Hoffman, P. F. 1969. Proterozoic paleocurrents and depositional hitory of the east arm fold belt, Great Slave Lake. Canadian Journal of Earth Science 6: 441-462.

51. Ivanov, A.V. 1963. Pogonophora. Academic Press, New York, pp. 394-444.

52. James, N. P. 1986. The St. George Group (Lower Ordovician) of western Newfoundland: tidal flat island model for carbonate sedimentation in shallow epeiric seas. Sedimentology 33: 313-343.

53. Kennard, J. M., and N. P. James. 1986. Thrombolites and stromatolites: Two distinct types of microbial structures. Palaios 1: 492-503.

54. Knoll, A.H. 1992. The early evolution of eukaryotes: A geological perspective. Science 256: 622-627.

55. Kruse, P. D., A Yu. Zhuravlev, and N. P. James. 1995. Primordial metazoan- calcimicrobial reefs: Tommotian (Early Cambrian) of the Siberian Platform. Palaios 10: 291-321.

56. Lee, J. J., S. H. Hutner, and E. C. Bovee, eds. 1985. An Illustrated Guide to the Protozoa. Society of Protozoologists, Lawrence KS.

57. Lesh-Laurie, G. E. and P. E. Suchy. 1991. Cnidaria: Scyphozoa and Cubozoa. Pp. 185- 266 in F. W. Harrison and J. A. Westfall, eds. Microscopic Anatomy of Invertebrates. Volume 2: Placozoa, Porifera, Cnidaria, and Ctenophora. Wiley-Liss, New York.

58. Li Guo-xiang, Yue-song Xue, and Chuan-ming Zhou. 1997. Late Proterozoic tubular fossils from the Doushnatuo Formation of Weng'an, Guizhou, China. Palaeoworld 7(1997): 29-37.

59. Little, C. T. S., R. J. Herrington, R. M. Haymon, and T. Danelian. 1999. Early Jurassic hydrothermal vent community from the Franciscan Complex, San Rafael mountains, California. Geology 27: 167-170.

60. Martin, H. 1965. The Precambrian Geology of South West Africa and Namaqualand. Rustica Press, Wynberg, South Africa.

61. McCaffrey, M. A., J .M. Moldowan, P. A. Lipton, R. E. Summons, K. E. Peters, A. Jeganathan, and D. S. Watt, D.S. 1994. Paleoenvironmental implications of novel C30 steranes in Precambrian to Cenozoic age petroleum and bitumen. Geochimica et Cosmochimica Acta, 58: 529-532.

62. McHugh, D. 1997. Molecular evidence that echuirans and pogonophorans are derived annelids. Proceedings, National Academy of Sciences, USA 94: 8006-8009.

63. Miller, R. M. 1983. The Pan-African Damara Orogen of South West Africa/Namibia. Pp. 431-515 in R.M. Miller, ed. Evolution of the Damara Orogen. Geological Society of South Africa Special Publication 11.

64. Narbonne, G. M. and H. J. Hofmann. 1987. Ediacaran biota of the Wernecke Mountians, Yukon, Canada. Palaeontology 30: 647-676.

65. Narbonne, G. M., B. Z. Saylor, B.Z., and J. P. Grotzinger. 1997. The youngest Ediacaran fossils from southern Africa. Journal of Paleontology 71: 953-967.

66. Nielsen, C., N. Scharff, and D. Eibye-Jacobsen. 1996. Cladistic analysis of the animal kingdom. Biological Journal of the Linnaean Society 57: 385-410.

67. Pflug, H. D. 1970a. Zur fauna der Nama-Schichten in Sdwest Afrika. I. Pteridinia, Bau und systematische Zugehorigkeit. Palaeontographica, Abteilung A 134: 226-262.

68. Pflug, H. D. 1970b. Zur fauna der Nama-Schichten in Sdwest Afrika. II. Rangidae, Bau und systematische Zugehorigkeit. Palaeontographica , Abteilung A 135: 198-231.

69. Pflug, H.D. 1972. Zur fauna der Nama-Schichten in Sdwest Afrika. II. Erniettomorpha, Bau und systematische Zugehorigkeit. Palaeontographica , Abteilung A 139:134-170.

70. Qian, Y. and S. Bengtson, 1989. Palaeontology and biostratigraphy of the Early Cambrian Meishuchunian Stage in Yunnan Province, South China. Fossils and Strata, 24: 1-156.

71. Richter, R., 1955. Die ltesten Fossilen Sd-Afrikas. Senckenbergiana lethaea 36:243- 289.

72. Riding, R. and A. Yu. Zhuravlev. 1995. Structure and diversity of oldest sponge-microbe reefs: Lower Cambrian, Aldan River, Siberia. Geology 23: 649-652.

73. Runnegar, B. N. 1992. Proterozoic fossils of soft-bodied metazoans (Ediacaran fauans). Pp. 999-1007 in J. W. Schopf and C. Klein, eds. The Proterozoic Biosphere. A Multidisciplinary Study. Cambridge University Press, Cambridge.

74. Ruppert, E. E. and R. D. Barnes. 1994. Invertebrate Zoology, Sixth Edition. Saunders College Publishing, Fort Worth. 1056 pp.

75. Sarg, J. F. 1988. Carbonate sequence stratigraphy. Pp. 155-182 in C. K. Wilgus and others, eds. Sea-Level Changes: An Integrated Approach. SEPM Special Publication 42.

76. Saylor, B.Z., and J. P. Grotzinger. 1996. Reconstruction of important Proterozoic- Cambrian boundary exposures through the recognition of thrust deformation in the Nama Group of southern Namibia. Communications of the Geological Survey of Namibia 11: 1-12.

77. Saylor, B. Z., J. P. Grotzinger, and G. J. B. Germs. 1995. Sequence stratigraphy and sedimentology of the Neoproterozoic Kuibis and Schwarzrand Subgroups (Nama Group), Southwest Namibia. Precambrian Research 73: 153-171.

78. Saylor, B. Z., A. J. Kaufman, J. P. Grotzinger, and F. Urban. 1998. A composite reference section for terminal Proterozoic strata of southern Namibia. Journal of Sedimentary Research 66: 1178-1195.

79. Schmidtling, R.C., 1995. Three-Dimensional Reconstruction of the Hydrospires of *Pentremites rusticus* (Echinodermata: Blastoidea), M.Sc. Geology Thesis, UCLA.

80. Sergeev, V.N., A. H. Knoll, and J. P. Grotinzger. 1995. Paleobiology of the Mesoproterozoic Billyakh Group, Anabar Uplift, northern Siberia. Paleontological Society Memoir 39: 1-37.

81. Smith, O. A. 1998. Terminal Proterozoic Carbonate Platform Development: Stratigraphy and Sedimentology of the Kuibis Subgroup (ca. 550-548 Ma), Northern Nama Basin, Namibia. Unpublished MSc. Thesis, Massachusetts Institute of Technology, Cambridge MA.

82. Soja, C. M. 1994. Significance of Silurian stromatolite-sphinctozoan reefs. Geology 22: 355-358.

83. Sokolov, B.S., 1997. Essays on the Advent of the Vendian System. KMK Scientific Press, Moscow, 156 pp. (In Russian)

84. Steele-Petrovich, H. M. and T. E. Bolton. 1998. Morphology and paleoecology of a primitive mound-forming tubicolous polychaete from the Ordovician of the Ottawa Valley, Candaa. Palaeontology 41: 125-145.

85. Steiner, M. 1994. Die neoproterozoischen Megaalgen Sdchinas. Berliner Geowissenschaftliche Abhandlungen E15: 1-146.

86. Turner, E. C., N. P. James, and G. M. Narbonne. 1997. Growth dynamics of Neoproterozoic calcimicrobial reefs, Mackenzie Mountains, northwest Canada. Journal of Sedimentary Petrology 67: 437-450.

87. Turner, E. C., G. M. Narbonne, and N. P. James. 1993. Neoproterozoic reef microstructures from the Little Dal Group, northwestern Canada. Geology 3: 259-262.

88. Turner, E.C., G. M. Narbonne, and N. P. James. In Press. Framework composition of early Neoproterozoic calcimicrobial reefs and associated microbialites, Mackenzie Mountains, N. W. T. in J. P. Grotzinger and N. P. James, eds. Carbonate Sedimentation And Diagenesis In The Evolving Precambrian World. Society of Economic Paleontologists and Mineralogists Special Publication 65.

89. Vanyo, J., and Awramik, S., 1982, Geophys. Res. Letts., v, 9, p. 1124-1128.

90. Vistelius, A. B., 1969, Preface, *J. Math. Geol.* 1:1-2

91. Walter, M. R., and G. R. Heys. 1984. Links between the rise of the metazoa and the decline of stromatolites. Precambrian Research 29: 149-174.

92. Werner, B., 1967. Morphologie, Systematik, und Lebensgeschichte von *Stephenoscyphus* (Scyphozoa Coronatae) sowie seine Bedeutung fr die Evolution der Scyphozoa. Zoologischer Anzeiger, 30 (Supplementband): 297-319.

93. Wray, J. L. 1977. Calcareous Algae. Elsevier, Amsterdam, 185 pp.

94. Xiao, S., A.H. Knoll, and X. Yuan. 1998. Morphological reconstruction of *Miaohephyton bifurcatum*, a possible brown alga from the Neoproterozoic Doushantuo formation, South China. Journal of Paleontology 72: 1072-1086.

Appendix A

Population Statistics

This appendix contains in tabular form the information displayed in Figure 1-12. Table A.1 was used to generate the graphs B and C, and Table A.2 was used to generate the graphs A, D, and E. Table A.1 displays the number of specimens identified as *Namacalathus*, *Cloudina*, and the number which could not be identified, in digital images of six cross sections from two rock samples (i.e., six frames in two image stacks, where each stack was generated in order to obtain tomographic reconstructions, as described in Chapter 2). One sample was collected from the Donkergange farm in the northern, Zaris subbasin, and the other from near the town of Vioolsdrif in the southern, Witputs subbasin. The exhaustive database of synthetic cross sections in Appendix B was used to identify *Namacalathus* fossils in the six images. The careful descriptions in Grant (1990) of *Cloudina* were used to identify examples of that genus. In cases where a single cross section was not adequate to make a positive identification, adjacent frames were examined to confirm or rule-out either morphology. The frames were chosen far enough apart so that none shared specimens in common, and this was carefully checked. The table shows that an overwhelming majority of fossils in each rock are examples of *Namacalathus*: 90% in the Donkergange sample, and 77% in the Vioolsdrif sample.

Table A.2 contains information regarding the size, orientation, and ornamentation of 51 *Namacalathus* specimens in the Vioolsdrif sample. The last column indicates one of three orientations for each specimen. "Vertical orientation" (V) describes fossils with a cross section parallel to bedding that exhibits the six or seven regular windows at the cup's mid-section (as in Figures 1-8C, D; recall that "windows" are the six or seven regular holes in the cup walls, as described in Chapter 1). This orientation can span approximately 22° before the diagnostic shape is lost, measured for the angle formed by the cup's symmetry axis and the bedding plane, at the cup's geometric center. "Horizontal orientation" (H) describes specimens with a cross section parallel to bedding that contains the cup's upper circular opening as well as the basal hole to which the stem attaches. This orientation can span an angle of approximately 25° before the diagnostic shape is lost, where this angle is measured in the same way. "Oblique orientation" (O) describes fossils that fit neither of the above categories; these span an angle of 43°. It follows that if there were no preferred orientation, 28% of fossils would have a vertical orientation, 24% horizontal, and 38% oblique. Instead, it is found that the vast majority (69%) have a horizontal orientation.

The sixth column in Table A.2 indicates whether each fossil has a pronounced inward-turning lip around its largest opening, and the seventh column indicates whether its windows have pronounced lips. A question mark is used to label ambiguous cases. It is clear that the vast majority of horizontally-oriented specimens appear to have no window lips, while the majority of vertical and oblique specimens clearly have this feature. This demonstrates that the window lips are very slight or non-existent when viewed in the first kind of cross section, and does not mean that they do not exist at all in specimens with a horizontal orientation. That is, the lips are more pronounced along one dimension, much like the lips of a human mouth (the choice of $\gamma_c(z_c)$ in Chapter 3 was informed by this observation). By contrast, all specimens in the Donkergange sample have pronounced window lips in every orientation, and hence this feature was included in the systematic description supplied in Chapter 1.

Table A.2 also indicates the diameter (d) and height (h) of each specimen, in cases were these quantities are measurable. The diameter is the maximum width of the cup (i.e., the distance between opposite walls oriented parallel to the symmetry axis). The height is measured from where the stem attaches to the farthest point along the cup's upper rim. Therefore, it is possible to estimate either quantity using a single cross section only in the case of fossils with a horizontal or vertical orientation. In cases where only oblique sections are contained within the data set, a maximum cup dimension in oblique cross section is measured instead ($d_{\max}$). If the specimen is only partially extant (i.e., if it is broken, or not contained entirely within the sample), then it is frequently not possible to make either one or both measurements, and this is indicated in the table by a question mark. The "aspect ratio" is the quantity d/h, and its average is $\alpha \equiv 1.18$ (for the cases in which d and h are both obtained). The "size", listed in the third column is an approximate measure of the cup's overall dimension. If d and h are known, the size is given by their average. If only d is known, the size is estimated by the average with d/α, and if only h is known, the size is estimated by the average with αh. If the fossil has an oblique orientation, then the size is estimated by $d_{\max}$. The mean size calculated for specimens listed in Table A.2 is 0.61 cm.

Table A.1: Numbers by type.

Frame label	Sample	*Namacalathus*	*Cloudina*	Unidentifiable
A	Donkergange	16	0	2
B	Donkergange	20	0	0
C	Donkergange	35	0	6
D	Vioolsdrif	18	3	3
E	Vioolsdrif	19	4	2
F	Vioolsdrif	30	5	3

Table A.2: Size, orientation, and ornamentation of 51 *Namacalathus* fossils in the Vioolsdrif sample.

Specimen	Diameter (d)	Height (h)	Size	(d/h)	Lipped Cup?	Lipped Windows?	Orientation
01	0.72	0.53	0.63	1.36	Yes	No	H
02	0.53	0.60	0.57	0.88	Yes	?	H
03	0.84	0.68	0.76	1.23	Yes	No	H
04	?	?	0.44	?	?	Yes	O
05	?	?	0.68	?	?	Yes	O
06	0.54	0.46	0.50	1.19	Yes	No	H
07	0.75	0.64	0.69	1.16	Yes	No	H
08	0.37	0.33	0.35	1.11	No	No	H
09	0.67	0.44	0.56	1.54	Yes	No	H
10	0.51	0.41	0.46	1.25	Yes	No	H
11	1.05	?	0.97	?	Yes	Yes	V
12	0.70	0.54	0.62	1.29	Yes	No	H
13	0.50	0.40	0.42	1.10	No	No	H
14	0.62	0.68	0.65	0.92	No	No	H
15	0.43	0.45	0.44	0.96	Yes	No	H
16	0.70	0.63	0.67	1.11	Yes	No	H
17	?	?	0.70	?	?	Yes	O
18	?	?	?	?	Yes	No	H
19	?	?	?	?	?	Yes	O
20	0.16	0.12	0.14	1.30	?	?	H
21	0.51	0.45	0.48	1.13	Yes	No	H
22	?	?	0.67	?	Yes	Yes	H
23	0.53	?	0.49	?	Yes	Yes	O
24	0.47	0.37	0.42	1.27	Yes	No	H
25	1.16	1.11	1.13	1.04	Yes	No	O
26	0.51	0.46	0.49	1.11	Yes	No	H
27	?	?	0.78	?	?	No	O
28	0.85	0.62	0.73	1.36	Yes	No	H
29	0.62	0.50	0.56	1.23	Yes	No	H
30	0.74	?	0.68	?	?	No	H
31	0.63	0.45	0.54	1.41	Yes	?	H
32	0.61	0.62	0.62	0.97	No	No	H
33	0.92	0.82	0.87	1.11	Yes	Yes	H
34	0.49	0.38	0.43	1.27	Yes	No	H
35	0.47	?	0.47	?	?	Yes	O
36	0.30	0.21	0.26	1.42	Yes	No	H
37	0.69	0.63	0.66	1.08	No	No	H
38	?	?	0.76	?	?	Yes	O
39	0.66	0.71	0.69	0.99	Yes	No	H
40	?	?	0.75	?	Yes	No	O
41	0.65	0.65	0.65	1.02	Yes	No	H
42	0.72	0.59	0.66	1.23	Yes	No	H
43	0.87	?	0.80	?	?	No	H
44	0.69	?	0.63	?	Yes	No	H
45	0.74	?	0.68	?	Yes	?	V
46	?	?	0.75	?	?	Yes	O
47	0.77	0.54	0.66	1.42	No	No	H
48	0.69	0.61	0.65	1.13	Yes	No	H
49	0.70	?	0.65	?	Yes	No	H
50	?	?	?	?	No	Yes	O
51	?	?	0.79	?	No	No	O

Appendix B

Database of Synthetic Cross Sections

This appendix contains a database of synthetic cross sections, obtained at random orientations through the mathematical models shown in Figures 1-10B and 1-10D. A complete description of the method used to generate these models is supplied in Chapter 3. The database is nearly exhaustive insofar as it contains the majority of cross sections that are in some way "obviously distinctive." That is, while an infinite number of cross sections can be generated, only certain cross-sectional morphologies are distinctive. The purpose of generating such a database is so that assemblages of *Namacalathus* can be easily identified in the field, where only two-dimensional cross sections are visible.

All cross sections were generated using the "SceneViewer" program for Silicon Graphics computers, normally used for viewing and manipulating three-dimensional data-sets stored in the so-called "inventor" format. (The software program used to articulate the models in three dimensions (Nuages) is able to save the articulated models in a variety of formats, including the inventor format.) The wireframe "draw-style" in SceneViewer is used to view a slice of the model between the near and far "clipping planes." These invisible parallel planes conceal all portions of the model that are nearer than the "near" plane and farther than the "far" plane. The model and both clipping planes can be translated and rotated within a three-dimensional Cartesian coordinate space. As the two planes are moved closer together, the model is viewed at progressively thinner increments until only a single cross section is visible. When the clipping planes are set 0.0001 distance units apart from each other, a very thin cross section of the model is obtained. An independent screen-capture utility is then used to store the cross section in a jpeg image file. The model can be moved into random orientations in order to obtain the whole range of possible cross sections.

