Interactive Visualization in Interdisciplinary Applications

Interaktive Visualisierungstechniken in interdisziplinären Anwendungsbereichen

Der Technischen Fakultät der
Universität Erlangen-Nürnberg

zur Erlangung des Grades

DOKTOR-INGENIEUR

vorgelegt von

Dipl.-Inf. Martin Meister

Erlangen – 2008

Als Dissertation genehmigt von
der Technischen Fakultät der
Universität Erlangen-Nürnberg

Tag der Einreichung:	24.01.2008
Tag der Promotion:	02.04.2008
Dekan:	Prof. Dr. J. Huber
Berichterstatter:	Prof. Dr. G. Greiner
	Prof. Dr. K. Hormann

Abstract

Visualization is any technique used to create meaningful and intuitive images to communicate information. Real-time 3D visualization techniques are commonly employed in engineering to support the interpretation of large amounts of complex data and to gain a deeper understanding of the underlying processes. In engineering, data most often originates from computer simulation experiments or sensors. Yet similar data exists in other fields of science and scientific visualization techniques can be of great advantage as well. We present applications and research in which new insights have been gained that were not possible before as these techniques are not well established in the respective fields of science. The contribution of this dissertation lies in the application and extension of real-time visualization and computer graphics techniques in interdisciplinary projects with partners from outside of the field of engineering.

Problems and their solutions are presented for several disciplines: in archaeology we inspect as well as interpret unique artefacts non-destructively, making an inside view of ancient pottery possible. Using computed tomography scans we offer methods for the illustration of complex fragmented pottery and find a new interpretation of the manufacturing process. Also in archaeology, we apply our techniques to the visualization of architectural reconstructions. Several examples are given that show how ancient structures can be reconstructed and then rendered in real-time. In palaeontology/oceanology we propose and apply computer graphics methods for the inspection and analysis of an endangered species of deep-sea corals. We describe the morphology of the corals and develop a mathematical model that is used to grow and evaluate artificial virtual reefs in order to evaluate their reef building and biodiversity potential. Finally, in biology/geology we use high-resolution terrain data to help calculate maps of sun radiation. Our results have been used for the creation of charts of potential natural vegetation in a project for the Federal Agency of Nature Conservation.

Kurzfassung

Unter Visualisierung versteht man jede mögliche Art und Weise, einfach zu verstehende Bilder zu erzeugen, um Information zu kommunizieren. Echtzeit 3D-Techniken werden herkömmlicherweise in den Ingenieursdisziplinen eingesetzt, um große Mengen komplexer Daten zu interpretieren und die darunterliegenden Prozesse besser verstehen zu können. Die betreffenden Daten kommen dabei meist aus Simulationsexperimenten oder stammen aus Sensoraufnahmen. Jedoch existieren ähnliche Daten in anderen Disziplinen und wissenschaftliche Visualisierungstechniken können dort ebenfalls zum Vorteil eingesetzt werden. In dieser Arbeit werden Anwendungsbereiche und Forschung präsentiert, in denen neue Erkenntnisse gewonnen wurden, die so vorher nicht denkbar waren, da sie ohne die eingesetzten Techniken nicht sichtbar wurden. Der Beitrag dieser Arbeit liegt in der Einbringung und Erweiterung von Vorgehensweisen und Techniken der echtzeit-fähigen Visualisierung und Computergrafik in nicht-ingenieurstechnische interdisziplinäre Projekte.

Probleme und ihre Lösungen werden für mehrere Disziplinen präsentiert: in der Archälogie werden antike Artefakte mit zerstörungsfreien Methoden untersucht und neu interpretiert. Mit dem Einsatz von Computertomographie wird es möglich, einen Blick in das Innere geschlossener antiker Keramik zu werfen und den restaurierten, komplex fragmentierten Erhaltungszustand aussagekräftig abzubilden und neue Hinweise über den Herstellungsprozeß zu erhalten. Ebenso in der Archäologie werden Visualisierungstechniken vorgestellt, um Rekonstruktionen antiker Bauwerke durchzuführen und darzustellen. Dabei werden geeignete Softwarelösungen aufgezeigt, mit denen diese Strukturen rekonstruiert und in Echtzeit dargestellt werden können sowie mehrere Beispiele vorgeführt. In der Paläontologie und Ozeanologie werden Vorgehensweisen vorgeschlagen, die zur Analyse einer vom Aussterben bedrohten Tiefseekorallenart nützlich sind. Zuerst werden mit Computertomografieaufnahmen Korallen erfasst und ihre Morphologie beschrieben. Danach wird ein mathematisches Modell entwickelt, das benutzt wird, um künstliche virtuelle Riffe zu erschaffen und auszuwerten, um Fragen des Riffbildungspotentials und der Biodiversität zu beantworten. Schlussendlich wird eine Anwendung in den Geowis-

senschaften präsentiert, in der hoch aufgelösten Karten eines Terrainmodells von Bayern verwendet werden, um genaue Sonneneinstrahlungskarten auszurechnen unter Berücksichtigung globaler Effekte. Diese Auswertungen helfen, Karten der potentiellen natürlichen Vegetation zu bestimmen, die in einem Projekt für das Bundesamt für Naturschutz angefertigt werden.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my supervisor Prof. Günther Greiner for his support and confidence in my work presented here. His professional advice helped me tremendously and I am grateful for the instructive time within his group. My colleagues were a great source of inspiration and we had good fun discussing and working on interesting scientific problems. Special thanks go to my office mates Dorit Merhof and Marco Winter.

I am grateful to Dr. Martin Boss of the Antikensammlung der Universität Erlangen-Nürnberg for illustrating archaeology issues and providing his expertise on pottery artefacts and architectural reconstruction. I am indebted to him for allowing me to use his reconstruction proposals and virtual models. The work of Alexander Bardosch on the second version of the Römerlager Marktbreit is also acknowledged.

Dominik Rietzel was very helpful with his incredible laser-sintering machine at the Lehrstuhl für Kunststofftechnik. Thank you for investing time in the fabrication a new museum exhibit for the Antikensammlung.

I would like to thank Prof. André Freiwald of the Lehrstuhl für Paläontologie and his staff, especially Chris Schulberg, for enlightening me on stony branching deep-sea cold-water corals.

Thank you to Heike Howein, Reiner Suck and Michael Bushardt at the Institut für Vegetationskunde und Landschaftsökologie for their support. I am grateful to Johannes Rauh for his committed work on QSolar.

This work is dedicated to my lovely wife Jennie and our dear son Robert.

Part I

Introduction

Chapter 1

Motivation

Interdisciplinary research is a field in which people from several disciplines come together to solve an interesting problem. This problem seldom is a common problem for all participants. More often one of the research areas provides methods that the others deem beneficial or advantageous. In the best case, the application of a new approach helps to find answers to important questions and problems. Interactive visualization is needed where there are huge amounts of data available and real-time user interaction is required to gain a better understanding of the underlying information. In our case, the data involved is three-dimensional, given as a regular lattice, triangular mesh or projection grid. In this thesis, we will present three cases in which our own field of work, computer graphics, has helped to gain insights in other disciplines. For each field, archaeology, palaeontolgy/oceanology and geosciences, we will give details of our contribution to interdisciplinary research and present the results of the application of interactive visualization techniques.

The outline of this thesis is as follows: in the following chapter we will briefly recall basic knowledge about computer graphics and scientific visualization. In the next part, applications are presented in which our methods have been successfully employed:

In archaeology, visualization traditionally is of great importance. From line and pencil drawings of finds, digs and reconstructions to photography and satellite imagery, archaeology has embraced new technologies that help to document and evaluate single artefacts as well as large-scale sites. In this thesis we will show new technologies can be used to support archaeological research, namely computed tomography (CT) scans, surface extraction and shading methods. We will give examples of non-destructive examination of pottery in which we obtain information on the manufacturing process. Additionally, for the purpose of a virtual museum we will show that multimodal rendering methods and frameworks are needed.

Furthermore, our work focuses on the reconstruction of large-scale buildings from few remains. With consumer technology such as personal computers and graphics accelerator cards, we show that existing computer game software can be efficiently used to display complex models of archaeologic interest. Here, global lighting solutions, dynamic 3D environments with collision detection and multi-texturing for details can help to illustrate new proposals on how to reconstruct buildings. In contrast to former approaches, these structures can be easily changed, explored by walking around in them and interaction with the building or "virtual" teachers can be integrated.

In paleontology and oceanology we present our contribution to the modelling of *Lophelia pertusa* corals. This stony branching deep sea coral species builds large-scale reef structures from its calcite skeletons. Deep sea corals are in acute danger from the trawling fishing industry while being largely unexplored or even charted. We present how to classify and evaluate the reef building potential of two main different budding types within the species. Since the role of this coral for the breeding stock of fish and other small animals as well as the contribution to the global carbonate-cycle is unknown, we shed light on issues like the spatial heterogenity of the coral branching matrix and the amount of calcite being bound by the coral.

Finally, in an interdisciplinary project with biologists and geologists we have contributed to the creation of maps of potential natural vegetation. There, we work on ways to quantify the amount of sun radiation, which is an important abiotic factor for extrapolating the development of a natural habitat. We show how to include global effects such as atmospheric absorption or shadowing on large grids for the example data of Bavaria and develop a tool that computes charts of sun hours for vegetational periods.

Chapter 2

Basic Concepts of hardware-accelerated Computer Graphics

2.1 Radiometry and Photometry

Colour and its perception is a complex matter. In the so-called *visible band* of wavelengths from 380 to 780 nm a wavelength of 450 nm is said to represent blue, 540 nm to be green and 650 nm is regarded as red. The correspondence between electro-magnetic wavelengths and a perceived colour lies in the human vision system having three different receptors especially sensitive to these wavelengths. As we will later see, these three receptors are seen as three independent signals and so many colour models represent colour with three independent values.

There are many other factors that influence our vision and it is easier to understand the abstractions that are present in computer graphics when also thinking of light in terms of photons. Each colour is a mixture of photons of different wavelengths. It is important to note that there are infinite combinations of photons of different wavelengths that the human eye perceives as one and the same colour. Usually, effects of polarization, flourescence and phosphorescence are disregarded.

The radiation we call colours is part of the wide field of radiometry that is concerned with optical radiation in a range of 3×10^{11} to 3×10^{16} Hertz (wavelengths of 0.01 to 1000 μm). Colours are combined from the above mentioned much smaller *visible band* and it is photometry that deals with the visible frequency spectrum and weights energies by the sensitivity of the human eye. To convert from radiometric entities to photometric units, values are multiplied with the *CIE photometric curve*. Despite from the focus on

the human eye and the transfer function between its units of measurement, the theories of radiometry and photometry do not differ substantially. Here, we will only give some very brief definitions of the main terms:

The basic relation between frequency and energy is defined by the *radiant energy* Q for a frequency ν with :

$$Q = h\nu \quad [J]$$

Radiant flux Φ measures the amount of energy per unit time that passes through space:

$$\Phi = \frac{\delta Q}{\delta t} \quad \left[W = \frac{J}{s}\right]$$

The terms *irradiance* E and *radiosity* B both denote the density of flux per area with the distinction that irradiance of the incident and radiosity is the exitant energy that a region can receive or emit.

$$E = B = \frac{\delta \Phi}{\delta A} \quad \left[\frac{W}{m^2}\right]$$

When projecting radiant flux that is received or distributed from a solid angle on an area we use the term *radiance*

$$L = \frac{\delta^2 \Phi}{\delta A^d \delta d} = \frac{\delta^2 \Phi}{\delta A \delta w^d} \quad \left[\frac{W}{sr \cdot m^2}\right]$$

2.2 Colour and Shading Models

The idea presented above of radiometric qualities does not directly transfer to computer graphics. Mostly, an intuitive approach to colour was needed in the past. The development of visual devices (e. g. monitors) in particular has encouraged a pragmatic approach towards colour since the displayable spectrum was and still is very limited. The omni-present RGB colour model that is commonly used to specify colour is a typical abstraction from the theory of wavelength distribution and photons. In the three colour channels only a relative quantity regarding an application specific maximum is stored. As mentioned above, the use of three independent values is motivated by the human eye's organization, with three different cone receptors in the retina.

Overview

An RGB colour is only a combination of three spectral bands, commonly thought to be associated with the three main wavelengths (red, green and blue) given above.

Another way of specifying colour for use in a computer is the $CMYK$ model, where the RGB channels are inverted and a common black part of the colour

K can be substracted, which is essentially only important for the printing industry in which it is cheaper to print black.

$$\begin{aligned} C &= 1 - R - K \\ M &= 1 - G - K \\ Y &= 1 - B - K \\ K &= min(1 - R, 1 - G, 1 - B) \end{aligned}$$

Often used in visualization is another idea of representing colour: the *HSV* colour model facilitates the composition of colour by specifying the hue, saturation and brightness of a colour. It is especially convenient to represent gradients through the hue of a colour here (linear interpolation), which led to the well-known temperature colour-encoding of values (a gradient from blue to red as an analogy from cool to hot).

For all colour models there are conversion functions into each other or into an absolute reference colour specification, the *CIE XYZ* colour space.

The manifold possibilities to represent colour and the different optical representations of the same colour on different media (scanners, CRT, LCD monitors, printers) gave rise to an international effort towards standardization. The International Color Consortium (*ICC* [62]) is a body of different producers of graphics software and imaging and printing industries. It defines *ICC-profiles* for compatibility between colour management systems. Within such a profile the colour space of a device is described in the absolute CIE-XYZ or CIE-Lab colour model. The use of such ICC-profiles is important to guarantee that an image that is made on one device looks the same (displayed or printed) on a completely different device. Most ICC-profiles consist of a 3×3 matrix and gamma curves. Lookup tables (LUTs) can also be specified but due to their size (> 1 MegaByte) are rarely used.

In the following, we will show a few colour models that are important for real-time rendering and formation of coloured images on the screen. In particular, we will show how to use them in the context of polygonal models that are comprised of vertices, normals at the vertices and face definitions. In this case, colour is formed by the surface being lit with a virtual light source. To describe material parameters of an object, i. e. the reflectivity of a surface, a BRDF (*bidirectional reflectivity distribution function*) representation is used. The BRDF is part of a more general way of describing light transfer in space, the *reflectance equation*:

$$L(\theta_o, \phi_o) = \int \int f_{\text{BRDF}}(\theta_o, \phi_o, \theta_i, \phi_i) L(\theta_i, \phi_i) cos(\theta_i) d\sigma(\theta_i, \phi_i)$$

which simplifies to

$$L(\theta_o, \phi_o) = f_{\text{BRDF}}(\theta_o, \phi_o, \theta_i, \phi_i) L(\theta_i, \phi_i) cos(\theta_i)$$

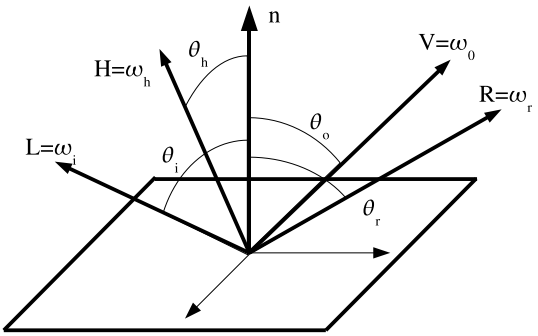

Figure 2.1: BRDF definition: incident angle θ_i corresponds to light direction vector $\vec{L}$ and viewing direction $\vec{V}$ is the outgoing light direction defined by angle θ_o. Note the vector of perfect reflection $\vec{R}$ with $\theta_r = \theta_i$.

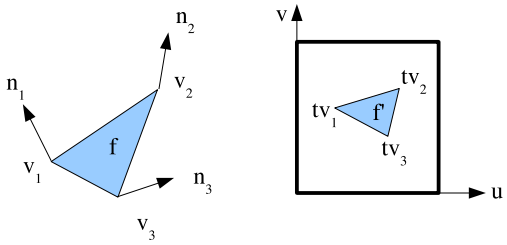

Figure 2.2: Polygonal meshes: vertices, normals and face definition. Right: two-dimensional texture space with texture vertices (tv) associated with each vertex in the (u,v) plane for the application of two-dimensional images (f') (textures) onto the face.

for a single light source. BRDFs describe how incoming radiance is related to outgoing radiance at a single surface point where light and viewing directions are known. Note that for each colour channel a separate BRDF is used and that the incoming light's energy is always attenuated by a cosine term to model the diffuse *Lambertian reflection* present in any surface. Fluorescence, Phorsphorescence and participating media cannot be reproduced with the BRDF model but are hardly needed, except for special cases.

In the following, several models are described that are commonly employed in real-time computer graphics.

Gouraud and Phong shading

Gouraud shading calculation is executed for each vertex of a polygonal mesh (the values k describe the surface's Phong BRDF parameters for this model). It is considered to be divided into three parts: diffuse, ambient and specular

reflection, and plays a crucial role in the OpenGL Graphics library:

$$\begin{aligned} L_{\text{out}} &= k_a L_{\text{ambient}} + k_d L_{\text{diffuse}} + k_s L_{\text{specular}} \\ L_{\text{diffuse}} &= L_{\text{in}} * \cos(\theta_i) \\ L_{\text{specular}} &= L_{\text{in}} * \cos(\theta_r)^{k_{\text{shiny}}} \end{aligned}$$

After calculating I_{out} for all three vertices $v1$, $v2$ and $v3$ (yielding I_1, I_2 and I_3, respectively) of a triangular face, rasterised pixels within the face are coloured by linear interpolation. The weights for the three vertex colours are determined by barycentric coordinates or bilinearly interpolated during a scanline rasterization step (one interpolation for the current scanline, one within the current scanline).

$$L_{\text{pixel}} = \rho * L_{\text{v1}} + \sigma * L_{\text{v2}} + \tau * L_{\text{v3}}$$

The Phong model introduces a per-pixel exact evaluation of the surface using Gouraud's equations. This is done by determining a normal for each pixel that is to be displayed and then the local Gouraud colour is calculated. With this extension, the computations become more expensive but also more exact, since specular highlights can be reproduced at an arbitrarily fine level only limited by the number of pixels displayed.

Torrance-Sparrow

A more refined model of local illumination was developed by physicists and introduced to computer graphics by Blinn [7, 141]. The Torrace-Sparrow model is designed to provide a better approximation of specular reflection and additionally introduces fresnel reflection at grazing angles. This is achieved by treating the surface of the object as perfectly reflective microfacets. The geometry of the facets and the direction of the illuminating light now determine the strength and the main direction of specular reflection. The normal $\vec{N}$ is perturbed to a new normal $\vec{H}$.

$$L_{\text{pixel}} = L_{\text{in}} \frac{F(\vec{L} \cdot \vec{H}) G(\vec{N} \cdot \vec{V}, \vec{N} \cdot \vec{L}) D(\vec{N} \cdot \vec{H})}{\pi (\vec{N} \cdot \vec{V})(\vec{N} \cdot \vec{L})}$$

The perturbation of the normals is described with the distribution D, self-shadowing and masking is modelled by a geometric attentuation factor G and with F a fresnel term is added that enhances specular reflection for grazing angles.

The distribution of microfacets D determines the roughness of the surface and a gaussian distribution is assumed.

$$D(\beta) = c_1 \exp^{-(\frac{\beta}{m})^2}$$

Cook-Torrance

The Cook-Torrance [27] model is an extension of the above model and uses a different more correct distribution function (Beckman's distribution):

$$D(\beta) = \frac{1}{m^2 \cos^4 \beta} \exp^{-(\frac{\tan \beta}{m})^2}$$

Here, for small m the microfacets vary very slightly on the surface by changing the surface normal and reflection is more strongly focused. For large values of m the slopes of the facets increase and the resulting rough surface leads to a more evenly distributed highlight.

2.3 Picture Synthesis

In the following we will shed light on how an image is created. Two major ways of synthesizing new imagery are state-of-the-art at the moment and a clear division exists between real-time techniques like rasterization and (mostly) offline techniques like raytracing. Despite this time trade-off, there is also a quality difference, with rasterised images generally being the less realistic and raytraced images being more close to reality. On both fronts there are advances, on the one hand ray tracing is being executed on high-speed parallelised computer networks to work in real-time [150], while extensive shaders and clever algorithm design has led to a huge increase in realism in real-time rasterization images.

2.3.1 Raytracing

Raytracing is a technique that sets up a virtual camera and a virtual pixel grid. As the name suggests, rays are shot through each pixel that the new image consists of (*forward raytracing* and *image-order*). Such a ray can now be sent through participating media, hit opaque or transparent objects or be mirrored or even absorbed somewhere in the scene. Surface effects such as transparency change the ray's direction as it becomes refracted. Mirroring causes new rays to be cast and their respective colour information is needed to composite into the final colour of the original ray, making the algorithm a recursive one. To include shadowing for each surface point that is hit a shadow ray is shot that is aimed at the light source. Shadowed surface points are easily detected if an intersection with another object occurs when following the ray. Accordingly, most of the processing time is spent on ray-object intersection calculations. Complex shadows, mirror effects and transparency are effects that are native to raytracing and justify its use in computer-generated imagery from advertising to cinema productions.

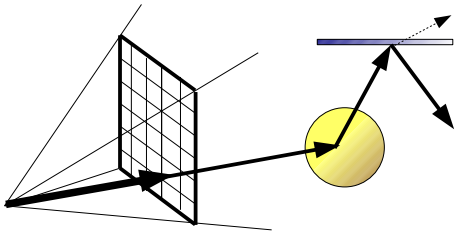

Figure 2.3: Forward raytracing: a view ray is mirrored and refracted (see dashed line) within the scene. Shadow rays are not included for clarity.

2.3.2 Rasterization

Rasterization also begins with a virtual camera and a virtual pixel grid. In contrast to raytracing, the objects are treated in sequential order to determine the contribution of each one to the final image (*object-order*). Each object is rastered in a scanline process. Objects can range from data grid volumes, quadrics functions to polygonal meshes and need only support an intersection test on them for processsing. A scanline is set up (usually a line of the output pixel) and the objects are examined for contributions to this line. A depth buffer has to be maintained in order not to overdraw already set pixels with pixels of objects that lie behind it but get rastered at a later stage. A general distinction is made between transparent objects and opaque ones. For greatest efficiency, opaque ones are sorted front-to-back and drawn. Additionally, transparent ones are sorted back-to-front and composited last (*painter's algorithm*). Effects such as mirroring or complex light interaction like soft shadows are not native to rasterization and special algorithms were developed to include them.

2.4 Hardware-accelerated 3D Graphics

Graphics cards and the libraries intended to programme them use the rasterization rendering paradigm. Since the conception of the first hardware solutions the nowadays standard PC cards generate an image from geometric primitives by converting these into a pixel representation. Generally, a graphics card is limited to working with basic piecewise linear surfaces known as triangles. Polygons with an arbitrary number of points on their circumference must be broken up into these basic primitives. After projecting in the current camera space and clipping to the current viewport, a triangle can be rasterised. Rasterization of triangles is done by dividing the viewport up into scanlines. The hardware then deals with each scanline successively to generate pixels from the given set of surfaces.

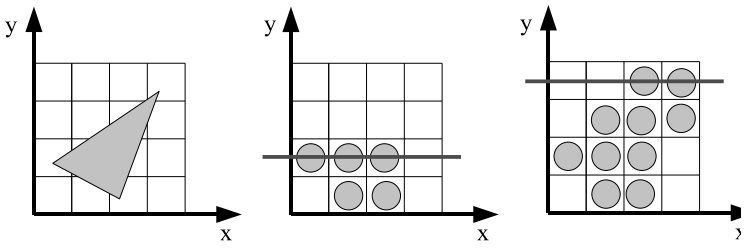

Figure 2.4: Rasterization: a triangle is converted to pixels, example scanlines for second and forth row given

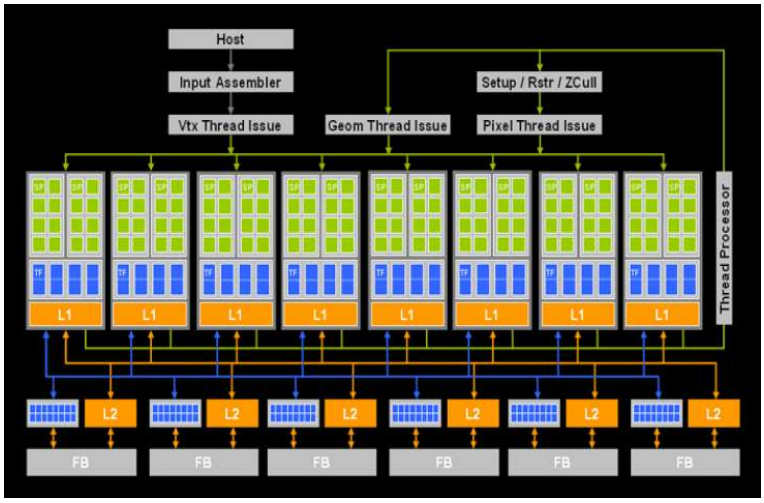

Figure 2.5: nVidia GeForce 8800: chip layout of a state-of-the-art graphics streaming processor

2.4.1 Graphics Cards

Hardware-acceleration for computer graphics has its origin in special graphics hardware for unix workstations developed in the 1960s. The first consumer graphics card able to rasterise polygons and perform transformations and lighting directly on the GPU (graphics programming unit) was the nVidia GeForce 256 (a NV10 chip), which was widely available for consumer PCs from 1999 on. At the time of writing, the dominating graphics card is the GeForce 8800 GTX, also from nVidia. To see the huge increase in computing power we will compare some of the key data in Table 2.1:

It is clear that the gain of several orders of magnitude in performance when rasterizing triangles has promoted the visualization of large datasets. It is now possible to simply render data when it is made available by a computation. However, specialised data structures are still needed to provide

Graphics card	Year	Triangles/sec.	Pixels/sec.	Memory
GeForce 256	1999	15 Million	480 Million	128MB
GefForce 8800 GTX	2007	> 2 Billion	38 Billion	768MB

Table 2.1: Comparison of GeForce 256 and 8800 as the result of graphics card development from 1999 to 2007 (source nVidia)

real-time simplifications of complex data as data generated by sensors, simulations or other applications is also growing at a similarly rapid pace. Since the increase in on-board memory is not very notable it severely limits the storage of large datasets on the graphics card. Real-time performance depends on the throughput of the computer system's bus (AGP or PCI Express) and a clever handling of the data (rendering algorithm) to decide which geometric primitives should be transferred and rasterised. As a consequence, in most brute-force rendering scenarios where data is simply generated, then streamed to the graphics card and subsequently rasterised, the full computational potential of the GPU is seldomly exploited fully.

2.4.2 Programming Libraries

For graphics programming two major low-level libraries or APIs (for *application programmer interface*) are used. OpenGL has its origins in the UNIX workstation world and has been continuously developed since the 1970s. Direct3d is a relatively new alternative to OpenGL, offering the same functionality but with a different scope on the platforms supported by its vendor, Microsoft. Both libraries offer an abstract view of the underlying hardware and the programmer's API calls are interpreted by an underlying hardware device driver supplied by the graphics hardware vendor. For this reason, subtle changes in the driver's implementation of registers, combiners or buffers can lead to differences in the performance and even stability of algorithms but are beyond the application programmer's control. nVidia's CUDA language promises to offer a new way of looking at the hardware of the new GeForce 8 series. With the introduction of the unified shader architecture it may be possible in the future to better control the load of processing units on the card directly.

2.5 Lighting Techniques

Lighting complex geometry is a task that is time-consuming and can be hardware-accelerated only partly. The theory of light distribution in computer graphics is commonly described by the rendering equation formulated

by Kajiya in [68] in a model where surfaces can emit light:

$$L_0(x, \vec{\omega}_0) = L_e(x, \vec{\omega}_0) + \int_{\Omega_{\vec{n}}} f(x, \vec{\omega}_i \rightarrow \vec{\omega}_0) L_i(x, \vec{\omega}_i) \cos\theta_i d\omega_i$$

It describes how at a point x in space in direction $\vec{\omega}_0$ the emitted radiance is composed of light directly outgoing from the point and, additionally, all incoming inscattered or directly hit light, be it from light sources or interreflected from other surfaces (for most surfaces, the first part of the sum will be zero, meaning they will never directly emit any light). The rendering equation models an infinite transfer of small quantiles of light between surfaces (interreflection) and a common name for it is *global illumination* as every surface can contribute to the final solution.

2.5.1 Direct Illumination

Solving the complete rendering equation is not the aim of direct illumination. Instead, an abstraction is computed in which the integral over all surfaces in the scene is missing and the radiance emitted from a point $L_e(x)$ is comprised of the light that reaches the point x from the existing light sources. One straightforward way to light virtual geometries is to compute the contribution of many virtual light sources for each pixel that is synthesised (for raytracing or rasterization). In the example below the colour model used is a purely diffuse Phong BRDF:

$$L_e(x) = \sum L_i \cos(\theta_i)$$

The final radiance of this point in space, x, is composed of a sum of each individual light source illumination calculation. Direct illumination, also for more complex material properties, is now mostly used in 3D real-time graphics. In raytracing for commercial applications, global illumination techniques are more commonly used.

2.5.2 Ambient Occlusion

Ambient occlusion (also called *sky light* or *dirt shader*) is a speed-up that simplifies the rendering equation to provide for a very fast and appealing way to mimick global influences (such as an overcast sky). Typically, the ambient occlusion of a point x is a value between 0 and 1, ranging from totally occluded (black) to unoccluded (white, or lit). The basic idea is as follows: a virtual half-sphere is set up around the x, pointing in the direction of the normal $\vec{n}$. The half-sphere is sampled several times and rays are cast into the scene. For each ray $\vec{r}$ that intersects another object of the scene

(maximum ranges can apply) the corresponding ray is said to be occluded. After averaging the number of total rays with the number of occluded rays the value of ambient occlusion is found:

$$L_{\text{Ambient Occlusion}} = \frac{\text{occluded rays}}{\text{number of rays shot}}$$

This value is most often multiplied with the local shading models involved to simulate the natural darkening of creases and corners (hence the name *dirt shader*). Interestingly for real-time graphics, once calculated, the averaged direction vector of the lit rays can be used as a *bent normal* for shading calculations with the standard colour models thus incorporating a multiplication with the ambient occlusion without having to explicitly store it in a texture.

2.5.3 Global Illumination

Global illumination (*radiosity* for diffusely reflecting surfaces) is a physically-based lighting technique that takes interreflections of surfaces into account and is seen as the sum of direct illlumination and indirect illumination. In contrast to simple local illumination calculations, additional steps are required that distribute the light that was emitted and reflected. Generally, this global interaction of surfaces leads to the most realistic lighting. Several methods exist to compute global illlumination and most of them are offline techniques not used in real-time rendering, although there is significant research on how to do so on graphics accelerator cards [30]. For a thorough discussion of all techniques available we recommend [135].