When fully assembled, the model described in Chapter 3 is composed of a single contiguous

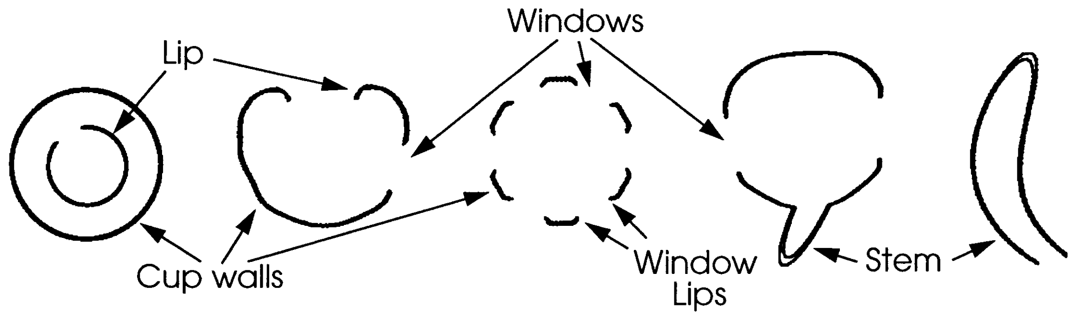

Figure B-1: The isolated components in cross sections represent portions of a contiguous fossil wall. In this figure, several features of the *Namacalathus* morphology are labeled.

wall that connects the stem, cup, and inward-turning lip. Cross sections of this morphology are normally composed of one or several isolated components which are in fact connected in the third spatial dimension. Cross sections observed in outcrop are divided into nine types. Eight of these are numbered 0 to 7 according to the number of components in the cross section (except in the case of Type 0), and the ninth type is called "Type F." (The Type F designation is explained later.) The eight numbered types refer to cross sections with thin walls of approximately constant thickness. Synthetic versions of the eight numbered types that were generated from the mathematical models are shown in Figures B-2 through B-9. The database contains a nearly-exhaustive set of these eight types, where each type is defined as follows. Type 0 cross sections consist of just one unbroken component that completely encloses a space (e.g., ring-shaped cross sections). Type 1 cross sections consist of just one component that does not completely enclose a space (e.g., goblets, cups, and broken rings). Type 2 through Type 7 refer to cross sections that have two through seven components, respectively. Several examples of these numbered types are shown in Figure B-1, where different components of individual cross sections are labeled according to the structures they represent in the three-dimensional model.

Type F ("filled") refers to cross sections through parts of the interior of a *Namacalathus* cup that were not initially filled by sediment, and which were eventually filled by calcite crystals (i.e., skeletal grainstones). Figure B-10 contains some examples of Type F cross sections observed in outcrop. Figure B-11 contains some examples of synthetic Type F cross sections, obtained by projecting an outer wall of the mathematical model onto a cross-sectional plane. This method assumes that the gravity field is perpendicular to the cross-sectional plane that receives the projection (i.e., which is parallel to the bedding plane). It is assumed that the space between the walls and this plane was not filled by sediment because the walls prevented sediment from falling into the regions directly beneath them. In fact, this method is only approximate, because the sediment is not expected to behave exactly in this manner. More accurate cross sections of Type F could be generated by thinning or

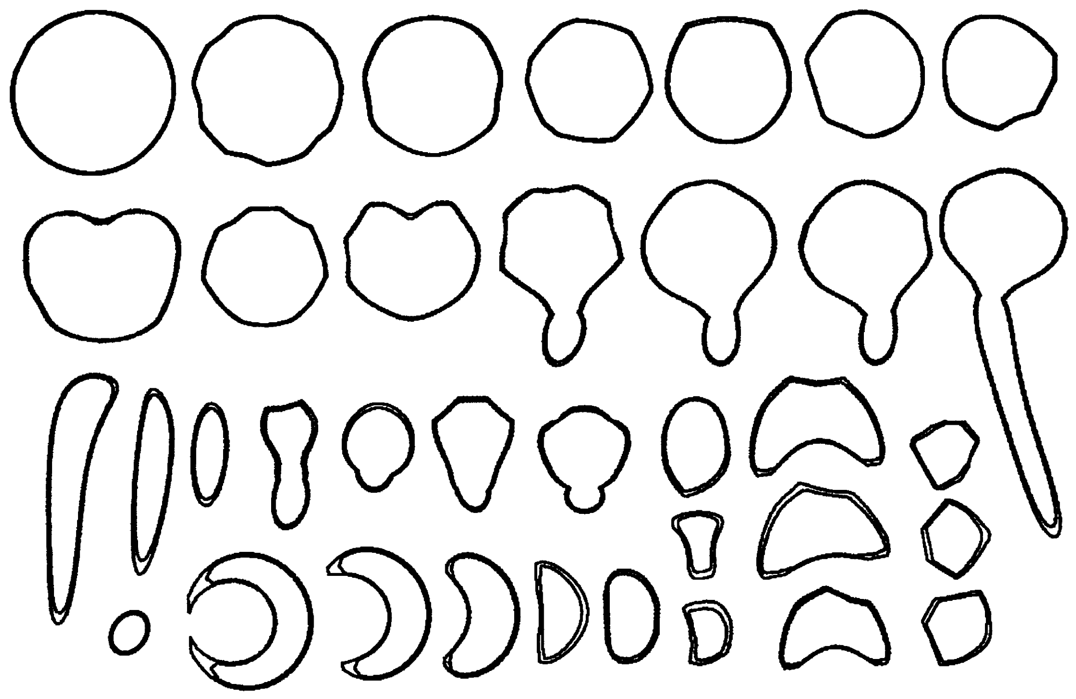

Figure B-2: Type 0 cross sections. Notice that the interior and exterior boundaries of the walls are clearly visible in several cases. The fossil wall appears to be thickest in oblique cross sections that "graze" the surface, seen here in the tips of crescent-shaped cross sections in the lower left.

eroding the walls of the cross sections shown in Figure B-11

Type F cross sections may have formed in other ways. For example, these might be the cross sections of fossil walls that have undergone massive diagenetic "fattening" or neomorphic recrystallization (see Figure 1-13). It is observed in digital images of serial cross sections, however, that Type F cross sections generally occur on just one side of an individual specimen, and within a region that is oriented parallel to the bedding plane. This observation favors the previous explanation whereby these cross sections represent interior regions that were not initially filled by sediment. Also, Type F cross sections could be interpreted as examples of a separate taxon if it were not for three observations. First, these occur in rocks containing abundant examples of the eight numbered types. Second, if the interior structure of a Type F cross section is "erased," what remains can generally be matched with a cross section in one of the eight numbered categories. Third, individual specimens with Type F cross sections have been reconstructed, and these are consistent with the morphology of a partially-filled *Namacalathus*. Since the varieties of Type F cross sections are possibly very great in number, Figure B-11 probably contains only a small subset.

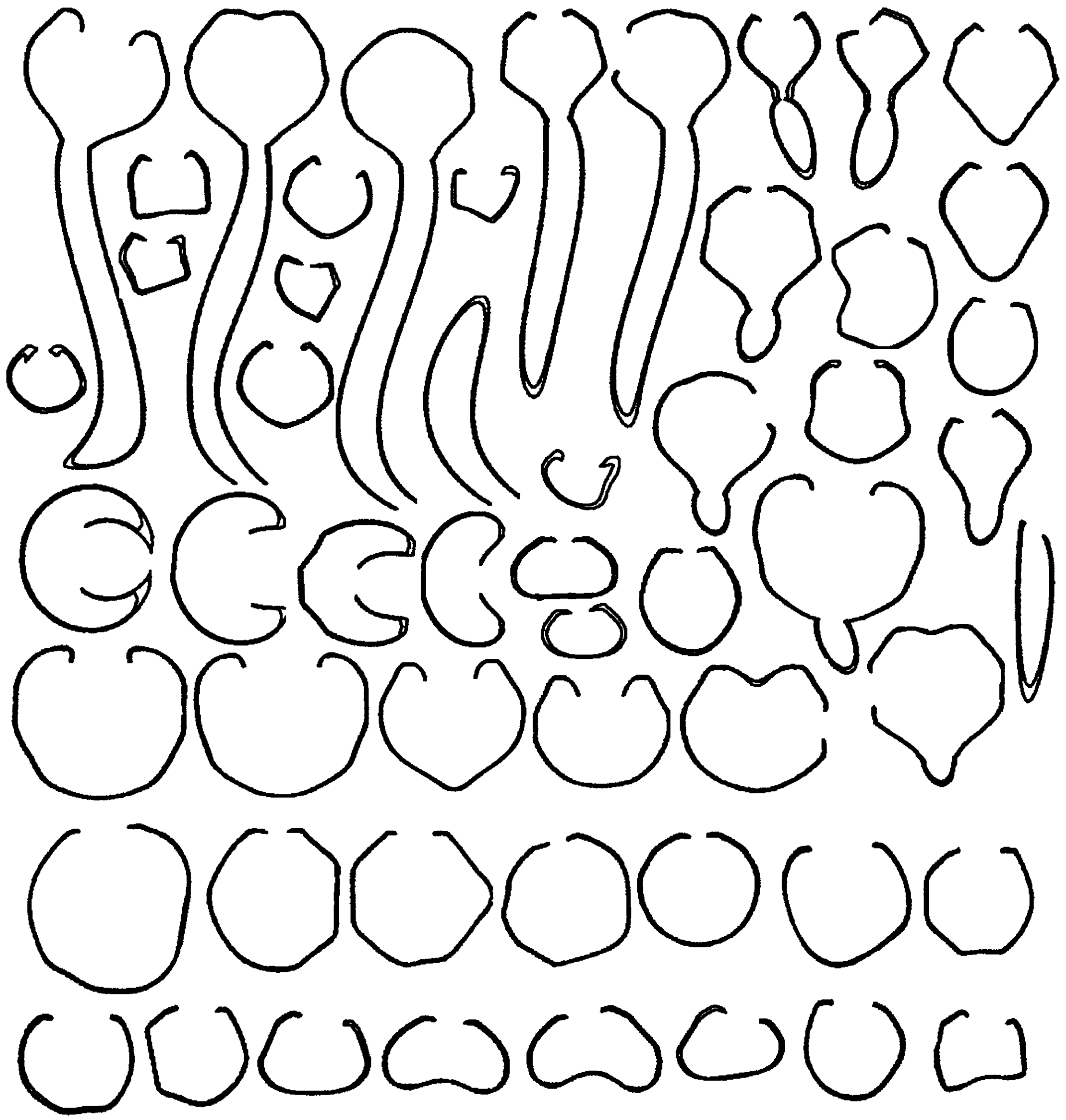

Figure B-3: Type 1 cross sections.

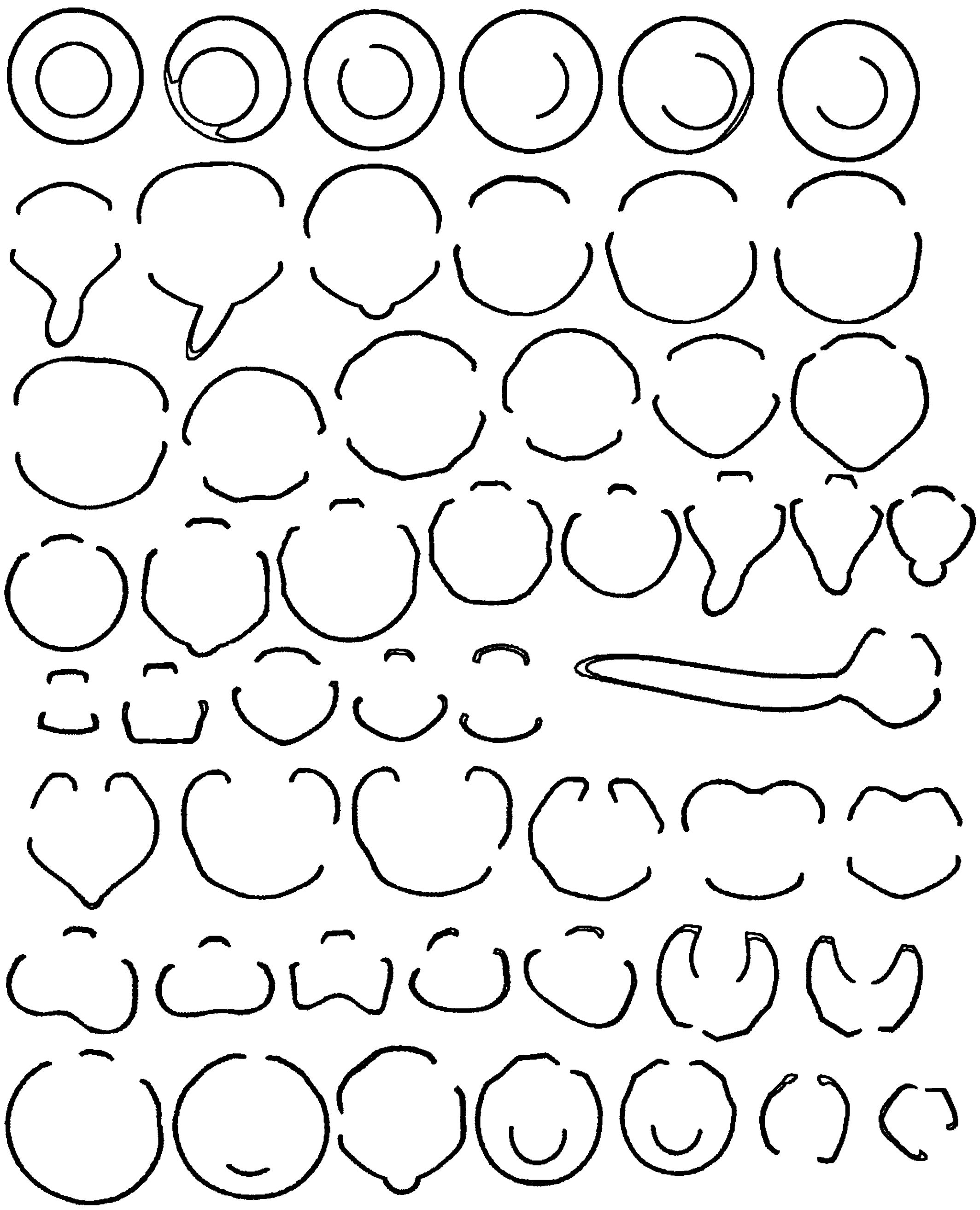

Figure B-4: Type 2 cross sections.

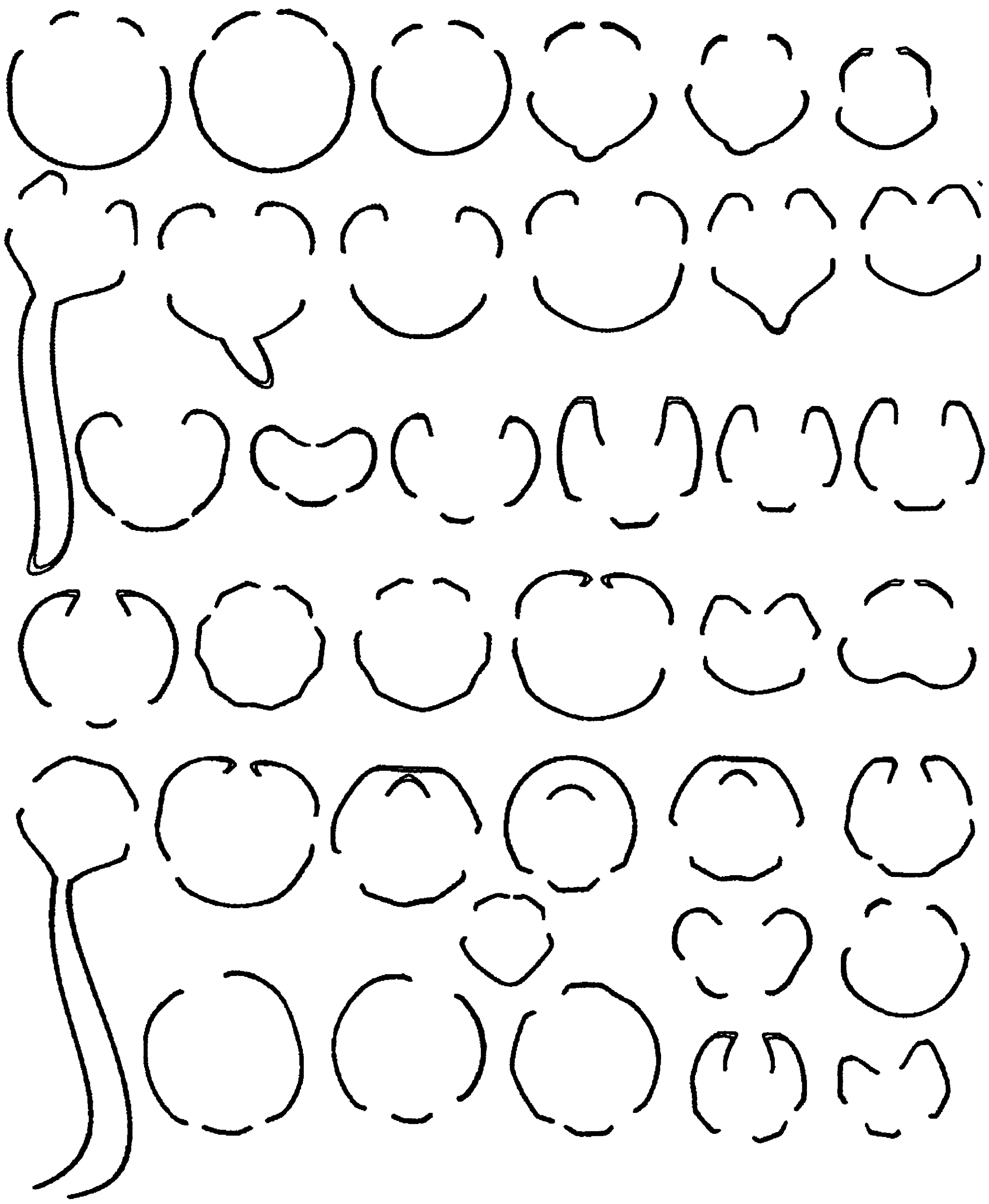

Figure B-5: Type 3 cross sections.

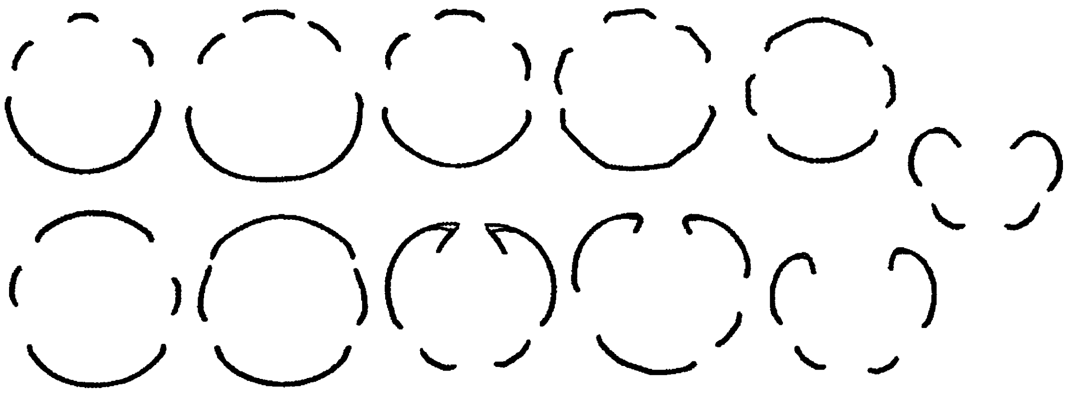
Figure B-6: Type 4 cross sections.

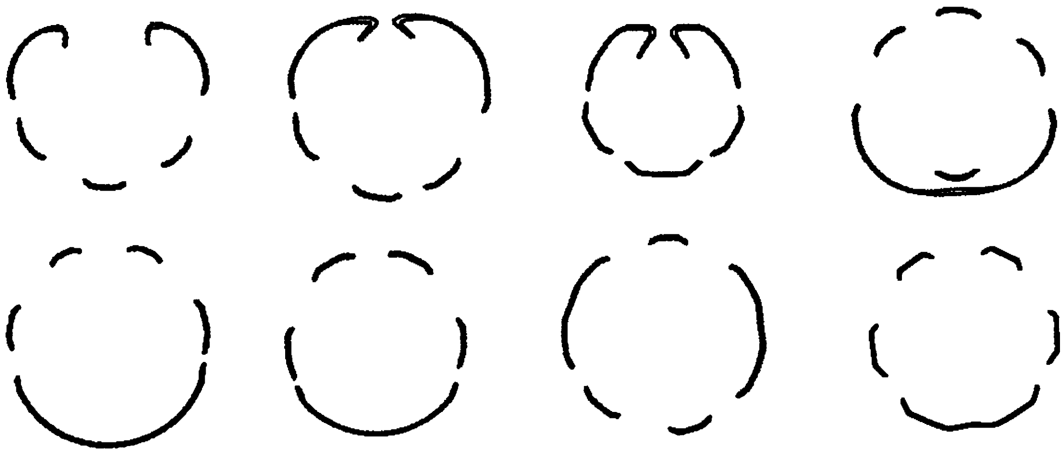
Figure B-7: Type 5 cross sections.

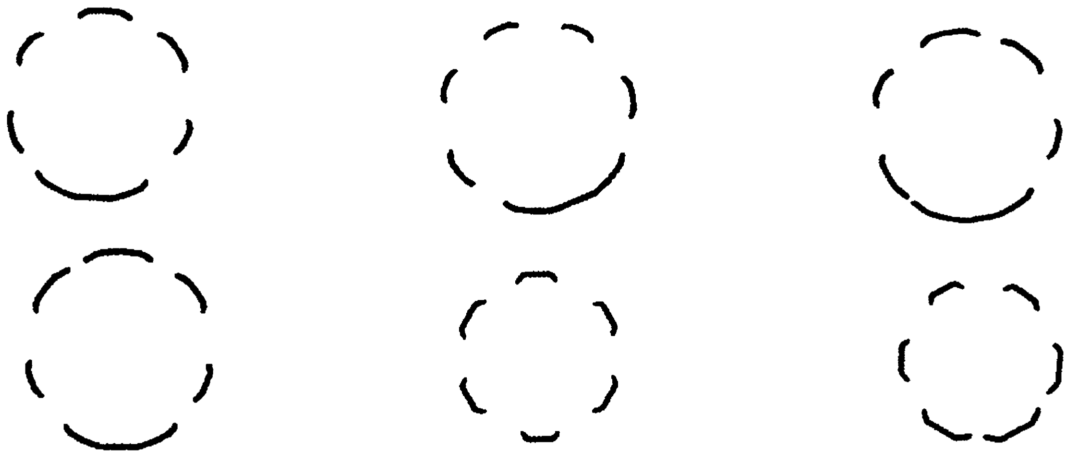
Figure B-8: Type 6 cross sections.

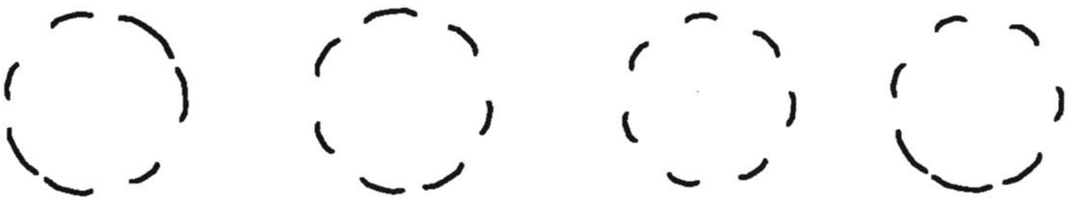

Figure B-9: Type 7 cross sections.

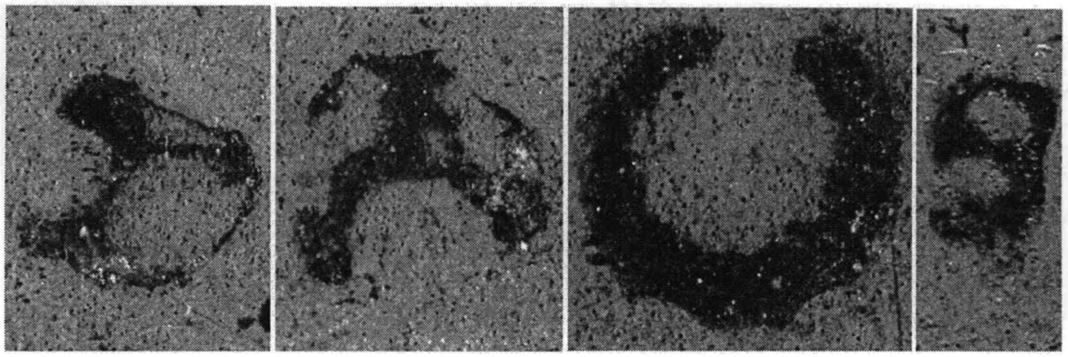

Figure B-10: Type F cross sections in outcrop.

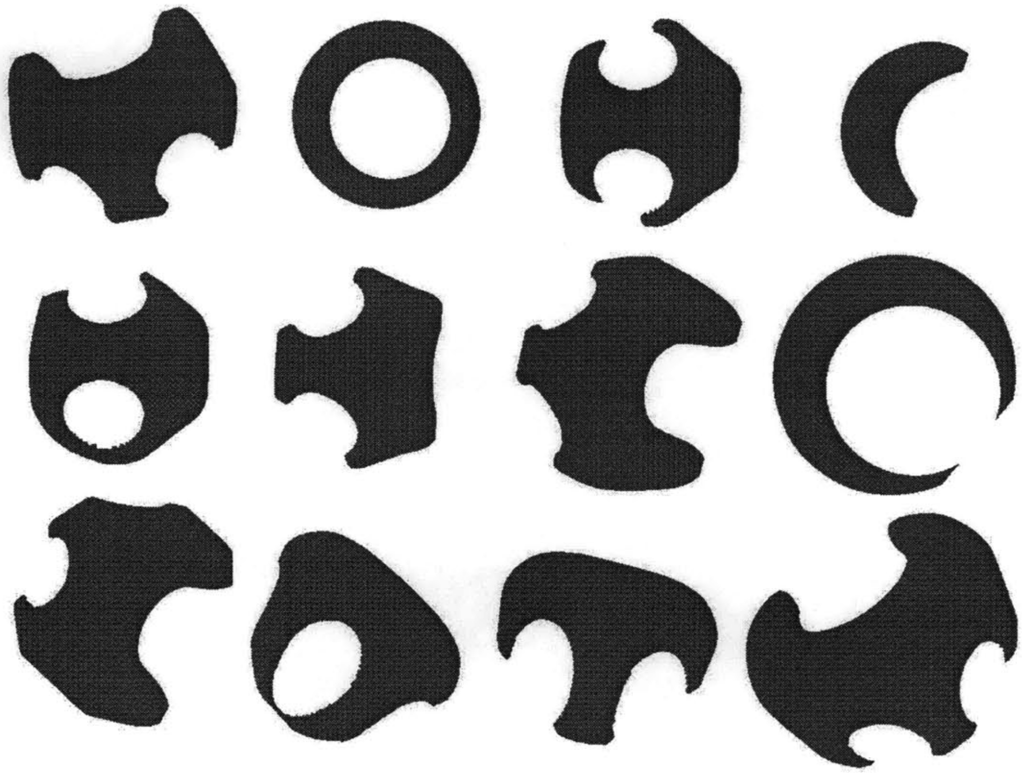

Figure B-11: Type F cross sections.

Appendix C

Documentation and Source Code

This appendix provides a documentation of the MATLAB scripts used to make the tomographic reconstructions according to the methods described in Chapter 2, and also the mathematical models described in Chapter 3. The scripts are divided into several types according to their function. Members of the first category are the so-called "database" scripts, used to view the stack of frames, to create and label movies (cropped stacks of jpeg images), and to display still images of tomographic reconstructions or the reconstructions themselves. In effect, these scripts are used to manage the digital image data and reconstructions by labeling, storing, and viewing data files associated with fossils listed in the specimen database (contained in an ASCII text file, mentioned in the section titled "Obtaining Serial Images," in Chapter 2). Members of the second category are the "processing" scripts, used to view and process the cropped stack of 24-bit color images of an individual fossil, in order to extract and save contour data. The third category are the "post-processing" scripts, used to manipulate the contour data once it has been obtained. Finally, the "morphology" scripts contain the mathematical description supplied in Chapter 3. All of the source code described in this appendix is printed in the last section.

The contours generated by the processing scripts and the post-processing scripts, and those generated using the morphology scripts, are formatted and stored in an ASCII "contour file" for use with the Nuages software package. Nuages is a separate program whose source code is not incorporated by any of the programs described in this appendix, and which accepts the ASCII "contour file" as input. The Nuages software can be obtained in compiled form on the World Wide Web at the following address: http://www-sop.inria.fr/prisme/personnel/geiger/nuages.html, where it is available for use with the following computer platforms: SunOS 4.1.4, SunOS 5.5, SGI IRIX 4.x, SGI IRIX 5.3, SGI IRIX 6.2, DEC Ultrix, IBM RS6000 AIX, and Linux. It should be noted that MATLAB and the MATLAB Image Processing Toolbox is available for the majority of computer platforms, including Macintosh, Windows, and most Unix-based operating systems.

Database scripts

The database scripts are used to manage the images, jpeg movies, and tomographic reconstructions of individual fossils listed in the specimen database. In what follows, each of the most important scripts is named and described. Each description contains a list of input arguments (if there are any), which must be defined before the script is run (alternatively, the user is sometimes prompted to enter some parameter values while the script is running). All of the scripts in this section were written for use with MATLAB and the MATLAB Image Processing Toolbox. Many of the procedures implemented in these scripts are described in the section entitled "Preparing the Images for Processing" of Chapter 2.

`initialize.m` — Uploads a list of locations of the jpeg and bmp files that constitute the stack of frames, and stores this list in an indexed array of string variables (called a "cell" array in Matlab). These lists are uploaded from two ASCII text files, which can be generated and labeled `jpg_filelist.lst` and `bmp_filelist.lst` using the following UNIX commands (assuming the image files are stored in subdirectories entitled `SET1`, `SET2`, `SET3`, etc., and listed in order of their position in the stack): `ls -l SET*/*jpg > jpg_filelist.lst`; and: `ls -l SET*/*bmp > bmp_filelist.lst`. This script also declares and initializes many global variables used by other scripts. [The input arguments to `initialize.m` are the string variables `jpg_filelist` and `bmp_filelist`, which should be given the names of the aforesaid ASCII text files. To use the same example, at the Matlab prompt one types: `jpg_filelist='jpg_filelist.lst'`.]

`showALLframe.m` — Used to view the jpeg image frame for a given frame number. [Input arguments: `Fn` (the frame number)]

`make.m` — Used to make a jpeg movie. Prompts the user to specify a rectangular region of interest (ROI) using a mouse. [Input arguments: `FIRST` (starting frame of the movie); `LAST` (ending frame); `INCREMENT` (number of frames skipped between *movie* frames]

`make_R.m` — Like `make.m`, except the region of interest is predefined in the variable `R`. This script is useful for generating movies of fossils for which an ROI is specified in the specimen database. [Input arguments: `R` (ROI), and {`FIRST, LAST, INCREMENT`} as in `make.m`]

`save_movie.m` — Used to save and label a jpeg movie.

`delete_movie.m` — Used to delete a particular jpeg movie.

`show_movie.m` — Used to display a particular jpeg movie.

`show_frame.m` — Used to display individual frames in a jpeg movie.

`show_pics.m` — Used to display still-image pictures of tomographic reconstructions, listed in the specimen database (and which are assigned the same labels as their jpeg movie counterparts: i.e., the labels assigned to individual fossils in the database).

`show_model.m` — Used to display the tomographic reconstruction of an individual fossil listed in the specimen database. The program "SceneViewer" (inventor format viewer) is used for this purpose.

Processing scripts

The Processing scripts are used to generate and then extract contours from a cropped stack of 24-bit images. The descriptions have the same format as that used in the previous section. All of the scripts in this section were written for use with MATLAB and the MATLAB Image Processing Toolbox. In this section only, scripts are listed in the order in which they are applied. Many of the procedures implemented in these scripts are described in the sections entitled "Preparing the Images for Processing," "Image Processing Techniques for Isolating Fossils," and "Obtaining Contours," in Chapter 2.

`register.m` — Prompts the user to point to each of four registration holes using a mouse (in clockwise order, starting from the upper-left). The user can magnify each point as much as 6 times in order to reckon the center accurately (i.e., in the present case, each 1 mm hole is approximately 13 pixels in diameter). The script records the positions of each hole in an indexed list.

`correct.m` — Calculates and records the translation, rotation, and scale corrections for each frame in the stack according to equations (2.1)–(2.8), using the registration hole positions obtained by `register.m`.

`makestack.m` — Creates a cropped stack of 24-bit color bitmap (bmp) images from which the contours will be obtained. [Input arguments: `FIRST` and `LAST` as in `make.m`, and `R` as in `make_R.m`]

`seeframe.m` — Used to view the bmp cropped image frame for a given frame number. [Input arguments: `Fn` (the frame number)]

`seeall.m` — Used to show the entire cropped bmp stack as a movie. Used along with `seeframe.m` to study the entire fossil, in order to make decisions about which components to include in the reconstruction, and which others to leave out.

• At this stage the user decides whether to obtain the contours in the same sequence as frames appear in the sample, or in the "piecemeal" sequence described in Chapter 2. In the former case, the script genframelist.m (described below) is skipped and nextframe.m is used to advance from one frame to the next. Alternatively, leapframe.m is used to advance between frames in the list generated by genframelist.m.

genframelist.m — Generates a list of numbers indicating the order in which frames are to be processed, and so that the reconstruction occurs in a "piecemeal" fashion. The first number in the list refers to the first frame in the stack, and then the last frame, the middle frame, the frame at a position equivalent to 1/3 of the stack's total depth, and then at the position 2/3, 1/4, 3/4, 1/5, 2/5, 3/5, 4/5, 1/6, etc. For example, if there are five frames in the stack [frequently there are hundreds], and if these are numbered $1, 2, 3, 4, 5$, then genframelist.m obtains the sequence: $1, 5, 3, 2, 4$. Recall that this enables the user to obtain a "complete" reconstruction after the first three frames, and where the model improves in vertical resolution while it is being assembled. In this way, the user can avoid processing all of the frames if it becomes clear that so much resolution is not required to obtain the morphology.

leapframe.m — Advances to the next frame in the list generated by genframelist.m, and clears and initializes some variables used by other scripts.

nextframe.m — Advances to the frame framenum + step, where framenum is the current frame, and step is user-defined (i.e., in total, a fraction 1/step of all frames are processed). [Input arguments: step]

histo.m — Performs the conversion from 24-bit color to grey-scale and the histogram adjustment (contrast stretching) discussed in Chapter 2. A suite of scripts called histo1.m, histo2.m, etc., perform the same function, except with a different value of y_{min1}, which determines the lower bound for the range of the mapping function. The script rehist.m can be used to perform the process using bounds that are entered manually. The result in all cases is a grey-scale image with relatively high contrast. The scripts isolcont1.m and isolcont2.m can be used to adjust the histogram differently in different regions of an image, which the user is prompted to select using a mouse.

focus.m — Performs an unsharp masking procedure, which is followed by a median filter process. This script is extremely useful for improving the contrast and resolution of badly focused images. Because this script can be omitted by ensuring that all images are focused correctly, it is not described in the text. For a detailed discussion of unsharp masking and median filtering techniques, see pages 191 through 197 of Gonzalez (1992).

binize.m — Applies the binary thresholding procedure described in Chapter 2, with the threshold value specified in the variable U. The value of U is adjusted until the fossil is separated from the surrounding material (i.e., is not in contact with objects in the surrounding material). [Input parameters: U]

getpolys.m — Prompts the user to select points along the boundary of fossil objects in the high-contrast grey-scale image obtained from histo.m, and generates a binary image with the outlined region in black. This script is used instead of binize.m if the fossil material cannot be separated by simple thresholding.

selector.m — Prompts the user to select objects in the binary image obtained with binize.m or getpolys.m, using a mouse. These objects are isolated from the surrounding material by means of the flood-filling procedure described in Chapter 2.

erase.m, eraser.m — Used to erase unwanted portions of a binary image. The user is prompted to choose the vertices of a polygon that encloses the region to be erased using a mouse. The polygon selection occurs with the binary image in the background in the case of erase.m, and with the high-contrast grey-scale image in the background in the case of eraser.m. The latter is used whenever the user wishes to insure that deletions do not eliminate real features in the original image of the fossil.

enhance1.m, enhance2.m — The first script applies a sequence of so-called "morphological operations" to the current binary image, including "single-fill" and the removal of so-called "spur" points (see Chapter 2). The script enhance2.m applies a "closing," which is a dilation followed by an erosion, using a 5×5 structural element (used primarily to connect objects that are separated by short distances and to fill small gaps; see Chapter 2).

linedraw.m, linedrawer.m, linesdraw.m, linesdrawer.m — The first script prompts the user to select the two endpoints of a straight line, drawn with a thickness (in pixels) given by the parameter THICKNESS, in order to connect two adjacent wall segments (which were separated artificially during a step in the processing). The script linedraw.m puts the binary image in the background while the endpoints are selected, while linedrawer.m makes use of the high-contrast image. The latter of these is used when the best accuracy is sought. The scripts linesdraw.m and linesdrawer.m differ from one another in the same regard, and in both cases multiple endpoints are selected in order that a connected poly-line is drawn instead. [Input arguments: THICKNESS]

undo.m — Used immediately following erase.m, eraser.m, enhance1.m, enhance2.m, linedraw.m, linedrawer.m, linesdraw.m, and linesdrawer.m in order to undo their effects.

collconts.m — Obtains the boundary points of objects in the current binary image (i.e., contours). Contours not exceeding a minimum length MINLEN are discarded. The contours are then displayed, and the user is prompted to select unwanted contours for deletion. [Input arguments: `MINLEN`]

`applycorr.m` — Applies the translation, rotation, and scale corrections in equations (2.12)–(2.17), obtained using `correct.m`, to the newly-obtained contour coordinates.