The most common technique that can be found in commercial software products for calculating the stable state of surfaces exchanging light is *photon tracing* since it is easy to integrate with other photo-realistic effects (subsurface scattering, caustics) in a ray-tracing methodology. To give a brief summary, photons are shot that bounce around the scene until they leave the scene or are absorbed by surfaces. If sufficient photons are shot into the scene and the hits and misses are averaged for all surface patches via interpolation, a realistic light distribution can be achieved and a radiosity solution is reached. Since this solution normally requires minutes to several hours to compute, the result is often stored in textures that can be used for real-time rendering (*light maps*).

Part II

Interdisciplinary Application

Chapter 3

Visualization in Archaeology

In archaeology most work is related to the excavation and restoration of archaeological sites and finds, publication of findings and, most of all, the scientific debate about the interpretation of the results of such digging campaigns, some of them re-examined even decades after their completion. Yet the whole range of work in this field seldom gets the attention of a wider public. The presentation of artefacts plays a dominant role in the media and museums/exhibitions form the image of the field of archaeology. Features of the archaeological dig site itself are only notable when there are large remains to be seen as, for example, in Rome. However, remains of such magnitude and state of preservation are rare in Northern Europe, especially when dealing with provincial Roman sites.

Displaying archaeological finds is a more difficult issue than it may seem at first glance. Of course, many of the physical remains can be displayed in a glass cabinet if size constraints, restoration condition and atmospheric preconditions (humidity, temperature) allow. However, there are other remains that also promise insights for any audience interested, for example post-holes of wooden buildings which cannot be displayed physically but may be shown digitally in a virtual, reconstructed space. Roman wooden legionary castles are a typical example that do offer spectacular multi-storey buildings if they are reconstructed, which is only possible virtually (as described in the next chapter).

Recently, the digital presentations of museums and institutes have been gaining importance as online presences in the world-wide-web are becoming state of the art. This raises difficult issues ranging from what to display to the more technical aspects of how to display an artefact. The limited dynamic range of the viewers (monitors), hardware requirements of data examiner software and the sheer size of recorded datasets demand specialised solutions on the interface between the two fields, computer graphics and archaeology.

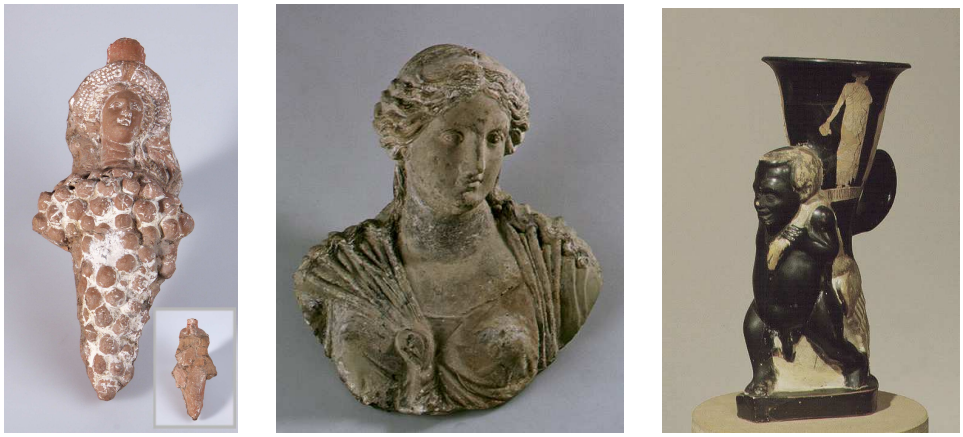

Figure 3.1: Three examined pottery pieces from the Parlasca Collection [15]: figural rhyton of a bunch of grapes (Egypt, 200 BC), Isis, goddess of fruitfulness (Asia Minor, 100 BC) and figural rhyton of a pygmy from the Sotades' painter (Greece, from approx. 450 BC).

In the following, we will report on our work on visualization of archaeological artefacts. The aim is to provide the archaeologists with new views on the data that clarify important questions about the objects. In our case, we will show results for new insights into the manufacturing process of ancient pottery.

Furthermore, we focus on our work on multi-modal representation of artefacts that were recorded with non-destructive CT-scans. For a virtual museum it is necessary to supply rendering techniques and a 3D framework that is capable of displaying diverse data. In the following, we will give an overview and a prototype example that enables the user to display datasets ranging from geometry, 3D volume data and finally light fields in a joint framework.

3.1 Non-destructive Examination

During the course of 2000, it became clear that the private collection of the former head of the department of Classical Archaeology at Erlangen University, Prof. Dr. Klaus Parlasca, would become part of the museum of the University (Antikensammlung) in Erlangen (for selected pieces see [15] [16] [36] [37] and the ones examined in Figure 3.1). Most of the works in this collection originate from the antiques trade around the Mediterranean and were acquired by Dr. Parlasca directly from the traders. This means that for most of these objects, their origin or the site where they were found and

Dataset name	Date	Origin	Publication	Resolution
Vine rhyton	200 BC	Egypt	p. 4 [15]	512 512 257
Isis	100 BC	Asia Minor	p. 6 [15]	512 512 269
Pygmy rhyton	450 BC	Attica	p. 78-79 [16]	512 512 312

Table 3.1: Acquired volume datasets (dynamic range 12 bit, unsigned short)

the context, i. e. the surrounding finds, location and geologic strata are lost and all interpretation relies purely on the artefact itself. This is an especially tricky but interesting field of research. First of all, the authenticity of an object has to be established and it has to be ruled out that the piece is a forgery, solely manufactured to look old and fetch a good price from tourists or other non-professionals. Secondly, it has to be established from where the artefact could have come, which period of time it comes from and how it typographically fits into the already known and catalogued artefacts. This can be very difficult as the objects generally get a "makeover" from the antiques traders to make them look more beautiful and complete. This is especially true in the case of original antiques (as opposed to forgeries) as they mostly are in bad shape when freshly dug up. These beautifications or sometimes even reconstructions are not always restricted to simply putting shards of a fragmented object together. In most cases, the objects get a general overall treatment and also minor imperfections are smoothed out. This raises a first important question: what actually is original in an object that was purchased at the antiques trade? Can we learn more about the object by using non-destructive methods and new ways of looking at the data?

In the following, we will show how we examined several of the artefacts of the Parlasca collection and report on the original state of the objects. The pieces examined are clay pottery pieces from the eastern Mediterranean dating from 450 to about 100 BC. We introduce a method of examination that yields new ideas about how these intricate figural objects were manufactured in the archaeological discussion below.

3.1.1 Data Acquisition

An archaeologist with his/her experience and training can easily spot plaster additions in a glazed clay vase because of discontinuities in the reflection properties of the varnish. Greek black- or red-figured potters, in particular, designed a remarkable finish for their vases from as early as 800 BC on, which even today cannot be properly reproduced by experimental archaeologists. It was achieved with a special pottery oven an a three-phase baking process. Simple black colour and glaze coating applied by many restorators

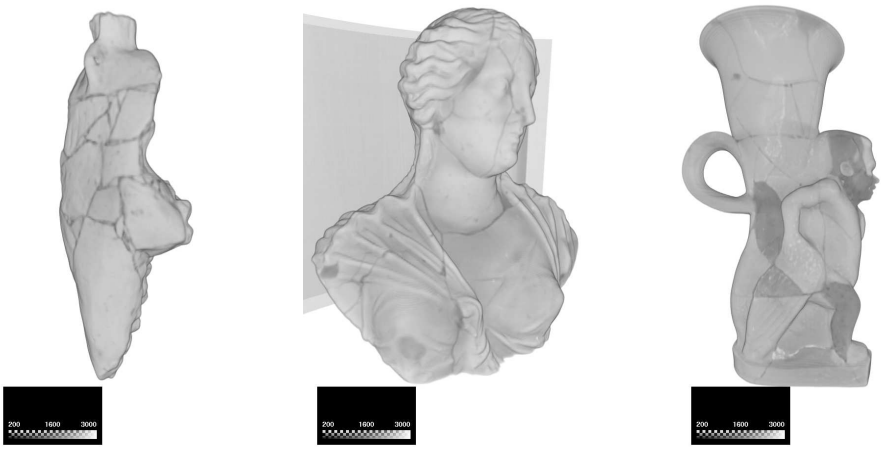

Figure 3.2: Volume rendering of Isis, vine rhyton (back side) and pygmy: fragmentation of the objects is obvious, which is not revealed in the photographs (see Figure 3.1 for colour images).

in the antiques trade cannot fool a trained eye here. For other objects that do not have this kind of elaborate finish, it is very difficult to spot. In our project, we set out to find the actual remains of these objects and we decided to use CT scanning (the device used was a Siemens Somatom Plus 4, see Figure 3.3). The reason is that we anticipated a lower absorption of the material for the plaster beautifications and higher absorption for those original parts that consist of the ancient clay. For Greek vases, especially attic ones, this assumption will always hold since the clay found there is rich in iron, making absorption ideal.

3.1.2 Visualization methods

After CT scanning, a volume dataset (3D lattice scalar field) is generated that can be used for the following procedures. There are various methods available to visualize scalar datasets. Firstly, direct volume rendering is used to inspect the fragmentation of the object. Although the scanned artefacts look near perfect on the outside, serious fragmentation is revealed upon rendering with a simple linear black and white transfer function (see Figure 3.2).

In the following, we will give an example of the techniques we applied with the example of the pygmy of the Sotades painter. Here, significant parts were added in a later reconstruction to prepare the object for sale, probably as early as in the 18th century. The volume renderings provide valuable insights into the object with the ability to see *through* the object. But

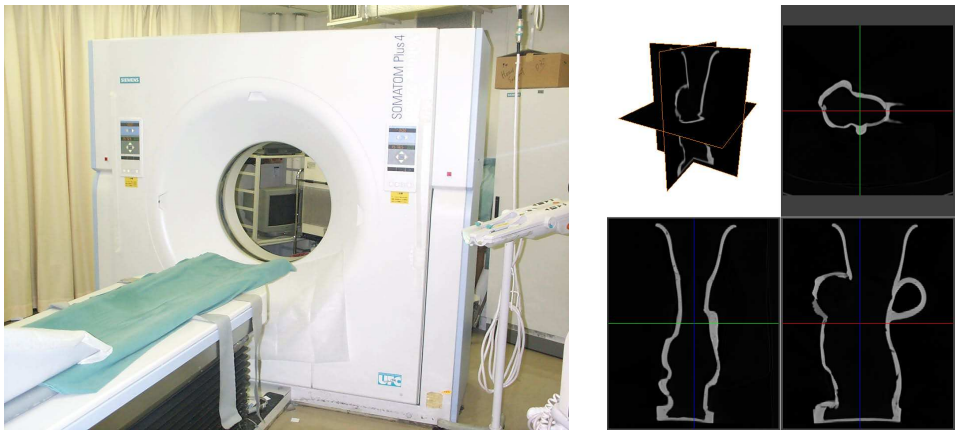

Figure 3.3: Left: Siemens Somatom Plus 4 CT scanner. Right: slices of the volume dataset of the figural rhyton of the Sotades' painter from the Parlasca Collection.

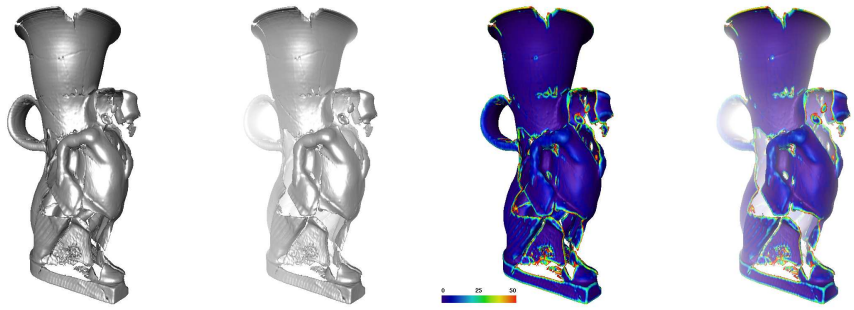

Figure 3.4: Rendering the original fragments with standard Phong illumination and mean curvature values (interval $[0 \dots 50]$ as a hue gradient colour map) with and without exponential fog in depth interval $[20 \dots 70]$.

hardware requirements for high quality volume visualization still remain high and we aimed for further uses of the data. On the one hand, triangular data enables us to execute measurements more exactly, on the other hand we have the conversion into the STL format in mind that is used for rapid prototyping. As we will later show, a real model of the original parts proves to be an excellent addition to augment the museum exhibit of a vase.

For the reasons given above, we extracted an iso-surface from the volume data. The iso-value has to be found manually with the help of a visualization tool (in this case *Amira*) that supports iso-surface extraction algorithms such as *marching cubes* [88]. Care has to be taken not to erode details away by selecting a value too high for the relatively thin thickness of the pottery. The help of an archaeologist is needed to ensure that a correct surface is extracted. The original iso-surface is very complex (over a million triangles) and surface decimation is applied to decrease the number of vertices (in our case $80,000$) and polygons while retaining all prominent features with the help of a discrete curvature based edge collapsing method [131].

In the case of the pygmy, standard shading methods fail to give a visually pleasing result (see Figure 3.9 top row) because of the messy appearance of the triangular mesh. We applied several shading methods to the mesh to obtain a clearer visualization. First of all, we tried to apply fog (fog colour $C_f = (1, 1, 1)$) to give the user an impression of depth, thus making gaps and holes lighter coloured. The fog applied is an exponential function that blends (blending factor f, see below) the colour of the fragments (C_u) in camera coordinates according to their depth value, z, linearly in an interval of $[\text{start} = 0.2 \ldots \text{end} = 0.7]$.

$$\begin{aligned} t &= \max\left(0.0, \min\left(1.0, \frac{\text{end} - z}{\text{end} - \text{start}}\right)\right) \\ f &= \exp(-t)^2 \\ C &= fC_u + (1 - f)C_f \end{aligned}$$

Secondly, we shaded the surface with the mean curvature values of the mesh to emphasise places on the surface where high curvatures are found because those places often coincide with the edges of the lines of the main fragmentation. We mapped different ranges ($[0 \ldots \text{maxcurv} = 50]$ in Figure 3.4) of mean curvature c_i of a vertex i to an HSV gradient. This is sometimes called a *physics colour map* because of the shading of low values in blue and a transition through yellow to high values in red as in a temperature analogy. The final colour C is a function of the mean curvature and yields a colour in the HSV colour space.

$$t = \frac{\text{maxcurv} - c_i}{\text{maxcurv}}$$

$$C = \text{HSV}(0.66 * t + (1 - t), 1, 1)$$

The visual complexity is still not significantly reduced and even with fog the overly colourful rendering is more confusing than clarifying.

Finally, we settled for ambient occlusion shading computed in the open-source *Blender* 3D software package. Ambient occlusion offers preferable visualization because the inner surfaces of the rhyton are rendered dark grey, the creases are darkened on the outside and the relatively undisturbed patches are displayed in an even bright colouration (see Figure 3.4 and Figure 3.9). The ambient occlusion solution was rendered into a texture (resolution $4,096 \times 4,096$) that can be mapped on the geometry in real-time for display. We repeated this process with the two other datasets, Isis and the vine rhyton, and, furthermore, used complex manual slicing techniques to make the inner surfaces visible.

3.1.3 Archaeological discussion of the findings

In the following, we will present the new analysis of the inspected objects from the point of view of an archaeologist. With the help of visualization methods that we have proposed, it is now possible to verify several points on the production techniques of the ancient pottery craft.

With the three objects from the antiques trade, different questions arise for an archaeologist:

- What is the original state of the fragments?
- Is the fragmentation original or has there been later damage?
- Does the fragmentation support the present interpretation of how the object was manufactured? Is there new insight?
- Is there information on the material and artistic quality of the piece: Is it commodity ware or an individual production?

The vine rhyton (Figure 3.5 and 3.6)

This object shows considerable fragmentation on the outside and thick plaster additions in between the grooves. On the outside it does not offer a clear view due to the complex clay patina. With the shading in Figure 3.5 several

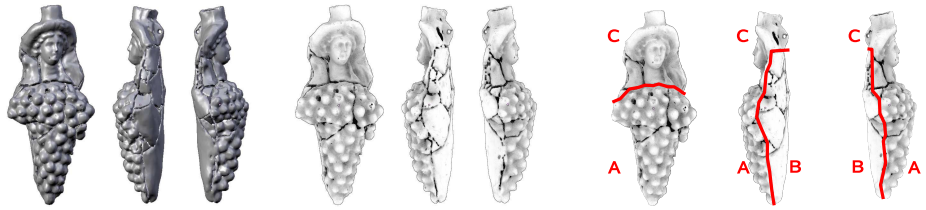

Figure 3.5: Rendering the original fragments with standard Phong illumination (from left). Using ambient occlusion (right) for the vine rhyton it becomes clear where the three main parts of the manufacturing process were joined. Notice the border of the back shell where it meets the modelled grape and the incision below the neck of the depicted deity, Dionysos.

things stand out: there are no major later additions and nearly all fragments are original. The fragmentation of the object is antique. The shape of the fragments show signs of heavy erosion. This suggests that the piece was broken during the period it was submerged, for example when destroyed with a collapsing tomb/building. Several centuries of erosion by water and other forces moving the earth can easily degrade low-quality clay (especially from Egypt). The manufacturing process is made clear from an inspection of the break lines and the inside surfaces (see Figure 3.6 for reference of the different main production parts):

1. The grape (part A) was manufactured first using a mould-pressing technique. Here, relatively moist clay is coarsely smeared into a mould. The mould is made from plaster and as such slowly draws water from the clay making it stable enough as to release it from the mould. This is a standard technique (similar to printing plates) for such shapes in the ancient pottery industry. Streak marks are visible in detail a.

2. The back side (part B) is made by hand and serves the sole purpose of closing the shape of the grape. This is made clear by the smooth surface. No smear or streaked lines appear and the imprint of the potters thumb is visible in detail b.

3. Both parts are joined together. Since the lines where the two shapes meet cannot be smoothed out from the inside (the cavity is too small) this forms the main later break line of the two pieces.

4. The deity Dionysos (part C) is manufactured with the mould-pressing technique and a back side is modelled manually. The hole in the neck for the future vessel is made.

5. The upper part (C) is set on top of the joined pieces A and B. The lines where the parts meet is closed. Again, this will be a further future

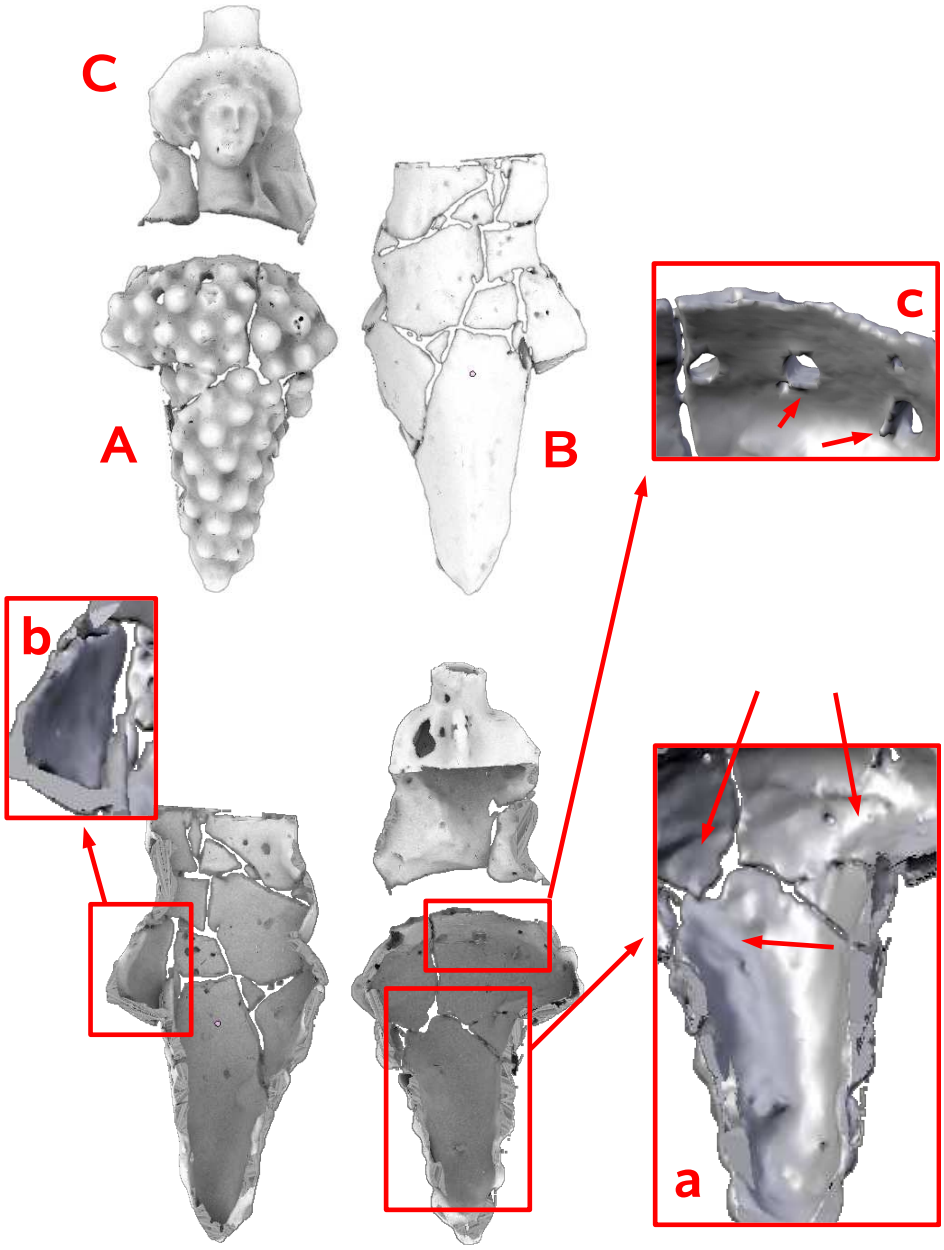

Figure 3.6: Non-destructive examination: The main lines of fragmentation indicate the way the piece was constructed: the view inside the vine rhyton is gained by cutting open the object virtually. Ambient occlusion light map on overall mesh and standard Phong local illumination for details of the triangulation. The composition of the object in parts A, B and C and details are shown: streak marks of the potter in a, imprint of the potter's thumb on b and indentations of the poked through holes in c.

Figure 3.7: The bust of Isis features deep crevices of the line where the two mould lines touched. They are less obvious with traditional Phong shading. Middle image shows ambient occlusion shading and enhances visual contrast for the crevices. To the right the dataset is virtually cut open to inspect the inside surface. Mean curvature visualization (physics colourmap) clearly shows that the inner surface is very rough and streak marks of the potter's hands in the clay are visible.

breaking point.

6. The holes are drilled into the grape as to facilitate its use as a libation vessel. This is made clear by the rim of the holes (detail c). As they were not smoothed out from part A, we conclude that they were made as a final step when the whole object was completed or at least parts A and B were joined. It also leaves room for the assumption that the shape of the grape was not primarily used for vessels of this kind as the positions of the holes are somewhat arbitrary. Maybe the mould also served different purposes, for example for the production of roof tiles or similar ornamentation.

As a conclusion, we find a new interpretation and a wealth of detail that finally lays open the production line from mould to final vessel. It also implies that possibly several potters cooperated by producing only the different parts (see A, B and D). Similar to modern industrial mass production, a production line where there is another person who solely puts objects together is imaginable.

Bust of Isis (Figures 3.7 and 3.8)

The object is comprised of mostly original pieces with the exception of some small to medium sized shards (e. g. in the back of the head). Several deep break lines (detail a) are obvious as well as small indentations of cuts (details b). Small indentations relate to damage after the piece was finished in the potter's baking oven. The absence of any sings of erosion and the very good

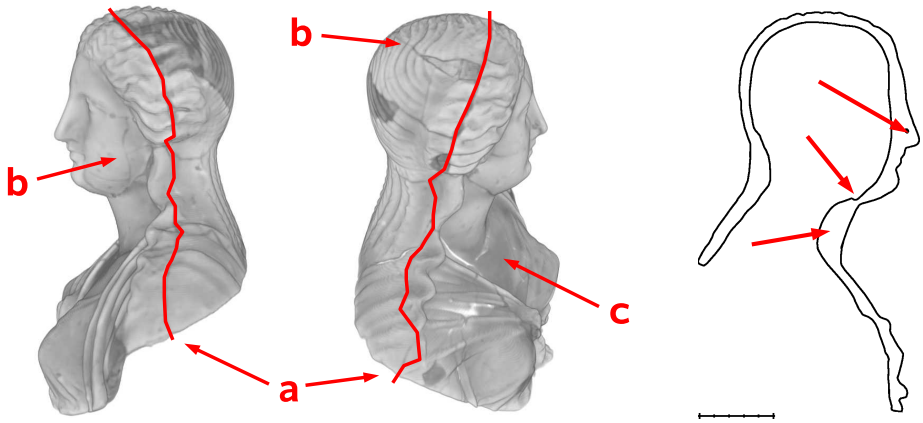

Figure 3.8: Deep crevices are found at the line where the two mould lines touched (a). Small later fragmentations are visible (b). Releasing the frontal clay from the mould seems to have broken the shape already in ancient times (see deep fragmentation lines, c). The cut (right) through the bust is of interest (see [96]), as it shows the distribution of the clay that was manually smeared in a plaster mould to produce the face (hence the uneven wall thickness, see red arrows).

fit of the shards lead us to believe that the damage was caused much later after production, possibly during the restoration of a collector in modern times. The deeper grooves relate to the structural composition of the piece. Contrary to former knowledge, apparently two shells derived from moulds were placed together. The frontal shape seems to have broken up upon release from the mould (see detail c). The meeting line of the two shapes on the inner side of the head shows a particularly deep groove. This is due to the potter not being able to smooth them out, as the opening through the neck into the head was too small to work the interior. Amazingly, the main groove passes through the hair and means that the moulding process was a complex one as the mould itself must have held many details (such as hairlocks and clothing, see d). The meeting line of the two shapes must have been subject to fine-scale manual reworking to make the two shapes match and the sculpted detail fit perfectly, as they do. Since the match is near perfect, we come to the conclusion that the moulds were created from an originally complete sculpted object. Then plaster moulds were taken from each half of the once complete, probably metal model. The artistic quality of the model is outstanding given that it is revealed to be a piece manufactured using moulds, i. e. inherently a mass-produced object. The profile cut given in Figure 3.8 shows a very uneven filling of the mould. Thick patches at the neck, the thin region under the chin and the solid nose are a great risk for the later baking process and prove that the mould was carelessly filled in a

very short time. Through the outer finish the craft of the potter becomes obvious with the high-quality moulds and the seamless blending of the two shapes on the outside.

Pygmy rhyton (Figure 3.9 and 3.10)

The rhyton depicting a pygmy is known to have later additions but it is so precious that the risk of damaging it is too great for a proper examination (there are only two more similar pieces and the Erlangen exemplar, therefore, is known world-wide). As it was purchased from a private collection that was set up in the 18th century, the restorations are believed to have been added at this time to make the fragmented object ready for display. Using ambient occlusion we obtain a convincing visualization of the original remains (see Figure 3.9). Many large parts are missing, such as the whole of the face, the side of the swan he carries over his shoulders and the right leg. The base of the rhyton is also not original, which was previously undiscovered. Many small incisions prove that the object was much more heavily fragmented after the baking process. It seems that the small damaged shards still offered a very good fit as the grooves are not very noticeable from the outside. As was expected, there is prove that the mouth of this vessel was modelled on a potter's wheel because of its perfect symmetry (see detail a in Figure 3.10).

Like the vine rhyton described above, for this rhyton it was assumed that the production process involved two or more moulds after which the form was modelled. Our research proves that this is not the case. There is no evidence of a deep groove, there are no streak marks on the inside and, most importantly, the overall very thin wall thickness rule out the mould-pressing theory. The object must have been sculpted manually by the potter. Possibly a cylindrical clay tube was put over the potter's arm and he/she applied pressure from within and shaped detail from the outside. Since the base of the object is missing it is plausible that the object was modelled as if the pygmy was standing on its head. After the rotationally symmetric mass ware mouth was produced, it was smoothly blended into the neck of the pygmy and the meeting line of the two shapes was bonded well since the inner as well as the outer surface could be worked due to its broad conic shape. To finish the piece, most probably a clay base was manufactured and put under the pygmy's feet (there are no remaining figural rhytons with an open base). But since the inner grove of this base was not accessible from the inside any more (the sculpture of the pygmy would have been deformed), this connection could not be executed as well as the rotary part and this, as in the case of the vine rhyton, might have been the reason for it to easily break off and, in this case, to be lost. The quality of the scultpure is very high and since it was produced manually it is unique. This goes well with the associated painting style that was found to belong to the workshop of the Sotades painter [60]. The Sotades painter is known for his very dynamic and individual pieces and it now becomes clear that all of his creations may

Figure 3.9: Rendering the original fragments with standard Phong illumination (top row). Using ambient occlusion (lower row), the fragmented original geometry of the Sotades painter's pygmy is much more visible and a better understanding of the three-dimensional nature of the object is won.

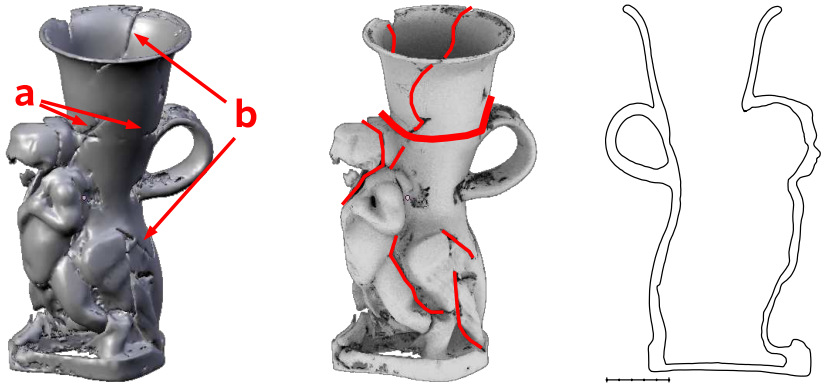

Figure 3.10: The rotationally symmetric mouth of the rhyton is added along break line (a). Later fragmentation is visible in (b) or the thin red lines. The profile cut through the object shows are very even and thin layer of clay (note that the later restorations are also included in the cut).

have been individual productions.