`storeconts.m` — Stores all current contours in one entry of a large cell array (i.e., an array whose entries contain arrays), where the entry corresponds to the current frame number.

`pstore.m` — Saves the cell-array updated by `storeconts.m` to a "`.cnt`" file, also called a Nuages contour file. This file is formatted for use with the Nuages software package, which is run separately. (The Nuages software is used to connect the points in contours obtained from adjacent frames using tetrahedrons, in order to articulate a three-dimensional reconstruction of an entire specimen.) The script records an elevation for each contour in the `.cnt` file, where the user is asked to supply the distance between contours. (This distance is obtained using the procedure described in the last section of Chapter 2.) A simple C program called `filter.c` can be used to eliminate every n^{th} coordinate from a `.cnt` contour file, in order to give the effect of "smoothing" the roughness of a tomographic reconstruction (where n is specified by the user). In order to articulate a reconstruction from a contour file called "`contour.cnt`" using Nuages, and in order to store this reconstruction in an "Inventor file" called "`contour.iv`," the following Unix command-line is executed: "`nuages contour.cnt -tri contour.iv -iv`." The Inventor file can then be viewed using the Silicon Graphics programs "SceneViewer" or "ivview." The contour file is an ASCII file has the following format, where any term surrounded by "<>" has the named quantity in its place. Ellipses (...) indicate that the pattern in the previous line is repeated many times.

```
S <number of frames>
v <total points in the contours obtained from frame 1> z <elevation of these contours>
{
<x position of 1st point in contour 1, frame 1> <y position of 1st point in contour 1, frame 1>
<x position of 2nd point in contour 1, frame 1> <y position of 2nd point in contour 1, frame 1>
...
<x position of nth point in contour 1, frame 1> <y position of nth point in contour 1, frame 1>
}
{
<x position of 1st point in contour 2, frame 1> <y position of 1st point in contour 2, frame 1>
<x position of 2nd point in contour 2, frame 1> <y position of 2nd point in contour 2, frame 1>
...
<x position of last point in contour 2, frame 1> <y position of last point in contour 1, frame 1>
}
...
{
<x position of 1st point in last contour, frame 1> <y position of 1st point in last contour, frame 1>
<x position of 2nd point in last contour, frame 1> <y position of 2nd point in last contour, frame 1>
...
<x position of last point in last contour, frame 1> <y position of last point in last contour, frame 1>
}
v <total points in the contours obtained from frame 2> z <elevation of these contours>
```

```
{
<x position of 1st point in contour 1, frame 2> <y position of 1st point in contour 1, frame 2>
<x position of 2nd point in contour 1, frame 2> <y position of 2nd point in contour 1, frame 2>
...
<x position of last point in contour 1, frame 2> <y position of last point in contour 1, frame 2>
}
...
```

Post-processing Scripts

The scripts described in this section are used to manipulate contours once they have been obtained and saved in a Nuages contour file (a `.cnt` file). Some of these operations are mentioned in the section entitled "Obtaining Contours" of Chapter 2. All of the scripts in this section were written for use with MATLAB and the MATLAB Image Processing Toolbox.

`pload.m` — Used to load all contours from a Nuages contour (`.cnt`) file into a large cell array.

`pstore.m` — See the description of `pstore.m` in the previous section.

`pview.m` — Used to view the contours obtained from individual frames. This script can also be used by other, more sophisticated scripts, to compare contours drawn with different colors and fill styles. [Input arguments: `cont` (contains a string that is formatted as follows: "`frame_number draw_style line_color red green blue`," where `frame number` is the frame number from which the contours were obtained, `draw_style` is a number 0 or 1 depending whether an outline or filled contour is preferred, `line_color` is the contour boundary color (from 1 to 16), and the quantities `red`, `green`, and `blue` are the intensities of red, green, and blue which determine the color of the contour's interior if it is filled.)]

`pmover.m` — Prompts the user to enter a starting frame number, which is denoted n. The script then displays contours in the $n-1$ frame using a solid red fill, contours in the current n frame using a blue outline, and contours in the $n+1$ frame using a green outline. A keystroke enables the user to change the value of n, and arrow keys are used to move the contour n with respect to the others if it is clearly misaligned (i.e., in which case an error was probably committed in the collection of registration points for that frame). Alternatively, the user can obtain the positions of registration holes again, if needed, using an abbreviated form of `register.m` (i.e., and then recalculate and apply the corrections).

`peruse.m` — Displays all contours in solid red fill as a movie.

`pedit.m` — Creates a binary image of the contours in a frame, and then enables the user to apply all of the binary image processing scripts to modify the solid binary image objects (e.g., `erase.m`, `linedraw.m`, `enhance2.m`, etc.). The contour is re-obtained and stored using `collconts.m`, `storeconts.m`, and `pstore.m`, as before.

`skeletons.m` — Creates a binary image of all the contours in a frame, and then obtains their skeletons (defined in Chapter 2). The user is then prompted to point with a mouse at the endpoints of segments that should be kept. Dots measuring two pixels on a side are put in these locations. Then, single-pixel spur-points are repeatedly removed until all that remains are the selected, pixel-thin lines that "summarize" the fossil walls (i.e., "spur points" are points that have only one neighbor which has the same color). Two dilations are then performed in order to thicken the walls again. The result is a smoothed, "schematic" version of the original contours. This can be repeated for all frames in the data set in order to obtain a relatively smooth surface in the final reconstruction. This procedure should only be used for specimens that have a constant, slender wall thickness throughout, and not for those which contain large interior spaces that were filled by calcite.

Morphology scripts

There are just three morphological scripts. These scripts are used to produce the mathematical model described in Chapter 3. It should be noted that the formulation that is implemented in these scripts differs slightly from the formulation supplied in the text. The main reason for this discrepancy is that the source code was assembled and developed in a piecemeal fashion according to practical considerations, and was then examined in order to produce a simplified mathematical description. (Please see the "caveat" supplied in the comments to the source code for the script "`model.m`" for additional instructions about the discrepancies.) The script `wall.m` contains the functions in the implementation that correspond to $\vec{\chi_1}(\psi, z)$ and $\vec{\chi_2}(\psi, z)$ (equations (3.16) and (3.24)), and can be used to draw an individual cross section of the cup, lip, or stem components. The input arguments of this script are the parameters of $\vec{\chi_1}(\psi, z)$ and $\vec{\chi_2}(\psi, z)$ discussed in Chapter 3. The script `wall.m` is called internally by another script, `model_maker.m`, whose arguments are the global and scale-invariant profile parameters, which are also described in Chapter 3. The script `model_maker.m` calculates all of the contour points and stores them in a Nuages contour file (`.cnt`). Nuages is then used to articulate a three-dimensional model. A third script, called `model.m` contains the definitions of all global and scale-invariant profile parameters which specify an individual model, and therefore must be run *before* `model_maker.m`. All three scripts are written for use with MATLAB (i.e., the Image Processing Toolbox is not used).

Source Code

This section contains the source code for all of the scripts described in this appendix. Comments begin with the percentage symbol "%." Lines that terminate with "!!!" are continued on the next line. (The symbol "!!!" must be removed from the code everywhere that it appears, and the corresponding lines must be re-connected in order for the script to run properly.) Within each sub-heading, scripts are listed in alphabetical order. It is recommended that the user type "help command" at the MATLAB prompt in order to determine the significance of an individiual command called command.

Database scripts

———delete_movie.m———

```
% Note that movies are being stored in the ./MOVIES directory.
specname = input('Enter specimen label (enclose in apostrophies): ','s');

delfile = sprintf('!rm -f MOVIES/%s*',specname);
eval(delfile);
```

———initialize.m———

```
fid1=fopen(jpg_filelist,'r');  % Opens the lists named in jpg_filelist
fid2=fopen(bmp_filelist,'r');  % and bmp_filelist.

files=cell(1000,2); % The cell-array in which the lists of files are stored.
V = cell(1000,20);  % The cell-array in which the contours are stored

i=1;
success=1;

% The following loop loads the file lists into an indexed array

while success==1
     [jpg_name,success]=fscanf(fid1,'%s\n',1);
     [bmp_name,success]=fscanf(fid2,'%s\n',1);
     files{i,1}=jpg_name;
     files{i,2}=bmp_name;
     i=i+1;
end

fclose(fid1);
%fclose(fid2);
numframes = i-2;

% Some variable initializations.

T=[];
U=0.3;
GAMMA=1;
framenum=0;
z=0;
CONTSAVED = 0;
CORRAPPLIED = 0;

framenum = FIRST-1;

fprintf(1,'\n\nSelect value for step. (Or run genframelist.m)\n\n');
```

———make.m———

```
frames=cell(1000,1);
```

```
[x,map5]=imread(files{Fn,1},'jpeg');  % Reads the first image file.
figure(50);
imshow(x);

fprintf(1,'Select reference points with mouse\n');

R=getrect;  % Prompts the user to select ROI.
delete(50);
y=imcrop(x,R);

i = FIRST;

% The following loop crops the ROI image from subsequent frames.

while i <= LAST
    [temp,map]=imread(files{i,1},'jpeg');
    CRPD=imcrop(temp,R);
    figure(10);
    s=sprintf('frame%d.jpg',i);
    imwrite(CRPD,s,'jpeg');
    imshow(CRPD);
    title(s);
    frames{i,1}=s;
    i = i + INCREMENT;
end
close(10);
```

————make_R.m————————————————————

```
% See make.m

frames=cell(1000,1);

i = FIRST;

while i <= LAST
    [temp,map]=imread(files{i,1},'jpeg');
    CRPD=imcrop(temp,R);
    figure(10);
    s=sprintf('frame%d.jpg',i);
    imwrite(CRPD,s,'jpeg');
    imshow(CRPD);
    title(s);
    frames{i,1}=s;
    i = i + INCREMENT;
end
close(10);
```

————save_movie.m————————————————————

```
% Note that movies are being stored in the ./MOVIES directory.
specname = input('Enter specimen label: ','s');

i = FIRST;
j = 1;

while i <= LAST
```

```
  newfile = sprintf('!cp frame%d.jpg MOVIES/%s%d.jpg',i,specname,j);

  eval(newfile);

  i = i + INCREMENT;
  j = j + 1;

end
```

——————**showALLframe.m**————————————————————

```
[temp,map]=imread(files{Fn,1},'jpg');
figure(199);
imshow(temp);
```

——————**show_frame.m**————————————————————

```
% Note that movies are being stored in the ./MOVIES directory.
specname = input('Enter specimen label: ','s');
framenumber = input('Enter frame that you wish to see: ');
newfile = sprintf('MOVIES/%s%d.jpg',specname,framenumber);
[temp,map]=imread(newfile,'jpeg');
imshow(temp);
```

——————**show_model.m**————————————————————

```
specname = input('Enter specimen label: ','s');

% The following four lines determine the names and
% quantity of files in ./MODELS with the inventor format
% extension *.iv.  The following loop displays these
% models with an external call to SceneViewer.

THEFILES = sprintf('ls -1 MODELS/%s*iv',specname);
[vvv,thefiles] = unix(THEFILES);
positions = findstr(thefiles,'.iv');
QUANTITY = size(positions);

empty = '';

for i=1:QUANTITY(2)

  temp_file = sscanf(thefiles,'%s^M');
  command = sprintf('!SceneViewer %s &',temp_file);
  eval(command);
  thefiles = strrep(thefiles,temp_file,empty);

end
```

——————**show_movie.m**————————————————————

```
% Note that movies are being stored in the ./MOVIES directory.
specname = input('Enter specimen label: ','s');

i = 1;

S = sprintf('ls -1 MOVIES/%s*jpg | wc',specname);
```

```
[s,w] = unix(S);

LASTNUM = sscanf(w,'%d');

for i=1:LASTNUM(1)

   newfile = sprintf('MOVIES/%s%d.jpg',specname,i);
   [temp,map]=imread(newfile,'jpeg');
   imshow(temp);

end
```

————show_pics.m————————————————————

```
specname = input('Enter specimen label: ','s');

% See show_model.m.  Notice that here the program xv is
% called externally, instead of SceneViewer.
% Note also that pictures are being stored in the ./PICS directory.

THEFILES = sprintf('ls -1 PICS/%s*jpg',specname);
[vvv,thefiles] = unix(THEFILES);

positions = findstr(thefiles,'.jpg');
QUANTITY = size(positions);

empty = '';

for i=1:QUANTITY(2)

  temp_file = sscanf(thefiles,'%s^M');
  command = sprintf('!xv %s &',temp_file);
  eval(command);
  thefiles = strrep(thefiles,temp_file,empty);

end
```

Processing scripts

————applycorr.m————————————————————

```
if (CORRAPPLIED == 0)  % A switch which keeps track of whether corrections
                       % have been applied.

T_new = cell(t,1);

delX = holes{framenum,2}(1);
delY = holes{framenum,2}(2);
S = holes{framenum,2}(3);
phi = holes{framenum,2}(4);
xA1 = holes{1,1}(1,1);
yA1 = holes{1,1}(1,2);

RR = [cos(-phi) -sin(-phi); sin(-phi) cos(-phi)];   % Rotation matrix.

for i = 1: z
     length = size(T{i,1});
     D = T{i,1}';
     del1 = [delX * ones(1,length(1)); delY * ones(1,length(1))];
     del2 = [(xA1-xmin) * ones(1,length(1)); (yA1-ymin) * ones(1,length(1))];
     DD = D + del1 - del2;
     DDD = S*(RR*DD) + del2;
     T_new{i,1} = DDD';
end

T_not = T;
T = T_new;
fastshow

CORRAPPLIED = 1;

elseif CORRAPPLIED == 1
       fprintf(1,'\n\nPositions Corrections have been applied already!\n\n');
end
```

————binize.m——————————————————————————

```
if HISTAPPLIED == 1  % A switch that indicates whether the histogram
                     % adjustment has been applied.

fprintf(1,'\n\nAdjust the threshold value U.\n\n');
U
bw=im2bw(IM,U);     % Converts the intensity image to a grey-scale image.
figure(2);
imshow(bw);
title('Binary Thresholded Image');

else fprintf(1,'\n\nApply the histogram adjustment!\n\n');

end
```

————collconts.m——————————————————————————

```
if SELECTED == 1              % A switch which can be used to indicate
                              % whether contours have been selected.

figure(5);
[D,H]=imcontour(bwsum,2);     % Obtains the contours and stores in the array D.
title('Contours Acquired');
E=D';
F=[];
F=cat(1,Mcoll,E);
Mcoll=F;                      % Adds the contours to the array ``Mcoll''
pause;
close(5);
AXIS_TEMP=axis;

% 't'  = Number of contours that can be stored
% 'MINLEN'  = Minimum length of contours (an input argument)

t=20

moves(framenum,1) = 0;
moves(framenum,2) = 0;

l=size(Mcoll);
T=cell(t,1);
```

```
b=1;
n=1;

Z(1)=Mcoll(1,1);
Z(2)=Mcoll(1,2);

% The following set of loops extracts the contour data from
% the array D, and removes redundant contours obtained by
% imcontour, as well as contours that are smaller than a
% minimum length.

for i=1:l(1)
        if abs(Mcoll(i,1)-Z(1)) > 2
           if abs(Mcoll(i,2)-Z(2)) > 1
              % Checks minimum length requirement:
              if b>MINLEN
              % Checks to see that size is not redundant
                 for k=1:n
                     c=size(T{k,1});
                     if c(1)+1==b
                        off=1;
                     end
                 end
                 if off==0
                    T{n,1}=N;
                    n=n+1;
                 end
              end
              N=0;
              b=1;
              Z(1)=Mcoll(i+1,1);
              Z(2)=Mcoll(i+1,2);
              off=0;
           end
        else N(b,1)=Mcoll(i,1);
             N(b,2)=Mcoll(i,2);
             Z(1)=Mcoll(i,1);
             Z(2)=Mcoll(i,2);
             b=b+1;
        end
    if n>t
       break;
    end
end
T
z=n-1;

else fprintf(1,'\n\nSelect binary objects first!\n\n');
end

showconts    % This script is listed elsewhere in the appendix, and is
             % used to display the contours.
```

———correct.m————————————————

```
% Number of registration holes per image:
numholes = 4
%------------------------------------------------------------
```

```
holes{1,2}(1) = 0;
holes{1,2}(2) = 0;
holes{1,2}(3) = 1;
holes{1,2}(4) = 0;

     xA1 = holes{1,1}(1,1);
     xA2 = holes{1,1}(2,1);
     xA3 = holes{1,1}(3,1);
     xA4 = holes{1,1}(4,1);
     yA1 = holes{1,1}(1,2);
     yA2 = holes{1,1}(2,2);
     yA3 = holes{1,1}(3,2);
     yA4 = holes{1,1}(4,2);

for i=2:numframes

     xi1 = holes{i,1}(1,1);
     xi2 = holes{i,1}(2,1);
     xi3 = holes{i,1}(3,1);
     xi4 = holes{i,1}(4,1);
     yi1 = holes{i,1}(1,2);
     yi2 = holes{i,1}(2,2);
     yi3 = holes{i,1}(3,2);
     yi4 = holes{i,1}(4,2);

% The following lines are used to calculate the corrections, by means
% of equations (2.1) through (2.8) in the text, and they are stored also
% in the cell array holes.

delX = xA1 - xi1;
delY = yA1 - yi1;

S2 = sqrt((xA1-xA2)^2 + (yA1-yA2)^2)/sqrt((xi1-xi2)^2 + (yi1-yi2)^2);
S3 = sqrt((xA1-xA3)^2 + (yA1-yA3)^2)/sqrt((xi1-xi3)^2 + (yi1-yi3)^2);
S4 = sqrt((xA1-xA4)^2 + (yA1-yA4)^2)/sqrt((xi1-xi4)^2 + (yi1-yi4)^2);

S = (S2 + S3 + S4)/3;

aaa = sqrt((xA1-xA2)^2 + (yA1-yA2)^2);
%bbb = sqrt((xA1-(S*(xi2 + delX - xA1)+xA1))^2 + (yA1-(S*(yi2+delY-yA1)+yA1))^2);
bbb = S*sqrt((xi1-xi2)^2 + (yi1-yi2)^2);

ccc = sqrt(( S*(xi2 + delX-xA1) + xA1 - xA2)^2 + (S*(yi2 + delY - yA1) + yA1 - yA2)^2);

phi = real((delY / abs(delY))*acos( (aaa^2 + bbb^2 - ccc^2)/(2*aaa*bbb) ));


holes{i,2}(1) = delX;
holes{i,2}(2) = delY;
holes{i,2}(3) = S;
holes{i,2}(4) = phi;

end
```