As this result is a major success, we will publish our findings both in the scientific community [29] and for the public. For the museum of the University of Erlangen (Antikensammlung) we have produced a rapid prototyped object (see Figure 3.12) with the aid of the Department of Plastic Materials (*Lehrstuhl für Kunststofftechnik*, FAU Erlangen-Nuremberg) to be able to exhibit the original fragments alongside the real object. With the applied selective laser sintering technique it is possible to create real-world objects by additively hardening slices of the object from plastic powder (see Figure 3.11). In our case a natural slicing direction is already given by the volume discretization (312 slices in z-axis with slice distance 1 mm). Since the real-world resolution of the laser is much higher than 1 millimetre we can obtain a clear view on the CT-Data without having to deal with artefacts from the prototyping method as well. As a further addition, a 3D computer installation for the view inside with volume or surface display is in planning. We are sure that with this key artefact the benefit of an interdisciplinay approach to research and museum installations becomes very clear and will be exploited more in the future.

3.2 Rendering of Multiple Modalities

In visualization, data has to be adapted to be renderable. Since graphics cards as well as the algorithms for display geometry (rasterization) work especially well with geometric primitives such as points, lines and triangles, it is always the case that data needs to be converted to fit the rendering

Figure 3.11: The "Sinterstation" of the *Lehrstuhl für Kunststofftechnik* to the left. To the right: the sintered form is brushed free from the unused plastic powder material.

Figure 3.12: Laser sintered Sotades rhyton: original parts have been restored with non-destructive means to be displayed alongside the original in the same scale. Since the size of the object was too big for the laser sintering machine, the artefact had to be cut into two parts (see line along the chest of the pygmy). To the far right a museum cabinet, showing the real object alongside the reconstruction.

needs. This means, for example, that it is straightforward to render a dataset that consists of unorganised points as they can simply be depicted as points. But with volume datasets alone, different algorithms had to be developed. In the 1990s with light field rendering, a lively scientific debate emerged about this new method of displaying visual data. Ranging from detailed geometric models and few pictures (*projective texturing* [33]) to full-range camera-arrays of a scene and no geometry information (*Lumigraphs* [52, 81]) different techniques exist that needed to display their information via a transition to geometric primitives (*mapping*).

We have contributed to the field of multi-modal rendering by showing how to integrate multiple Lumigraphs into a state-of-the-art rendering framework [157]. This is especially interesting to the interdisciplinary field as for the same objects multiple datasets are recorded and many different rendering algorithms are needed that have to work seamlessly together.

Our approach is a generic rendering system that incorporates different modalities by using specialised renderers. The proposed system focuses on the combination of the outputs of different rendering algorithms and is not primarily data-driven as are most other frameworks. We combine the various renderers for geometry, volumes and unstructured lumigraphs in a hierarchical structure, enabling us to use groups of renderers and flexibly join their outputs sequentially via the use of *pbuffers* or *framebuffer objects* to combine tree layers. By imposing several programming guidelines as rules of behaviour for rendering algorithms we ensure valid output after each rendering step. Additionally, we show the extensibility of our framework by giving hints on how to integrate common rendering techniques, with a focus on multiple unstructured lumigraph rendering.

3.2.1 Introduction and Motivation

Today many applications and algorithms exist that provide combined visualizations of different modalities. Most of all, this is true for the interdisciplinary (for example, cultural heritage or medical) field of application. With medical applications, systems have been implemented that support rendering of a subset of CT, MR, DTI, ultrasound or other information, in order to visualize different parts of the human anatomy. These systems focus on providing a meaningful visual combination of the given datasets. Therefore, different rendering algorithms are applied, such as direct volume rendering or visualization of isosurfaces or fibers [104, 99, 156]. While these systems are highly optimised for their particular purposes, they in general are not designed for flexibility concerning their rendering strategies. So, adding or even replacing existing rendering capabilities to fit a new application scenario may often cause significant problems and delays in software development.

For this reason, we develop a rendering framework (based on the work of [148]) which features a modular design of different rendering components. It allows the application programmer to build an application specially adapted for a certain visualization scenario by simply assembling from these components. It supports easy integration of new rendering algorithms. We will show how this can be achieved for the example of the unstructured lumigraph (ULG) [18], which we will use for the rendering of multiple light fields. Generally, rendering multiple ULGs is of interest to all applications in which only parts of a scene can be acquired in a light field or where the scene is comprised of several light fields. We find that our approach is best suited for facilitating a smooth transistion between multiple instances of ULGs and other visualizations and give implementation details.

Our motivation for research is illustrated with a practical example set in the field of archaeology. The above mentioned datasets and their many representations already require more than one rendering technique in order to be displayed (photos, volume data, triangular data). Additionally, beginning in 2001, objects of cultural heritage became the subject of light field acquisition and rendering methods because of their complex reflective nature and the need for a very colour-accurate display [57, 145]. The above mentioned pygmy rhyton was additionally captured in a surface light field to capture the intricate reflection properties of the clay varnish (see [158] for the original idea and [148] for the implementation) in 2005.

In a pilot study, a surface light field of the pygmy was computed with the aim of creating a virtual museum. The long-time goal for the use of surface light fields is to create a true three-dimensional presentation that can accurately depict the colour reflection properties of the exhibits of the real museum. In contrast, most collections and museums still rely solely on still and video images for their online web presentations and very little 3D content is available. In most cases, interactive photo scrolling and zoom is provided. See [13] for the Antikensammlung for a pure picture collection. Compared to the most prominent representatives ([17] for The British Museum in London and [89] for the Louvre, Paris) similarly little interactivity is provided. In our work we will contribute to the foundations that will enable virtual museums to show their three-dimensional contents in real-time.

3.2.2 Previous Work

In the following we provide an overview of the research done in the fields of light field rendering and visualization software. Thereby, we focus on the support of multiple image-based representations.

Image-based Rendering: Light fields

Image-based rendering is a now well-established technique that has seen

several key concepts developed over time. Starting with the definition of the plenoptic function [1], the terms *light field* [81] and *lumigraph* [52] were coined for a dimensional reduction of complexity. In the following years real-time applications such as view-dependent texturing [33], surface light fields [158, 23] and ULGs were presented. All these techniques have slightly different prerequisites concerning the density of sampling or additional geometry for rendering. Currently, the ULG approach may be seen as the most general regarding the requirements of the underlying light field and possible availability of proxy geometry.

Visualization Frameworks and Image-based Techniques

Modern visualization software and libraries (examples include VTK, Amira, OpenDX, OpenInventor and many more) include support for different basic geometric primitives, such as points, lines and triangles. For these there are counterparts in graphics programming languages such as DirectX or OpenGL for immediate use. Furthermore, the need for direct volume rendering as an excellent, hardware accelerated means of displaying scalar data volumes has been thoroughly researched and implemented. The combination of different volume rendering methods with geometric primitives has been established and is now state-of-the-art in many applications [125].

Research has been done to integrate scene visualization with different kinds of image-based rendering methods although several had been proposed before the efficient hardware accelerated light field rendering techniques described above were available [93, 3, 5]. A difficult task is how to realise the transition of one image-based scene representation into another that represents a different part of the global scene. Mainly panoramic views, concentric mosaics or omni-directional video have been considered.

Little has been published on how to integrate image-based light field rendering techniques with the state-of-the-art geometric primitives of traditional visualization [156]. While it can easily be imagined how a surface light field could be integrated into a present day rendering system, the same issue is not so obvious in the ULG case. Our contribution for light field rendering lies in the integration of ULGs (we follow the implementations in [40, 149]) into a rendering framework based on OpenGL. This way we facilitate interaction such as shadowing or blending between triangular meshes, volumes and an arbitrary number of ULGs. To the knowledge of the authors such a common code base is currently not available in scientific or commercial visualization tools.

3.2.3 Rendering Framework

In order to combine the visual outputs of both unstructured lumigraph renderings and the renderings of other data (meshes, volumes), a framework

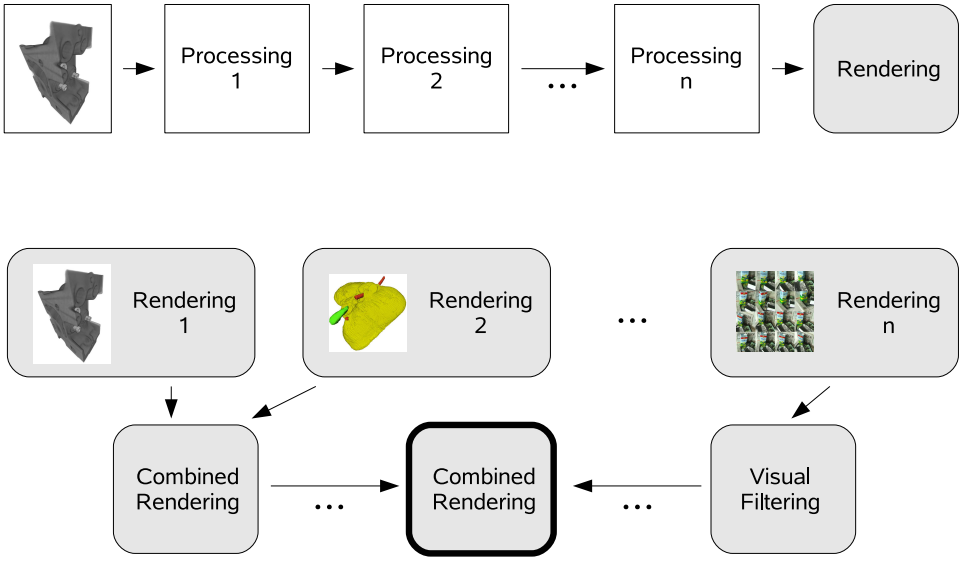

Figure 3.13: A schematic picture of the different focal points of traditional visualization systems (top) and our approach (bottom).

is developed that deals with the visual combination of different renderer outputs. Figure 3.13 depicts the main difference between established visualization frameworks and our approach:
Classic frameworks implement a visualization pipeline that has datasets as input and produces a visual representation of that data. In most cases, there are additional processing steps in between that convert or modify the input data. This may also lead to a tree-like processing structure that supplies several rendering algorithms with data. However, the main focus of classic approaches lies on the intermediate processing steps, and rendering is "just" the finishing part of the pipeline (Figure 3.13, top).

In contrast, our approach can be considered as a layer on top of the classic frameworks. Given the visual results of different rendering algorithms, our rendering framework handles the visual combination of these results. This combination can even be done hierarchically, and intermediate processing of the different visualizations (*visual filtering*) is also supported. Thus, our framework focuses on the handling of the different rendering algorithms, or the *renderers*. The basic motivation is that instead of providing specialised renderers that generate specific visual combinations, e. g. of triangle meshes and 3D volumes, the framework aims to provide a rendering environment for arbitrary algorithms which concentrate on their own visualization. Combination of these rendered results is then done by grouping elements of the framework (see section 3.2.3).

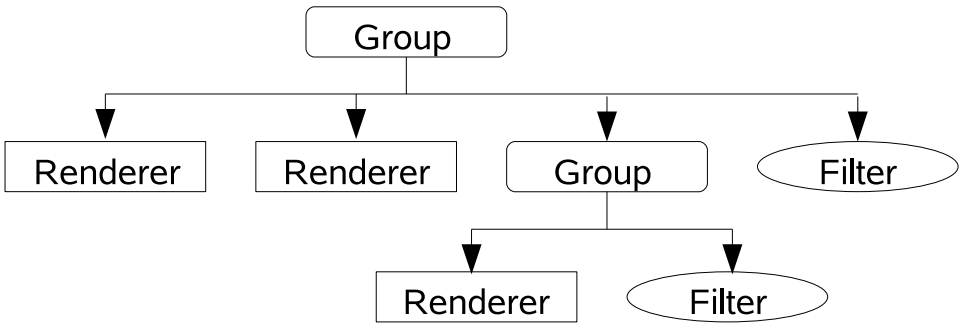

Figure 3.14: An example of a renderer tree showing the three different types of renderer modules.

Such a goal can hardly be achieved without putting a "sandbox" around each renderer that separates the different visual results. Fortunately, OpenGL supports this idea through its *p-buffer* and *framebuffer object* extensions. They allow the application to switch between different OpenGL contexts / framebuffers for (offscreen-) rendering and afterwards reuse the results as textures, shadow maps, etc. So the first idea would be to render in different buffers and to combine them in a final step.
However, these extensions produce an additional burden for the graphics hardware and can slow down the rendering process significantly. Therefore they have to be used carefully when creating visual combinations in realtime. The consequence was to allow the user to incorporate knowledge about the current rendering application. Instead of a fully automatic rendering system, the application programmer has to design the rendering workflow on his/her own, using available rendering components and combining them in a hierarchical way. Thus, the framework supports the programmer in designing arbitrary rendering systems while giving him/her full control over the performance and quality of the final visual results.

The Renderer Tree

The main structure that controls the visualization process is called the *renderer tree*. This structure is organised in a hierarchical manner and contains the different renderer modules that create the resulting image of the scene. Figure 3.14 depicts a schematic example of a renderer tree.

For rendering, the tree is traversed in depth-first order, usually from left to right (see Figure 3.14). However, parent nodes are free to change the rendering order as well as the number of render passes of their children.
By default, all modules direct their visual output to the standard OpenGL framebuffer since this is the fastest way to create results. However, if needed, they may also make use of "temporary buffers" (such as p-buffers, framebuffer objects or depth sprites) at an intermediate stage, e. g. if a multi-pass

algorithm has been implemented. This is especially true for parent nodes in the renderer tree, since they are responsible for providing a meaningful visual combination of their children's output.
It is now up to the application programmer to select the appropriate renderer modules and to combine them in a way that suits the particular application. For this goal, he/she may choose from a set of different renderer types which are described in the following.

Outer Nodes: Renderers and Filters

One class of the outer nodes (or leaves) of the renderer tree is the renderer objects that actually create images from datasets. There are several renderers implemented into the framework that visualize different data structures: besides algorithms for rendering 3D volumes and triangle meshes, there are implementations available for light field rendering [40, 149] and distance visualization [156]. The results of these renderers can be further processed using two other types of renderers: filters and renderer groups.
Filters are defined as renderers which do not create images from scene data but rather use existing images created from other renderers and manipulate them. One example is an implementation of a so-called luminance filter that simply converts the current coloured image into a grey-scaled one. For debugging purposes, an OpenGL buffer filter has been implemented that overlays the colour buffer with the data from the "invisible" buffers, such as alpha, depth or stencil. An example of the usage of this filter is given in Figure 3.17.

Inner Nodes: Renderer Groups

The inner nodes of the renderer tree are called *renderer groups* and may contain several other renderer modules. They combine the rendered results of their children and propagate them to their parent node. The way of calling the underlying renderer modules and the combination of their visual results depends on the respective renderer group.
Several renderer groups have already been implemented. The simplest renderer group only calls the child renderers and combines their output using the OpenGL depth test. This is useful for grouping renderers and succeeding filters, and for organising the whole renderer tree.
A more sophisticated group is the so-called *occlusion group*. This class provides algorithms for blending two rendered images, the one already written into the framebuffer (image A) and the combined image of its children's results (image B), with respect to their corresponding depth buffers. Results are such that the parts of image B occluded by image A will either be fully visible or will be shown in a semi-transparent manner, depending on the algorithm selected by the user. The algorithm for semi-transparency has been described in detail in [156], where it was used to realise the blending between light field and distance-mapped geometry. Figure 3.15 shows an

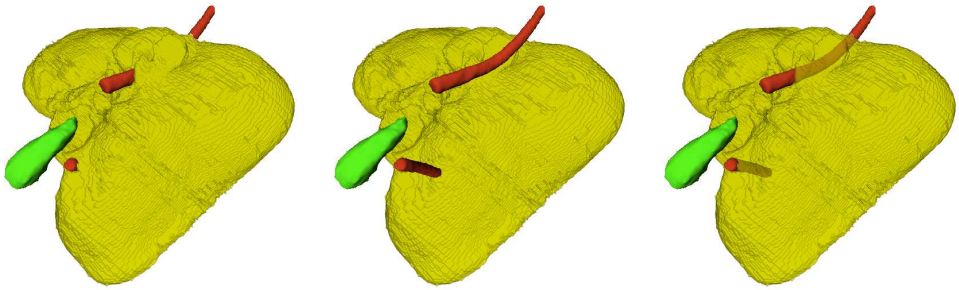

Figure 3.15: Medical application example: A liver phantom dataset consisting of a binary segmented liver volume, a triangulated gall bladder and two triangulated tubes. The tubes can be made either completely visible (middle) or semi-transparent (right) using the occlusion group.

example of the usage of this group.

Another useful renderer group is the so-called *shadow group*, which enables shadowing on the output of its child renderers. The main idea behind this group is based on a two-step algorithm related to hardware-accelerated shadow mapping [132] and deferred shading [56]. In the first step, the scene is rendered from the light's position, i. e. the renderers are called with the current camera position set to the light position. Then, the shadow map is created by storing the resulting depth buffer image in a depth texture. However, the second step differs from the classic approach. Using original shadow mapping, one would render the geometry to be shadowed from the normal camera's position using automatic texture coordinates generation and projective texturing to create a shadowed result. Instead, since the shadow group only has knowledge of the rendered buffers of its child renderers, we use the depth buffer contents that were created by rendering the scene from the camera view point. This buffer then represents the visible surface of the scene generated by the child renderers. Then, we project each pixel of this depth buffer back into the shadow map and perform a standard shadow test using a fragment shader. Shadowing itself is done by darkening the corresponding pixels on the screen if necessary. Figure 3.16 shows an example of the shadow group.

Initialization and Workflow

The rendering framework is initialised by building the renderer tree. This is done by the application programmer by choosing the appropriate rendering components and connecting them in a tree-like manner. The root node, which must be a renderer group in our implementation, registers its direct child renderers for later execution. If a child itself is a renderer group, it will register its own child renderers, and so on. This way, the complete tree

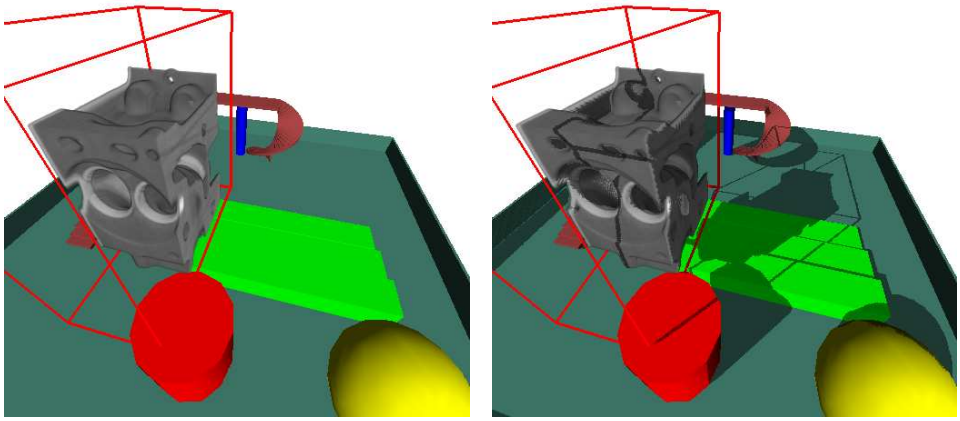

Figure 3.16: A scene consisting of triangulated meshes and a volume rendering with (right) and without (left) shadows.

is built, although some things should be respected. First, a renderer group should always be an inner node, i. e. it should have registered child renderers. Second, renderers normally are called in the order of registration at their parent, so the first registered renderer should not be a filter component. Figure 3.14 shows a schematic drawing of a renderer tree where these points are fulfilled (note that the leftmost renderers were registered first).

After the framework has been set up, images are generated by calling the rendering method of the renderer tree. When rendering the whole tree, it is traversed in a depth-first order. The root node calls the rendering methods of its child renderers, which will call the corresponding methods of their respective children, and so forth. If a renderer group's rendering method is called, it is completely up to the internal algorithm of the group how the visual results of its children will be merged with the existing result of previously called renderers. Renderers will always render to the current OpenGL context and framebuffer, which will override parts of previously generated results. If this is not desired by the group, it is responsible for either temporarily saving the current framebuffer content or switching to temporary buffers (p-buffers or framebuffer objects) for image creation. After its children have been called, the renderer group must combine their results with the original framebuffer content.
The behaviour of the remaining objects is straightforward: renderers will execute their implemented rendering algorithms and produce visual results, and filter components, which are called afterwards, will process these results and modify them according to their algorithms.
To summarise, the combination of visual results is done solely by the regis-

tered renderer groups in a hierarchical way. However, the rendering framework provides some convenient tools that simplify that task. First, an implementation of depth sprites allows for easy storage and restoration of the framebuffer content. Second, a "p-buffer stack" is provided, which contains information about the current nested OpenGL context hierarchy. That enables renderer groups and multi-pass renderers to temporarily switch to a new context for working and safely restore the previous one afterwards.

Rules for Renderer Integration

Although the framework aims to visually combine arbitrary rendering algorithms without restricting them in their generality, experience has shown that there has to be a small set of rules that every implemented algorithm has to respect in order to guarantee smooth combinations with other renderers. These rules address the treatment of the different OpenGL buffers in particular since these are the basis the framework uses for the different visual combinations. Our guidelines presented here are the core programming rules that an implementor must respect in his/her code.

- **Preservation of Results:**
  Rendering algorithms must not destroy the visual results of preceeding renderers. If a specified initial state of the OpenGL buffers is required, e. g. at the different steps of a multi-pass method, then these algorithms have to use the aforementioned "temporary buffers" to create their visualization and afterwards must integrate it into the existing results. Filters are an exception to that rule since it is their main purpose to modify visual results.

- **Colour-depth Consistency:**
  The common depth buffer should only be modified in those parts where the colour buffer is also filled with new information. The other way round, changes in the colour buffer should also lead to a meaningful modification of the depth buffer contents. Figure 3.17 shows an example of colour-depth consistent volume rendering.

- **Stencil Buffer Invariability:**
  Renderers must not change the stencil buffer contents for succeeding renderers. If stencil operations are necessary, they have to be encapsulated inside a temporary buffer.

- **Render State Preservation:**
  Renderers should not take any specific OpenGL state for granted. They are responsible for setting their own state at the beginning of the rendering process and reverting it to the initial state when finishing. However, the implemented algorithms are free to modify the OpenGL

Figure 3.17: Consistent buffer usage in the case of volume rendering. Note that colour buffer (left) and depth buffer (middle) give a consistent impression of the scene. In contrast, the picture on the right shows a possible depth buffer footprint of the volume rendering without respecting colour-depth consistency (image zoomed out).

state for the duration of the rendering process. This is especially useful for renderer groups when applying effects to the results of their children.

3.2.4 Integrating Various Rendering Algorithms

After describing the general idea and details for the inner nodes, we proceed and discuss how three of the most common rendering algorithms can be adapted to fit into the framework. The following visualization algorithms are implemented in our framework and respect the integration rules mentioned above.

Meshes

Mesh rendering can be integrated straightforwardly, as it fulfils most renderer rules in the first place. However, there are some points to keep in mind, although they are normally taken for granted. A mesh rendering algorithm must enable the GL depth test and adjust it such that a colour-depth consistent result is guaranteed, which is true for the default settings of the depth test. Additionally, when using multi-pass algorithms such as depth-peeling [39], the intermediate steps of these algorithms have to be done in a temporary buffer, and the final result has to be merged with the existent colour buffer content.

Volumes

Common hardware-accelerated volume rendering algorithms use viewer-

aligned textured slices to visualize volume data. In general, they can easily be used together with mesh rendering and provide a reasonable visual result by rendering meshes first and volumes afterwards. However, they rarely provide colour-depth consistent results (see figure 3.17, right).
A simple solution to this problem is to use the GL alpha test and discard all fragments that are occluded anyway. This way, far more consistent results are achievable, as can be seen in the middle of Figure 3.17. However, this does not solve the problem of semi-transparent fragments in the visual volume, which cause noticable artifacts when rendered together with intersecting geometry. Here, the rendering framework enables the programmer to change the order of rendering algorithms and to design the rendering pipeline such that volume rendering occurs last.

Light Fields: Unstructured Lumigraphs

For consistent combination with other objects in the framework we need to use a lightfield rendering technique that supplies depth information for each image contained. To acquire depth for our light field, 3D locations of feature points can be generated and a depth map can be obtained from shape-from-motion methods [129][41]. Alternatively, we capture the depth map with specialised hardware such as a photonic mixer device (PMD) camera [159]. Finally, for efficient rendering it is convenient to store a sparse triangulated version of the 2D depth map (local proxy [149]).

To synthesise a new view we proceed as in [40] by creating a triangular mesh of the light field scene oriented at the camera (screen mesh). Screen mesh construction has to be done for each ULG for each frame and requires the synthesis of coherent depth values for the whole extent of the light field that is assigned to each ULG. Screen mesh resolution is adapted to the extent of the light field scene by pruning away unneeded vertices (see Figure 3.18).

According to [40] the vertices of the screen mesh have to be corrected for depth. This is done by selecting a number of cameras whose depth information is considered relevant for the view. When rendering with local geometric proxies, the local meshes can be rendered into a backbuffer and the lookup can be done for each viewer-aligned screen mesh vertex. There is always a number of screen mesh vertices for which no valid depth value can be determined. This is the case when no information is present in the light field for a ray sampling beyond the bounds of the recorded scene. For these vertices the corresponding depth has to be set to a meaningful value and is crucial for the interaction with other geometry in a rendering framework. We decide to set the invalid depths to a heuristic value based on the diagonal of the bounding box of the light field scene. This bounding box is found by intersecting camera frustums in a preprocessing step. We then proceed to delete screen mesh triangles if all vertices of a triangle are at invalid depths and only keep those connected to the valid part of the screen

Figure 3.18: Screen mesh generation: Top left original and right pruned mesh. Lower row: side-view. Notice the alpha-blended triangles towards the background.

Figure 3.19: *Milk*-light field: captured from a hand-held camera it shows a detailed part of a desktop scene recorded with 16 images. See right for the four cameras used for blend field construction (frustums coloured for better visibility)

Figure 3.20: Smooth transitions between two unstructured lumigraphs (*Milk*-light field) combined with two lucys (polygonal surfaces). Right: correct shadowing in our rendering framework

mesh (see Figure 3.18 top right). The only remaining triangles with invalid vertices are found at the border of the light field. These are alpha blended towards the invalid region and so provide a smooth transistion to the framebuffer contents that may have been set already by preceeding renderers (see Figure 3.18 bottom right for a side-view).

After generating a depth-corrected screen mesh for each ULG, a *blend field* needs to be constructed as in the previous approaches [18, 40]. The blend field describes the weighting of the number of cameras that are considered for blending together to yield the new view (see Figure 3.19). The final image is rendered by projective texturing of the camera images onto the screen mesh with additive blending. Blend values for the border of the screen mesh are calculated as normal but appear to fade out due to the alpha compositing employed (see Figure 3.18 bottom right).

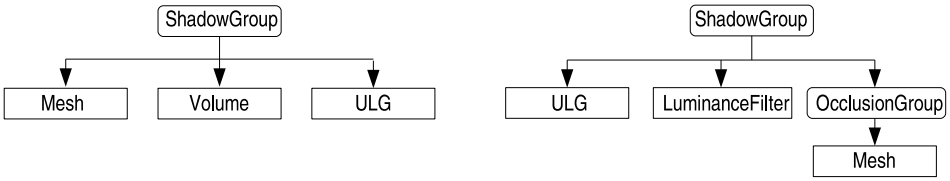

Figure 3.21: The corresponding renderer trees for Figure 3.22 (left) and 3.23 (right).

3.2.5 Conclusion

We have introduced a rendering framework that handles the visual combination of different renderer outputs. Furthermore, we have shown that our framework is adjustable to specific visualization demands and can be easily extended with new rendering algorithms. Figure 3.20 shows an example of several ULGs, highly complex triangle meshes and our shadow group described above. Further examples of different renderer trees and their visual results can be found in Figure 3.21 and 3.23.

The framework described in this paper is already being successfully used in minimally-invasive surgery of the abdomen, where ULGs and CT-data are rendered to provide an enhanced view of the operating field [156]. Furthermore, we are convinced that the flexibility of our approach enables it to be used in many other fields of application.

In the future we plan to employ the framework developed to show artefacts examined at the beginning of the chapter in the context of a virtual museum. Here, it is of great benefit to be able to synthesise light fields of exhibits with complex reflection properties as well as traditional geometry, e. g. for displaying the actual walls of the museum or a virtual display cabinet.

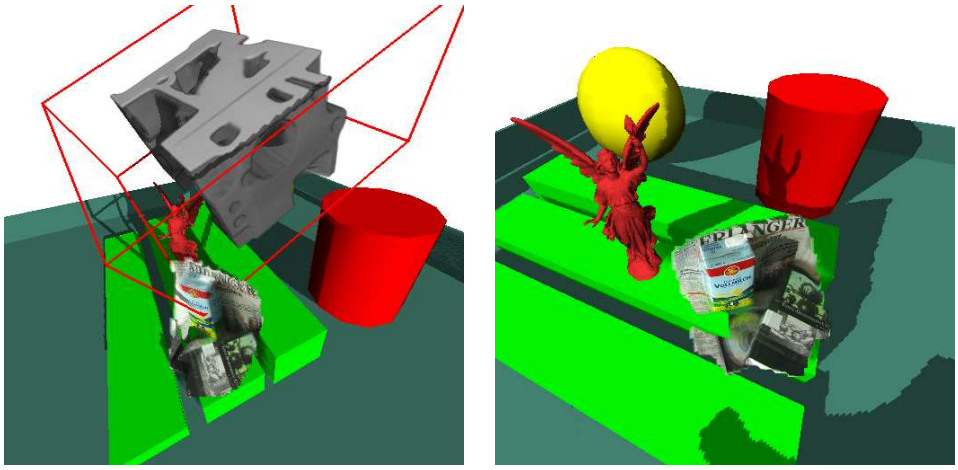

Figure 3.22: A combination of light field, volume (left) and mesh renderers, all encapsulated in a shadow group. Note that the light field is embedded smoothly into the scene, casting shadows onto the geometry.

Figure 3.23: A simple scene consisting of light field and mesh renderers, again embedded into a shadow group. Furthermore, the mesh renderer is embedded into an occlusion group, allowing the visualization of the geometry parts that are otherwise occluded by the light field. The figure on the right shows the additional usage of the luminance filter on the light field.

Chapter 4

Archaeological Reconstruction

The most frequently seen and commercially exploited form of archaeological display is 3D reconstructions of ancient structures. Mostly, virtual reconstructions are carried out for presentation in the media for example on television and for large-scale exhibitions. The most prominent example may the be latest Troy dig by controversial but famous archaeologist Manfred Korfmann, in which a complete virtual Troy was constructed from old and new finds and featured in the exhibition *Troja - Traum und Wirklichkeit* [143].

An important part of the work of an excavating archaeologist is to offer a detailed description of the work on the site. Additionally, an interpretation of the findings is presented alongside all the actual information about the dig. Part of the interpretation of such finds is the reconstruction of man-made structures to visualize their former form. What then follows is the discussion within the archaeological community about the validity of the interpretation and the correctness of, for example, the reconstructions carried out. It is here that virtual reconstruction methods can play a vital role in research. Giving the researcher the possibility to quickly change parts of the building, cut through it and have same-scale comparisons with different structures from the same cultural time and region provides the basis for a more plausible argumentation for changes in the original reconstruction proposal.

Nevertheless, while the benefits for public display are no longer questioned [136], the credibility and usefulness of virtual images for research is debated in the archaeological community. The main problem is identified as one of detail rather than overall impression. As stated in [48, 126], it is impossible to know every detail of an ancient building and even for very well known structures and short periods of time, for example the Roman Forum, there are unbridgeable gaps that need to be filled plausibly. Regarding the recon-

struction of large sites, special care is needed not to suggest inhabitation where actually no remains are proven. In the case of virtual Troy, a few buildings that were carelessly inserted below the fortress mound have led to a heated scientific dispute which the German media described as *the new war of Troy*.

In the following, our analysis of several examples of reconstructions carried out between the museum (Antikensammlung) and the Chair of computer graphics at the University of Erlangen. Our focus is on coarse-scale reconstruction of the main structure of a building where only post-holes or indentations of walls remain. In one of our cases, a legionary fortress at Marktbreit, the only visible marks are dark discolourations (post-holes and walls) in a field. Here, educated guesses are needed and the idea is to give an overall idea of the building complex. Small-scale detail such as plaster ornamentation or other fine-grain features are of no importance here, mostly because none of these features were found. Furthermore, a discussion of how virtual models can be inexpensively used in real-time on the low-performance hardware of any personal computer, as well as the important dispute in the archaeologist community about the accuracy and responsibility of digital models, follows.

4.1 Modelling of Structures

After excavation, large-scale structures present a problem for archaeologists. Since the protective layer of earth is removed, the site becomes subject to either preservation or renewed protective burial to protect it from erosion. This is true of the crumbling of the walls of Pompeii as well as the remains of Roman fortresses under Bavarian soil. Additionally, it is especially tricky to display the awe-inspriring constructions of our ancestors to the public. Few museums are big enough to house a Roman temple, or even a bronze-age one. For example, the Ishtar-Gate at the Pergamon Museum in Berlin gives an impression of the size of the museum rather than that of ancient Babylon. While hardly fitting into the room of the museum, the ancient magnificient impression of a solid ceramic tiled structure rising over 14 metres high from a lofty procession street sided by brick city walls is completely lost on the onlooker.

Typically, such problems of architectural context are solved and the public is educated with models. Traditionally, models have been built with sticks, rods, suspended wires, plaster and a bit of colour for a finished look. Often, the scale is 1/100, thus displaying a real length of one metre with a mere centimetre. As we will later see, with these models the issue of accuracy is a surprisingly small one for archaeologists. In the following, it will transpire that there are multiple uses for virtual models where they are better suited

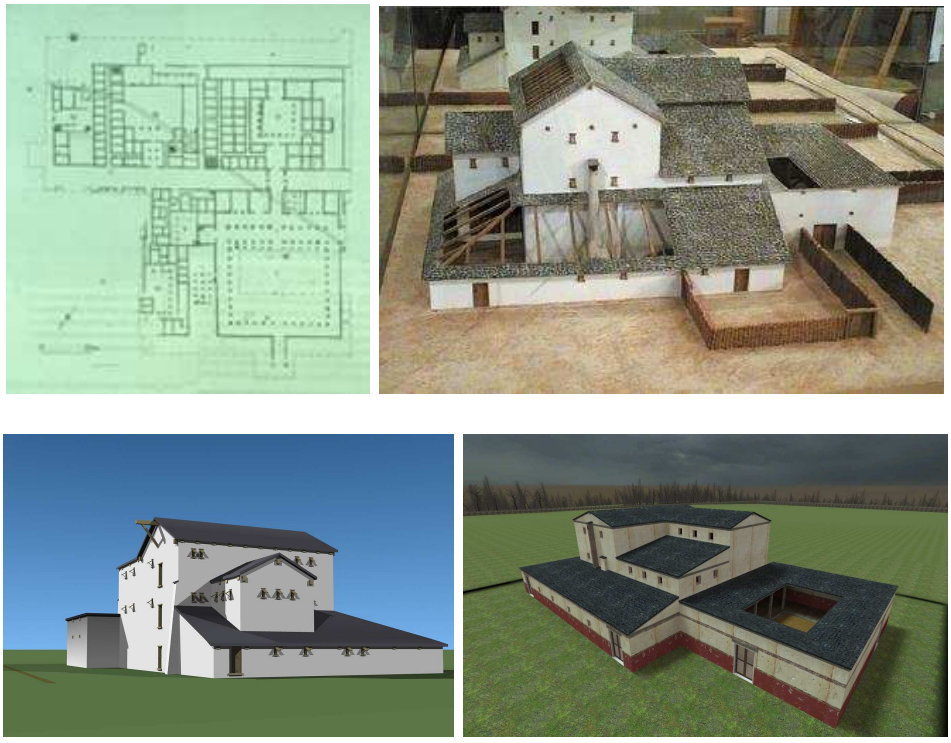

Figure 4.1: Reconstructed model of the *Fabrica* of Roman camp Marktbreit. Top row: plan of walls and post-holes, plaster and wood model (scale $1/100$ by F. Höchsmann [152]). Lower row: virtual building modelled in *LightWave* by Dr. Martin Boss and game engine (Half-Life2), modelled in solid modeller *Valve Hammer Editor* by Alexander Bardosch.

Figure 4.2: Initial data to start modelling from: orthophoto of dig site and reconstruction drawing

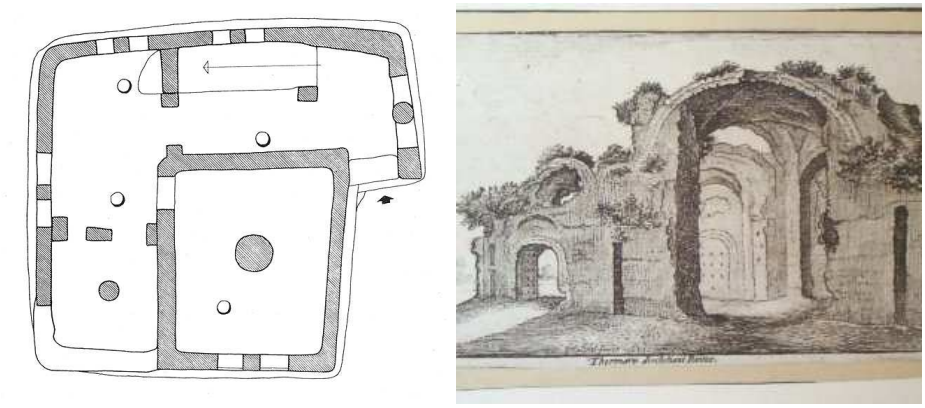

Figure 4.3: Modelling data: wall plans (house of Archanes in Crete) and old drawing (Wenzelaus Hollar, 17th century)

or of the same instructive qualities as real wood-and-plaster models can be.

For an archaeologist there are several kinds of data to start modelling from (see Figures 4.2 and 4.3):

- Orthographic photographs (on-site, aerial or satellite)
- Wall plans (wall indentations, post-holes)
- Reconstruction plans of the full building
- 3D Survey data (3D registered point data from scanners)
- Old Photograph or drawings/lithographies

- Literary evidence for unknown, destroyed or inaccessible sites (e. g. see famous traveller Pausanias' work [109] from 200 BC)

Depending on the material that is consulted, there are several techniques to create a model:

- *Extrusion* of (scanned in) outlines of walls and subsequent insertion of doors, windows and roofs
- *Triangulation* of survey data
- Modelling from scratch by an experienced modeller (often a digital artist and not an archaeologist)

In our work we focus on the reconstruction of buildings where there are very little remains of the original building substance. The site of the Roman camp at Marktbreit [112] is comprised only of indentations of walls and post-holes in the ground. Different but inherently similar sites are found all over Europe, most notably where the local population has either burned a primarily wooden place down or used a stone structure as a quarry for their own houses. An example of the latter case is our reconstruction of the palace of Pylos in Greece where only the stone foundations of the first half metre of wall are preserved.

For the afore-mentioned types of sites, wall plans can easily be drawn. Reconstructing the ancient building is done with the knowledge of similar sites, ancient building plans or simply by common sense, i. e. by finding plausible solutions to possible setups. It is precisely in this scenario, the finding and easy realisation of feasible arrangements in which virtual reconstruction is most advantageous. Unlike a reconstruction of a well-known site (for instance, the Roman Forum or Colliseum in Rome), here an inexperienced modeller can present new interpretations or his own interpretation without the risk of being caught up in detail issues or exaggerated accuracy demands.

4.1.1 CAD Design

Computer aided design (CAD) has become a standard for industrial blueprints and the creation of virtual worlds. Available digital modelling packages such as 3dsMax, Maya, Blender or Lightwave (there are, of course, many more) and professional modelling software such as AutoCad exist to create virtual structures. Since modelling in AutoCad is mostly done by engineers or architects, we will only consider the first mentioned applications that are commonly used by digital artists, and give several examples.

Constructing a virtual building needs skills quite different from that of a typical draftsman. Drafting skills are common among archaeologists as it

is seen as a necessity to draw especially when working on a dig. Digital modelling skills can be readily acquired but the learning curve for each individual software is high. In short, points need to be placed in a 3D world and polygons need to be defined on them. Also, working from simple primitives is possible. Visual quality is produced by applying textures to a model. To texture models, high quality (i. e. minimal distortion) uv-coordinates need to be generated, which is a non-trivial task still being researched within the computer graphics community.

The following list gives the main points that need to be kept in mind in order to produce good models:

- maximum number of polygons and points (geometric complexity of the model)
- good-naturedness of the polygonal surfaces, i. e. no hanging vertices or holes in the triangulation (quality of the model)
- creation of minimal stretch texture coordinates (quality of the texturing)
- selection of meaningful (archaeologically correct) textures
- seamless blending of textures and possible pre-lighting (ambient occlusion or global illumination in light map)

In cooperation with Dr. Boss of the University's museum (Antikensammlung) in Erlangen, several buildings have been reconstructed using 3D software, most notably the Roman camp at Marktbreit and the *Palace of Nestor* in Pylos. For the former, *Fabrica*, *Principia*, *Praetorium*, main gate and wall sections were reconstructed with Lightwave (see Figure 4.1 left). The original data (plan of walls indentations and postholes) was imported into the modeller through overlay.

Problems encoutered during these reconstructions were modeller issues with the 3D software itself. Duplicated points, quadrilateral faces with deviating vertex normals and holes in the triangulation can easily occur when no experience in modelling is present. Furthermore, during the first such project the time spent on the model prohibited further work on any texture coordinates. So, only primary colours were applied to geometric patches so as to visually separate wood (brown) from plaster (white) or roof (red, see Figure 4.1 and 4.5). The resulting models were not textured and subsequently looked "flat", i. e. gave the buildings a rather plain look. This look is actually favoured by many archaeologists [44] because of its visual clarity.

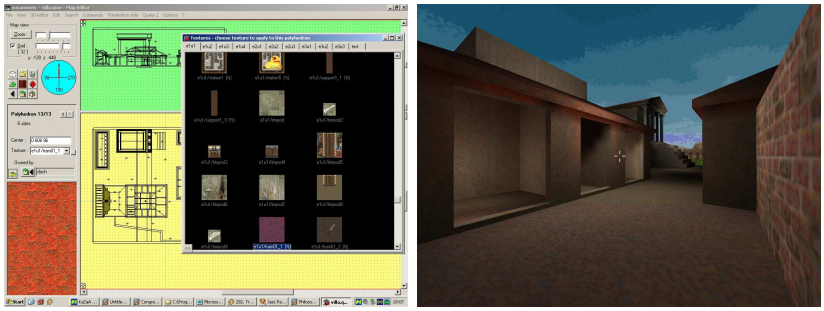

Figure 4.4: Screenshot from open source solid modeller *QuArk* and game view for Quake2-Engine: study of a Roman *domus* and temple

4.1.2 Constructive Solid Geometry Modelling

Constructive solid geometry (CSG) is a modelling technique used to create digital content. With the help of simple, convex primitives, such as boxes, spheres, prisms, pyramids or cylinders, complex objects can be formed. These primitives are considered to be solid, meaning that a function exists for which a 3D point can be tested as to whether it is inside or outside the primitive. By construction, the intersection of the planes the solid is made of defines the primitive. This way, addition and subtraction of objects can be executed efficiently. Furthermore, information exists about the final model. There is a well-defined inside and outside and texture coordinates are straightforwardly set since mapping the primitive is easy.

A modeller does not have to deal with points and polygonal faces but simply with solid objects, making the construction process more intuitive. Constructive geometry has many uses, mainly in design. There are advantages also for rendering purposes. Computer game engines started using CSG solid modelled maps from as early as Doom (1993) and they are still in use with 3D shooter games (2004: Half-Life2, 2005: Quake 5, 2007: Unreal Tournament 3). Primarily, binary space partitioning (BSP [43]) can be efficiently computed from the half-spaces that the walls of the solids are made from. Additionally, hints for the internal collision detection and physics engine structures can readily be extracted from a building that was constructed with CSG. Furthermore, realistic lighting calculations (radiosity) can be calculated from the efficient spacial data structures. In the open source community, the similarity of many game titles has led to large-scale software development for editing these games. With *Quake Army Knife* or short *QuArK*, a semi-professional modelling tool is available that enables the creation of maps for more than 21 major 3D computer games (see Table 4.1.2).

CSG modelled geometry has its advantages for use in archaeological re-

Game title	Year
Quake	1996
Quake2	1997
Half-Life	1998
Quake3	1999
Medal of Honor	2000
Return to Castle Wolfenstein	2001
Doom3	2004
Half-Life2	2005

Table 4.1: Open source CSG mapping tool *QuArk* supports more than 21 major 3D game titles. Listed above are some of the most popular 3D games editable with *QuArk*.

construction in the easy construction process and the possible use in game engines, making interactive display of the most complex buildings possible without the need for specialised visualization software or browser plugins.

Case study: Polygonal modeller versus CSG

As an example, the polygonal model of the *Palace of Nestor* in Pylos (see Figure 4.7) was created in two weeks modelling time by Dr. Martin Boss in the Lightwave modeller. It features very exact extruded wall plan geometry but no texturing or global illumination. Additionally, a lot of manual work went into finding suitable views of the object, animating video paths and the rendering time of the images for still images or animation [11]. Quite differently, the solid geometry model was created with *Valve Hammer Editor* (supplied for the editing of Half-Life levels from the game vendor) in two days (see Figure 4.7). Its geometry is not as exact as the coordinate space is gridded, but the available texture library makes up for it with visual realism. Additionally, Half-Life delivers real-time framerates and a physics engine that makes it possible to walk round the palace. Also, a fly mode is offeredas well as precomputed global illumination. Compared to the static renderings of the Lightwave model, this is a preferable reconstruction. It offers a much higher flexibility when the CSG model needs to be changed since the CSG solid objects can intuitively be understood as wall segments, column pillars or ceiling panels. Despite from the map building process (explained below) there are no preprocessing times for newly found views.

In the following, we will discuss how to render reconstructed ancient buildings at interactive framerates and present more examples of our work.

4.2 Interactive Display

A virtual model of a building alone does not improve insight or promote its features. In the past, digital models have been used to render still images or movies displaying a virtual flyby through a building [11]. Comparing old and new techniques, still images are the same as drawings by draftsmen as they are static, offer high resolution for detailed examination and are, thus, generally accepted among archaeologists. As mentioned before, with movies the same problems hold as with artefact presentation. Moving pictures offer no control for the viewer as he/she is forced along a restrained path. The notion of time is introduced and it is difficult for the onlooker not to miss important features. The insufficient resolution makes a proper examination even harder. In short, movies are not an adequate medium for research or teaching.

In short, we conclude that users of digital models need control over the image synthesis process. This includes the view they choose to take of the object and furthermore, but less importantly, the way they want to look at it (rendering styles, camera parameters, lighting). Since real-time graphic display of models of moderate complexity are feasible with even the most inexpensive graphics hardware, we suggest always using a 3D viewing application.

There are several problems that immediately surface when giving a user total control over what he/she sees:

- What does he/she expect to see and how is the model presented (appropriate texturing, lighting, sense of scale)?
- Are the controls that are used to change the view intuitive?
- How can the user move his/her virtual position and how is it constrained?
- Can a sense of direction and location be established?
- Is it possible to focus on and access relevant information?
- Does the user feel immersed in the environment and is it desirable to be immersed?

In the following, we will describe methods used to solve the above mentioned problems. Firstly, any 3D library or software can be utilised to simply display virtual models. But questions remain about whether the amount of required development and the rendering quality are sufficient to create virtual worlds. Finally, a solution using computer game engines is found to be preferable, its application is discussed and accuracy issues are mentioned.

4.2.1 Visualization Toolkits

A polygonal virtual model of a building contrived from points and polygons can be easily displayed in any 3D software. Conversion between various extisting graphics formats means that almost any graphics library can be used. Apart from the modelling software itself, which uses DirectX or OpenGL previews for the modeller to have visual feedback, stand-alone applications often need to be adapted to special needs to display ancient structures. This necessary programming is mentioned in many publications that deal with the development of such visualization software [44].

There are several well-suited 3D application (and visualization) frameworks that offer converters for geometry files and texture support:

- OpenInventor
- OpenSG Open Scenegraph
- Java3D
- VRML
- VTK

With these libraries and a bit of user interface programming a technically sound 3D model (regarding the triangulation, texture coordinates and appropriate textures) can be displayed. Real-time performance of the model display relies strongly on the graphics hardware involved since these libraries do not employ any speed-ups for rendering complex single-textured geometry. In other words, they simply render any polygon soup they are fed with. Furthermore, standard techniques such as multi-texturing, lighting or shadow mapping are not necessarily included and need to be implemented. With modern embedded graphics chips the untextured display of the *Fabrica* model in Figure 4.5 is no problem with regard to geometry alone (50k faces and 25k vertices). This situation was quite different in 1996 when the model was first designed. Nowadays, high resolution multi-textures (4,096×4,096 RGBA) pose the greater problem for these chips.

Apart from rendering speed, visual quality is another important issue for the display of virtual reconstructions. Texturing to give hints about the material properties alone does not solve the problem of a model looking flat. The reason for the synthetic look of a model is that the lighting situation is unsatisfactory. Several techniques for *texture baking* that manipulate the texture atlas can be used to alleviate this problem where only one texture can be shown (see Figure 4.6):

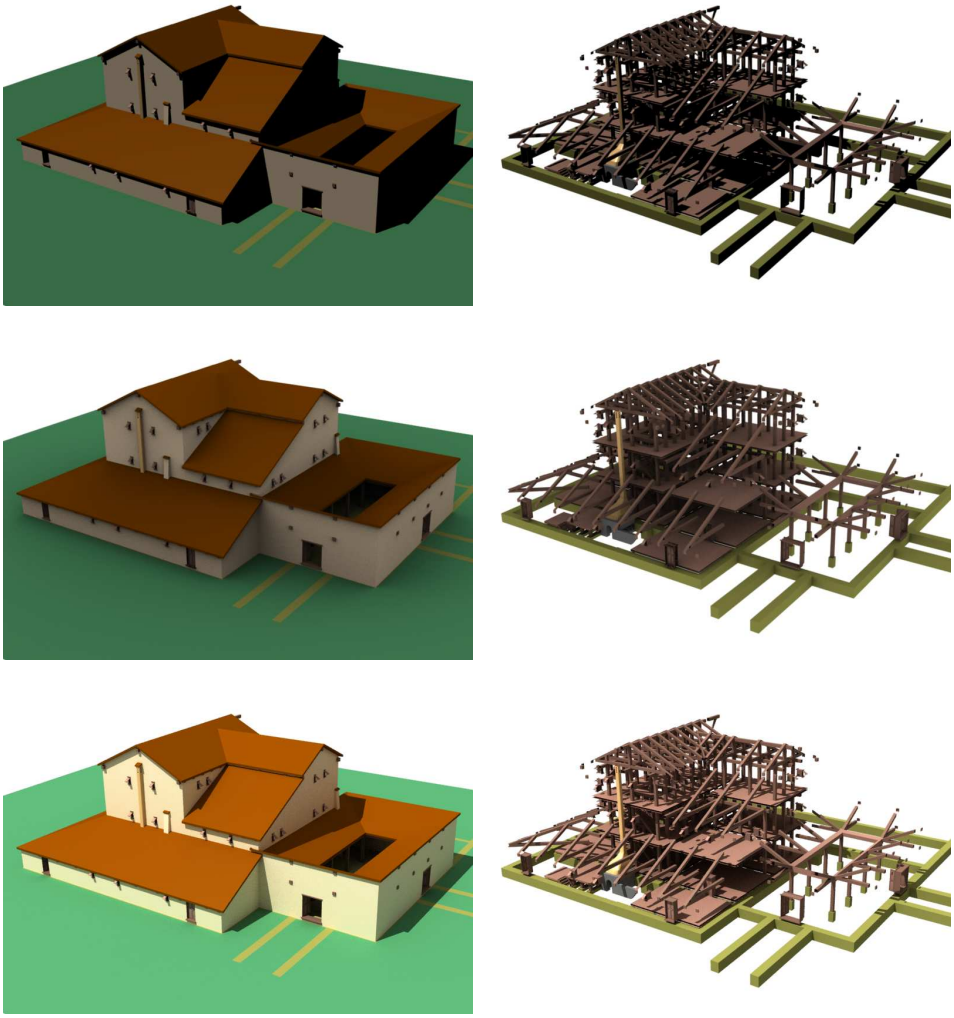

Figure 4.5: Shading the *Fabrica* model. From top: Shadowed Gouraud shaded, ambient occlusion and global illumination (photon tracing in YajRay Renderer). Global illumination can be seen as a mix of direct shading with hard shadows and ambient occlusion, except for the colour bleeding visible in the greenish tinge of the lower walls (see top right picture).

Right column: Visualization of the wooden supports and excavated foundations (in light green) show the complexity of the building and reconstruction.

- Shading with ambient occlusion (*sky light* or *dirt shader*): good for visual depth because it puts emphasis on broadly exposed patches and darkenes creases. Preprocessing time is fast and real-time are algorithms available for certain scenes [72]. No light sources need to be set up, environment maps can be used for use of realistic pre-recorded light situations.

- Shading with a global illumination solution: requires copious amounts of processing time, usually in a preprocessing step. Until now no usable high quality real-time algorithms available for scenes of moderate complexity. Effects of the lighting have a high degree of realism with sometimes unexpected behaviour (colour bleeding, very high dynamic range needs to be mapped to displayable brightness range). Usually the setup of light emitting surfaces is a non-trivial task.

Light transfer dependent on the placement of doors and windows is of great interest in archaeological research. Since the only light source in antiquity was sun radiation and candles, computer graphics outdoor illumination techniques can be applied successfully. Here, heuristics (ambient occlusion) or realistic solutions to light transfer can help tremendously to, on the one hand, add to the realistic look of the model and, on the other hand, add useful information for the archaeologist. While the techniques exist and are widely implemented in commercial software, no general automated solution for real-time rendering is provided. In the case of both ambient occlusion or global illumination, most software solutions provide for static renderings where the required lighting mode can be switched on. Rendering to a light map texture for real-time display is a relatively new extension to most rendering packages (examples include Lightwave and Blender), no doubt driven by the gaming industry in which light maps are widely used. Care has to be taken that appropriate texture coordinates need to be found and a sufficient resolution for the light map is chosen to reduce aliasing artefacts and allow for enough texels per surface.

Controlling user movements and looks is crucial for the inspection work an archaeologist will need to do. The graphics programming libraries mentioned above do supply interaction styles with the user. However, these mostly include trackball features and a fly mode. While this is sufficient for an experienced user like a digital artist using 3D rotation and translation in his daily work, these forms of interaction with structural data such as buildings do seem rather out of place. It is more appropriate to simply walk around the place and, for details, maybe use a zoom lens for close ups just as one would in real life. Such a walk mode is not provided by most visualization libraries and programming is indispensable. There are several reasons for this. Firstly, a force of gravity is not present and secondly, it is not clear just from file import in what direction this force should work. Furthermore,

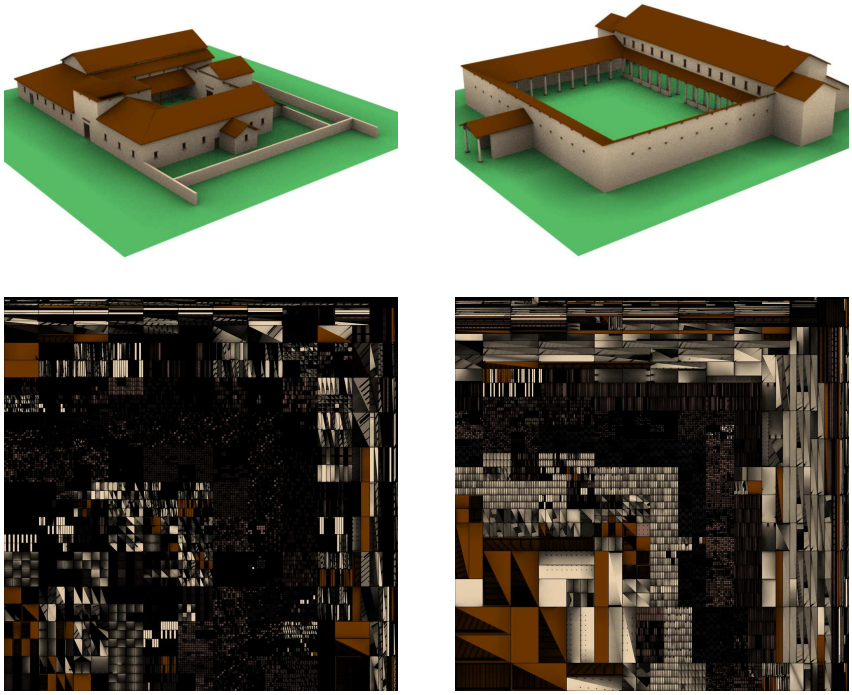

Figure 4.6: Texture baking global illumination for display of virtual models with 3D viewers. Roman camp Marktbreit *Praetorium* and *Principia*. Reconstruction by Dr. Martin Boss. Lower row: corresponding texture maps encoding primary colour and ambient occlusion.

costly collision detection with the walls would be necessary to inhibit the user from walking through them.

4.2.2 Game Engines for Teaching and Research

The exhibition "Die Römer zwischen Alpen und Nordmeer" in Rosenheim [151] during the year 2000 offered the opportunity to experiment with real-time and interactive virtual 3D reconstructions. There, the Roman legionary fortress at Marktbreit was presented to a wider range of viewers. In view of the huge number of schoolclasses visiting the exhibition and the fact that most children today are familiar with computer games, a modification based on a first person action shooter (FPS), Quake2, was developed [152].

Before we go on to discuss the pros and cons of game engines as 3D visualization tools, a definition of the notion of a game engine is apt. An engine is the core module of a software package that deals with the rendering of geometry from the memory of the computer or graphics hardware directly

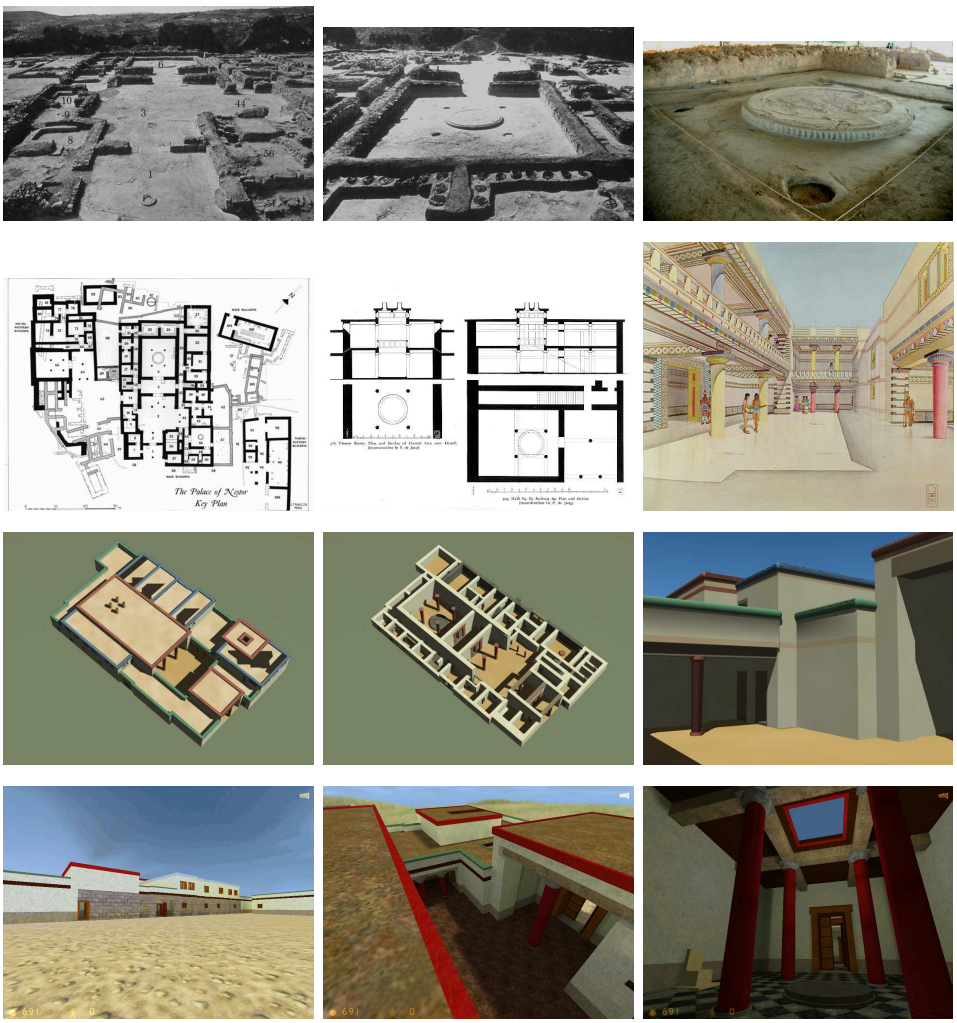

Figure 4.7: The *Palace of Nestor* in Pylos, excacated by C. Blegen in 1952 and drawn by his draftsman Piet de Jong for publication in 1966 [6].

Top: actual on-site findings of walls (the famous throne room to the right).

Second row: excavator's floor plan left and reconstruction drawing as plan and perspective colouration.

Third row: reconstructions in Lightwave on the basis of a new reconstruction proposal by Dr. Martin Boss.

Bottom row: virtual solid geometry model in Half-Life game engine (1998). See throne room to the far right.