————delconts.m————————————————————

```
% This script is called internally by showconts.m
% Used to delete the contours listed in toBdel.
```

```
size2Bdel=size(toBdel);

for i=1: size2Bdel(2)
    T{toBdel(i),1}=[];
end

sizeT=size(T);

for h=1:size2Bdel(2)

   for i=1: sizeT(1)
       sizeTspec=size(T{i,1});
       if sizeTspec(1)==0
          for j=i: sizeT(1)
             if j+1 < sizeT(1)
                T{j,1}=T{j+1,1};
             end
          end
       break;
       end
    end
end

z=z-size2Bdel(2);
T
```

———enhance1.m———

```
bwnew1=bwsum;
bwnew2=bwmorph(bwnew1,'spur',1);  % Removes spur points.
bwnew3=bwmorph(bwnew2,'fill',1);  % Fills isolated points.

bwsum_old=bwsum;
bwsum=bwnew3;
figure(200);
imshow(bwsum);
title('Enhanced Binary Image for Contourization');
fprintf(1,'\n\nType "undo" to Undo.\n\n');
```

———enhance2.m———

```
se = ones(5,5);   % Defines the structural element used to perform the
                  % erosion and dilation --- a 5x5 array of 1s.
bw1 = bwsum;
bw2 = dilate(bw1,se);  % Performs the dilation
bw3 = erode(bw2,se);   % Performs the erosion

bwsum_old=bwsum;
bwsum=bw3;
figure(200);
imshow(bwsum);
title('Enhanced Binary Image for Contourization');
fprintf(1,'\n\nType "undo" to Undo.\n\n');
```

———erase.m———

```
figure(199);
imshow(bwsum);
title('Current Binary Image for Contourization');
```

```
HHH=0;
[mask,XX,YY]=roipoly;         % Prompts the user to select a region to be
                              % erased (overprinted with 0s).
NEW=mfilter2(HHH,bwsum,mask);
bwsum_old=bwsum;
bwsum=im2bw(NEW,0.5);
imshow(bwsum);
title('Current Binary Image for Contourization');
fprintf(1,'\n\nType "undo" to Undo.\n\n');
```

——————**eraser.m**——————————————————————————————

```
figure(219);
imshow(J);
title('Erase unwanted portions');
HHH=0;
[mask,XX,YY]=roipoly;         % Prompts the user to select a region to be
                              % erased (overprinted with 0s).
NEW=mfilter2(HHH,bwsum,mask);
bwsum_old=bwsum;
bwsum=im2bw(NEW,0.5);
figure(220);
imshow(bwsum);
title('Current Binary Image for Contourization');
fprintf(1,'\n\nType "undo" to Undo\n\n');
```

——————**fastshow.m**——————————————————————————————

```
% This script is called internally by showconts.m, and is used to
% rapidly display all current contours.

figure(299);
clf
axis(AXIS);
axis ij

for i=1:z
    line(T{i,1}(:,1), T{i,1}(:,2));
end

title('Current contours');
```

——————**focus.m**——————————————————————————————

```
% This script has several input arguments, where these are specified in
% leapframe.m and nextframe.m

h = fspecial('unsharp',UM);
J5 = filter2(h,J);            % Applies an unsharp masking.

if filter_type == 1
   J6 = wiener2(J5,[NR NR]);  % Applies a Weiner filter.
elseif filter_type == 2
   J6 = medfilt2(J5);         % Applies a median filter.
end

figure(355);
imshow(J5);
title('Unsharp Masking');
```

```
figure(356);
imshow(J6);
title('Noise-Reduced');

IM = J6;
```

————genframelist.m————————————————————

```
first = input('First frame number: ');
last = input('Last frame number: ');
number_of_frames = last-first+1;
clear framelist;


framelist(1) = first;
framelist(2) = last;
place = 3;

base = 3;

while base < number_of_frames,

  for j = 1: base-1,
  curr  = first + j*round((1/base)*(number_of_frames-1));

    if isempty(find(framelist==curr)) & (curr < last)
       framelist(place) = curr;
       place = place + 1;
    end
  end

  base = base+1;;
end

if length(framelist) ~= number_of_frames
   fprintf(1,'\n\n Error! NOT ALL FRAMES WERE ADDED TO framelist!\n\n');
end
```

————getpolys.m————————————————————

```
fprintf(1,'\n\nMust use a gray scale image for polygon selection.\n\n');
figure(140);
imshow(IM);                      % Displays the high-contrast image
title('Polygon Selection');
[bww,NN,MM]=roipoly;             % Prompts the user to outline shapes in the
                                 % high-contrast image.
BWW=double(bww);
bw=1-BWW;
imshow(bw);
title('Selected Polygon');
```

————histo.m————————————————————

```
LESSTHAN = 50;          % This is the value of y_min1 (see text)
GREATERTHAN = 2;        % This if the value of y_min2 (see text)

% Note that the variable I contains the grey-scale image.

A12=double(I(:,:,1));   % The Red color component
```

```
B12=double(I(:,:,2));   % The Green color component
C12=double(I(:,:,3));   % The Blue color component

D12=A12.*B12;
I2=D12/(256*256);       % Obtains the product image. (see text)

[CTS,BIN]=imhist(I2,3000);
sizeCTS=size(CTS);

% The following while loops obtain the lower and upper bounds for the
% histogram adjustment, using LESSTHAN (y_min1) and GREATERTHAN (y_min2).

i=1;
while CTS(i)<LESSTHAN
      i=i+1;
end
ll=i/sizeCTS(1);

j=i+20;
while CTS(j)>GREATERTHAN
     j=j+1;
end

hh=j/sizeCTS(1);

J=imadjust(I2,[ll;hh],[0;01],GAMMA);  % Performs the histogram adjustment.

figure(101);
imhist(J);
title('Contrast-adjusted Histogram');

% Display contrast-enhanced image and original image:

figure(102);
imshow(I2);
title('Original');
figure(103);
imshow(J);
title('Enhanced');

% Some additional variable initializations.

Mcoll=[];          % The contours will be stored in this variable.
IM=J;              % The high contrast images is passed to IM.
HISTAPPLIED = 1;   % Switch that indicates that the histogram adjustment
                   % has been applied.
```

————isolcont1.m————————————————————

```
% Enables the user to perform a region-specific histogram adjustment,
% where the bounds for this adjustment are specified manually.
% After this script is run, the script isolcont2.m is run.
% Then, after running binize.m and selector.m, the user should run reset_isol.m

figure(87);
imshow(IM);
title('ROI contrast adjustment');
[mask1,NNN,MMM]=roipoly;
OO=mfilter2(O,IM,mask1);
```

```
OOO=IM-OO;
imshow(OOO);
title('ROI contrast adjustment');
figure(88);
imhist(OOO);
%axis([0 1 0 1000]);
title('Enter values for ll, hh & GAMMA; then isolcont2');
fprintf(1,'\n\nEnter values for ll, hh, & GAMMA; then isolcont2\n\n');
```

————isolcont2.m————————————————————

```
% See the comments for isolcont1.m
% After running binize.m and selector.m, the user should run reset_isol.m

OOOO=imadjust(OOO,[ll;hh],[0;01],GAMMA);
close(88);
figure(87);
maskdb=double(mask1);
invert=1-maskdb;
OOOOO=OOOO+invert;
imshow(OOOOO);
title('ROI contrast adjustment');
IM=OOOOO;
```

————leapframe.m————————————————————

```
% This script moves to the next frame in the list generated by
% genframelist.m, and many important parameters are initialized.

if (CONTSAVED == 0)
   fprintf(1,'\n\nSave the contours before moving on!!\n\n');

   elseif (CORRAPPLIED == 0)
     fprintf(1,'\n\nApply the position corrections!!\n\n');
     else

%-----------------------------------------------------------------
% Parameter initializations:
%-----------------------------------------------------------------
new=1;                      %[Clean slate for selector.m]
Mcoll=[];                   %[Erase stored contours, initialize Mcoll]
close all;                  %[Close all windows]
MINLEN = 20;                %[Minimum allowed length of contours.]
filter_type = 1;            %[Noise reduction scheme for focus.m: 1=wiener, 2=median]
UM = 0.3;                   %[Unsharp masking alpha parameter for focus.m]
NR = 3;                     %[Noise reduction factor for wiener scheme in focus.m]
Old=T;                      %[Stores contours of previous frame for reference]
z_old=z;                    %[Store number of contours in previous frame]
THICKNESS = 1.0;            %[Thickness of line in linedraw.m tool]
CONTSAVED = 0;              %[Resetting switch (are contours saved?)]
CORRAPPLIED = 0;            %[Resetting switch (are positions corrected?)]
HISTAPPLIED = 0;            %[Resetting switch (is the histogram adjusted?)]
SELECTED = 0;               %[Resetting switch (are binary objects selected?)]

curr_index = find(framelist==framenum);
framenum = framelist(curr_index+1);

fprintf('\n\n Current frame is: %d/%d\n\n',curr_index+1,number_of_frames);
```

```
[I,map5]=imread(frames{framenum,1},'bmp');
imshow(I);

end
```

————linedraw.m————————————————————

```
% This script draws a rectangular line with length and orientation chosen by
% the user, and a THICKNESS specified in THICKNESS.  The user is prompted to
% select the endpoints of the line. The current binary image is shown in the
% background.

THICKNESS
figure(199);
imshow(bwsum);
title('Select end-points for the line:')
[Xx,Yy] = getpts;

% The following calculations obtain the orientation of the line.

mslope = (Yy(2) - Yy(1))/(Xx(2) - Xx(1));
mprime = -1/mslope;

x1 = Xx(1);
x2 = Xx(2);
y1 = Yy(1);
y2 = Yy(2);

b1 = y1 - mslope*x1;
b2 = y1 - mprime*x1;
b3 = y2 - mprime*x2;

A1 = mprime^2 + 1;
B1 = -2*(x1 + mprime*y1 - mprime*b2);
C1 = (x1^2 + y1^2 + b2^2) - (2*y1*b2 + THICKNESS);

x3 = (-B1 + sqrt(B1^2 - 4*A1*C1))/(2*A1);
y3 = mprime*x3 + b2;

A2 = mprime^2 + 1;
B2 = -2*(x2 + mprime*y2 - mprime*b3);
C2 = (x2^2 + y2^2 + b3^2) - (2*y2*b3 + THICKNESS);

x4 = (-B2 + sqrt(B2^2 - 4*A2*C2))/(2*A2);
y4 = mprime*x4 + b3;

Xxx = [x1 x3 x4 x2];
Yyy = [y1 y3 y4 y2];

linemask = roipoly(bw,Xxx,Yyy);
bw_new1 = double(bwsum) + double(linemask);
bw_new2 = im2bw(bw_new1, 0.5);
bwsum_old = bwsum;
bwsum = bw_new2;
figure(199);
imshow(bwsum);
title('Current Binary Image for Contourization');
fprintf(1,'\n\nType "undo" to Undo.\n\n');
```

———linedrawer.m———————————————

```
% This script draws a rectangular line with length and orientation chosen by
% the user, and a THICKNESS specified in THICKNESS.  The user is prompted to
% select the endpoints of the line. The current high-contrast image is shown
% in the background.

THICKNESS
figure(199);
imshow(J);
title('Select end-points for the line:')
[Xx,Yy] = getpts;

% The following calculations obtain the orientation of the line.

mslope = (Yy(2) - Yy(1))/(Xx(2) - Xx(1));
mprime = -1/mslope;

x1 = Xx(1);
x2 = Xx(2);
y1 = Yy(1);
y2 = Yy(2);

b1 = y1 - mslope*x1;
b2 = y1 - mprime*x1;
b3 = y2 - mprime*x2;

A1 = mprime^2 + 1;
B1 = -2*(x1 + mprime*y1 - mprime*b2);
C1 = (x1^2 + y1^2 + b2^2) - (2*y1*b2 + THICKNESS);

x3 = (-B1 + sqrt(B1^2 - 4*A1*C1))/(2*A1);
y3 = mprime*x3 + b2;

A2 = mprime^2 + 1;
B2 = -2*(x2 + mprime*y2 - mprime*b3);
C2 = (x2^2 + y2^2 + b3^2) - (2*y2*b3 + THICKNESS);

x4 = (-B2 + sqrt(B2^2 - 4*A2*C2))/(2*A2);
y4 = mprime*x4 + b3;

Xxx = [x1 x3 x4 x2];
Yyy = [y1 y3 y4 y2];

linemask = roipoly(bw,Xxx,Yyy);
bw_new1 = double(bwsum) + double(linemask);
bw_new2 = im2bw(bw_new1, 0.5);
bwsum_old = bwsum;
bwsum = bw_new2;
figure(199);
imshow(bwsum);
title('Current Binary Image for Contourization');
fprintf(1,'\n\nType "undo" to Undo.\n\n');
```

———linesdraw.m———————————————

```
% This script draws connected rectangular lines with length and orientation chosen by
% the user, and a THICKNESS specified in THICKNESS.  The user is prompted to
% select the endpoints of the lines in a sequence.  The current binary image is
% shown in the background.

THICKNESS
figure(199);
imshow(bwsum);
title('Select end-points for the line:')
[Xx,Yy] = getpts;
clear length

bwsum_old = bwsum;

for i = 1: length(Xx)-1

mslope = (Yy(i+1) - Yy(i))/(Xx(i+1) - Xx(i));
mprime = -1/mslope;

x1 = Xx(i);
x2 = Xx(i+1);
y1 = Yy(i);
y2 = Yy(i+1);

b1 = y1 - mslope*x1;
b2 = y1 - mprime*x1;
b3 = y2 - mprime*x2;

A1 = mprime^2 + 1;
B1 = -2*(x1 + mprime*y1 - mprime*b2);
C1 = (x1^2 + y1^2 + b2^2) - (2*y1*b2 + THICKNESS);

x3 = (-B1 + sqrt(B1^2 - 4*A1*C1))/(2*A1);
y3 = mprime*x3 + b2;

A2 = mprime^2 + 1;
B2 = -2*(x2 + mprime*y2 - mprime*b3);
C2 = (x2^2 + y2^2 + b3^2) - (2*y2*b3 + THICKNESS);

x4 = (-B2 + sqrt(B2^2 - 4*A2*C2))/(2*A2);
y4 = mprime*x4 + b3;

Xxx = [x1 x3 x4 x2];
Yyy = [y1 y3 y4 y2];

linemask = roipoly(bw,Xxx,Yyy);
bw_new1 = double(bwsum) + double(linemask);
bw_new2 = im2bw(bw_new1, 0.5);
%bwsum_old = bwsum;
bwsum = bw_new2;
figure(199);
imshow(bwsum);

end

title('Current Binary Image for Contourization');
fprintf(1,'\n\nType "undo" to Undo.\n\n');
```

———**linesdrawer.m**———————————————

```
% This script draws connected rectangular lines with length and orientation chosen by
% the user, and a THICKNESS specified in THICKNESS.  The user is prompted to
% select the endpoints of the lines in a sequence.  The current high-contrast image is
% shown in the background.

THICKNESS
figure(199);
imshow(J);
title('Select end-points for the line:')
[Xx,Yy] = getpts;
clear length

bwsum_old = bwsum;

for i = 1: length(Xx)-1

% The following calculations obtain the orientation of the line.

mslope = (Yy(i+1) - Yy(i))/(Xx(i+1) - Xx(i));
mprime = -1/mslope;

x1 = Xx(i);
x2 = Xx(i+1);
y1 = Yy(i);
y2 = Yy(i+1);

b1 = y1 - mslope*x1;
b2 = y1 - mprime*x1;
b3 = y2 - mprime*x2;

A1 = mprime^2 + 1;
B1 = -2*(x1 + mprime*y1 - mprime*b2);
C1 = (x1^2 + y1^2 + b2^2) - (2*y1*b2 + THICKNESS);

x3 = (-B1 + sqrt(B1^2 - 4*A1*C1))/(2*A1);
y3 = mprime*x3 + b2;

A2 = mprime^2 + 1;
B2 = -2*(x2 + mprime*y2 - mprime*b3);
C2 = (x2^2 + y2^2 + b3^2) - (2*y2*b3 + THICKNESS);

x4 = (-B2 + sqrt(B2^2 - 4*A2*C2))/(2*A2);
y4 = mprime*x4 + b3;

Xxx = [x1 x3 x4 x2];
Yyy = [y1 y3 y4 y2];

linemask = roipoly(bw,Xxx,Yyy);
bw_new1 = double(bwsum) + double(linemask);
bw_new2 = im2bw(bw_new1, 0.5);
%bwsum_old = bwsum;
bwsum = bw_new2;
figure(199);
imshow(bwsum);

end

title('Current Binary Image for Contourization');
fprintf(1,'\n\nType "undo" to Undo.\n\n');
```

———makestack.m———

```
% See the comments for make.m and make_R.m

frames=cell(1000,1);

i = FIRST;

while i <= LAST
    [temp,map]=imread(files{i,2},'bmp');
    CRPD=imcrop(temp,R);
    figure(10);
    s=sprintf('frame%d.bmp',i);
    imwrite(CRPD,s,'bmp');
    imshow(CRPD);
    title(s);
    frames{i,1}=s;
    i = i + 1;
end
close(10);

xmin = R(1);
ymin = R(2);

numberframes = LAST-FIRST;
```

———nextframe.m———

```
% This script moves to the next frame in the stack, at an increment ''step''
% from the current frame number.

if (CONTSAVED == 0)
   fprintf(1,'\n\nSave the contours before moving on!!\n\n');

   elseif (CORRAPPLIED == 0)
     fprintf(1,'\n\nApply the position corrections!!\n\n');
     else

%-----------------------------------------------------------------
% Parameter initializations:
%-----------------------------------------------------------------
new=1;                      %[Clean slate for selector.m]
Mcoll=[];                   %[Erase stored contours, initialize Mcoll]
close all;                  %[Close all windows]
MINLEN = 20;                %[Minimum allowed length of contours.]
filter_type = 1;            %[Noise reduction scheme for focus.m: 1=wiener, 2=median]
UM = 0.3;                   %[Unsharp masking alpha parameter for focus.m]
NR = 3;                     %[Noise reduction factor for wiener scheme in focus.m]
Old=T;                      %[Stores contours of previous frame for reference]
z_old=z;                    %[Store number of contours in previous frame]
THICKNESS = 1.0;            %[Thickness of line in linedraw.m tool]
CONTSAVED = 0;              %[Resetting switch (are contours saved?)]
CORRAPPLIED = 0;            %[Resetting switch (are positions corrected?)]
HISTAPPLIED = 0;            %[Resetting switch (is the histogram adjusted?)]
SELECTED = 0;               %[Resetting switch (are binary objects selected?)]

framenum=framenum+step     %[Advance by one step]

[I,map5]=imread(frames{framenum,1},'bmp');
```

```
imshow(I);

end
```

————pstore.m————————————————————

```
% Stores the contents of V in a *.cnt file for use with Nuages.
% (i.e., according to the format given earlier in this appendix)

totalsize = length(V);

savefile = input('Filename: ','s');
reply = input('Apply an altitude scale factor? (y/n) [n]: ','s');
switch reply
      case 'y', scalefactor = input('Enter altitude scale factor: ');
      otherwise, scalefactor = 1;
      end

reply = input('Has an altitude array been specified already? (y/n) [n]: ','s');
switch reply
      case 'y', isThereZ = 1;
      otherwise, isThereZ = 0;
end

fid = fopen(savefile,'w');

if isThereZ == 1
   SS = length(Z);
   elseif isThereZ == 0       % Counts the number of non-empty entries in V
   SS = 0;
   for kk = 1: totalsize
     if isempty(V{kk,1}) ~= 1
        SS = SS+1;
     end
   end
end

fprintf(fid,'S %d\n',SS);

currentframe = 1;

for kk = 1:SS,

while isempty(V{currentframe,1}) ==1
      currentframe = currentframe + 1;
end

if isThereZ == 0
   Z(kk) = scalefactor*currentframe;
end

message = sprintf(' Storing contours in section %d/%d...\n',kk,SS');
fprintf(1,message);

sizes = 0;
for sz = 1: 20,
  siz = size(V{currentframe,sz});
  if siz(1) == 0
     sz = sz-1;
     break;
  else
  sizes = sizes + siz(1);
  end
end

fprintf(fid,'v %d z %f\n',sizes,Z(kk));


for kkk=1:sz,
    fprintf(fid,'{\n');
    nsize = size(V{currentframe,kkk});
    for kkkk = 1: nsize(1),
    fprintf(fid,'%f %f\n',V{currentframe,kkk}(kkkk,1),V{currentframe,kkk}(kkkk,2));
    end
    fprintf(fid,'}\n');
end

currentframe = currentframe + 1;
end % for kk=...

fclose(fid);
```

————register.m——————————————————

```
% This script prompts the user to mark the positions of registration holes
% in each image of the stack, and records these positions in the cell array
% called ''holes.''

% Number of registration holes per image:
numholes = 4

holes=cell(1000,2);
more off

[x,map5]=imread(files{1,2},'bmp');
figure(50);
imshow(x);
zoom on;
zoom out;
fprintf(1,'\n\nSelect reference points with mouse. Left button zooms in.  !!!
       Right zooms out.\n\n');

i = 1;

s = sprintf('Frame Number %d',i);
for j=1:numholes
    switch j
       case 1,
       fprintf(1,'\n\n%s\n',s), input('UPPER LEFT --- ')
       case 2,
       input('UPPER RIGHT --- ')
       case 3,
       input('LOWER RIGHT --- ')
       case 4,
       input('LOWER LEFT --- ')
    end
    [X,Y]=getpts;
    holes{1,1}(j,1) = X;
```

```
   holes{1,1}(j,2) = Y;
   zoom out;
end
   s1 = sprintf('X values (should be ~equal) = {%f, %f}; {%f, %f}', holes{i,1}(1,1), !!!
        holes{i,1}(4,1), holes{i,1}(2,1), holes{i,1}(3,1)');
   s2 = sprintf('Y values (should be ~equal) = {%f, %f}; {%f, %f}', holes{i,1}(1,2), !!!
        holes{i,1}(2,2), holes{i,1}(3,2), holes{i,1}(4,2)');

   fprintf(1,'%s\n%s\n\n',s1,s2);

for i=2:numframes

   [x,map5]=imread(files{i,2},'bmp');
   figure(50);
   imshow(x);

   for j=1:numholes
   switch j
      case 1,
      s = sprintf('Frame Number %d',i), fprintf(1,'\n\n%s\n',s), input('UPPER LEFT --- ')
      case 2,
      input('UPPER RIGHT --- ')
      case 3,
      input('LOWER RIGHT --- ')
      case 4,
      input('LOWER LEFT --- ')
   end
   [X,Y]=getpts;
   holes{i,1}(j,1) = X;
   holes{i,1}(j,2) = Y;
   zoom out;
   end

   s1 = sprintf('X values (should be ~equal) = {%f, %f}; {%f, %f}', holes{i,1}(1,1), !!!
        holes{i,1}(4,1), holes{i,1}(2,1), holes{i,1}(3,1)');
   s2 = sprintf('Y values (should be ~equal) = {%f, %f}; {%f, %f}', holes{i,1}(1,2), !!!
        holes{i,1}(2,2), holes{i,1}(3,2), holes{i,1}(4,2)');

   fprintf(1,'%s\n%s\n\n',s1,s2);


end

more on
```

————rehist.m————————————————————

```
% This script enables the user to perform a manual histogram adjustment
% (i.e., one for which the upper and lower bounds are specified manually)
% See also the comments for histo.m.

LESSTHAN = 100;
GREATERTHAN = 5;

LL = input('\n\nSet lower bound = ');
HH = input('\nSet upper bound = ');

J7=imadjust(J,[LL;HH],[0;01],GAMMA);
J = J7;
```

```
figure(103);
imshow(J);
title('Contrast-adjusted Histogram, 2nd Pass');
```

——————reset_isol.m——————————————————————————

```
% See the comments for isolcont1.m and isolcont2.m.  This script merely
% retrieves the original high-contrast image.  The variable IM which stores
% the current high contrast image is modified through the isolcont1.m and
% isolcont2.m procedures.

IM=J;
```

——————seeall.m——————————————————————————

```
for i=1:numberframes

   newfile = sprintf('frame%d.bmp',i);
   [temp,map]=imread(newfile,'bmp');
   imshow(temp);

end
```

——————seeframe.m——————————————————————————

```
[temp,map]=imread(frames{Fn,1},'bmp');
figure(99);
imshow(temp);
```

——————selector.m——————————————————————————

```
SELECTED = 1;  % A switch used to determine whether binary objects have
               % been selected already.

bwA=bw;
figure;
imshow(bwA)
title('Binary Object Selection');
bwB=bwfill(8);      % Performs the binary flood-filling operation.
close;
AA=double(bwA);
BB=double(bwB);
CC=BB-AA;           % Subtracts the original image from the flood-filled image
                    % in order to obtain just the flood-filled region.

if new==0
   BWsum=BWsum+CC;
else
   BWsum=CC;
end

figure(199);

bwsum=im2bw(BWsum,1);
imshow(bwsum);
title('Current Binary Image for Contourization')
new=0;
```

————showconts.m————————————————————

```
% This script is called internally by collconts.m, and is used to display and
% select the contours obtained by that procedure.

figure(299);

clf
axis(AXIS);
axis ij
toBdel = [ ];

for i=1:z
    line(T{i,1}(:,1), T{i,1}(:,2));
    i
    reply=input('Keep this contour --- y or n? [Y] :','s');
    switch reply
           case 'n', new = [toBdel,i], toBdel = new
    end
end

title('Current contours');

delconts  % This script eliminates from T the contours listed in toBdel.
fastshow  % This script shows all of the selected contours, at once.
```

————storeconts.m————————————————————

```
% Store the current contours in an entry of the cell-array V.

for i=1:z
   V{framenum,i} = T{i,1};
   Z(framenum) = framenum;
end

CONTSAVED = 1;  % Switch that indicates that the contours have been stored.
```

————undo.m————————————————————

```
% Undoes certain previous operations by recalling an older version of the image.

bwsum=bwsum_old;
figure(199)
imshow(bwsum);
title('Current Binary Image for Contourization');
```

Post-processing scripts

————pedit.m————————————————————

```
% This script generates a binary image of the contours stored
% in the entry of the cell-array V numbered framenum, and then enables the
% user to edit the contour with the customary processing tools
% (called internally), and then re-collects and stores these contours
% in the cell array V.

MINLEN = 0;           % Minimum-allowed  contour length
THICKNESS = 3;        % Thickness of lines for line-drawing tools.
SCALE_FACTOR = 1;     % Scale factor by which binary image of
                      % contours should be scaled.
SIZE = 400;           % Dimension of square binary image of contours.
more off;

startframe = input('Starting frame: ');
framenum = startframe;
gonextframe = 1;

while gonextframe == 1,

  if isempty(V{framenum,1}) ~= 1

     bw = zeros(SCALE_FACTOR*SIZE);
     bwcoll = zeros(SCALE_FACTOR*SIZE);

     figure(40);

     % The following loop obtains the number of contours in the cell-array
     % index that is numbered framenum.

     for sz = 1: 20
        siz = size(V{framenum,sz});
        if siz(1) == 0
        sz = sz-1;
        break;
        end
     end

     for i=1:sz

       XXX = V{framenum,i}(:,1);
       YYY = V{framenum,i}(:,2);

       % The following line makes a mask in the shape of the contours in
       % the frame framenum.

       shapemask = roipoly(bw,SCALE_FACTOR*XXX,SCALE_FACTOR*YYY);
       bwcoll = double(bwcoll) + double(shapemask);
     end

     figure(40);
     imshow(bwcoll);

     bwsum = bwcoll;
     continue = 0;
     goahead=1;

     % The next loop prompts the user to select either one of several
     % editing options, which make internal calls to the scripts
     % enhance.m, erase.m, linesdraw.m, etc.

     while continue == 0;
        reply = input('[S]kip frame, [R]esize next, [E]rase tool, E[n]hance, !!!
                [L]ine tool, [U]ndo, [C]ontinue: [Continue] : ','s');
        switch reply
          case 'e', erase;
          case 'n', enhance2;
          case 'l', linesdraw;
          case 'r', SIZE = input('Enter windowsize (in pixels): ');
          case 's', goahead = 0; continue = 1;
          case 'u', undo;
          otherwise, continue = 1;
        end
     end


    % The remainder of this script obtains contours from the edited
    % binary image, in the usual way, with much of this code borrowed
    % from collconts.m.

if goahead == 1

     bwsum_old = bwsum;
     off = 1;
     z_old = 0;
     SELECTED = 1;
     Mcoll = [];
     collconts;

     continue = 0;
     while continue == 0,
        reply = input('[R]eselect contours or [S]ave contours : [Save] : ','s');
        switch reply
          case 'r', showconts;
          otherwise, for k = 1: sz
                     V{framenum,k} = [];
                     end, storeconts, continue = 1; Old = T;
        end
     end

end % if goahead == 1

gohead = 1;

  end  %end of if isempty(V{framenum,1}) ~= 1 ...

  framenum = framenum+1;

end
```

————peruse.m————————————————

```
% Used to peruse the contours stored in the cell array V.  Contours are
% displayed in solid red as a movie, which pauses at each frame.  It is
% highly recommended that Figure 30, in which the contours are displayed,  be
% radically enlarged in order to see the contours clearly.

figure(30);

continue=1;
direction=1;
```

```
jjj = input('Starting frame: ');

if jjj > 1
   jjj = jjj-1;
end

while continue == 1,
jjj = jjj + direction;

if isempty(V{jjj})~=1

clf;
cont = sprintf('%d 1 b 10 0 0',jjj);
pview

% The following loop obtains the number of frames in the current
% entry of the cell array V.

for sz = 1: 20
  siz = size(V{jjj,sz});
  if siz(1) == 0
     sz = sz-1;
     break;
  end
end

sss = sprintf('Frame: %d    Contours: %d',jjj,sz);
title(sss);
reply=input('[b]ackward, [q]uit, or [f]orward? [forward] :','s');
    switch reply
           case 'b', direction = -1;
           case 'q', continue = 0;
           otherwise, direction = 1;
    end

end
end
```

————pload.m————————————————————

```
% Loads contours from a *.cnt Nuages contour file into the cell-array V.

cntfile = input('Enter the Nuages contour filename: ','s');
fid=fopen(cntfile,'r');        % Opens the contour file.
S = fscanf(fid,'%*c %d');      % Reads the first line, and thereby obtains
                               % the total number of frames S.
V = cell(S,20);                % A cell array is declared that can contain as
                               % as many as 12 contours for each of S frames.

% The remainder of this script parses the contour file format that is given
% earlier in this appendix.

for j=1:S

   M = fscanf(fid,'%c %d %c %f');
   if j > 1
      M = fscanf(fid,'%c %d %c %f');
   end
   vv = M(2);
   Z(j) = M(4);
   k = 0;
   i = 1;
   h = 1;
   G = fscanf(fid,'%c',1);

   while k < vv,
         h = 1;
         count = 2;
         while count == 2
               [L,count] = fscanf(fid,'%f %f',2);
               if count  == 2
                  V{j,i}(h,1) = L(1);
                  V{j,i}(h,2) = L(2);
                  k = k+1;
               end
               h = h + 1;
         end
    if k < vv
       G = fscanf(fid,'%c %c',2);
    end
    i = i+1;
    end

G = fscanf(fid,'%c',1);
end
```

———pmover.m———————————————————

```
% Used to adjust the relative positions of contours. For each frame, the
% next (higher) contour is displayed in blue outline, the current contour
% in solid red, and the previous contour in solid green. The position of the
% red contour can be adjusted using the 4 directions represented by:
%
%                         [r]
%                      [d]   [g]
%                         [v]
%
% It is strongly recommended that Figure 30 be enlarged in order that
% the distinction between the three contours is easily apparent.  Note that
% the contours are displayed using internal calls to pview.m


figure(30);

continue=1;
direction=1;

framenum = input('Starting frame: ');

while continue == 1,

if isempty(V{framenum})~=1

clf;

stay =1 ;

while stay == 1,
lastone = framenum - 1;
    while isempty(V{lastone,1}) == 1
          lastone = lastone - 1;
          if lastone == 0
             break;
          end
    end

% The following loop finds the first non-empty cell array.

nextone = framenum + 1;
    while isempty(V{nextone,1});
        if nextone == 999
             break;
          end
          nextone = nextone +1;
       end

clf
cont = sprintf('%d 1 b 0 10 0',lastone);  % Specifies the arguments to pview.m,
                                          % which is used to display the contours.

if lastone ~= 0
   pview
end

cont = sprintf('%d 1 b 10 0 0',framenum);
pview
cont = sprintf('%d 0 b 0 0 0',nextone);

if nextone ~= 999
   pview
end

for sz = 1: 20                 % Determines the number of contours for the
  siz = size(V{framenum,sz});  % current contour-layer.
  if siz(1) == 0
     sz = sz-1;
     break;
  end
end


sss = sprintf('Frame: %d    Contours: %d',framenum,sz);
title(sss);
reply=input('[r,d,v,g]move, [b]ackward, [q]uit, or [f]orward? [forward] :','s');
    switch reply
           case 'b', rt = 0, dn = 0, direction = -1, stay=0;
           case 'q', rt = 0, dn = 0, continue=0, stay=0;
           case 'v', rt = 0, dn = 1, stay=1;
           case 'd', rt = -1, dn = 0, stay=1;
           case 'r', dn = -1, rt = 0, stay=1;
           case 'g', rt = 1, dn = 0, stay=1;
           otherwise, rt = 0, dn = 0, direction = 1, stay=0;
    end


for jjj=1:sz
```

```
       tmp=V{framenum,jjj}(:,1);
       V{framenum,jjj}(:,1)=tmp+rt;
       tmp=V{framenum,jjj}(:,2);
       V{framenum,jjj}(:,2)=tmp+dn;
end



  end
 end  % if isempty(...

framenum = framenum + direction;
end  % while contine ==
```

————pswap.m————————————————————

```
% This script is not listed in the appendix.
%
% It is used to copy the cell array VV to V so that the p*.m tools can be used to
% maniupate the refined-skeleton contours.  (That is, notice that skeletons.m
% stores the final results to the cell-array VV).

if pswapped == 1
   V = V_backup;
   pswapped = 0;
elseif pswapped == 0
    V_backup = V;
    V = VV;
    pswapped = 1;
end
```

————pview.m————————————————————

```
% Command for displaying a single contour stored in the cell-array V. Takes as
% in input argument the string cont, formatted as follows:
%
% cont = '[frame_number] [linetype] [edgecolor] [r] [g] [b]'
%
% where [linetype] = 0 (for edge) and 1 (for solid)
%       [edgecolor] = rgbcmywk (i.e,. one of these)
%       [r] [g] [b] = each is a value between 1 and 10
%

zoom on;
axis square;
axis off;

% The next set of commands is used to parse the input argument cont.

INSTRUCT = sscanf(cont,'%d %d %c %d %d %d');

framenumber = INSTRUCT(1);
fill_type = INSTRUCT(2);
color0    = INSTRUCT(3);
color1s    = sprintf('[ %d %d %d ]',INSTRUCT(4)/10,INSTRUCT(5)/10,INSTRUCT(6)/10);
color1 = eval(color1s);
color2    = sprintf('%c-',color0);
for sz = 1: 20
  siz = size(V{framenumber,sz});
  if siz(1) == 0
     sz = sz-1;
     break;
  end
end

hold on

% The following loop uses the fill and plot commands to draw the contours.

   for i = 1:sz

   if ( fill_type == 1 )
       fill(V{framenumber,i}(:,1),V{framenumber,i}(:,2),color1);
   elseif ( fill_type == 0 )
       plot(V{framenumber,i}(:,1),V{framenumber,i}(:,2),color2);
   else
      fprintf(1,'\n\n Fill type must be 1 or 0. \n\n');
      break
   end
   axis(AXIS);
   axis ij;
   end
```

————skeleton_storeconts.m————————————————

```
for i=1:z
   VV{framenum,i} = T{i,1};
   Z(framenum) = framenum;
end

CONTSAVED = 1;
```

————skeletonize.m————————————————

```
% This script is called internally by skeletons.m, and accomplishes the
% skelton refinement that is "managed" by skeletons.m.

bw1 = bwmorph(bwsum,'skel',Inf);  % Performs the skeletonization.
go_on = 0;
bw1_oldest = bw1;

while go_on == 0,

DOTSIZE = 5;

figure(199);
imshow(bw1);
title('Select end-points of lines to be preserved:');
[X,Y] = ginput;

for i = 1:length(X)

% The following lines create a large dot (a mask) and prints them over the
% end-points that the user has selected.

Xxx(1) = X(i) ;
Yyy(1) = Y(i) + 1;
Xxx(2) = X(i) + 1;
```

```
Yyy(2) = Y(i) + 1;
Xxx(3) = X(i) + 1;
Yyy(3) = Y(i);
Xxx(4) = X(i) + 1;
Yyy(4) = Y(i) - 1;
Xxx(5) = X(i);
Yyy(5) = Y(i) - 1;
Xxx(6) = X(i) - 1;
Yyy(6) = Y(i) - 1;
Xxx(7) = X(i) - 1;
Yyy(7) = Y(i);
Xxx(8) = X(i) - 1;
Yyy(8) = Y(i) + 1;

linemask = roipoly(bw,Xxx,Yyy);
bw_new1 = double(bw1) + double(linemask);
bw_new2 = im2bw(bw_new1, 0.5);
bw1_old = bw1;
bw1 = bw_new2;
figure(199);
imshow(bw1);
title('Current Binary Image for Contourization');
end

reply = input('[R]edraw end-points or [C]ontinue? [Continue]: ','s');

switch reply
     case 'r', go_on=0; bw1 = bw1_oldest;
     otherwise, go_on=1;
end

end  % while go_on == 0,

% The following lines remove spur points, in order to eliminate the
% remaining, unwanted segments.

bw2 = bwmorph(bw1,'spur',Inf);
repeatlut = 2;
f = inline('sum(x(:)) >= 1');
lut = makelut(f,2);

for i=1 : repeatlut
    bw2 = applylut(bw2,lut);
end

figure(41);
imshow(bw2);
bwsum = bw2;
```