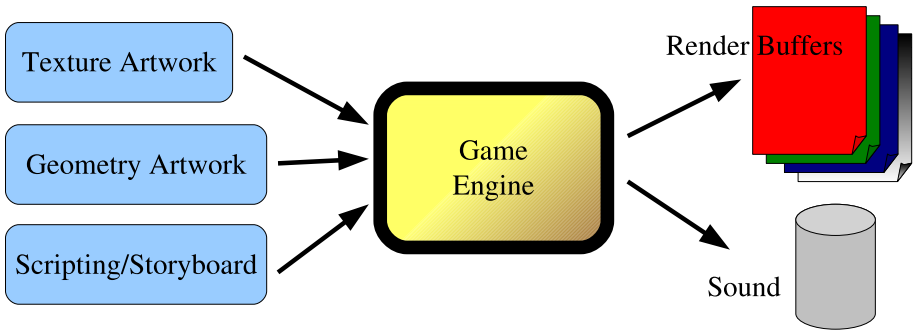

Figure 4.8: Game engines and their use in programming games

to the screen (see Figure 4.8). It creates a transparent layer that wraps around the hardware and enables the user to process geometric shapes.

In the game industry this division of, on the one hand, geometry and texture artwork and, on the other hand, the render engine existed from early on. Games manufacturers licence graphics engines for their software and do not have to programme low-level graphics functionality at all. Instead, they concentrate on the nature of their game (scripting pre-defined behaviour and artificial intelligence) and the visual artwork, such as geometric shapes and textures by digital artists. This is often mirrored in the fact that game production studios employ only a few programmers but many artists who take care of scripting, story development, texture and geometry artwork. Editing software is provided alongside most engines (for example *Valve Hammer Editor* in Figure 4.4 for solid modelling engines of the Quake and Half-Life series and many more similar titles). Consequently, the gaming community is able to create modifications, changing the appearance and behaviour of these with the aid of the game manufacturer, but the engine cannot be altered. These modifications and the ease with which they are created have recently produced a whole field of new games designed by pure amateurs. These so-called *mods* rely only on a specific game being installed and solely add new models of enemies, weapons and items, as well as texture artwork and scripting of the newly implemented behaviour.

Archaeological reconstruction and visualization can benefit from this situation, as we presented to researchers at CAA in Vienna [97]. Most archaeologists will want to change the geometry displayed and hide most of the gaming aspects. While a modification normally focuses on changing gameplay, textures and geometry, for educational purposes it is necessary to make shooting and killing impossible, thus transforming the game into a visualization tool. As in Quake2, modified dynamic link libraries (dll) need to be programmed that make the player invulnerable and hide default weapons, stat bars and headup displays (hud). For Half-Life and most newer games,

Construction of the map file describing the CSG geometry	*map*
Computation of BSP tree	*bsp*
Portal visibility	*prt*
Potentially visible and hearable sets	*vis*
Radiosity calculation for realistic lighting and calculation and storage in lightmaps	*rad*
Accumulation of the data in the bsp file	*bsp*

Table 4.2: Steps to generate a map for common 3D games that use binary space partitioning. In brackets the associated file names of the data generated. The tools required for each individual step are supplied by the games manufacturer and a variety of modifications by the user community can be found on the internet.

such changes no longer require programming but rather scripting work or simple commands via the built-in console of the engines.

Technologically, the above mentioned games all use binary space partitioning for a hierarchical representation of the geometry data. We will give a very brief outline of how a level for such games is created (see Table 4.2). For a more thorough discussion of the terms mentioned below we refer the reader to a series of books called the *Game Programming Gems*.

With the CSG description of the geometry of the map (the *map*-file), brushes are created and stored in a modelling tool (see *Valve Hammer editor* or *QuArK*). A brush is a three-dimensional structure, e. g. a cube or another solid convex polyhedron. After that, a binary space-partitioning (*BSP*) algorithm calculates a hierarchical data structure for the 3D scene that will be used for speeding up the following steps (*bsp*-data). After that, *portals* (*prt*-data) are calculated that help in the later real-time rendering of the map file by storing neighbouring access possibilities in the tree for fast lookup (the *prt*-file). The next step calculates the *potentially visible set* and the *potentially hearable set* and stores its results in a map (*vis*-data). Then, radiosity lighting is baked into lightmaps in a time-consuming steps (*rad*-data). After this last step, all the above generated information is stored in the final *bsp-file* after the already present binary space-partitioning data. This whole process can last from minutes for a small map-file and low quality settings to several tens of hours for large and complex maps with high light map resolution and global illumination settings (e. g. high number of indirect light bounces).

Several points can be put forward that support the case for using computer games in archaeological visualization of reconstructed ancient buildings. In the following list we will group these points in three categories, *rendering* as a technical aspect, *immersion* for audio-visual quality and *utility* as

a description of useful add-ons that would need programming when using stand-alone graphics libraries from scratch.

- state-of-the-art graphics: efficient rendering of complex multi-textured geometry with lighting effects in real-time (**rendering**)
- sound is triggered depending on undergrowth and interaction. Footsteps during a walk-through add to realism (**rendering, immersion**)
- physics rigid body simulation that enables a player to walk around and climb steps, fall down ladders and bump into walls (**rendering, immersion, utility**)
- scripting engines are included facilitating an interactive, i. e. responsive environment. Events can be triggered, textboxes can appear depending on time and location (**immersion, utility**)
- texture artwork is included in the game, for example the *.pak*-Files in the Quake-series games (**immersion, utility**)
- looking around is mapped to mouse movement. From a fixed point where the user stands he/she is able to scan his/her environment by virtually turning his/her head. Virtual zoom lenses can be scripted by changing the field-of-view of the camera (**immersion, utility**)
- 3D content is present in games that can be reused to augment a scene, for example trees and bushes for vegetation, or chairs, tables and other accessories (**immersion, utility**)

For the above reasons we conclude that for archaeological visualization purposes the benefits gained from using computer game engines should be considered. With an inexpensive game (typically 50 Euros) a toolkit is purchased that enables the creation of reconstructions. Of course, modelling skills are still required, be it for solid modellers or polygonal modellers. But several other key abilities need not be present when carrying out a reconstruction with a game. There is no need to learn a programming language (as there would be with the aforementioned 3D libraries) to re-implement the features we listed above under *rendering*. The textures present are mostly sufficient, especially for the coarse-grain visualization that we carried out. This is an advantage because projects can be kept small, no digital artists need to be employed as a basic kind of *immersion* can be guaranteed. Scripting languages (for example the omni-present lua) can be learned easily and allow for a dynamic, time or context-driven events (*utility*).

In the following we present different application scenarios in which we used computer games for public audiences, teaching in the class room and research to show how easy it is to put our ideas into practice.

Public audience: Roman legionary camp Marktbreit

In the 1980s a large Roman military camp was found by aerial prospection of the area around Marktbreit and parts of it were excavated [112]. It is seen as a legionary stronghold that was able to house up to 24,000 men for a military expedition of Tiberius against the Germanic emperor Marbod in 6 AD. The camp was never inhabited long and was burned down by its builders after the Roman position at Marktbreit was given up because of an uprising in Pannonia in 9 AD. Traditional reconstruction drawings were done after the excavation by Martin Boss (see [112] for details). 10 years later, for the above mentioned exhibition at Rosenheim [151], the modified game Quake2 offered four virtual spaces (so-called maps), displaying various places within the camp (maingate, Fabrica, Principia and Praetorium), which could be explored in detail through free individual movement (Figure 4.11). In retrospect, the computer installation proved to be a good addition to the real plaster models which stood nearby (Figure 4.1). Additionally, the difference between virtual and physical reconstruction became very evident. While plaster models always stay the same scale, size and texture, the virtual model can easily be altered, individual view points can be found and the materials can be changed at the click of a button. It is notable that children in particular were drawn to the interactive exhibit as they excelled in the gaming skills of navigating in 3D using mouse and keyboard. Jumping off the gate's towers or exploring the many storeys of the Fabrica proved to be much fun.

Student seminar workshop: Attic oil mills at Legrena

The Greek farmhouse at Legrena was reconstructed in Quake2 (see Figure 4.9) for a lecture at the University of Erlangen. The unexcavated site was chosen because it is hardly visible today and its few remains are unattractive for tourism as it lies in the vicinity of the famous sights at Sunion. Nevertheless, from an academic point of view the site is of great importance since the economic wealth of Athens in classical times was based on such oil mills. The spatial arrangement of the various buildings and their functional relationship was evaluated. In cooperation with the students, a new interpretation was found for the controversial issue of the great tower bases near the mills. It seems that the traditional interpretation that the towers were used for defensive purposes is misleading. It is more probable that the towers were simply lookout points for surveying the oil plantation. Since the thickness of the walls and foundations do not support a very high structure, it is improbable that anything other than the surrounding area could have been seen as the nearby rolling hills obscure the view of larger distances.

Research: The Palace of Nestor in Pylos

With the so-called *Palace of Nestor* in Pylos, Greece [6] the subject to be investigated was the number of storeys at various locations throughout the

Figure 4.9: Oil mills at Legrena: the tower remains do not support the previous interpretation as a defensive structure

building (see Figure 4.7). In reality, only the foundations of the palace are preserved and the remains of several staircases give a hint of further floors (see Figure 4.7 top row). During the 1950s the excavator Carl Blegen and his draftsman Piet de Jong used these remains in their reconstruction drawings and duplicated the layout of the ground level at the level above. The new interpretation by Dr. Boss finds it more likely that larger rooms, such as the central megaron (the so-called Throne room) and its anteroom, are higher than the narrow chambers in the wings. On the other hand, some parts of the palace, such as later storage annexes, certainly lacked an upper level. The results of this revised reconstruction are theoretically discussed in [11] with traditional 3D modelling in Lightwave. For our publication in [97], we remodelled the structure in Sierra's *Valve Hammer Editor*. With the lightmap based radiosity lighting the model performed and looked better than the renderings done in Lightwave. Our presentation of the model in real-time on a consumer laptop was even more convincing as the user could actually climb the staircases and get a view for himself, be it from the lower storeys, galleries or even rooftop. To stay close to a scientific visualization we chose not to overload the building with textures of wall paintings that were found. Instead, we painted the different proposed storey heights in different colours: red for the highest room (the greek *megaron* actually means large or high room), green for galleries of the one-storey roofs and brown for the first storey. This way, our reconstruction resembled more a sketch than a fully fledged painting of a fictional reality, especially with regard to the upper floors. For contrast, see Figure 4.7 second row from above to the right for a traditional drawing. The disavdantage of such proposals is the insecurity of the colours, decorations and composition of the ornamentation and, most importantly, it does not add to the credibility of the reconstruction itself.

Research: The Throne of Apollon at Amyklai

Amyklai is a famous religious site in ancient Greece well-known in antiquity for its chair-like building around an archaic pole statue representing the deity Apollon. It represents a good example of how a reconstruction of a structure can be done where very little fragmentary architectural pieces

remain. Reconstructions have to interpret the description of Pausanias, a famous Roman who travelled Greek sites in 200 AD and described them in his book *Hellados Periegesis* (there, III:18.9 [109]). The difficulty lies in the different opinions on how to deal with the instruction that Pausanias gives and the limited archaeological evidence. As pointed out in our reconstruction in [12], former reconstructions (see Figure 4.10) often misinterpret the structure itself and deviate too much from the original description. Based on a comparison with the other literary reports of similar thrones with a high degree of certainty this rules out the previously done drawings because of the scale and arrangement of the architectural elements. We will not go into further details from the archaeologic discussion and continue to describe why our visualization with computer games proved an advantage. With solid modelling, textures can be assigned easily and we used it to show the many mythological scenes described by Pausanias (see Figure 4.10 lower rows left). All this imagery needed to be visible to be meaningful and to be described by Pausanias. The visibility and the scale of the drawings could be easily checked with our approach. Other reconstructions lack this plausibility and therefore credibility.

4.2.3 Accuracy Issues

Digital models of archaeological reconstructions or even virtual worlds differ in many ways from their plaster and wood counterparts as seen in Figure 4.1. Plaster models are built to illustrate spatial relationships and used mostly in museums to illustrate a possible context in which the exhibits were formerly used or found. Their static nature and miniature detail suggest that the display is more of a toy for children than hard science. It is explicitly clear that the model is a fictional setup.

With 3D visualization different rules apply. Partly because of available archaeometric data and the involvement of a computer, expectations are high. While a plaster model never will be the exact scaled image of a digging plan, the assumption is that the models in the computer should therefore be. It then sometimes is seen as disappointing when visually "messy" archaeometric data is visualised and used for texturing and modeling. Since measured data in the field suffer from noisy sensors and scanners, the data has to be worked over by an expert to be of use for display. Commonly, point clouds are obtained that need to be triangulated. While this step is largely automated, smoothing and decimation steps for these triangular surfaces and careful registration of unconnected partial views of a whole must be executed. The so acquired datasets can only serve as a basis for architectural reconstruction and the real use of archaeometric data is much more a means of recording a present day state for preservation in a modern eletronic way.

In our case, with the reconstruction of the Marktbreit camp we noticed

Figure 4.10: The *Thronos of Apollo Hyakinthios* in Amyklai:

Top row from left: reconstruction by Theodor Pyl in 1852, closely following the instructions by Pausanias [116] albeit in the style of his time. Martin (in Fichter [42] Figure 40) visualizes a large peristyle area without any evidence from Pausanias. Prückner reconstructs a massively oversized building in 1992 [113] not comparable to similar descriptions and archaeological finds.

Lower two rows: Interpretation by Dr. Boss that was checked for consistency with the ancient description, placement of the narrated imagery (see the backrest and armrests), static plausibility and the excavated remains. To the left artefacts of the gridded coordinate space in Quake2 can be seen in the main supports. To the right rendering (untextured) in Morrowing after 3ds file import.

Figure 4.11: Engine comparison for maps of Roman camp Marktbreit. Screenshots of wall gate, Fabrica, Principia and Praetorium: top two rows Quake2 visualization (year 2000) with a gridded coordinate space of 8 cm. Bottom rows Half-Life2 reconstruction by Alexander Bardosch in 2007, gridded coordinate space now below 1 cm. Note the better light map resolution and higher resolution detail texturing (including normal maps) in Half-Life 2. Additionally, Quake2 is a colour mapped engine meaning that at most 256 colours (the 8-bit palette of the game) can be displayed.

that due to the gridded coordinate space of the early Quake2 engine map generators, the smallest distinguishable unit we could use was 8 cm. This restriction meant that small-scale detail could not be modelled. Compared to plaster models, a unit of 8 cm would translate to a mere 0.8 mm. The presented newer game engines no longer have these restrictions and some even support curved surfaces, such as Bezier patches. In Figure 4.11 the same Roman camp, redone in Half-Life2 three years later, shows a very differently textured and lit but also geometrically more detailed and exact reconstruction. Without a doubt, despite the basic idea being still the same, the mere passage of time brought along better looking games that immediately benefit our previous work. Since our reconstructions are done in a game-independent way (CSG descriptions) they can be reused again and again for each new generation of the above mentioned games, making use of new texturing and lighting methods as soon as they are state of the art.

Reconstructions done on a computer are often criticised for the following reasons:

- They suggest detail in geometry or textures that is not present (no evidence).
- Controlling mouse and keyboard for navigation is a learning process for some users.
- The distinction between actual finds and reconstructed material is not present or not clearly visible.

In our case where we reconstruct from wall and post-hole indentations the common points of criticism are easily dispatched. On the one hand, we favour textures that are plain but fit the point (e. g. the plasterwork on the walls as visible in the reconstruction of Pylos). On the other hand, no texture at all reduces the realistic feel that the patinated plaster models offer. Geometric additions are similarly difficult and again have to be plausibly filled. This can be done by either referring to similar objects and copying their design or by using very abstract shapes as we have done in Pylos and the other reconstructions. Distinguishing between actual finds and reconstruction (in the case of Pylos the excavated walls in Figure 4.7 top row) is easily done by inserting geometry in a special colour that can be switched on or off to show the actual situation compared to the idealistic reconstructed shape. In our experiments this was not done, firstly because our focus was the reconstruction of the whole building and, secondly, because it would have been impossible to depict the excavated detail in the gridded coordinate space of the games we used for modelling and visualization.

In the future, CAD data from archaeometry can be used with games that facilitate direct 3D data input. We used *Morrowind*, a role-playing game,

as a testbed for such a purpose, depicting the throne of Apollo at Amyklai (as also explained in [97]). The structure was remodelled in 3dsMax and imported with the tools the game manufacturer supplied. This facilitates the inclusion of actual GIS digging data combined with the game's stunning realism (weather system, extensive vegetation) as can be seen in Figure 4.10. In contrast to the first person action shooter games we have used so far, Morrowind is a role-playing game that focuses on different gaming aspects. In role-playing a lot of time is spent talking to so-called *NPCs* (non-player characters) and items are acquired and studied. This is especially convenient to exploit for archaeological visualization since the fixation on killing is not present. In role-playing, the user can act as an archaeologist and actually pick up artefacts, examine them as well as talk to people. For example, books (or scrolls) could be placed that show the previous reconstruction drawings. Placed in an ancient setting the user could learn directly from the people living in the area and accompany them to a religious site for a sacrifical celebration. It is possible to think of virtual ancient tourism that retraces Pausanias' steps [109] around the great sights of Greece. Accidentally, one might even meet the famous traveller and talk to him directly and so learn from about the pecularities and characteristics of the surveyed object in comparison to other sites.

Chapter 5

Modelling synthetic Deep Sea Coral Reef Growth and Simulation for *Lophelia pertusa*

Coral reefs are commonly associated with tropical areas. Many important and well known reefs (such as the Great Barrier Reef off the east coast of Australia) lie in the tropical belt in the Pacific Ocean. Indeed, these complex calciferous structures are home to a great variety of life includig hundreds of different species (high *biodiversity*). Largely unknown are reefs common to all oceans, especially the Atlantic where they are found at great depths.

Deep sea corals build large-scale structures at depths of over 1,000 metres where no light can reach them (*aphotic* conditions) and temperatures of well below 6 degrees Celsius. News of these organisms originally came from fishermen off the Norwegian and Irish coasts. Ever so often, a set of stony corals was caught up in and posed a danger to the fishermen's nets. The organisms were first described in the 18th century by Linne and Gunnerus [85, 53]. But until the 19th century, it was commonly believed that the bottom of the oceans represents a lifeless zone (the *Azoic* theory by Edward Forbes). This became outdated with the explorations of Charles Thomson, who brought up the various life-forms of the deep by dredging heavy containers over the surface of the seabed. Thomson's four cruises in 1868 [20] (see Figure 5.1) established impressive evidence of life in the deep sea from great depths of over 4,000 metres. From then on, systematic research into cold water stony branching (*scleractinian*) corals began and we will focus in the following on the most common of these corals, the genus *Lophelia* and species *pertusa*.

In modern times, oceanwide visual inspection of *Lophelia pertusa* in the north Atlantic has revealed several characteristic colonial growth habits as a

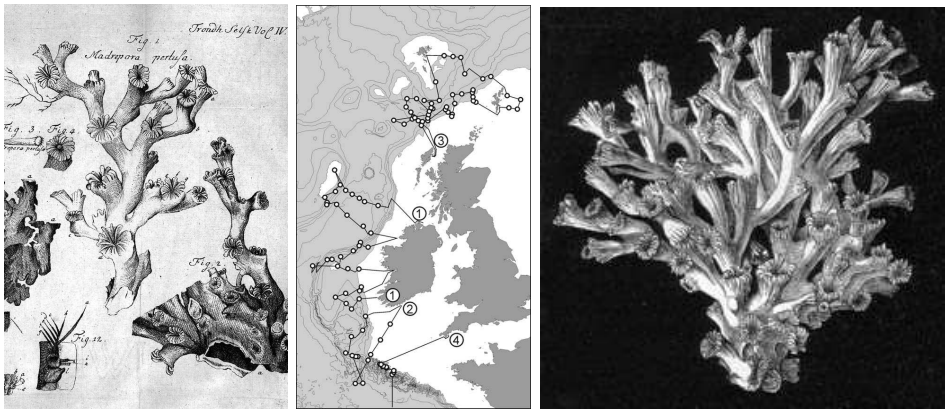

Figure 5.1: Research into stony branching corals is a relatively new phenomenon. From left: Johan Ernst Gunnerus' drawing of corals in 1768 [53], the four cruises of Thomson's HMS Porcupine and a drawing of *Lophelia pertusa* from his expedition [20].

manifestation of skeletonisation in different environmental settings. In order to understand the variety of ecophenotypes and budding patterns that are realised by *Lophelia* in nature, we analyse the basic growth and branching patterns per polyp generation. In the following, we will describe a methodology and a pilot study that we executed in 2003/2004 with the Chair of Palaeontology at Erlangen University in order to investigate these issues by means of 3D visualisation with computed-tomography.

5.1 *Lophelia pertusa* and Coral Reefs

For an introduction to the coral in question, we closely follow [46, 47, 139, 106]: *Lophelia pertusa* is a colonial coral with one or more polyps (a colony). Little is understood about its basic biology and despite the obvious budding (cloning) of polyps, nothing is known about sexual reproduction or the larvae the species produces. The polyps need a calciferous seabed to settle and continue to deposit a hard skeleton made from aragonite (a special calcite) throughout their lifetime. The many skeleton branches of the dead and living polyps (so-called *matrix*) increase the spatial heterogenity of the seabed. These skeletons form structures from small patches of a few metres width to reefs[1] many hundreds of metres in size (see Figure 5.2). The coral lives

[1]There is still debate about what cold water coral structures should be called. We stick to the same name as is used for their tropical kin, *reef*, that is defined by the European Habitats Directive (92/43/EEC) [38] as a submarine, biogenic concretion which arises from the sea floor and which supports a community of animals. Note that the term reef is used for smaller colonies as well as largest agglomerations of smaller reefs.

Figure 5.2: Complex branching structures of *Lophelia* colonies hint at a genetic building plan. Right: side scan sonar image of a coral reef. In (a) in intact coral reef is shown, in (b) a similar reef was destroyed by deep-sea trawling. Images from [45, 87], side scan sonar courtesy Dr. Andrew Wheeler, University of Cork.

at depths of 100–3000 metres and temperatures of 4–8 degrees Celsius and an individual polyp reaches 10–15 years in age, while the reef itself lives for hundreds of years. The aragonite skeleton is surrounded by living tissue called *coenosark* which interconnects the polyps for organic distribution processes within the colony. The coenosark is surrounded by tissue called *ectoderm* that protects the branches against bio-eroders and other dangers to the skeleton (inorganic solving processes). Polyps feed on the filtration of plankton from the surrounding currents and do not possess symbiotic algae as tropical reefs due to the aphotic conditions the coral lives in (no photo-synthesis). The distribution of these reefs is global but still unknown in detail. The reefs along calciferous flanks of the continental shelfs or along the plough marks of glaciers of the last ice ages are prominent. Relatively precise information exists for the north east Atlantic where the reefs are in acute danger from the deep-sea trawling methods of the fishing industry. Typically, reefs are surrounded by loose sediment.

Cold-water coral reefs develop over long periods of time, usually hundreds, sometimes thousands of years. For example, the Sula Ridge reef in Norway is thought to have been growing since the end of the last ice age (10,000 years ago). The corals grow very slowly and the presence of a reef therefore indicates a stable, low disturbance environment. North Atlantic reefs are dominated by *Lophelia* corals. The largest *Lophelia* complex at Sula is over 14 km long and grows up to 35 metres high from the sea bed. High levels of biodiversity, comparable to tropical reefs, are found among the coral matrices with 800 species being a conservative estimate. 23 kinds of fish are believed to aggravate at reefs and many raise their young in the complex branching structures for protection, which explains the interest of the fishing industry. Due to trawling, Norwegian reef shelfs are believed to have been destroyed by up to 50 per cent by now and the slow growth of about 4–25 mm a year severely limits the ability of these reefs to regrow in a short time

Figure 5.3: Known *Lophelia* distribution map and the danger from deep sea bottom trawling (images from [46]).

(hundreds of years are estimated [45, 51]). In 2008, a United Nations campaign (*International Year of the Reef* 2008) will try to raise awareness of the issues endangering the species, namely pollution (oil and gas exploration), trawling fisheries (see [51] for details) and global warming.

5.1.1 Methodology

As seen in Figure 5.4 *Lophelia pertusa* can be found in various types of skeletal configurations. There is still no standard way of describing the different patterns it can grow in. This is unfortunate and a deterministic way to differentiante "subspecies" concerning growth patterns is desired. Until now *Lophelia* has been described in the literature as *compact and bushy* or *elongated and slender*. We presented a way of dealing with the problem in a mathematical way at ISDSC 2003 [98]. Our aim is to describe the differences in branching and provide a system to ease classification. We propose addressing the problem in the following way (see Figure 5.5 for a graphical representation of the approach):

- Data Sample Collection
- CT-Scanning
- 3D-Model Extraction
- Virtual Skeleton
- Formal Grammar (L-Systems)
- Classification
- Applications: Quantitative Analysis, Reef Building Potential, Fluid Simulation

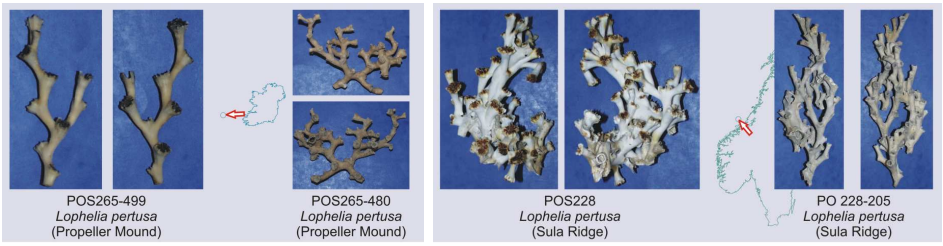

Figure 5.4: Our working samples from two prominent sites: left two different morphologies from Propeller Mound (west of Ireland). Right: two totally different branching structures from Sula Ridge (Norway). Samples brought up by Andre Freiwald's Poseidon cruises (POS 228 and 265) in 2000 with the remote operating vehicle JAGO.

In the following, we will detail the feasibility of several steps of the pipeline given in Figure 5.5 and suggest how to realise the overall procedure. We have worked in detail on a pilot study with few samples from prominent reef areas (Propeller Mound and Sula). We show the steps from CT-scanning to the virtual skeleton. Additionally, we present results of reef simulation with L-Systems and the applications for quantitative analysis and the reef building potential.

Data Collection and 3D Model

In order to research branching patterns, we need to take data samples for examination. We decided to measure the objects with computed tomography. With a CT-scan, a three dimensional scalar field of absorption coefficients is acquired. Although no information about the actual absolute density of the object can be given, the data is sufficient to evaluate key features. Compared to medical data where the separation of materials is often very difficult and of great importance, CT data from corals have high contrast since the main material found is aragonite ($CaCO_3$). This material gives good absorption results and is similar to human bone.

After CT scanning, methods developed in the field of computer graphics can be applied to visualize the data set. Volume rendering techniques are generally divided into indirect and direct methods. In our approach we focus on an iso-surface extracted from the data volume (indirect volume rendering). This can be done with the marching cubes algorithm [88]. A suitable iso-value has to be selected for surface calculation. This numerical value can be estimated by incorporating the object's mean density and its weight:

For each sample, its weight w is known. A volume can be computed from aragonite density ($2.95g/cm^3$).

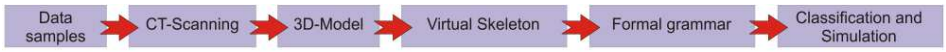

Figure 5.5: Our proposed methodology discussed in [98] for finding a way to classify and evaluate different ecophenotypes within the species *Lophelia pertusa*.

$$V = \frac{w}{\rho} = \frac{w \quad [g]}{2.95 \quad [\frac{g}{cm^3}]}$$

Furthermore, the number of volume cells N needed can be calculated with the volume of a voxel cell V_{voxel}.

$$N = \frac{V}{V_{\text{voxel}}}$$

The volume is then binarised into values of 0 (water) and 1 (coral) to match N voxel being above the binarization threshold. The iso-value is then selected at 0.5. With marching cubes the output is a triangular mesh with linear interpolated geometry along the edges of the cells of the volume that resemble the iso-value.

The advantage of an iso-surface consisting of vertices and triangles is that these primitives can be efficiently displayed on consumer graphics hardware found in every computer nowadays. A triangular mesh can be examined for angles, lengths and areas that can be obtained from its piecewise linear structure. Furthermore, triangles can be easily coloured and manipulated to visualize properties of the data while the display is being accelerated by graphics hardware.

Virtual Skeleton and Grammar Extraction

From the 3D model we propose to extract a skeleton either manually or automatically. A virtual skeleton is a linear shape description that can be used to extract an L-System grammar of the coral that captures the complex branching pattern. For a sufficiently large collection of corals such grammar descriptions can be used to better differentiate between the different budding patterns and classify newly found samples. This can be accomplished by finding representatives of the most prominent classes of branching patterns. A new sample can then be compared and classified by finding the smallest distance to the representatives.

For an implementation of this pipeline with evolutionary algorithms see [75] where the very same idea is used for the classification of blood vessel branching patterns on the retina of the human eye. There, the idea is to detect pathologic deviations between healthy and diseased systems. Also, patients

are monitored and pathologic variations are detected over a longer period. For more information see the GREDEA (*Grammatical Retina Description with Evolutionary Algorithms*) project [76]).

We discussed our proposition (see Figure 5.5) at ISDSC in 2003 with the palaeontologist and oceonologist community [98]. Great interest was expressed in our idea to incorporate a genetic building plan in the form of an L-System. This sets us apart from other computer-based approaches [67, 100, 101], which mainly focus on accretive growth by nutrient distribution in fluid flow. There, for flow visualization Lattice-Boltzmann methods are favoured that can only be used for very small domains (single corals) because of the very fine grid which is necessary for the high Reynolds numbers involved. Due to its randomness, accretive growth is useful only for very few corals and sponges. In the case of *Lophelia*, the thin, elongated skeletal tubes supporting the living polyps are found in a large variety of branching patterns and variations (branch length, calys diametres, branching angles). Here, it is of paramount importance to find a better way to classify the diversity within the species *pertusa* and we argue in favour of mathematical models such as L-System grammars that can parametrise this genetic variation.

In the following, we will give details of our pilot study using the four samples from the Poseidon cruises (Figure 5.4). In a first step, we are able to calculate and animate the process of growth with shading colour and transparency while incorporating basic assumptions about coral growth. Our model gives information on key features of the basic structure of the coral, such as branching angles, velocity of growth and its age at various stages.