——————skeletons.m——————————————————————

```
% Skeletonization of contour data by summarizing wall-contours in terms of
% smooth lines of constant thickness.  This is accomplished by first reproducing
% the contour patters of each wall in a binary image file, and then using
% skeleton erosion to obtain a thin line that summarizes the wall.  Then large
% points are placed at the ends of each line in order that, when spur-erosion is
% applied, those end-points and the points connecting them, remain.  Then a simple
% look-up operation is applied to thicken the lines.  The contours are then
% collected and stored in the cell-array variable VV.  The cell-array VV can be
% stored to a *cnt file using first pswap.m to move VV to V, and then pstore.m.

THICKNESS = 3;
SCALE_FACTOR = 1;
more off;

reply = input('Initialize the dataset variable VV? (Y/N) [N]: ','s');

       switch reply
          case 'y', VV = cell(1000,20);
          otherwise, continue = 1;
        end

startframe = input('Starting frame: ');
framenum = startframe;
gonextframe = 1;

while gonextframe == 1,

  if isempty(V{framenum,1}) ~= 1

     % The following lines generate a binary image.

     bw = zeros(SCALE_FACTOR*400);
     bwcoll = zeros(SCALE_FACTOR*400);

     figure(40);

     for sz = 1: 20
        siz = size(V{framenum,sz});
        if siz(1) == 0
        sz = sz-1;
        break;
        end
     end

     for i=1:sz

       XXX = V{framenum,i}(:,1);
       YYY = V{framenum,i}(:,2);

       % The following lines generate a binary image from the current
       % set of contours stored in V.

       shapemask = roipoly(bw,SCALE_FACTOR*XXX,SCALE_FACTOR*YYY);
       reply = input('[A]dd or [S]ubtract the next contour: [Add] ','s');
       switch reply
          case 's', signn = -1;
          otherwise, signn = 1;
        end
       bwcoll = double(bwcoll) + signn*double(shapemask);
       figure(40);
       imshow(bwcoll);
     end

     figure(40);
     imshow(bwcoll);

     bwsum = bwcoll;
     continue = 0;
     goahead=1;

     % The following loop enables the user to edit the contours using the
     % tools erase.m and linesdraw.m, in order to connect adjacent segments,
     % for example.

     while continue == 0,
        reply = input('[S]kip this frame, [E]rase tool, [L]ine tool, [U]ndo, !!!
                [C]ontinue: [Continue] : ','s');
        switch reply
          case 'e', erase;
          case 'l', linesdraw;
          case 's', goahead = 0; continue = 1;
          case 'u', undo;
          otherwise, continue = 1;
        end
     end

if goahead == 0
for nnn = 1: sz
   VV{framenum,nnn} = V{framenum,nnn};
        end
end

if goahead == 1

     bwsum_old = bwsum;

     skeletonize;
     enhance2;
     continue = 0;
     while continue == 0,
        reply = input('[R]eskeletonize or [C]ontinue : [Continue] : ','s');
        switch reply
          case 'r', bwsum = bwsum_old; skeletonize;
          otherwise, continue = 1;
        end
     end

     Mcoll = [];
     collconts;

     continue = 0;
     while continue == 0,
        reply = input('[R]eselect contours or [S]ave contours : [Save] : ','s');
        switch reply
          case 'r', showconts;
          otherwise, skeleton_storeconts, continue = 1; Old = T;
        end
     end

end % if goahead == 1

gohead = 1;

  end  %end of if isempty(V{framenum,1}) ~= 1 ...
  framenum = framenum+1;

end
```

Morphology scripts

————model.m——————————————————————

```
% Important caveat regarding the implemented formulation of the mathematical model:
% ------------------------------------------------------------------------------
%
% This script contains the definitions for the global dimension parameters and
% profile parameters for the model labeled ''A'' in Chapter 3 of the thesis.
% In fact, the implementation here (in this script and the two other morphological
% scripts, model_maker.m and wall.m) is not identical to the mathematical formulation
% supplied in that chapter.  The resulting model, however, is equivalent (except for
% the implementation of equations 3.22 and 3.33, where a simplified version was used).
% That is, the mathematical description in Chapter 3 describes the resulting model,
% even if the description itself is not implemented.  The main reason for this
% discrepancy is that a source code is assembled and developed in a piecemeal fashion
% and with mainly practical considerations in mind.  It is was practical, however,
% to use the implemented forumation for the purposes of explanation.  That is,
% after a successful model was obtained, the code was examined in order to obtain
% a simplified mathematical description: i.e., simplified for the purposes of explanation.
% The code was not then subsequently modified in order to match the text's formulation
% exactly. Instead, it bears a close semblance.
%
% The primary differences reside in the definition, specification and names of
% profile- and dimension parameters, and consequent changes for the shape of
% equations 3.16 and 3.24.  The source code here is provided in order to assist the
% reader with an implementation of the mathematical model, as some devices will be
% helpful.  If the reader chooses to implement the model by herself, she is encouraged
% to implement the description in chapter 3 directly using her own source code,
% with references to this code for clues about how certain mechanics of the implementation
% are realized.  Alternatively, this code may be used in its current form to generate
% an accurate mathematical model, which can be described using the formulation in Chapter 3.

%-----------------------------------------------------------------------------
% Variable initializations:
%-----------------------------------------------------------------------------

MODEL = cell(1000,20);  % Cell-array in which all specifications are stored.
double Z;

%-----------------------------------------------------------------------------
% Universal parameter values:
%-----------------------------------------------------------------------------
SMOOTH_TYPE      = 2;
SMOOTH_POINTS    = 10;
NUMLAYERS        = 605;
dT               = 2/1E2;    % Wall Resolution
dD               = 1/1E3;    % Profile Resolution

%-----------------------------------------------------------------------------
% Model specification
%-----------------------------------------------------------------------------

% All profiles are generated for an interval from [0,1] and will be scaled
% using an interpolation routine to find values for each layer in the model.
% -- Morphological parameters ----------------------------------------------------

NUMSIDES          = 7;          % Number of sides.
NUMWALLS          = 6;          % Number of walls.
THICK             = 0.1;        % Wall thickness throughout.
MAX_RAD           = 6;          % Maximum radius of exterior wall.
OVAL              = 0.6;        % Fraction of side-wall occupied by oval holes.
SIDECAV           = [0.01 0.99];% Interval-fraction where side-curvature is 1.
CORNCAV           = [0.3 0.75]; % Interval-fraction where corner-curvature is MAX_CORNCAVITY
                                % (i.e., as fraction of cup height from top).
MAX_CORNCAVITY    = 0.50;       % Maximum corner curvature.
OPENING_RADIUS    = 0.55;       % Fractional position of opening radius
                                % (as fraction of MAX_RAD).
TOPLIP_SIZE       = 0.12;       % Fractional size of inner lip (as fraction of MAX_RAD)
HOLE_POSITION     = [0.3 0.75]; % Fractional position of holes from top of cup.
TOPLIP_POSITION   = [0 0.09];   % Fractional vertical range/position of large lip.
MAXDIAM2HEIGHT    = 1.24;       % Maximum diameter to height-of-cup ratio.
STEM2CUP          = 1.7;        % Length of stem to height-of-cup ratio
MIN_CORNSIZE      = 0.2;        % Minimum size of corner throughout.
BASE_STEMRAD      = 1.1597;     % Inner radius at base of stem (i.e., where it attaches)
XYSIZE            = 10;         % Horizontal extent of stem's XY trajectory in top-view.
STEM_ATTACH_OFFSET= [0 0];      % Position of basal stem attachment;
MIN_SSLOPE        = -0.1;       % Minimum side slope.
CSLOPE            = 0;          % Constant (zero) corner slope throughout.
DEPWIDTH          = 0.07;       % Fractional size of side-slope-induced depression
                                % surrounding holes.
RADPROFILE_SCALE  = 0.1;        % Fraction of BASE_STEMRAD over which radial profile
                                % is allowed to vary.

% -- Main exterior wall of cup: ------------------------------------------------

D = [0:dD:1];  % Standard domain interval for profiles.

%...(Radial profile)...........................................................

P = [

  369.5000
  373.5000
  383.5000
  387.5000
  392.5000
  401.5000
  409.5000
  425.5000
  442.5000
  459.5000
  474.5000
  489.5000
  502.5000
  515.5000
  535.5000
  555.5000
  573.5000
  594.5000
  614.5000
  630.5000
  640.5000
  650.5000
  655.5000
  665.5000
  670.5000
  672.5000
  675.5000];


Q = [

  173.0411
  153.0600
  142.0705
  129.0828
  122.0894
  113.0980
   98.1122
   81.1283
   73.1359
   66.1426
   61.1473
   55.1530
   52.1558
   47.1606
   45.1625
   43.1644
   42.1653
   46.1615
   49.1587
   52.1558
   60.1482
   65.1435
   74.1350
   84.1255
   94.1160
  104.1065
  113.0980];

Q = 250 - Q;
Q = Q - 0.57*min(Q);  % 0.57 determined from profile image;
                      % This ensures that the stem aperture is correct, where it
                      % attaches to the base of the cup.

P = 800 - P;

P = P' - min(P);
P = P/max(P);
Q = Q'/max(Q);

POLY_DEGREE = 10;
c =  polyfit(P,Q,POLY_DEGREE);

switch POLY_DEGREE
 case 2, Q=c(1)*D.^2+c(2)*D.^1+c(3);
 case 4, Q=c(1)*D.^4+c(2)*D.^3+c(3)*D.^2+c(4)*D+c(5);
 case 6, Q=c(1)*D.^6+c(2)*D.^5+c(3)*D.^4+c(4)*D.^3+c(5)*D.^2+c(6)*D+c(7);
 case 8, Q=c(1)*D.^8+c(2)*D.^7+c(3)*D.^6+c(4)*D.^5+c(5)*D.^4+c(6)*D.^3+c(7)*D.^2 !!!
         +c(8)*D+c(9);
 case 10, Q=c(1)*D.^10+c(2)*D.^9+c(3)*D.^8+c(4)*D.^7+c(5)*D.^6+c(6)*D.^5+c(7)*D.^4 !!!
```

```
        +c(8)*D.^3 +c(9)*D.^2+c(10)*D+c(11);
end

RAD_PROFILE = Q/max(Q);

figure(11);
clf;
subplot(4,3,1), plot(D,Q,'r-');
axis([0 1 0 1]);
title('Cup: radial profile');

%...(Hole-half-diameter profile)...........................................

Q = zeros(1,length(D));
FIRSTPT = floor(HOLE_POSITION(1)*length(D));
LASTPT  = floor(HOLE_POSITION(2)*length(D));
D_TEMP = linspace(0,1,LASTPT-FIRSTPT+1);

% Ellipse centered at 0.5 + HOLE_POSITION(1), minor axis OVAL:

Q(FIRSTPT:LASTPT) = Q(FIRSTPT:LASTPT) + (OVAL^2*(1-((0.5-D_TEMP)/(0.5)).^2)).^(0.5);

HOLE_PROFILE = Q;

figure(11);
subplot(4,3,2), plot(D,Q,'m-');
axis([0 1 0 1]);
title('Cup: hole-half-diameter profile');

%...(Side curvature profile)...............................................

Q = exp((5/SIDECAV(1))*D-5);
i = 0;
for i = 1:length(D)
    if D(i) > SIDECAV(1) & D(i) < SIDECAV(2)
       Q(i) = 1;
    elseif D(i) >= SIDECAV(2)
       Q(i) = (1-exp(5/(1-SIDECAV(2))*(D(i)-SIDECAV(2))-5));
    end
end

CAVS_PROFILE = Q;

figure(11);
subplot(4,3,3), plot(D,Q,'g-');
axis([0 1 0 1]);
title('Cup: side curvature profile');

%...(Corner curvature profile).............................................

Q = MAX_CORNCAVITY*exp((5/CORNCAV(1))*D-5);
i = 0;
for i = 1:length(D)
    if D(i) > CORNCAV(1) & D(i) < CORNCAV(2)
       Q(i) = 0.5;
    elseif D(i) >= CORNCAV(2)
        Q(i) = -(MAX_CORNCAVITY/(1-CORNCAV(2)))*D(i) + MAX_CORNCAVITY/(1-CORNCAV(2));
    end
end
```

```
CAVC_PROFILE = Q;

figure(11);
subplot(4,3,4), plot(D,Q,'g-');
axis([0 1 0 1]);
title('Cup: corner curvature profile');

%..(Corner size profile)..............................................................

Q = ones(1,length(D));
FIRSTPT = floor((HOLE_POSITION(1)-DEPWIDTH)*length(D));
LASTPT  = floor((HOLE_POSITION(2)+DEPWIDTH)*length(D));
D_TEMP = linspace(0,1,LASTPT-FIRSTPT+1);

% 1-Ellipse centered at 0.5 + HOLE_POSITION(1), minor axis OVAL:

Q(FIRSTPT:LASTPT) = Q(FIRSTPT:LASTPT) - ((1-MIN_CORNSIZE)^2*(1 - !!!
                    ((0.5-D_TEMP)/(0.5)).^2)).^(0.5);

CORNER_PROFILE = Q;

figure(11);
subplot(4,3,5), plot(D,Q,'c-');
axis([0 1 0 1]);
title('Cup: corner diameter profile');

%..(Wall thickness profile)...........................................................

Q = THICK*ones(1,length(D));
THICK_PROFILE = Q;
figure(11);
subplot(4,3,6), plot(D,Q,'b-');
axis([0 1 0 1]);
title('Cup: wall thickness profile');

%..(Side slope profile)...............................................................

Q = zeros(1,length(D));
FIRSTPT = floor((HOLE_POSITION(1)-(DEPWIDTH))*length(D));
LASTPT = floor((HOLE_POSITION(2)+(DEPWIDTH))*length(D));
D_TEMP = linspace(0,1,LASTPT-FIRSTPT+1);

% Simple sloping lines across midpoint of holes.

FIRSTHALF_INDEX = floor(length(D_TEMP)/2);
TOT = ((HOLE_POSITION(2)+ DEPWIDTH) - (HOLE_POSITION(1)-(DEPWIDTH)));
MID = (HOLE_POSITION(1)-DEPWIDTH) + ((HOLE_POSITION(2)+ DEPWIDTH) - (HOLE_POSITION(1)- !!!
      (DEPWIDTH)))/2;
Q1 = 2*MIN_SSLOPE*D_TEMP(1:FIRSTHALF_INDEX);
Q2 = 2*MIN_SSLOPE + -2*MIN_SSLOPE*D_TEMP(FIRSTHALF_INDEX+1:length(D_TEMP));
Q3 = [Q1,Q2];
Q(FIRSTPT:LASTPT) = Q(FIRSTPT:LASTPT) + Q3;

SIDESLOPE_PROFILE = Q;

figure(11);
subplot(4,3,7), plot(D,Q,'m-');
axis([0 1 -1 1]);
```

```
title('Cup: side slope profile');

%..(Corner slope profile).............................................................

Q = CSLOPE*ones(1,length(D));
CORNSLOPE_PROFILE = Q;

figure(11);
subplot(4,3,8), plot(D,Q,'m-');
axis([0 1 -1 1]);
title('Cup: corner slope profile');

%..(Storing Specifications)...........................................................
%
% All profile parameters are stored in the MODEL cell array.
% The format of the array is given by: MODEL(Wall#,Parameter#)

MODEL{1,1} = [1];                     % Cup-type component/wall
MODEL{1,2} = [0 1];                   % Wall occupies whole cup height
MODEL{1,3} = MAX_RAD*RAD_PROFILE;     % Radial profile for exterior wall
MODEL{1,4} = HOLE_PROFILE;            % Hole diameter profile
MODEL{1,5} = CAVS_PROFILE;            % Side curvature profile
MODEL{1,6} = CAVC_PROFILE;            % Corner curvature profile
MODEL{1,7} = CORNER_PROFILE;;         % Corner-size profile (0.2 throughout)
MODEL{1,8} = THICK_PROFILE;           % Lip-size profile (THICK throughout)
MODEL{1,9} = SIDESLOPE_PROFILE;       % Side slope profile
MODEL{1,10}= CORNSLOPE_PROFILE;       % Corner slope profile

MODEL{2,1} = [1];
MODEL{2,2} = [0.02 1];
MODEL{2,3} = MODEL{1,3} - (THICK+0.05);
MODEL{2,4} = MODEL{1,4};
MODEL{2,5} = MODEL{1,5};
MODEL{2,6} = MODEL{1,6};
MODEL{2,7} = MODEL{1,7};
MODEL{2,8} = zeros(1,length(D));
MODEL{2,9} = MODEL{1,9};
MODEL{2,10}= MODEL{1,10};

% -- Circular opening at top: --------------------------------------------------

P = [

   41.9857
   44.0374
   45.5761
   48.1407
   52.2440
   57.3732
   62.5024
   68.1444
   73.2736];

Q = [

  104.6934
   99.5636
   93.4079
   86.2262
   79.5574
   73.9147
   69.2979
   66.7330
   65.1940];


P = P' - P(1);
P = P/max(P);
Q = Q' - min(Q);
Q = Q/max(Q);

POLY_DEGREE = 4;

c =  polyfit(P,Q,POLY_DEGREE);

switch POLY_DEGREE
 case 2, Q=c(1)*D.^2+c(2)*D.^1+c(3);
 case 4, Q=c(1)*D.^4+c(2)*D.^3+c(3)*D.^2+c(4)*D+c(5);
 case 6, Q=c(1)*D.^6+c(2)*D.^5+c(3)*D.^4+c(4)*D.^3+c(5)*D.^2+c(6)*D+c(7);
 case 8, Q=c(1)*D.^8+c(2)*D.^7+c(3)*D.^6+c(4)*D.^5+c(5)*D.^4+c(6)*D.^3+c(7)*D.^2 + !!!
         c(8)*D+c(9);
 case 10, Q=c(1)*D.^10+c(2)*D.^9+c(3)*D.^8+c(4)*D.^7+c(5)*D.^6+c(6)*D.^5+c(7)*D.^4 + !!!
          c(8)*D.^3 +c(9)*D.^2+c(10)*D+c(11);
end

Q = Q/max(Q);

TOPLIP_PROFILE = Q;

figure(11);
subplot(4,3,9), plot(D,Q,'c-');
axis([0 1 0 1]);
title('Cup: circular opening inner lip profile');

CAVS_PROFILE = 0.01*ones(1,length(D));
CAVC_PROFILE = 0.01*ones(1,length(D));
THICK_PROFILE = 0.2*ones(1,length(D));

MODEL{3,1} = [1];               % Cup-type component/wall
MODEL{3,2} = TOPLIP_POSITION;   % Fractional position of top lip from top of cup;
MODEL{3,3} = OPENING_RADIUS*MAX_RAD-TOPLIP_SIZE*MAX_RAD*(1-TOPLIP_PROFILE);
                                % Radial profile of lip (must start with 1)
MODEL{3,4} = zeros(1,length(D));% Hole diameter profile (zero size throughout)
MODEL{3,5} = CAVS_PROFILE;      % Side curvature profile (circular throughout)
MODEL{3,6} = CAVC_PROFILE;      % Corner curvature profile (circular throughout)
MODEL{3,7} = zeros(1,length(D));% Corner-size profile (zero size throughout)
MODEL{3,8} = zeros(1,length(D));% Lip-size profile (zero size throughout)
MODEL{3,9} = zeros(1,length(D));% Side slope (zero throughout)
MODEL{3,10}= zeros(1,length(D));% Corner slope (zero throughout)

MODEL{4,1} = [1];
MODEL{4,2} = [TOPLIP_POSITION(1)+0.02 TOPLIP_POSITION(2)];
MODEL{4,3} = THICK + MODEL{3,3};
MODEL{4,4} = MODEL{3,4};
MODEL{4,5} = MODEL{3,5};
MODEL{4,6} = MODEL{3,6};
MODEL{4,7} = MODEL{3,7};
MODEL{4,8} = zeros(1,length(D));
```

```
MODEL{4,9} = MODEL{3,9};
MODEL{4,10}= MODEL{3,10};

% -- Stem -----------------------------------------------------------------------

%..(XY Trajectory Profile).....................................................

Q = 0.5*(D.^10 - D.^9 + 0.7*D.^2);

figure(11);
subplot(4,3,10), plot(D,Q,'r-');
axis([0 1 0 1]);
title('Stem: XY trajectory');

TRAJECTORY_PROFILE = Q;

%..(Inclination Profile)......................................................

Q = 0.4*(D.^10 - D.^9 + 0.7*D.^2);

figure(11);
subplot(4,3,11), plot(D,Q,'g-');
axis([0 1 0 1]);
title('Stem: inclination profile');

INCLINE_PROFILE = Q;

%..(Radial profile)...........................................................

RISE = 2;
Q = RADPROFILE_SCALE*atan(RISE*D)/atan(RISE);
figure(11);
subplot(4,3,12), plot(D,Q,'m-');
axis([0 1 0 1]);
title('Stem: radial profile');

STEMRAD_PROFILE = Q;

%..(Storing Specifications)...................................................

MODEL{5,1} = [2];                 % Stem-type component/wall
MODEL{5,2} = STEM_ATTACH_OFFSET; % X and Y coordinates of center-of-basal stem attachment.
MODEL{5,3} = TRAJECTORY_PROFILE; % XY-trajectory profile
MODEL{5,4} = INCLINE_PROFILE;    % Inclination-angle profile as fraction of pi/2
                                 % (the angle is calculated by subtracting this from pi/2:
                                 % i.e., small angles means nearly vertical)
MODEL{5,5} = BASE_STEMRAD+BASE_STEMRAD*STEMRAD_PROFILE; % Hole diameter profile

MODEL{6,1} = [2];
MODEL{6,2} = STEM_ATTACH_OFFSET;
MODEL{6,3} = TRAJECTORY_PROFILE;
MODEL{6,4} = INCLINE_PROFILE;
MODEL{6,5} = MODEL{5,5} + (THICK + 0.05);
```