5.2 Growth Visualization

In order to understand the growth patterns we decided to analyse the specimens at our disposal for their own growth behaviour. Looking at the complexity of the branching structures, it is interesting to see how the coral has evolved. Unfortunately, no certain conditions of the habitat can be established since salinity, temperature, currents and nutrient distribution vary considerably within both reefs, Propeller Mound and Sula, and were not recorded. As a consequence, assumptions are made on how a coral grows and how a budding polyp behaves [139]. The resulting growth times need to be evaluated and, therefore, visualization techniques can be applied.

5.2.1 Definition of a Virtual Skeleton

To visualize growth we need a basic understanding of the morphology. The triangular mesh that we generated from the volume dataset is too complex

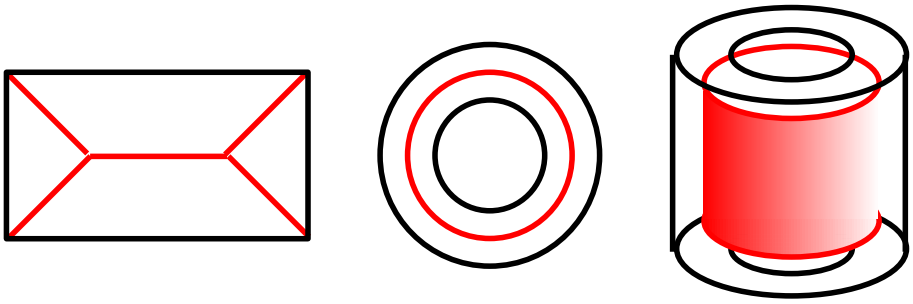

Figure 5.6: The skeleton of a rectangle and a doughnut in 2D (red lines). In 3D the extruded doughnuts skeleton are made up of medial surfaces (the inner red cylinder).

to work with directly. Since the branching structure we need to examine resembles a tree we propose using a hierarchical tree-like data structure to deal with it. In our case, a virtual tree of the coral skeleton is constructed as a branching linear virtual skeleton (Figure 5.8). In Computer Science, skeletonization exists as a term for calculating such structures. A virtual skeleton is a reduction of dimensionality while retaining relevant structural information for the overall object (Figure 5.6). Blum described how to find the skeleton by computing the medial axes of an object with a prairie fire analogy: if a fire were to be lit around the boundaries of the object which burns inwards, it would extinguish at the medial lines [8]. His proposed distance transformation computes the medial axis and therefore the skeleton of an object. For each inside element of volumetric data the nearest outside element of the space around the object is found and the distance is stored. The medial axes are found where there are equidistant distances to outside elements which can be detected by a very small distance gradient. This distance to the boundary elements relates to the radius of a *maximum inscribed circle*. The medial axis alone and circles drawn at each element facilitate a reconstruction of the overall shape of an object (virtual skeleton). Hence, a virtual skeleton is an equivalent description of the shape.

For two-dimensional objects a simple linear description (the medial lines) remains after skeletonization. Additionally, information is stored at each medial line point that corresponds to how far away the next surface is (maximum inscribed circle). Unfortunately, for solid 3D objects such as our recorded CT data volumes, a reduction in dimensionality yields not only line segments but also surfaces. This is easily understood with the following example: a doughnut shape in 2D has a circle as medial line. Extrude the doughnut to 3D and the shape now is a hollow cylinder (see Figure 5.6 right). The set of inner elements of a cylinder where there are equal distances to the boundary elements is, again, a cylinder, this time with zero thickness

(an unoriented medial surface).

Principally, three different methods are known that work on a discretization of the data. In our case, such a discretization is given because the coral data is already present in a voxel grid produced by the CT-scanning step. For our purposes we evaluated several algorithms to generate the virtual skeleton [28]:

- *detecting ridges* in distance maps of the boundary points
- calculating the *Voronoi diagram* generated by boundary points
- *thinning* of the object by erosion layer by layer

All of these methods have proven to be insufficient for a fully automatic approach. Although to the human eye there is a linear tree-like branching structure to a coral, many cavities are found inside the coral with openings to the outside making automatic filling impossible. Furthermore, the calyxes (the openings) of the corals lead to too many medial surfaces. Since there is no established easy way to convert medial surfaces into medial lines (so-called second-order skeletonization [28]), a different approach needed to be taken.

There are geometry-based approaches in the literature [128, 69] that try to find the skeleton by using only triangles as input. In [128] a distance volume for geometry shrinking is used and during the shrinking process triangles are collapsed if their edge lengths become too small. The final remaining graph again yields medial surfaces. For further automatic processing, this idea is not well-suited, especially since through Laplacian smoothing on the mesh, the virtual skeleton is guaranteed to have the topology of the outer shape of the object. In our case, this leads to holes where only the tree structure is wanted.

In [69] a hierarchical way of finding meaningful cuts through an object is used to find a skeleton by using the barycentres of the segmentation. Unfortunately, the approach suffers greatly from a strong dependence on the parametrization (edge lengths, number of triangles) of the underlying mesh. For meshes of over 100,000 triangles, the approach takes an overly long time and does not produce a segmentation usable for skeletonization. The so-called meaningful cuts on the outside rely on surface angles to detect cuts. This cannot always be related to meaningful skeleton bones segmentation on the inside.

For the above reasons we implemented a user-driven skeleton design approach. Manual placement of spheres aids the user in placing important structural points in the coral. The spheres can be hierarchically connected and their radii describe the thickness of the structure. This time-consuming

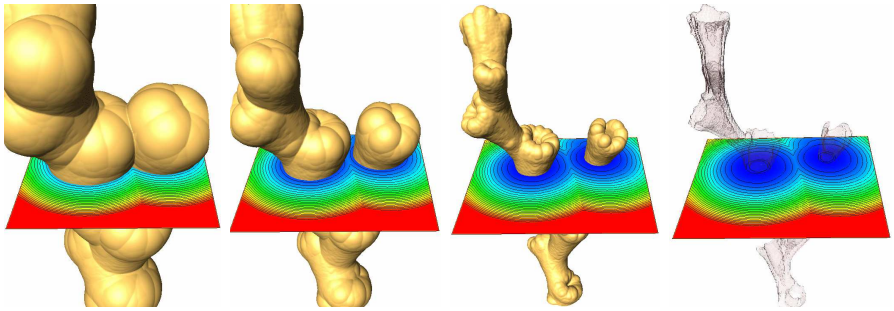

Figure 5.7: Distance transform based skeletonization: isosurfaces of the distance field shown. To the right: volume where medial voxels are found. The calyx structures are clearly visible.

process is far from optimal regarding the time it takes to create a manual skeleton branch. In Figure 5.8 two skeletons are shown that were created by manual placement. The advantage of a manual placement is that the user's special knowledge can be integrated in the designed tree. This is useful for incorporating basic assumptions about branching behaviour, giving special nodes a time key and for determining where self-replicating branching patterns are found.

5.2.2 Animating Coral Growth

Retracing growth is important to check whether former assumptions hold true when examined with real samples. We implement a model with which we can visualize two important parameters of palaeontologic interest: growth length per year and growth speed of newly branched polyps. A basic assumption in coral research is that each year the polyps branch off new polyps by cloning themselves (so-called *budding* [139]). We therefore assign each newly budded polyps skeleton an age that is incremented one year to its "father". This father then grows until its final calyx size is reached. Generally, this growth is much slower compared to the new polyp(s) that has branched off. Its initial growing speed is assumed to be higher than the remaining growth of the father in the literature. The reason is that the newly branched polyp needs to feed its own digestive system and therefore competes with its father for the food resources drifting by in the currents around it.

In our research, we can make this behaviour visible. We assign a relative age in years to each newly budded polyp in Figure 5.9 during the manual skeleton building step. The distance d in mm between two branching points is relevant for determining growth speed v:

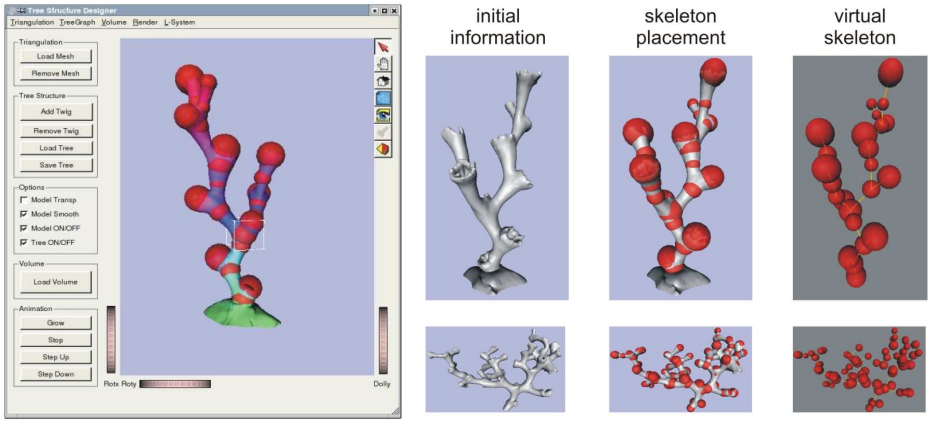

Figure 5.8: Manual helper application to construct a virtual skeleton and two examples for our specimen. The yellow lines describe the growth direction and the red balls depict the thickness of the structure. The whole coral can be reconstructed from this information and the relevant branching information formerly given as a surface before is simply described differently.

$$v = \frac{d}{t} \qquad \text{in} \qquad \frac{\text{mm}}{\text{year}}$$

The assignment of the age of the father's calyx is difficult since there are different hypotheses about their growth while these polyps live on for several years. We assume that the growth in length is slower after budding and set the rate of growth of the father to a quarter of its former growth speed.

$$v_{\text{after budding}} = 0.25 \cdot v_{\text{before budding}}$$

From the defined skeleton we can easily determine the branching speed by measuring the distance to the next budding point (when the newly branched polyp itself is budding).

To illustrate the growth process, we animate the shading of the age and visualize the progression from start to end of the sample's lifetime. We begin by assigning each vertex an individual age t_{p_i} by projection on the nearest growth axes. Each vertex p_i at each time step t is then assigned an opacity $\alpha = o(p_i, t)$, meaning that the vertex is either visible ($\alpha = 1$) when already grown or invisible ($\alpha = 0$):

$$o(p_i, t) = \begin{cases} 1.0 & t >= t_{p_i} \\ 0.0 & t < t_{p_i} \end{cases}$$

Coral	Mean Branch length	Mean Branch angles	Mean Calyx length	Mean Calyx angles	Mean Growth speed
POS265-499	12.7	44.6	5.2	22.2	10.1
POS265-480	11.6	52.5	4.5	42.6	6.6

Table 5.1: The two examined specimens from the same location show variations in mean branching angles, lengths and growth speeds. All measurements are based on the assumption that budding takes place at roughly the same time each year. Branch length is the length of a polyp's skeletal growth from birth to budding. Calyx length is the growth length of the father after budding. Growth is given in millimetres and speeds in millimetres per year.

With linear interpolation of alpha values on triangles we give the impression of a growing coral while a transparent shading of the overal structure gives hints about the future shape (Figure 5.9 right). An animated sequence can be displayed at any time by incrementing the time step.

We have shown a way to give visual feedback on assumptions about coral colony growth. A manual software approach was developed to help the specialist user to build a virtual skeleton. With this virtual skeleton, hypotheses about growth speed and development times for branching behaviour can be evaluated. Additionally, mean branch and calyx lenghts, radii and growth times can be extracted for further use. Animations can be created with ease and help visualize the time-dependent developmental process.

5.3 Virtual Reef Simulation

One main part of our work is the mathematical model for *Lophelia* corals. We assume that the branching structure is part of a genetic layout that can be formulated with general rules. L-Systems are widely used for such a purpose in computer graphics. In the following, we will give details of our pilot study and show how we suggest using regular grammars to describe shape and use these descriptions for the growth and analysis of virtual reefs. We will not explain how a grammar can be described as sufficient methods derived from evolutionary programming have been presented in the literature [76, 75], but rather focus on the computer graphics aspects of turtle graphics and the quantitative analysis of virtual reef modelling with L-Systems, which can be used as a testbed for new coral colony morphologies by palaeontologists.

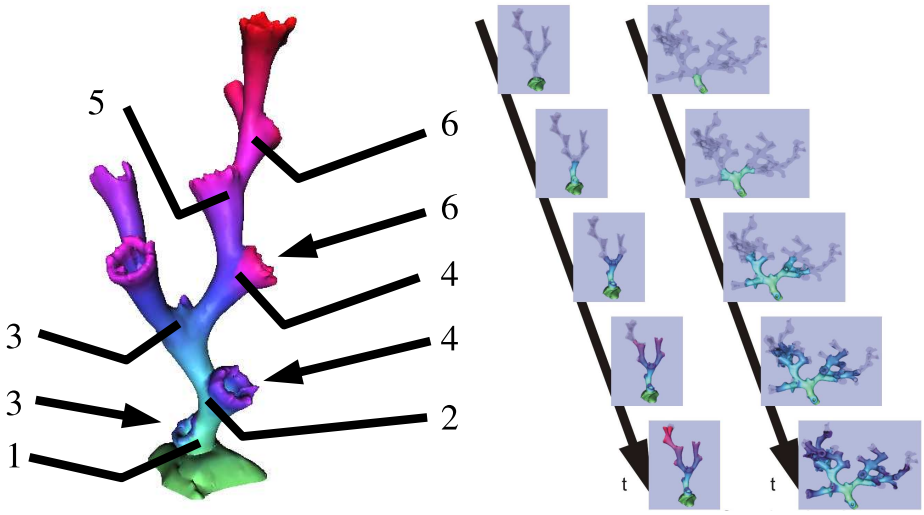

Figure 5.9: Visualizing coral age: same colours indicate same age. Left: Assigning coral age (given in years). Arrows point to the calyxes that grow slower after budding. Angled lines indicate branching-off points. In our manual skeleton approach, the age of each branching point needs to be given while the growth speed of the father's final calyx is dependent on its former growth speed. Right: Animating coral skeleton growth by linear interpolation of age along triangular faces. Shading from blue (early in time) to red (late in time) in the seventh year. The coral on the left is about 7 and the one on the right 13 years old.

5.3.1 Basics of L-Systems and Morphological Modelling

L-Systems were introduced by Lindenmayer (hence the name, Lindenmayer-Systems) in 1968 to model developmental processes in multi-cellular organisms [83]. Emphasising topological relationships between cells and larger plant modules, L-Systems are a new way of expressing organic growth as a function of time that manipulates space. The use of such L-Systems to generate not only cellular life but plant life in general has been extensively researched. Aristid Lindenmayer once pointed out how he envisaged tackling the complex processes that lead to the formation of complex organisms [84]:

> The development of an organism may [...] be considered as the execution of a 'developmental program' present in the fertilised egg. The cellularity of higher organisms and their common DNA components force us to consider developing organisms as dynamic collections of appropriately programmed finite automata. A central task of developmental biology is to discover the underlying algorithm for the course of development.

Prusinkiewicz [115] focused on computer graphics and the generation of plants with stunningly realistic results. While it has become increasingly easy and convenient to design and understand growth on the computer, one important problem remains. Generation of synthetic plants is always a deterministic even if pseudo-randomly altered process. The modelling of real plants with L-Systems, meaning the derivation of L-System rules from an example in nature, is still an unsolved problem.

Introduction to L-Systems

Working similar to grammars of formal languages as invented by Chomsky [25], an L-System is a string rewriting mechanism that produces new words from given rules and an alphabet that form the grammar. We start by comparing Chomsky's notion of a grammar and the resulting language with L-systems:

$$G = (V, \Sigma, P, S)$$

The grammar G is comprised of a starting variable, or symbol, S. The Variables V can be substituted with the left sides of the production rules specified in P. Any symbol that cannot be substituted (i. e. a terminal symbol) is defined in the alphabet of the grammar, Σ. A word of the above defined language is any string that can be derived from the starting symbol and production rules until all variables have been substituted by terminal symbols.

Depending on the nature of the production rules, grammars can be split into categories, defining their complexity as a language. We will solely use deterministic, context-free languages (*DOL-systems*) here for the purposes of plant generation. Although we will use the Chomsky-notation for L-Systems it is important to note that all context-free Chomsky languages can be produced by deterministic, context-free L-Systems grammars while on the other hand, there are languages produced by *DOL-systems* that cannot be produced by Chomsky grammars. A complete survey of the variety of formal languages is beyond the scope of this text and we refer the reader to [130].

Example of a context-free Chomsky grammar:

$$\begin{aligned} G &= (V, \Sigma, P, S) \\ V &= \{A, B\} \\ \Sigma &= \{a\} \\ P &= \{A \rightarrow AB, A \rightarrow a, B \rightarrow a\} \\ S &= A \end{aligned}$$

The only way to form words in the above grammar is to use the starting symbol $S = A$ and then sequentially apply the rules. To detail how Chomsky grammars and L-Systems differ, we will show a possible derivation of the word *aaaaa* for the above language:

$$\begin{aligned} S &\rightarrow A \\ A &\rightarrow AB \\ AB &\rightarrow ABB \\ AAB &\rightarrow ABBB \\ AAAB &\rightarrow ABBBB \\ AAAAB &\rightarrow aBBBB \\ aAAAB &\rightarrow aaBBB \\ aaAAB &\rightarrow aaaBB \\ aaaAB &\rightarrow aaaaB \\ aaaaB &\rightarrow aaaaa \end{aligned}$$

So, a formal language in its basic form rewrites a string (the left side of a production rule) into a new string (the right side of a rule). While Chomsky grammars do this sequentially, L-Systems do so in parallel and this is the main distinction between the two. To show this concept, let us evaluate the word *aaaaa* produced with the grammar given above again, this time with

an L-System. In deterministic L-Systems, for each member of the alphabet there is at most one production rule. The next example defines an L-system grammar in the DOL way:

$$\begin{aligned} G &= (V, \omega, P) \\ V &= \{a, b\} \\ \omega &= a \\ P &= \{a \rightarrow ab, b \rightarrow a\} \end{aligned}$$

Here, we call the starting symbol a generator or axiom, ω, we have an alphabet (V) and production rules (P). Confusingly, in L-System literature the symbol V that refers to the variables in Chomsky grammars here denotes the alphabet, since all production rules have the form $P \subset V \times V^*$. Note that the identity production ($a \rightarrow a$, $\forall a \in V$) is implicitly contained in the set of productions.

The main difference is now the formation of the language: any parallel application of production rules yields a new string. Although this string will contain variables, it is defined to be a word of the language defined by the L-system.

$$\begin{aligned} S &\rightarrow a \\ a &\rightarrow ab \\ ab &\rightarrow aba \\ aba &\rightarrow abaa \\ abaa &\rightarrow abaaa \\ abaaa &\rightarrow aaaaa \end{aligned}$$

See line three: here, two production rules were used: one that alters variable a and one that alters b at the same time. This is not possible with Chomsky grammars. As can be plainly seen, the parallel use of production rules yields a faster convergence or faster build up of the final word. This idea of parallelism is of paramount importance for a simulation of growth. As an example, buds on a plant will open simultaneously and not sequentially, as Chomsky's grammars suggest.

Still missing, of course, is any graphics element in the above example. Graphical representations (or visualizations) can be derived by interpreting the results of such a production sequence in a geometric way. One way to do this is by using so-called *turtle graphics*. A *turtle* is a virtual pen that traces its way through two or three dimensional space. The turtle can be

Figure 5.10: Three steps of the parallel evaluation of a simple L-System for the well-known Koch curve

placed in a position and then given commands of the sort *turn* 45° *to the left* or *move* 2 *spaces in the current direction*. The sequence of all commands given is called the *operation stack* of the turtle.

Considering the next grammar given below, we introduce basic commands for our turtle to sketch the idea of turtle graphics in two dimensions. Let $n = 5$ and $\delta = 45$:

F	move the turtle	n	paces
$+$	turn the turtle's head	δ°	to the left
$-$	turn the turtle's head	δ°	to the right

An example of the resulting graphical rendering (see Figure 5.10) of the turtle for a simple grammar G:

$$G = (\{F\}, \{F\}, \{F \to F + F - -FF\})$$

The first (left hand) curve of Figure 5.10 is generated by string-replacing the starting symbol:

$$F \to F + F - -F + F$$

The second (middle) curve replaces all Fs with the production rule, yielding:
$F + F - -F + F \to$
$F + F - -F + F + F + F - -F + F - -F + F - -F + F + F + F - -F + F$
And then again for the last (right hand) curve, shortened here.

$$F+F--F+F+F+F--F+F--F+F--F+F+F+F--F+F \to \ldots$$

The starting symbol, in the above case F, is called the *generator* for an L-System, as it can differ from the then applied rules significantly and prescribe a basic shape to the object. An example of this is the Koch snowflake that uses a triangle (three Fs, turned 60 degrees to the left) as a starting symbol. Another observation is that the curve occupies the same space with each interation. This method of constructing curves is called a *rewriting system* in L-System literature and edge and node rewriting are distinguished. In short, this means that each iteration (i. e. parallel execution of production

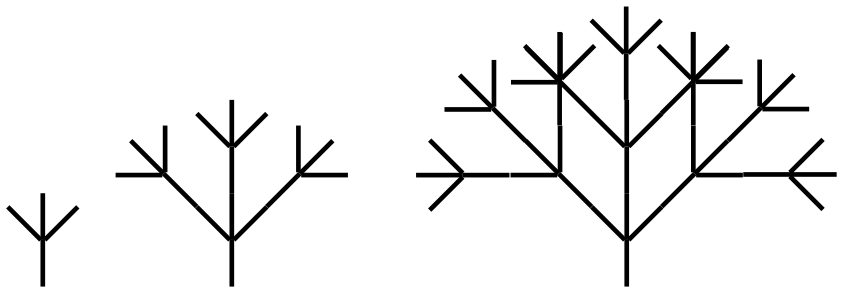

Figure 5.11: Three steps of parallel evaluation of a branching L-System

rules) leads to a substitution of the currently placed primitives (edge or node) with the newly produced symbols in its place. It is this behaviour that likens L-Systems to *fractals* and *self-similar* contructions. This way it is possible to construct space-filling curves, for example.

The graphical results are still far from similar to any plant structure. We need to introduce a typical plant behaviour to achieve a greater variety. An L-System is called a *tree OL-system* if production rules replace a predecessor by a successor *axial tree* where the starting node of the predecessor is identified with the successor's base and the ending is the successor's top. The definition of *axial tree* comes from graph theory and refers to a rooted tree that has edges that are ordered, labelled and directed. At each node of an axial tree at most one straight segment is distinguished, while there are several side elements. Axial trees can be best represented by inserting the ability to save the turtle's current stack of operations (the symbols are brackets, [and]). We can now finally produce shapes that are immediately recognisable as plants (bracketed OL-systems). This is illustrated in Figure 5.11 below and shows how this idea aids the graphical illustration of plant growth:

A very simple branching system can be generated by the grammar G, using the parameters for F and $+/-$ as above:

$$G = (\{F\}, \{F\}, \{F \to F[+F][-F]F\})$$

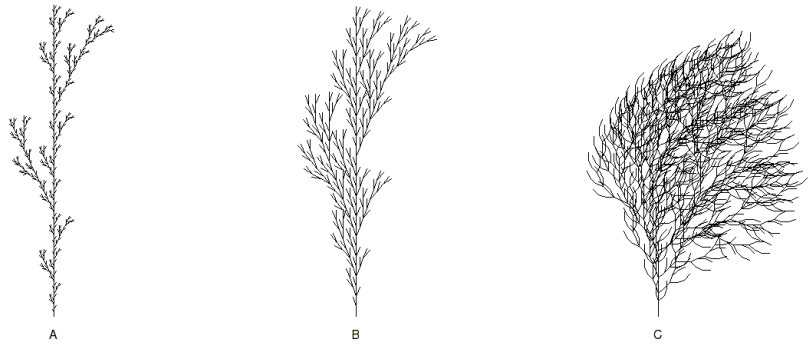

Figure 5.12: 2D plants created with edge rewriting bracketed DOL-Systems

Further examples are given in Figure 5.12, their grammars are given below (all have the same axiom, F).

$$A: \quad F \rightarrow F[+F]F[-F]F$$

$$B: \quad F \rightarrow F[+F]F[-F][F]$$

$$C: \quad F \rightarrow FF-[-F+F+F]+[+F-F-F]$$

The extension to 3D turtles is straigtforward and involves the definition of a current coordinate system. Each rotation or translation is defined with respect to this system. Stack saves here preserve the current coordinate system and position.

Morphological Modelling

Many extensions to the original L-Systems [115] have been proposed. Ranging from external forces (e. g. *tropism*), message passing from root to leaf (context-sensitive L-systems), stochastic parametrised production rules or branching patterns, a rich variety for different purposes has been developed.

Although it is relatively simple to experiment with different L-System grammar parts to achieve novel shapes, it is not very intuitive. The change introduced by a basic production rule influences large parts of the generated figure and might yield totally unexpected morphologies. It is therefore very difficult to construct an L-System that captures a known form, for example an apple tree one might see in a garden. Building L-Systems that

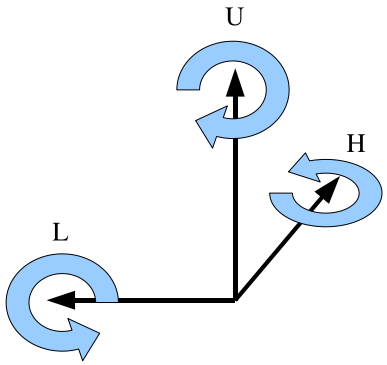

Figure 5.13: 3D turtle coordinate system and the rotation axes: H, L and U indicate orientation *head*, *left* and *up*

match given structures or sequences of structures representing developmental processes is called the *inference problem* since it is desirable to infer the grammar from a given example. Only a few papers on molecular modelling have proposed algorithms and these have little practical use for the creation of real plants from example. The complex is seen as a machine learning problem by most researchers. One seeks to train a machine to recognise a certain structure. In our case, solutions presented recently exploit genetic and evolutionary programming approaches for machine learning and succeed in finding appropriate grammars [76, 75]. We will sketch why this is not a trivial problem. Firstly, it is not clear how many production rules our parametrised DOL grammar should have and, secondly, there are too many grammars with the same number of production rules that can produce this same word that we encounter. Genetic programming is used to find an optimal configuration of production rules. Then, evolutionary strategies are used to find a meaningful parametrization of the newly found production rules. In our limited test case we did not implement this part of the proposed methodolody. Instead, we rely on expert knowledge and provide tools for a user-oriented reef modelling process.

5.3.2 L-System Grammars Design

The principal steps in developing an L-system for a particular (biological) species are given [110] below:

1. analyse the biological object
2. define informal rules
3. define L-system axiom and rules

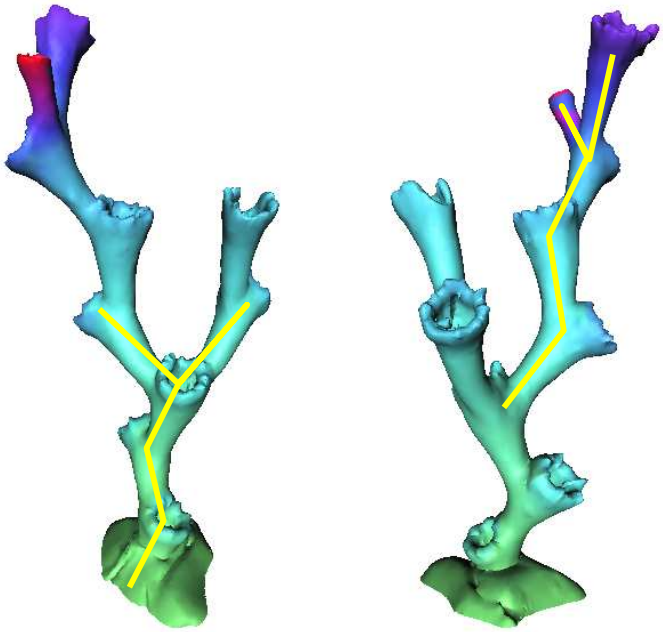

Figure 5.14: In our manual approach, hints on self-similar structures can be given.

4. do a computer simulation to generate the strings (i. e. words of the language)
5. translate the strings into graphical output
6. compare the artificial object with the real object
7. correct the L-system and repeat the above steps until conversion

Since we expect much simpler branching patterns than in other botanical examples, a manual approach is feasible for our pilot study. There is no data for our samples on factors that are assumed to be growth related (salinity, temperature, depth, nutrient abundance and velocity of flow). Therefore, we need to abstract from the exact lengths and diameters of the coral skeleton. What is more interesting is the general branching behaviour and how it influences the overall reef building potential of the coral. Since the same species is able to produce a great variety, we want to find out how we can quantify this behaviour.

For our examples, test grammars were extracted by parsing the tree shown in Figure 5.8. As a first test, each user-defined node was converted into a

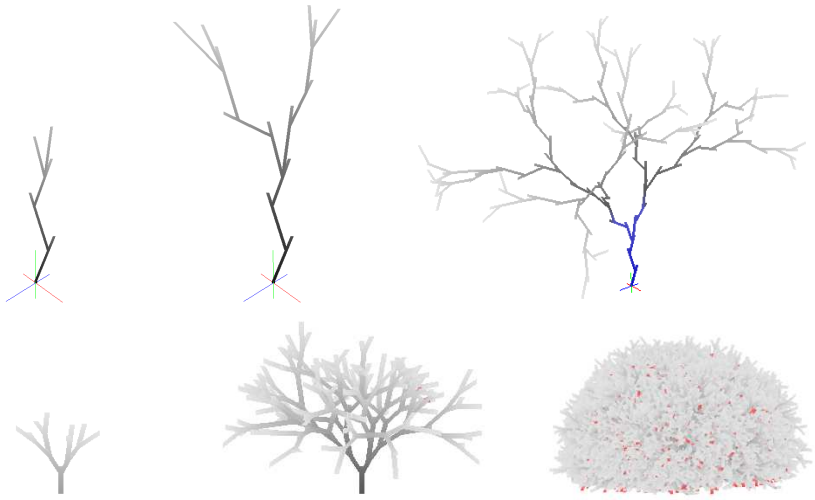

Figure 5.15: Grammar A: turtle line graphics of our stochastic grammars A and B developed in Tables 5.2 and 5.3 for 4, 8 and 17 generations (each generation resembles a year of growth) to resemble the common classification *elongated and slender* (upper row) and *compact and bushy* (lower row). Both images on the right scaled for comparison. Upper right: shaded in blue is the "dead zone" of protectionless aragonite skeleton (occluded at lower right). Collisions are shown in red.

grammar symbol. Each measured quantity is represented with the respective turtle motion that is deduced from the linear graph. See the number following the turtle instructions r,p,h for the heading, R and L for growth diameter and length in Table 5.2.

The extracted grammar in Table 5.2 is, of course, able to reproduce this one coral. Since we are interested more in the budding behaviour than in the exact values, we simplify the grammar. Our aim is, for example, to use just one production rule for the creation of the polyps calyxes after branching (symbol $S1$ in second row). Errors in turtle branching angles can be accepted. The resulting user-designed minimal grammar (second row in Table 5.2) is much more suitable and can be understood more easily.

Since our aim is to build reefs, we need a degree of pseudo-random behaviour in our coral grammars. A fine-grain control of user-supplied input for modelling known structures is achieved through the introduction of pseudo-randomly parametrised production rules. This leads to a greater variety in plants and gives the illusion of differences in growth while retaining the basic branching patterns. In our case, the production rules can be augmented by simply stating how they are affected by a random number generated with a normal distribution. For example, the length can be given

Coral	Extracted grammar
POS265-499	S0 -->h67.0602 p-77.6253 r-67.5397 R4.72009 F5.89495 [S1][S2] S1 -->h8.99106 p-2.06323 r-19.915 R4.53425 F3.03927 S2 -->h-52.3183 p-30.8915 r-1.97128 R4.30178 F12.0476 [S3][S17] S3*-->h23.127 p30.1767 r-37.1936 R5.37163 F5.15017 [S4][S5][S8] S4 -->h-2.10426 p15.593 r-13.0204 R4.85352 F5.4735 S5 -->h-41.8986 p-9.94447 r-18.2503 R3.78407 F5.91641 [S6][S7] S6 -->h3.81765 p36.5183 r4.09547 R6.06548 F10.4154 S7 -->h-6.3318 p-17.422 r-5.94594 R5.34951 F7.3294 S8 -->h-24.4708 p-36.0713 r49.4082 R4.31792 F13.545 [S9][S10] S9 -->h-25.5863 p-40.7605 r-13.021 R5.07104 F9.79203 S10-->h7.35656 p12.1696 r-52.6652 R4.99554 F7.81304 [S11][S16] S11-->h-21.1912 p-6.6356 r42.9578 R4.1038 F14.3989 [S12][S13] S12-->h21.9528 p31.7628 r-10.3086 R4.56841 F8.12339 S13-->h-1.8939 p-3.57798 r-22.02 R4.74401 F5.489 [S14][S15] S14-->h-49.289 p16.9036 r-2.39238 R2.80628 F13.7392 S15-->h30.592 p-41.3406 r11.327 R5.9384 F9.68613 S16-->h36.6942 p-13.1688 r-3.67959 R6.11436 F6.58411 S17-->h-0.295876 p-11.732 r13.2032 R5.07853 F7.22172
User-assisted simplified minimal grammar	S0-->r180 F0.5 S6 S1-->F5 R4.5 S2-->h-40 F12 R5 [S3] S1 S3-->r90 h40 F12 R5 [S5] [S4] S1 S4-->h20 r180 p20 F12 R5 [S2] S1 S5-->h-20 p20 F12 R5 [S2] S1 S6-->h20 F12 R5 [S2] S1
User-assisted stochastic grammar	S0-->r(180,180) F0.5 S6 S1-->F(5,1.0) R4.5 S2-->h-40 F(12, 2.0) R5 [S3] S1 S3-->r90 h40 F(12,2.0) R5 [S5] [S4] S1 S4-->h20 r180 p(20,-2) F(12,2.0) R5 [S2] S1 S5-->h-20 p(20,2) F(12,2.0) R5 [S2] S1 S6-->h20 F(12,2.0) R5 [S2] S1

Table 5.2: Parametrised grammar extraction from sample POS265-499: each node is converted into a symbol (S). Additionally, turtle head rotation (h,p,r for heading pitch roll), radius (R) and branch length (F) are recorded. The asterisk at $S3$ indicates a self-similarity feature. Second row: minimised grammar respecting self-similarity at branching node $S3$. Third row shows our definition of a stochastic grammar. For each turtle movement symbol a mean value and a variance are given in brackets. Both mean and variance can be taken from our examination in Table 5.1.

User-assisted stochastic grammar	S0-->R4 F(11,3) [S2] [S3] S1 S1-->R4 F(4, 1.5) S2-->r(45,15) h(22,5) p(22,5) R4 F(11,3) [S2][S3] S1 S3-->r(45,15) h(-22,5) p(-22,5) R4 F(11,3) [S2][S3] S1

Table 5.3: Parametrised grammar B shows a typical budding pattern of two new polyps every generation.

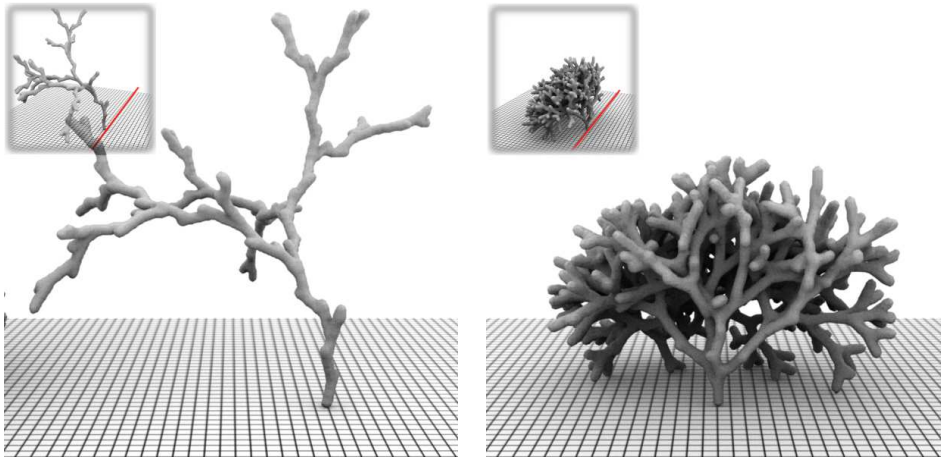

Figure 5.16: Isosurface of a volumetrization of our two main test grammars (pruned along the red line for clarity) after 20 and 11 iterations. The branching behaviour of grammar A (left) is different and leads to a totally different spatial structuring of the seabed compared to grammar B (right).

with $F(2.5, 0.5)$ and therefore, a random value is generated once this rule is applied with a gaussian centred at 2.5 and a variance of 0.5 (see lower row in Table 5.2).

The above grammar captures a budding pattern of 1–1–2 with a split in two polyps every three generations, as can be seen in Figure 5.14 and Figure 5.15. We repeat the process for another common branching behaviour. This time, every generation two polyps are branched off. The grammar is again very simple as can be seen in Table 5.3 and the resulting grammar in Fig 5.15.

5.3.3 Virtual Reef Building

After our test coral L-Systems have been defined, we can use these to grow reefs by settling one or more on a virtual underwater rock bed. Our interest lies in the reef building potential of the coral and its statistic properties. To

Reef building potential factors	
age	of the reef in years i. e. generations. This is the standard of comparison.
polyp seeding	describes the average distance between each polyp at the start of a reef.
weight	of the reef as a sum of all branches indicate how much $CaCO_3$ is bound within the reef.
complexity	of the matrix of branches for the potential for biodiversity.
living polyps	that are able to branch.
sensitivity	to bioeroders for branches where the ectoderm has receded and the structure itself is subject to attacks.
stability	of the structure by counting collisions between branches.

Table 5.4: Factors for measuring the reef building potential of a coral L-System

measure the potential to build reefs, we set out to find the density of the branches (*complexity of the matrix* and accessibility for fish), the number of branchable *living polyps* for each generation, the total number of branches (*sensitivity to bioeroders*) and the number of collisions with the reef as a measure for *stability*.

Assumptions for coral growth include the way the L-System grammar is used. We limit the application of grammar rules to at most once per year. Grammar production rules are executed simultaneously, meaning that all branches are thought to grow in parallel. Time is discretised so that each year yields a generation and only one production rule is used. The temporal resolution is therefore limited to one year. Polyps which, in reality, continue to thicken their skeleton over their life-time are assumed to produce the whole weight of their branches in one year. Collisions of branches lead to one or two polyps dying off. Their calyxes are thought to merge (as can be observed in real-life corals) and form a stable arogonite connection. Collision between corals or within the same coral are not distinguished and both are attributed to a stabilisation of the reef. The protective tissue layer (ectoderm) is said to exist around the skeletal branches for 12 years (same age as a polyp). Branches without it are shaded in blue to visualize the so-called "dead zone". The weight is estimated by assuming a homogenous distribution of aragonite throughout each branch. Each such tubular section is estimated to have $2mm$ wall thickness and to have a radius according to grammar rule specification. For a branch of 10 cm length and radius of 5

Assumptions for virtual Reef Building	
budding	new branches are budded according to L-System grammar rules. Each year a new grammar production is used and budding is thought to occur at the same point in time, i. e collisions in growth form in parallel.
time resolution	at most one production rule is used for a polyp per year. Changes in thickness of the branches (accretive skeleton deposition) are ignored as they do not alter the overall branching structure
collision	leads to the death of a polyp and produces a joint calcite bond enforcing the coral structure. Collision within a coral or different corals are not distinguished
dead zone	after a life of 12 years the ectoderm of the branches is assumed to be dead or receded making the structure vulnerable to bio-erosion or inorganic dilution processes.

Table 5.5: Assumptions for the application of L-Systems for virtual reef building.

mm this amounts to 1.7 g. In later experiments, only the weight will be given, the volume can be computed using aragonite density.

Collision Detection

We apply a bounding volume hierarchy to speed up costly collision detection. A bounding volume hierarchy is a tree composed of bounding volumes. At a given node, a bounding volume encloses all the bounding volumes of its children. In our case, no special tree needs to be constructed and we can use the internal hierarchical coral structure that itself holds all branches with relationships from generation to generation. At each barycenter of a sub-branch a bounding sphere (or axis-aligned bounding box) is fitted. For collision detection, the hierarchy is descended from top node to leaf nodes. The number of collision tests therefore are a logarithmic function of the number of all branches that need to be taken into account. With each growing step the bounding sphere or box of a tree node is enlarged for the newly produced size so that the above stated bounding volume property holds (i. e. all children are contained within the father's bounding volume).

There are two primitives especially suitable for the purpose of bounding volume generation: spheres and axis-aligned bounding boxes (see Figure 5.17).

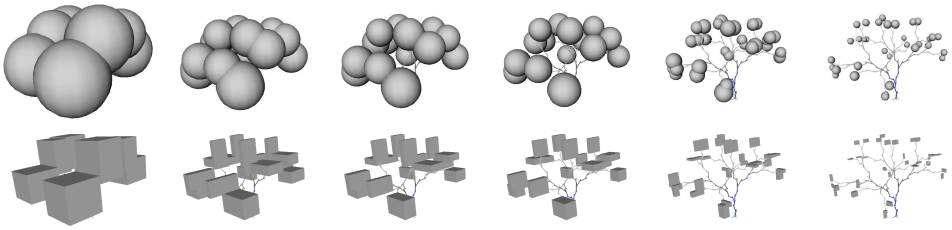

Figure 5.17: Sphere or axis-aligned bounding hierarchies for six hierarchy levels from generations 11 to 16.

Both can be enlarged very fast and intersection test with branch lines are efficiently calculated.

Experiments and Quantitative Measurements

We have designed and evaluated two main grammars (A and B) with the input of palaeontologists. The two sample designs show a main distinction their branching patterns: the first coral produces two branches every generation and the second a pattern of 1–1–2 as in Figure 5.15.

The experiment is executed in five phases:

1. **Setup** of the parameters of the experiment.
2. **Polyp Settlement** on the rock bed.
3. **L-System Rule Application** with one production execution per generation and year.
4. **Reef Evaluation:** Analysis of the reef counts living, dead and collided branches and computes the numbers given in Table 5.4
5. **Rendering** of branches with age-dependent line shading, marking of the dead zone.

Setup

To execute the experiment, parameters need to be chosen. Firstly, the grammar capturing the genetic branching behaviour is selected. The number of corals to be grown needs to be given. The pseudo-random starting points for the corals is given by a uniform distribution that is cropped to form a circle. This is apparent in side scan sonar imagery of the seabed. Coral colonies seem to form in circular patterns (see Figure 5.2 right). These agglomerations can develop into large complexes, typically one or a few metres across with tens to hundreds of living corals [45].

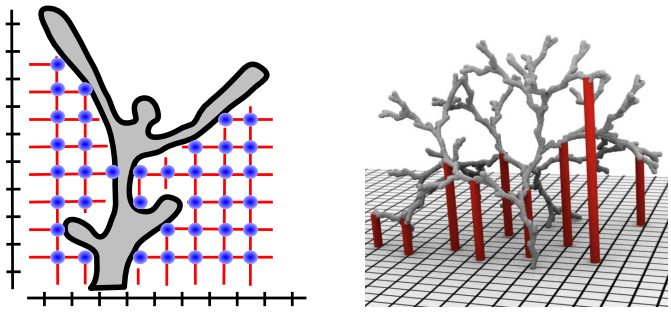

Figure 5.18: Measurement of the biodiversity factor V by counting volume elements that are shielded from top by coral branches (red lines).

Polyp Settlement

Our experimental setting is composed of a virtual seafloor that is thought to be of calciferous rock bed. Calcite is important for the polyps to settle and start building a skeleton. In our case, the whole seafloor can be settled and is completely level. Collision with the seafloor leads to the dying off of the polyp. In experiments we prohibit settlement in special areas of the seafloor (see Figure 5.16 for such pruning) for testing purposes.

L-System Rule Application

The corals produced are aware of their own structure and grammar (i. e. genetic programme). At each production step a coral branch node evaluates its own grammar symbols and produces either new polyps, an elongated branch for itself or it remains unchanged and continues to feed until its internal age counter has elapsed. Each year produces one generation and only one grammar production step is executed.

Reef Evaluation

Here, exact collision points are detected for newly produced branches. The death of collided polyps and calcification of exact collision points is registered and the dead zone of branches is propagated. The spatial density of the branch matrix is evaluated by counting the empty voxels bounded at the top by branches. This measures the complexity of the branch matrix and yields the protected inner reef volume V in litres. V is calculated from a discretization of the reef in mm^3, which allows for the upward tracing of voxel spaces that are bounded at the top by the coral. These areas provide protection from predators for smaller fish as the view from the top is blocked by coral branches (for explanation of the procedure see 5.18). V is therefore an important value for the possible bio-diversity among the protected cavities in a reef.

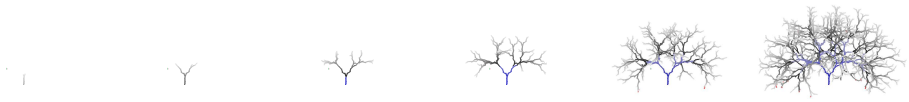

Figure 5.19: Experimental setup with one coral (grammar A) for 5, 10, 15, 20, 25 and 30 generations.

age	l	w	h	w	B_t	B_p	C	V
5	0.25	0.045	0.25	0.01	9	3	0	$0.52 * 10^{-3}$
10	0.29	0.88	0.29	0.049	39	8	0	$11 * 10^{-3}$
15	0.35	0.136	0.35	0.18	139	32	0	$67 * 10^{-3}$
20	0.4	0.18	0.4	0.58	443	96	0	0.26
25	0.46	0.23	0.46	1.77	1401	254	5	0.98
30	0.5	0.28	0.5	5.48	4323	902	18	3.11

Table 5.6: Experiment 1 for a reef composed of a single coral (grammar A)

Rendering

For clarity and speed of visualization we decide to depict the corals with a simple line drawing. With linear grey-scale shading we can colour the direction of growth by shading the starting point of a branch darker than the end point. For the assumed 12 living generations we successively draw the younger branches in lighter colours and linearly interpolate for the length of the branch. For a given generation G_i at year i the most recent 12 generations use starting grey colour c_s and end grey colour c_e depending on the age a of the branch (minimum age is 1):

$$\begin{aligned} c_s &= (1 - \frac{(a-1)}{13}) * 0.9 \\ c_e &= (1 - \frac{a}{13}) * 0.9 \end{aligned}$$

Experiment 1:

Our first experiment examines a reef made up of a single coral colony. The number of total branches is given in B_t, number of newly branched polyps in last generation B_p and the number of collisions with C. Dimensions l, w, h of the reefs are given in metres. Weight of the whole structure is given in w.

Experiment 2:

In our second experiment, we set up 50 corals of grammar A, which branches in a 1–1–2 pattern. On a domain of 1.5 m^2 we distribute 50 corals and

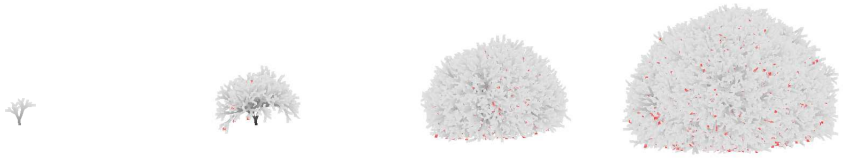

Figure 5.20: Experimental setup with one coral (grammar B) for 5, 10, 15 and 20 generations.

age	l	w	h	w	B_t	B_p	C	V
5	0.53	0.051	0.53	0.052	46	24	0	$7.2 * 10^{-3}$
10	0.58	0.11	0.59	1.6	1474	732	36	0.37
15	0.64	0.17	0.65	29.7	29,902	12,393	2659	3.9
20	0.69	0.23	0.711	183.5	211,174	54,960	20,750	12.3

Table 5.7: Experiment 1 for a reef composed of a single coral (grammar B)

simulate for 5, 10 and 15 generations.

We repeat the same setup with grammar B, which branches in two polyps every generation. Again, the placement of corals is pseudo-randomly distributed.

Results

In the following, we will discuss our findings in the tables and images given above. We proceed to detail problems accompanying our approach and refer to future work.

In our simulation in the first experiment we have achieved similar shapes, weight and dimensions for small colonies. Compared to the samples, we have created heuristic L-System grammars that capture branching features and yield imagery similar to real life (compare to Figure 5.2). For grammar A we have simulated to a greater age than for grammar B because of the smaller number of branches. As expected, the different grammars deliver different exponential growth of branches due to their branching pattern. Collision as

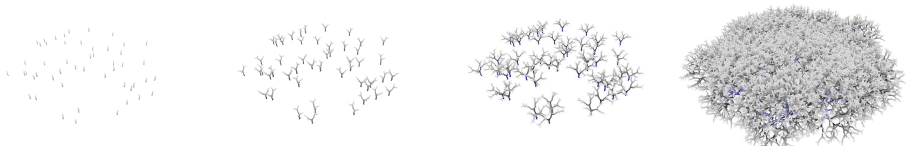

Figure 5.21: Experimental setup for grammar A for generations 5, 10, 15 and also for 30 years.

age	l	w	h	w	B_t	B_p	C	V
5	1.3	0.05	1.4	0.5	450	150	0	0.03
10	1.3	0.1	1.5	2.4	1950	400	0	0.6
15	1.4	0.15	1.5	8.8	6936	1592	9	3.3
30	1.6	0.3	1.6	238.2	190,414	37,062	1471	117.1

Table 5.8: Experiment 2 for grammar A for 50 corals distributed in a circle with radius 0.75 m.

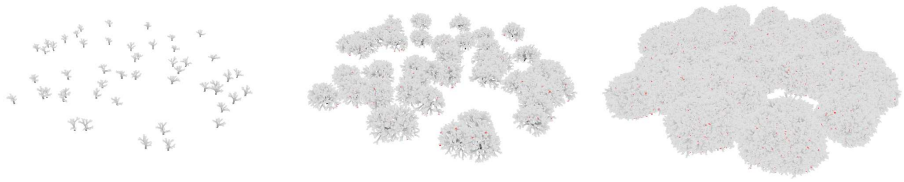

Figure 5.22: Experimental setup with grammar B shows the spatial distribution of the 50 corals (5, 10, and 15 generations). Note the low volume for B at generation 15, albeit with many more branches than with our test grammar A at generation 30.

age	l	w	h	w	B_t	B_p	C	V
5	1.4	0.07	1.4	2.6	2300	1200	7	0.04
10	1.4	0.13	1.5	78.6	7,2146	35,004	2295	1.7
15	1.5	0.190	1.5	1159.4	1,222,196	461637	133,383	12.1

Table 5.9: Experiment 2 for grammar B for 50 corals distributed in a circle with radius 0.75 m: note the expected exponential increase in branches.

a major factor limiting the exponential increase is negligible in the case of grammar A. For B collision happens on a larger scale. The complexity of the matrix, V, is comparable for generation 30 for A and generation 15 for grammar B while in the first case with very few branches and less overall weight.

In the second experiment with grammar A a greater variety is observed with fewer collisions and overall active branches compared to grammar B (see generations 30 and 15). The biodiversity measure V for type A yields greater values, since the branching matrix is less dense. This may be due to the overly dense matrix that develops with type B. Since we are not considering volume elements smaller than $1mm^3$ in our discretisation, we may miss small scale enclosures. In our opinion, these very small-scale biotopes can be safely neglected.

Summing up, when looking at the number of collisions we conclude that the reef potential of type B is much greater than that of type A. For type A high values of V hint at a greater biodiversity for larger-scale animals (fish, crabs), which is more valuable for the fishing industry. Unfortunately, these structures seem to be the most vulnerable as less inter-coral bonding through collisions occurs. The weights computed in our experiments have to be seen as an upper boundary and tend to be too high when compared with real samples. Nevertheless, it gives an insight into the magnitude of $CaCO_3$ that is chemically bound by coral reefs in a certain time-frame. The importance of these numbers is linked to the world-wide carbonate-cycle in which the atmosphere and hydrosphere exchange large quantities of carbonate in a continuous closed loop. The exact workings of this global mechanism are still subject to scientific research but one of the questions is how much carbonate is bound by marine organisms which deposit hard calcite or aragonite skeletons and shells. Here, it is hypothesised that global warming is accelerated if such marine organisms are driven to extinction by the fishing industry.

5.3.4 Discussion, Problems and Future Work

We have investigated two common concepts of how to distinguish *Lophelia pertusa* budding patterns. On the one hand, the terms *elongated and slender* (grammar A) and on the other *compact and bushy* (grammar B) are commonly used to describe the main differences encountered. In a pilot study we have described how these two concepts can heuristically be put into L-Systems for analysis and the formation of virtual structures. With collision detection and pseudo-random parametric growth rules we are able to establish convincing structures very similar to the reefs found by remote vehicle or remote sensing (side scan sonar) inspection of the oceans. With

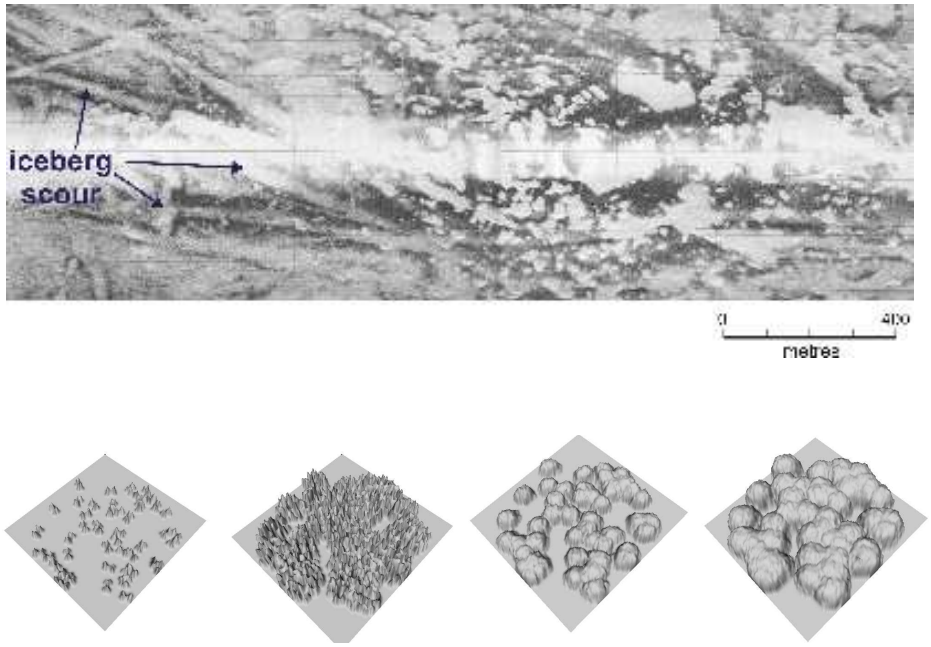

Figure 5.23: Top row left: cold water reefs settle on the caciferous rockbed uncovered by the plough marks of a receeding ice age glacier [26]. Lower rows: Virtual side scan sonar imagery by displacing a tesselated plane according to coral growth. Test grammar A to the left and grammar B to the right for 10 and 20 generations. For comparison see also Figure 5.2 to the far right.

our model, examinations of these structures are possible and we have provided measures to quantify the reef building potential of the two abstract concepts (A and B). By discretising the resulting geometry we are able to offer different visualization of our reefs. Line graphics can be directly used or an iso-surface can be shaded with ambient occlusion techniques (see Figure 5.16). Additionally, imagery similar to side-scan sonar can be computed by tesselating a plane and converting height values of the coral into a displacement map for the plane. The likeness with structures measured in real-life with remote sensing is reasonably close (see Figures 5.2 and 5.23).

The observations presented cannot yet be validated since a comparison with reef complexes of the observed size is not feasible. Examinations of coral colonies and reefs in situ are done by remote operating vehicles which capture video imagery or break off samples for analysis so as not to damage the structure. Our reef complexes measure over $1m^2$ in area and resemble realistic reef sizes for young structures. Unfortunately, samples of this size cannot be collected from the seafloor and the lack of validation may hide the fact of over-modelling on our part.

In the future, to further extend this work it is necessary to find ways to incorporate outer factors into coral growth. While it is interesting to evaluate different coral branching typologies, it remains unconnected to the reasons why the growth patterns of *Lophelia pertusa* vary so much. Here, oceanologist research is well under way, either with dedicated laboratories surveying the growth of artificially spawned corals or with video camera installations directly at the reefs. There, data about salinity, temperature, nutrients and fluid flow can also be measured and, in the future, a thorough explanation and dependencies between the mentioned factors may be found. With more information we are optimistic that our model can be augmented to produce coral reefs for special underwater setups.

Chapter 6

Application in Geosciences

Sun radiation is an important abiotic factor for all life on this planet. In biology, hypotheses about the distribution of vegetation rely strongly on abiotic factors. Data about climate, geology, relief (height) and water conditions play a predominant role when determining how well a habitat is suited for specific flora. Description and classification of vegetation is strongly dependent on the habitat. Each vegetational type shows a different socialisation with plants especially adapted to special living conditions. As an example, cold and moist as well as warm and dry sites are home to different but at the same time habitat-typical plant communities.

In a joint project with biologists and geologists (our partner is the *Institut für Vegetationskunde und Landschaftsökologie, IVL* [64]) we investigated one of the abiotic factors influencing habitats. Direct sun radiation is one of the most important aspects of plant growth and habitat development. Taking account of terrain features, surrounding landscape, latitude and atmospheric absorption we calculate an estimate of the direct sun radiation received at ground level. This analysis is used to with the task of determining the potential natural vegetation (pnv) of a habitat and software is developed to aid our industrial partner in the creation of maps of sun radiation for use in geographic information system (*GIS*) applications.

6.1 Problem Statement

To calculate maps of sun radiation data is needed for computation. In our case, such data is available as digital terrain model we will explain later. Additionally, the biologists need to collect information about habitats such as water conditions, geologic constitution of the ground and many more facts. In the following, we will sketch the task and introduce basic concepts. We begin with a definition of pnv.

6.1.1 Potential Natural Vegetation

For the description and classification of vegetation it is important to evaluate the local characteristics of the biological habitat. The German Federal Agency of Nature Conservation's pnv project sets out to estimate the potential natural vegetation of Germany. The name pnv was coined by Tüxen in 1956 [144] and a more precise definition was given by Trautmann [142]. Recent developments and research can be found in [73] and [55].

For our purposes we define the term *pnv* in the following way:

- The term pnv is a theoretic vegetational state. For each site of a surveyed area the most sophisticated vegetation is projected under current habitat conditions.
- For each surveyed area human interaction is assumed to be nonexistent. This refers to cultivation and agricultural utilisation of all kinds. Anthropogenic influences (e. g. air and ground pollution) are assumed to persist and remain constant.
- The pnv of an area is an equilibrium state of interaction with its habitat and wholly dependent on conditions such as climate and geology as well as anthropogenic influences. Man-made interaction is not reversed by pnv (examples include the fact that regulated waterways stay the same, dams do not break and river do not change course).
- There is no temporal dimension to pnv as it is neither the climax nor last stage of a natural succession process. It is derived solely from habitat conditions at present. If these conditions are changed then the pnv also changes instantly. In this way, pnv constructions vary depending on the time they were carried out. So, pnv maps made in the 1960s differ considerably from more recent ones and each constitutes assumptions and knowledge at its creation time.
- The way the future pnv is established is not through a mathematical model. The gathered data is used to compare the surveyed site for which the pnv is to be projected with similar undisturbed sites in Northern Europe. Alongside other considerations from the previously mentioned factors the vegetation present in that site is used to predict the pnv of the concerned region.

For a very short definition we refer to the Federal Agency of Nature Conservation of Germany (*Bundesamt für Naturschutz, BfN*):

> Potentielle natürliche Vegetation:
> Vegetation, die sich unter den gegenwärtigen Umweltbedingungen ohne Eingriffe des Menschen von selbst einstellen würde.

Name	Equatorial radius	Polar radius	Oblateness
Bessel 1841	6,377,397.155 m	6,356,078.985 m	1 : 299,153
Hayford 1924	6,378,388.0 m	6,356,911.946 m	1 : 297.0
WGS84 1984	6,378,137.0 m	6,356,752.315 m	1 : 298.257

Table 6.1: Reference ellipsoids through the ages

This short definition translates into English as: potential natural vegetation is the vegetation that would naturally emerge under present environmental conditions if human interaction with the habitat ceased.

6.1.2 Basic Concepts of Geodesy, Cartography, Astronomy and Physics

The aim of our research is the computation of maps of sun radiation to augment the generation of maps of pnv. To obtain information about the sun and its properties, several scientific disciplines are involved. In the following, we rely on results from these disciplines. We will sketch the necessary topics to make some of the concepts involved clearer. Beginning with the problems of the shape of the earth (geodesy) we will continue by showing mapping procedures (cartography) and finish with observations on astronomy and physics for background of the natural sciences involved.

Geodesy

Geodesy is a scientific discipline concerned with the measurement and representation of the Earth and its gravitational field. One of the main issues in Geodesy is the search for a suitable reference body usable as a coordinate frame for the oblate ellipsoid shape of our planet. In the past, several so-called *reference ellipsoids* have been proposed (see Table 6.1). The first such ellipsoid was set up by Sir Isaac Newton in 1665, followed by differently sized ones created by Laplace, Gauß and Bessel and many more.