————model_maker.m————————————————————————

```
% (Please read the comments supplied at the beginning of model.m)
%
% This scripts is used to draw and store contours sampled at regular intervals
% using wall.m and according to the parameters specified in model.m (which must
% be run before this script.)
%===============================================================================
% Variable initializations
%===============================================================================
V = cell(1000,20);
V_TEMP = cell(2);
POL_V_TEMP = cell(2);

CUPHEIGHT = 2*MAX_RAD/MAXDIAM2HEIGHT;
TAILENGTH = STEM2CUP*CUPHEIGHT;
TOTALENGTH = CUPHEIGHT + TAILENGTH;
INTERVAL = TOTALENGTH/NUMLAYERS;
LASTCUPLAYER = ceil((CUPHEIGHT/TOTALENGTH)*NUMLAYERS); % Frame number of the last cup layer.
PROPER_AXIS = [-(MAX_RAD+1) MAX_RAD+1 -(MAX_RAD+1) MAX_RAD+1];

%================================================================================

figure(10); clf;

for l = 1: NUMLAYERS

   whichlayer = sprintf('Processing contours in layer %d/%d...\n',l,NUMLAYERS);
   fprintf(1,'%s',whichlayer);
   figure(10); clf;


   v = 1;

   for w = 1:NUMWALLS/2;

      Z(l) = l*INTERVAL;
      COMPONENT = MODEL{2*w-1,1};

      switch COMPONENT

        % ========= The first case in the switch loop is used to draw cup and lip walls.===

        case 1,
             clear V_TEMP;

             if l >= MODEL{2*w-1,2}(1)*LASTCUPLAYER & l <= MODEL{2*w-1,2}(2)*LASTCUPLAYER

                % Notice that interp1 is used to interpolate between points supplied
                % in the profiles.

                curr_position = (l - (MODEL{2*w-1,2}(1)*LASTCUPLAYER))/ !!!
                ((MODEL{2*w-1,2}(2)-MODEL{2*w-1,2}(1))*LASTCUPLAYER);
                RADIUS = interp1(D,MODEL{2*w-1,3},[curr_position],'linear');
                HOLESIZE = interp1(D,MODEL{2*w-1,4},[curr_position],'linear');
                SIDE_CONCAVITY = interp1(D,MODEL{2*w-1,5},[curr_position],'linear');
                CORNER_CONCAVITY = interp1(D,MODEL{2*w-1,6},[curr_position],'linear');
                CORNERSIZE =  interp1(D,MODEL{2*w-1,7},[curr_position],'linear');
                LIPSIZE = interp1(D,MODEL{2*w-1,8},[curr_position],'linear');
                SIDE_SLOPE = interp1(D,MODEL{2*w-1,9},[curr_position],'linear');
                CORNER_SLOPE = interp1(D,MODEL{2*w-1,10},[curr_position],'linear');

                wall;
                axis(PROPER_AXIS);
```

```
            if HOLESIZE > 0
    V_TEMP{1} = Xc;
               V_TEMP{2} = Yc;
            else
               V{l,v}(:,1) = Xc';
               V{l,v}(:,2) = Yc';
            end
         v = v+1;
         end

         if l >= MODEL{2*w,2}(1)*LASTCUPLAYER & l <= MODEL{2*w,2}(2)*LASTCUPLAYER

            RADIUS = interp1(D,MODEL{2*w,3},[curr_position],'linear');
            HOLESIZE = interp1(D,MODEL{2*w,4},[curr_position],'linear');
            SIDE_CONCAVITY = interp1(D,MODEL{2*w,5},[curr_position],'linear');
            CORNER_CONCAVITY = interp1(D,MODEL{2*w,6},[curr_position],'linear');
            CORNERSIZE =  interp1(D,MODEL{2*w,7},[curr_position],'linear');
            LIPSIZE = interp1(D,MODEL{2*w,8},[curr_position],'linear');
            SIDE_SLOPE = interp1(D,MODEL{2*w,9},[curr_position],'linear');
            CORNER_SLOPE = interp1(D,MODEL{2*w,10},[curr_position],'linear');

            wall;
            axis(PROPER_AXIS);

            if HOLESIZE > 0
V_TEMP{1} = [V_TEMP{1}, fliplr(Xc)];
               V_TEMP{2} = [V_TEMP{2}, fliplr(Yc)];
               [POL_V_TEMP{1},POL_V_TEMP{2}] = cart2pol(V_TEMP{1},V_TEMP{2});

               TOPANGLE = 2.5*(2*pi/NUMSIDES);
               BOTANGLE = 1.5*(2*pi/NUMSIDES);

               clear CNTEMP1;
               clear CNTEMP2;

               for i = 1: length(POL_V_TEMP{1})
                   if POL_V_TEMP{1}(i) < TOPANGLE & POL_V_TEMP{1}(i) > BOTANGLE
    [CNTEMP1(1,i), CNTEMP1(2,i)] = pol2cart(POL_V_TEMP{1}(i), !!!
                   POL_V_TEMP{2}(i));
                   end
               end

               j = 1;

for i = 1: length(CNTEMP1(1,:))
                   if CNTEMP1(1,i) ~= 0 & CNTEMP1(2,i) ~= 0
         CNTEMP2(1,j) = CNTEMP1(1,i);
                      CNTEMP2(2,j) = CNTEMP1(2,i);
                      j = j+1;
                   end
               end

               ROTANGLE = 2*pi/NUMSIDES;
               RRRR = [cos(ROTANGLE) -sin(ROTANGLE); sin(ROTANGLE) cos(ROTANGLE)];

for i = 1: NUMSIDES
                   V{l,v-2+i} = CNTEMP2';
                   CNTEMP2 = RRRR*CNTEMP2;
```

```
                end
             else

                V{1,v}(:,1) = Xc';
                V{1,v}(:,2) = Yc';
           end
           v = v+1;
         end

     % ========= The second case in the switch loop is used to draw stem walls. ===

case 2,

          if l > LASTCUPLAYER
             MINOR = interp1(D,MODEL{2*w-1,5},[(l-LASTCUPLAYER)/(NUMLAYERS - !!!
                     LASTCUPLAYER)],'linear');
             X_pos = (l-LASTCUPLAYER)/(NUMLAYERS-LASTCUPLAYER);
             Y_pos = interp1(D,MODEL{2*w-1,3},[(l-LASTCUPLAYER)/(NUMLAYERS - !!!
                     LASTCUPLAYER)],'linear');
             POSITION = STEM_ATTACH_OFFSET + XYSIZE*[X_pos Y_pos];
             MAJOR = MINOR./sin(pi/2 - (pi/2)*interp1(D,MODEL{2*w-1,4},[(l - !!!
                     LASTCUPLAYER)/(NUMLAYERS-LASTCUPLAYER)],'linear'));
             [THETA,RADD] = cart2pol(X_pos,Y_pos);
             wall;
             axis(PROPER_AXIS);

             V{1,v}(:,1) = Xc';
             V{1,v}(:,2) = Yc';

             v = v+1;

             MINOR = interp1(D,MODEL{2*w,5},[(l-LASTCUPLAYER)/(NUMLAYERS - !!!
                     LASTCUPLAYER)],'linear');
             X_pos = (l-LASTCUPLAYER)/(NUMLAYERS-LASTCUPLAYER);
             Y_pos = interp1(D,MODEL{2*w,3},[(l-LASTCUPLAYER)/(NUMLAYERS - !!!
                     LASTCUPLAYER)],'linear');
             POSITION = STEM_ATTACH_OFFSET + XYSIZE*[X_pos Y_pos];
             MAJOR = MINOR./sin(pi/2 - (pi/2)*interp1(D,MODEL{2*w,4},[(l - !!!
                     LASTCUPLAYER) / (NUMLAYERS-LASTCUPLAYER)],'linear'));
             [THETA,RADD] = cart2pol(X_pos,Y_pos);
             wall;
             axis(PROPER_AXIS);

             V{1,v}(:,1) = Xc';
             V{1,v}(:,2) = Yc';

        end

     end
 end

end
```

————wall.m————————————————————

```
% (Please read the comments supplied at the beginning of model.m)
%
% This script contains the drawing procedure used to produce the inner and
% outer boundaries of the cup-, lip-, and stem walls.
```

```
%

%=========================================================================
% Cup and Lip walls (n-symmetric walls with breaks, and circles)
%=========================================================================

if COMPONENT == 1  % This means that a cup wall is being drawn.

figure(9);
clear T;
T = 0:dT:1;

C1 = SIDE_CONCAVITY;
C2 = CORNER_CONCAVITY;
C3 = CORNERSIZE;
C4 = NUMSIDES;
C5 = SIDE_SLOPE;
C6 = CORNER_SLOPE;

Y1_TEMP = (1./( sin(pi/2 + C1*(1-C3)*pi/C4) - C5*cos(pi/2 + C1*(1-C3)*pi/C4))) / !!!
          (1./sin(pi/2 + pi/C4));
Y2_TEMP = 1./(sin(pi/2 + C2*C3*pi/C4) - C6*cos(pi/2 + C2*C3*pi/C4));

DISPLACEMENT = Y2_TEMP - Y1_TEMP;

Y1 = (1./( sin(pi/2 + C1*(1-T)*pi/C4) - C5*cos(pi/2 + C1*(1-T)*pi/C4)))/(1./sin(pi/2 + !!!
     pi/C4)) + DISPLACEMENT;

subplot(2,2,1), plot(T,Y1,'g-');
axis([0 1 0 2]);
title('Corner Profile');

Y2 = 1./(sin(pi/2 + C2*T*pi/C4) - C6*cos(pi/2 + C2*T*pi/C4));

hold on;
subplot(2,2,1), plot(T,Y2,'r-');
axis([0 1 0 2]);
title('Corner Profile');

hold on;
NUMPTS = size(T);
Y3 = ones(NUMPTS(2));
subplot(2,2,1), plot(T,Y3,'k');
axis([0 1 0 2]);
title('Corner Profile');
AXIS = axis;
hold off;

sizeT = size(T);

clear YY;

CUTOFF = 1-HOLESIZE;

i = 1;
while T(i) < CUTOFF;

     if Y2(i) < Y1(i)
        YY(i) = Y2(i);
     else
        YY(i) = Y1(i);
     end;
i = i + 1;
end


branch_size = i-1;

% -- Adding inner lip for joining to inner wall ------------------------------

LIP = linspace(YY(i-1)-LIPSIZE/RADIUS,YY(i-1),sizeT(2)-i+1);
LIP = fliplr(LIP);

for j=i-1:sizeT(2)
    T(j) = T(i);
end

YY = [YY,LIP];
subplot(2,2,2), plot(T,YY,'.');
AXIS2 = axis;


% ---- Polynomial smooth fitting (SMOOTH_TYPE = 1) (NOT USED) ------------------

if SMOOTH_TYPE == 1

c =  polyfit(T,YY,POLY_DEGREE);

         switch POLY_DEGREE
case 1,
YY = c(1)*T.^1 + c(2);
case 2,
YY = c(1)*T.^2 + c(2)*T.^1 + c(3);
case 3,
YY = c(1)*T.^3 + c(2)*T.^2 + c(3)*T + c(4);
case 4,
YY = c(1)*T.^4 + c(2)*T.^3 + c(3)*T.^2 + c(4)*T + c(5);
case 5,
YY = c(1)*T.^5 + c(2)*T.^4 + c(3)*T.^3 + c(4)*T.^2 + c(5)*T + c(6);
case 6,
YY = c(1)*T.^6 + c(2)*T.^5 + c(3)*T.^4 + c(4)*T.^3 + c(5)*T.^2 + !!!
                              c(6)*T + c(7);
                case 7,
        YY = c(1)*T.^7 + c(2)*T.^6 + c(3)*T.^5 + c(4)*T.^4 + c(5)*T.^3 + !!!
                              c(6)*T.^2 + c(7)*T + c(8);
                case 8,
        YY = c(1)*T.^8 + c(2)*T.^7 + c(3)*T.^6 + c(4)*T.^5 + c(5)*T.^4 + !!!
                              c(6)*T.^3 + c(7)*T.^2 + c(8)*T + c(9);
                case 10,
                        YY = c(1)*T.^10 + c(2)*T.^9 + c(3)*T.^8 + c(4)*T.^7 + !!!
                              c(5)*T.^6 + c(6)*T.^5 + c(7)*T.^4 + c(8)*T.^3 + c(9)*T.^2 + !!!
                              c(10)*T + c(11);

end   % end of switch POLY_DEGREE
end


% --- Spline ``smoothing'' -----------------------------
```

```
if SMOOTH_TYPE > 1

clear TTT;
clear YYY;
clear TTTT;

switch SMOOTH_TYPE,
       case 2, method = 'spline';
       case 3, method = 'cubic';
       case 4, method = 'linear';
end

TTT = linspace(0,T(branch_size),SMOOTH_POINTS);
YYY = interp1(T(1:branch_size),[YY(1:branch_size-1),YY(end)],TTT,method);

TTTT = 0:dT/2:TTT(end);

YY = interp1(TTT,YYY,TTTT,method);

T = TTTT;
end


% ----- Rescaling Cup wall to actual size ---------------------------

P = YY;
subplot(2,2,2), plot(T,P);
axis(AXIS2);
title('Half-Side');
P = RADIUS*P;

%-----------------------------------------------------------------
% Generating a single side, and then all sides:
%-----------------------------------------------------------------

SIDE_Y = [P,fliplr(P)];
SIDE_X = [T,2-fliplr(T)]/2;
hold off;
subplot(2,2,3), plot(SIDE_X,SIDE_Y,'.');
axis([0 1 0 RADIUS+1]);
title('Entire side');

BASELINE = SIDE_X;
PERIM = SIDE_Y;
for i = 1:NUMSIDES-1,
    PERIM = [PERIM,SIDE_Y];
    BASELINE = [BASELINE,SIDE_X+i];
end;

[Xc,Yc] = pol2cart(2*pi*BASELINE/NUMSIDES,PERIM);

end % end of if COMPONENT === 1 conditional

%=====================================================================
% Stem walls (ellipses)
%=====================================================================

% Takes as arguments: CENTER, THETA, MINOR, MAJOR
```

```
if COMPONENT == 2 % A stem wall.

t = [0:0.01:2*pi];

Xe = MAJOR*cos(t);
Ye = MINOR*sin(t);

RRR = [cos(THETA) -sin(THETA); sin(THETA) cos(THETA)];
XYe = [Xe; Ye];
RXYe =  RRR*XYe;

Xc = RXYe(1,:) + POSITION(1);
Yc = RXYe(2,:) + POSITION(2);

end

%=========================================================================
% Plotting the walls
%=========================================================================

figure(10);
hold on;
plot(Xc,Yc,'.');
title('Plotted Walls');
axis equal;
axis off
hold off
```