These so-called *geoids* describe a hypothetical Earth surface that ideally extrapolates the mean sea level with gravity assumed to work perpendicular to it. Exact knowledge of the gravity feld (see Figure 6.1) is needed to compute exact digital terrain models. Derived from the reference, each point on the Earth's surface can be uniquely described by spherical coordinates (latitude, longitude and height value) as depicted in Figure 6.1 to the right.

Cartography

In cartography, the curved ellipsoid 3D shape of the Earth needs to be mapped to two dimensions. Over the centuries, many projections have been proposed and all are based on some form of projection of the ellipsoid to

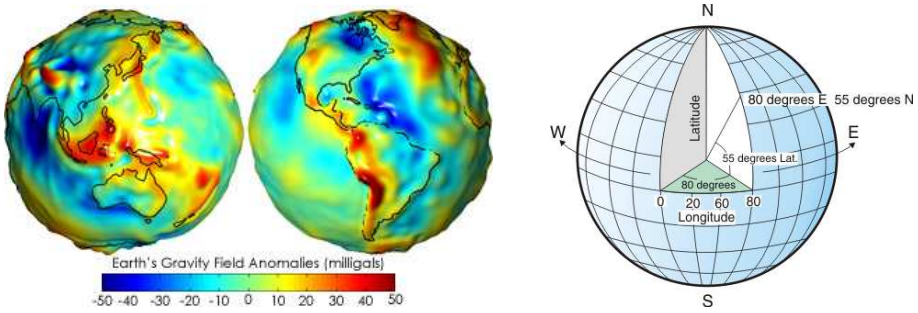

Figure 6.1: The Earth's continually changing gravity field shown as a deviation from the regular WGS84 geoid shape (image from NASA Gravity Recovery and Climate Change). To the right: geocentric spherical coordinates are used to describe a point at a surface. East, West and North and South are used to delimit longitude to a range of $(0 \dots 180)$ and latitude to $(0 \dots 90)$.

a body where a 2D parametrization is known: cylinders, cones or planes (azimutal projections). Furthermore, the centre of projection is varied from orthographic, stereographic or gnomic (for Earth-centred) and a distinction is made between mapping bodies that tangent or intersect the reference ellipsoid.

In 1569, Gerhard De Kremer developed an isogonic (angle-preserving) map in his work on a world map named *"Nova et aucta orbis terrae descriptio ad usum navigantium"*. Famous for its usefulness for navigation, it is now known as the *Mercator*-projection. It is a tangent cylindrical projection where the axis of the cylinder coincides with the rotational axis of the Earth (as opposed to the traversal cylindrical mercator projection) as can be seen in Figure 6.2 left. Angles are preserved but areas are not (polar regions are mapped to infinity, Greenland is the same size as Africa).

For more accurate mapping of small-scale regions and a mixture of preserved angle and area, the Gauß-Krüger (GK) projection was introduced in 1923 and uses the Bessel ellipsoid for reference. It soon became the standard German reference for carthography. In the GK system, 120 traversal cylinders are used according to the Mercator method, one for each three degrees of longitude. At each third meridian, a tangential cylinder is responsible for the mapping of 1.5 degrees longitude to the west and east of it. Each point on the Earth's surface can be described by coordinates on the respective traversal cylinder.

Astronomy

Ever since the dawn of mankind celestial motions have been of great interest and importance. Ancient civilizations are known to have searched for the meaning of the many configurations of stars and moon, the notion of time,

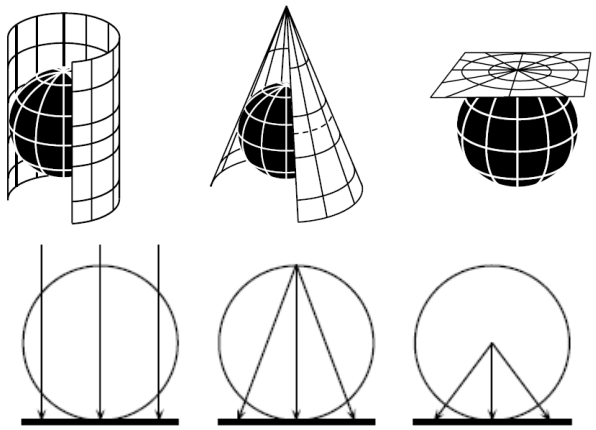

Figure 6.2: Mapping needs a reference 2D parametrizable space. The ellipsoid is projected onto such a body and different projection methods (see lower row) can be used (orthographic, stereographic and gnomic).

the cyclic occurences of seasons. Generally, the relevance of a reliable calender for agriculture and, therefore, survival gave astronomy the number one place in sciences in nearly all great ancient civilizations. We cannot give a thorough review here but refer the reader to [102, 95].

From the complicated motions of celestial bodies we need to derive simple formulas that can be incorporated into our software. It is not in our scope to provide similar accuracy or completeness as is found in other astronomical products (e. g. the open source *Celestia*).

To compute the vector of incidence of the parallel sunlight $\vec{l}(p,t)$ at a given point in time at any surface point in 3D space we need the following variables:

- the observers position (longitute, latitude and height or, alternatively, true 3D "space" coordinates), p
- the current time as in the human calender, t, determining the position of the sun in angles to the viewers coordinate system or, in our case, as absolute 3D space coordinates.

Since the Earth's rotational axis is pitched (producing winters of low and summers of high radiation) the angle of declination is derived from the time t. A pitch of -23.5 is assumed at the winter and 23.5 at the summer solstice (20th/21st of June and 21st/22nd of December). Declinations of 0 degrees are found in spring and autumn (20th/21st of March and 22nd/23rd of September). We use the library from [95] to compute $\vec{l}$. The angle of

incidence is then given by a scalar product of $\vec{l}$ and the relief surface normal $\vec{n}$.

$$\cos\phi = \vec{l} \circ \vec{n}$$

Abstractions of our model include the fact that variations in the pitch of rotational axis as well as the exact space-time definitions are ignored. Furthermore, the distance to the sun is assumed to be constant.

Physics

The sun is a very large hydrogen-helium fusion reactor, producing about 385×10^{24} megawatts of power. A small part of this energy is emitted as radiation (radiant flux) and again a small part of it directly hits the Earth after travelling for 8.3 minutes in space. Above the atmosphere, a density of flux (irradiance) of 1.3 kilowatts (the so-called *solar constant* E_0) per square metre can be measured. Although there are variations to this amount we assume it to be constant. About one third is reflected back into space. On its way through the atmosphere the power diminishes. About 25 per cent is absorbed by chemical and thermal processes in the different atmospheric layers. Only 25 per cent of the above amount hits the surface directly, while the rest is distributed diffusely after colliding with air particles.

$$E_0 = 1.367\frac{kW}{m^2}$$

The sum of direct and diffuse radiant power is called global radiation. It is composed of direct and indirect radiation, which depends on weather and climate-related factors such as clouds, air pressure and temperature. For a reliable forecast we focus on direct radiation, which solely depends on the angle of incidence, a simple model for atmospheric absorption, shadowing and the relief of the surface.

When passing through the atmosphere, three different effects are considered for radiation: reflection (ρ), absorption (α) and transmission (τ).

$$\rho + \alpha + \tau = 1$$

After substraction of the reflected part, we are interested in the transmitted amount. To attenuate the incoming energy given by the solar constant we choose to employ a simple absorption model. According to the laws of absorption, radiant flux is absorbed exponentially with $\Phi(x) = e^{-\gamma \cdot x}$ for a depth x and medium-dependent value γ. The traversed length of radiation through the atmosphere of thickness d is again dependent on the angle of incident radiation ϕ as seen from a surface point p.

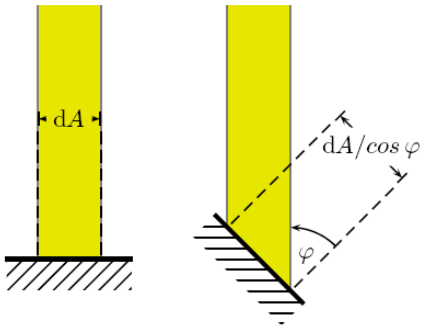

Figure 6.3: The terrain relief influences the amount of radiation that is received at each surface point.

The influence of the terrain is modelled via the surface normal, $\vec{n}$ and the projected area (see Figure 6.3). This is expressed with Lambert's reflection law that expresses that the perceived intensity (I) of radiation is dependent not on viewing but on the cosine of the angle between radiation source and surface normal.

$$I = I_{\text{radiation source}} \cdot \cos\phi$$

For simplification, we use his definition originally devised for illumination strength to make the flux density $E_{\text{incident}}(\vec{l}, \vec{n})$ dependent on the surface angle.

6.2 Maps of Direct Sun Radiation

In 2005 the Computer Graphics Group and IVL set up an interdisciplinary project [121] to combine pnv project data and computer graphics approaches to implement a tool to compute maps of direct sun radiation. Different periods of time can be observed (e.g. vegetational periods) for the mapping of the *sun hours*. In biology, a sun hour is counted when the amount of received energy exceeds $200 W m^{-2}$ in an hour. Therefore, we developed a software tool in which regions of arbitrary size are selectable as domains for the computations and an import/export into standard GIS software is provided.

In the following, we will detail the steps for radiation map computation. We continue with an analysis of the data, the shadowing algorithm and our approximation of atmospheric absorption.

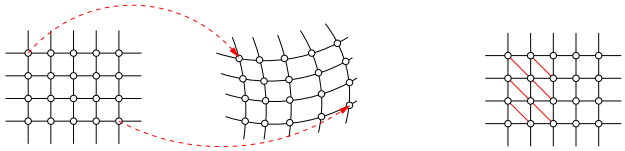

Figure 6.4: Change of geometry, constant topology. Right: regular triangulation scheme

6.2.1 Digital Terrain Model Data

Terrain relief data is given in the form of a digital terrain model (*dgm*) for Bavaria. Bavaria is the south-eastern part of Germany and covers nearly $71,000\ \mathrm{km}^2$ (latitude $9-14$, longitude $46-51$ degrees). With its lowest height at 107 m (Kahl am Main) and the highest peaks in the Alps (Zugspitze, $2,962$ m) is home to interesting terrain features.

We work with height data given as a 2D array representing Gauss-Krüger coordinates (also called German Grid). The example dataset of Bavaria has a resolution of $7,200 \times 8,000$ grid points (grid size ≈ 50 m). One of our first tasks is to remap the given grid to true 3D space coordinates (for details and the employed algorithm see [59]). To facilitate a fast lookup of neighbours we keep the vertices in memory along the original organisation of the grid (topology). This is feasible as the only slightly curved nature of the geometry still supports its interpretation as a grid. The 3D coordinates of our model of the earth are intuitively situated in space, where they are illuminated by the sun's (parallel) rays. At each vertex v normals for our terrain can be estimated using the given topology (Figure 6.4, right). From the reference geometry, an ellipsoidal normal $\vec{s}_v$ is given as a normalised vector normal to the tangent plane of the Bessel ellipsoid at a surface point (see Figure 6.5).

6.2.2 Sun Irradiance and Shadowing

Our model used to determine the amount of radiance received by terrain is Lambert's cosine law. The normalised light vector $\vec{l}$ and normalised surface normal $\vec{n}_v$ of vertex v yield $\vec{l} \circ \vec{n}_v = \cos\phi$ with ϕ being the angle between the two vectors. To decide whether the two vectors enclose an angle greater than $|90|$ degrees (light hits the back face of the terrain) we test for the sign of the scalar product. This lambertian term provides us with an upper bound for the maximum irradiance, E_r, that is available when using the solar constant E_0 as maximum value:

$$E_{\text{incident}}(\vec{l}, \vec{n_v}) = E_0 \cdot \max(\vec{l} \circ \vec{n}_v, 0)$$

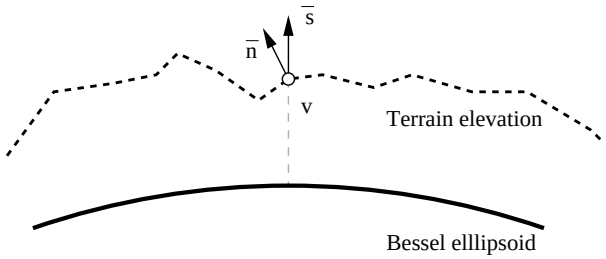

Figure 6.5: Data at a vertex v: averaged vertex fan normal $\vec{n}_v$ and ellipsoidal normal $\vec{s}_v$ (normalised position vector of v)

For now, we can compute the flux density in W/m^{-2} at instants of time. To compute the energy density $H(t_0, t_1)$ over time we integrate over larger periods from t_0 to t_1 by taking discrete measurement steps and adding them up with the trapezoidal intergration rule:

$$H(t_0, t_1) = \int_{t_0}^{t_1} E_{\text{incident}}(\vec{l}(v,t), \vec{n}_v)$$

Energy densities that exceed $200 W m^{-2}$ per hour are counted as sun hours. The temporal sampling is arbitrary and remains as a parameter for the user. Since the transition of the terrain and sun vector is very gradual we had few problems with as little as one measurement per hour.

Shadows

More sampling steps are necessary when shadows are calculated since slight changes of angles can lead to stark contrasts (several evaluations per hour are needed). To increase the precision of our direct radiation prediction we include global illumination effects in the form of shadows. Computing shadows is traditionally done by checking whether the sun can be seen from a surface point (shadow ray test [43]). Since our grid is very dense, we can discretise this test by walking along the grid.

We can make the shadow computation efficient by traversing the grid in a suitable way. To avoid testing much of the same geometry again we re–use shadow information. Introducing a quick *early reject* and *early accept* for shadow, we test the shadow information of the few local neighbours (mostly two are sufficient) in light direction $\vec{l}$. These can be quickly found based on the assumption that our topology information can be used for our transformed geometry.

Starting the computation from the boundary vertices and working inwards by using a grid-walk best suited for the current light direction (we distinguish eight directions, see Figure 6.6) all points are already processed that can

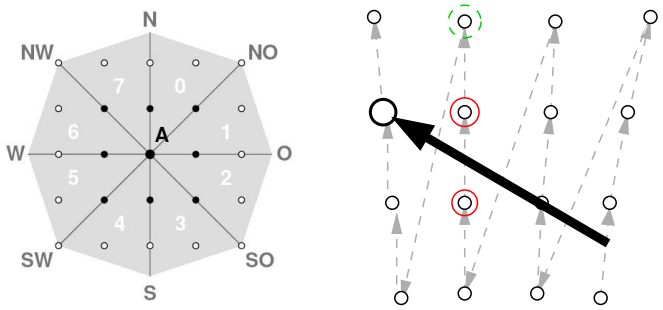

Figure 6.6: Left: the eight grid walk directions depend on the direction of the sun. Right: Grid with processing order and light vector showing the two neighbours used for the early accept/reject tests. The shown provessing order guarantees that the needed two neighbours for the early shadow tests have already been tested for shadow themselves.

contribute to the shadowing of the current vertex. The shadow lookup is done by examining the distance of the nearest vertices v_i in light direction $\vec{l}$. This is done by projecting them on a plane defined by the position of the current vertex v and the light vector with the normal $\vec{s}_v$ (see Figure 6.7) derived from the Bessel reference ellipsoid. Distances smaller than $d = 0$ are below the plane and do not cast shadow, distances greater than zero will shadow the current vertex.

$$d = \vec{n}_l \circ (\vec{v}_i - \vec{v}) = \begin{cases} \leq 0 & : \text{ does not cast shadow} \\ > 0 & : \text{ casts shadow} \end{cases}$$

Early accept is reached when the neighbours cast shadow on the vertex v. Early reject holds when two neighbours are not in shadow and do not cast shadow on v. For all other cases we need to test against further neighbours. Table 6.2 gives an overview of the performance of our shadow tests for different typical scenarios. The mean visited neighourhoods (mvn) is the number of such tests in light direction. In the Alps scenario the medium shadow length of 11.91 tests (mvn) per shadow ray is much higher than in lowlands. This corresponds with the greater height of the peaks of terrain and the resulting longer shadows.

6.2.3 Atmospheric Absorption

Lambert-Beer's law is used to get an estimate of the amount of energy absorbed on its way through the atmosphere ($\Phi(h) = e^{-\lambda h}$). For an incoming solar flux density of $E_0 = 1367 W m^{-2}$ and known received energy measurements ($E' = 1000 W m^{-2}$ at air mass $\psi = 48.2°$) we can derive a formula for

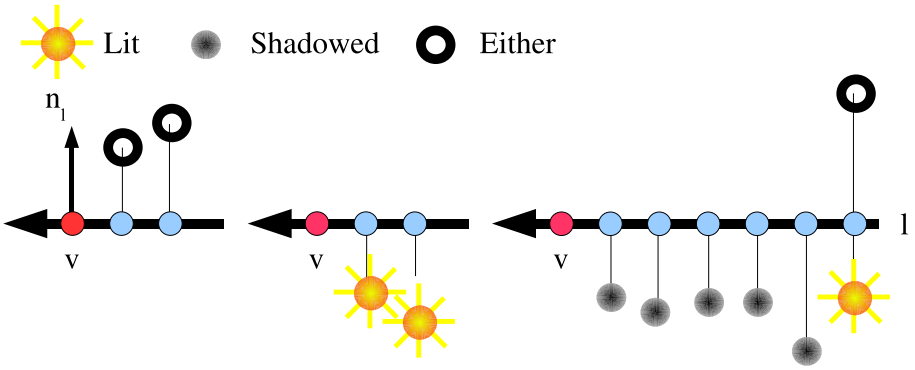

Figure 6.7: Shadow tests: left and middle easy cases (early accept and reject). To the right complex case: the traversed neighbours in light direction lie below the plane and a final decision is reached when an unshadowed vertex below or a vertex above the plane is found. Distances d are shown as thin black lines projecting to the plane.

Area	Grid points	Time	accept	reject	mvn
Alps	995 x 711	15.07. at 12 am	3,005	639,999	1.18
Alps	995 x 711	15.01. at 9 am	489,858	153,081	11.91
Lowland	995 x 711	15.07. at 12 am	0	702,336	1.002
Lowland	995 x 711	15.01. at 9 am	8,502	693,793	1.02

Table 6.2: The effect of early accept/reject tests: mean visited neighbourhoods

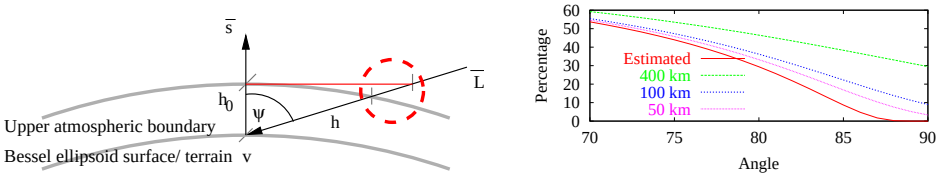

Figure 6.8: Left: Approximation of traversed atmospheric length. Right: Lambert-Beer factor for differing atmospherical thicknesses h_0 (closeup for angles greater than 70 degrees)

an atmospheric correction factor and the corrected energy $E_{\text{AtmosCorr}}(\vec{L}, v)$. We simplify the traversed atmospheric length h_0 to h for computational speed (see Figure 6.8).

$$\begin{aligned} h &= \frac{h_0}{\cos(\psi)} \\ E &= E_0 \cdot e^{-\lambda h} = E_0 \cdot e^{-\frac{\lambda h_0}{\cos\psi}} \\ E_{\text{AtmosCorr}}(\vec{l}(v,t), \vec{n}_v) &= E_{\text{incident}}(\vec{l}(v,t), \vec{n_v}) \cdot e^{-\frac{\lambda h_0}{\vec{l}(v,t) \circ \vec{s}_v}} \end{aligned}$$

The errors introduced by this approximation are negligible due to the thinness h_0 of the atmosphere compared to its curvature for small values of ψ. We retain stability in spite of the error because of the exponential function (see Figure 6.8 right). Note that the traversed atmospheric length h does not have to be computed explicitly. For accuracy, we provide an exact calculation of the traversed atmospheric length in our tool as well. Traversed length is then given by

$$h = r\cos(\psi) + \sqrt{r^2(\cos(\psi))^2 + 2rh_0 + {h_0}^2}$$

and is dependent on the atmospherical thickness h_0. We implement this to fine-tune the model.

Shadow Test Cases: Lowlands and Alps Region

To determine the sum of radiation we have to integrate the measurements distributed over the period of time that we observe. Our model can be evaluated at intervals specified by the user and the results are linearly interpolated for the given interval to form the integration. A typical region of interest for computation is about $10,000\,\text{km}^2$ in size and consists of roughly four million grid points (about the dimensions of the map TÜK 200, CC 7126 Nürnberg) or regions as small as $50\,\text{km}^2$. We will give computation results for both lowland and mountainous regions. The samples taken indicate

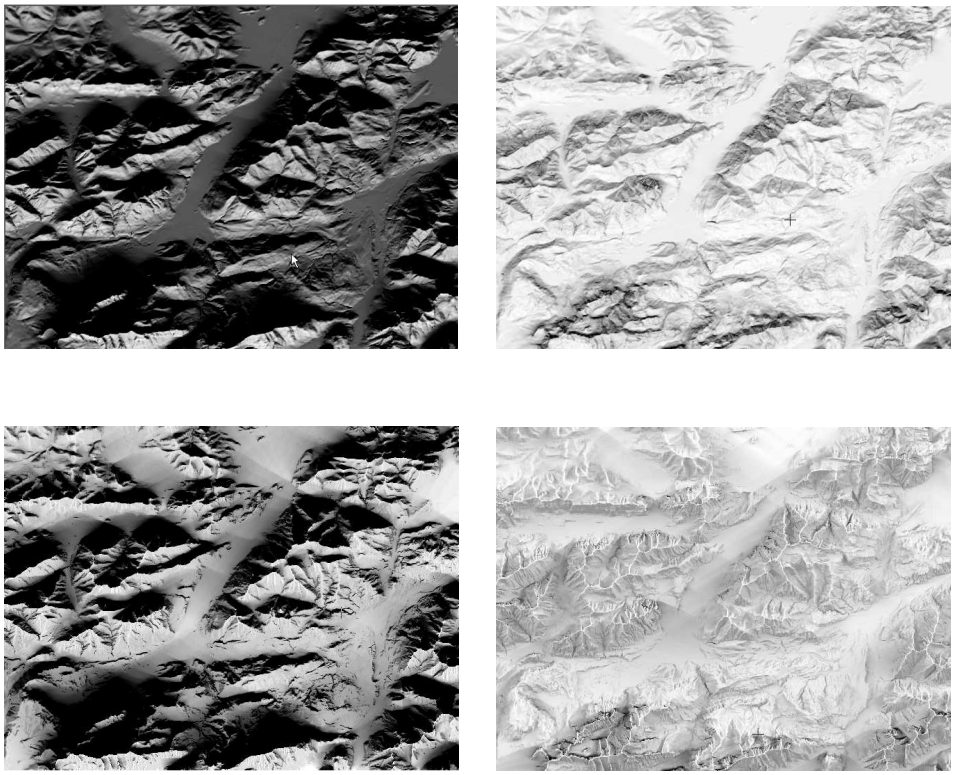

Figure 6.9: Alps region used in Table 6.2, evaluated for one day each 10 minutes. Top row shows sun radiation intensity in gray-scale: to the left January and to the right July. Lower row shows shadowing maps (white corresponds to no shadow, black to all shadow) for the above mentioned one-day evaluation. The lighter the grey colour the more sun is seen on average. These maps can be easily extracted from our approach and can be seen as a natural form of ambient occlusion maps for terrain.

Sample time	Samples	Shadow time	Total time	Time/sample
15.07. 12 am	1	3.99 sec	5.25 sec	5.25 sec
15.01. 9 am	1	4.41 sec	5.61 sec	5.61 sec
March to Oct.	3041	13,332.15 sec	17,283.7 sec	5.68 sec

Table 6.3: Nürnberg TÜK 200 region, $10,071.2\text{km}^2$, $2,222 \times 1,813$ grid points

Sample time	Samples	Shadow time	Total time	Time/sample
15.07. 12 am	1	0.49 sec	0.87 sec	0.87 sec
15.01. 9 am	1	2.21 sec	2.49 sec	2.49 sec
March to Oct.	3019	2,853.44 sec	4,500.34 sec	1.49 sec

Table 6.4: Alps region (*Koralpe*) $1,103\text{km}^2$, 888×497 grid points

how often the model with its size in grid points was evaluated (hardware: P4, 3.6 GHz, 2GB RAM). Generally, a greater amount of work for shadow tracing results in longer rendering times per grid sample point. This is especially visible in the morning and evening hours of winter (see Tables 6.3 and 6.4).

6.3 Results

We have shown a way to efficiently compute high-resolution sun radiation maps. Large grids that are semi-regular can be exploited for their grid structure. In our case, a very exact and fast raytracing algorithm was adapted.

In the following, we will describe our results and the benefit for the pnv calculation.

6.3.1 Implementation

For our interdisciplinary partner we have implemented our methods in a software application called *QSolar* with platform-independent libraries OpenGL and QT [117]. We have worked according to the following considerations for our software product:

- Portability for platforms Linux/Unix, Windows and Mac OS X
- Object-oriented design for easy exchange of algorithms, data structures and core modules

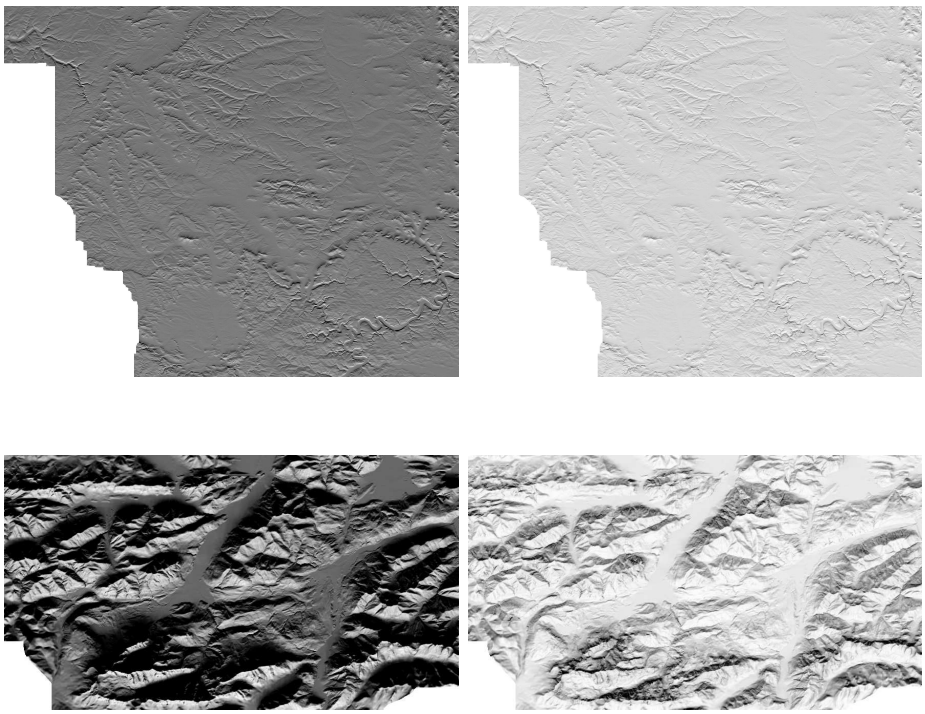

Figure 6.10: Lowlands of the Nürnberg TÜK 200 region and the *Koralpe* as in Tables 6.3 and 6.4. Grey-valued comparison for the energy.

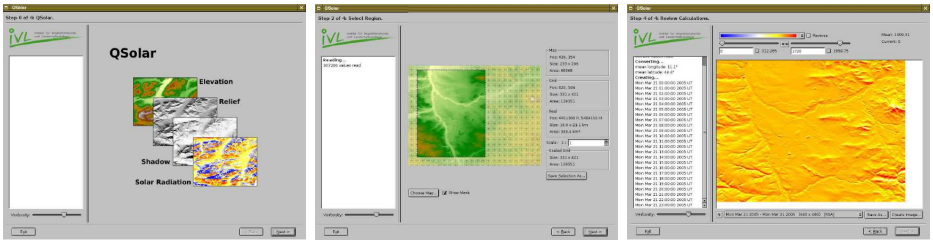

Figure 6.11: Screenshots from QSolar. Left: welcome screen, in the middle step 2 and to the right step 4 is shown.

- User friendliness: a comfortable graphical user interface (GUI) was designed for our partner aimed at an easy workflow
- Programmer friendliness: a GUI library was used (QT) that supports graphical design of user interfaces (*QT Designer*) and platform-independent compiling (*qmake*)
- Online help system and code documentation

QSolar guides the user by proceeding in four simple steps:

1. Load digital terrain model
2. Select region of interest
3. Setup parameters for experiment
4. Display the results, choose colour map
5. Check and save results or go back to 1), 2) or 3)

6.3.2 Example Images

In the following (Figures 6.12 to 6.17) we will give results for typical analyses done with our software, *QSolar*. Three areas are selected and our simulated, absolute energy density maps are presented.

1. Nürnberg area with the Regnitz and Pegnitz rivers and the Moritzberg mountain at 603 m.
2. Region around Garmisch-Partenkirchen with Germany's highest mountain, the Zugspitze
3. The whole of Bavaria.

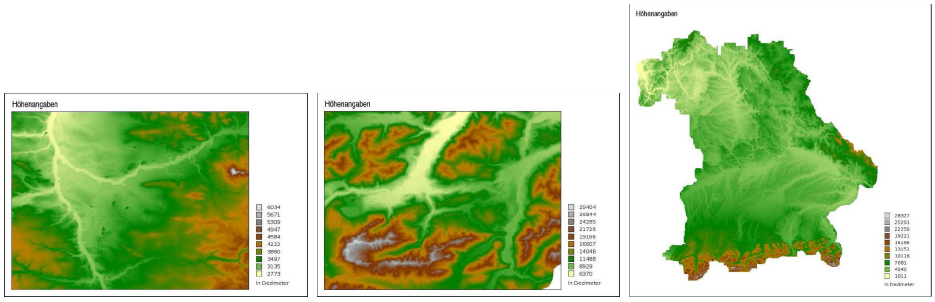

Figure 6.12: Height reliefs of our three example scenarios. From left: Nürnberg, Alps and the whole of Bavaria.

For each of these examples we will show direct radiation maps for the days 21st March (8:00 and 12:00 o'clock) with one single evaluation. Then, we will present mean values for the months of March, June, September and December with a resolution of an evaluation every 10 minutes. In the following pages, we show a map of a vegetational period ranging from the beginning of March to mid-November (10 minute evaluations every 10 days).

Our partner IVL uses the radiation maps for the evaluation of the potential natural vegetation of an area. In Figure 6.18 the different factors influencing the pnv decision are shown. Additionally, courtesy of IVL, we show one of the results of their work to be published in [19] in Figure 6.19.

6.3.3 Discussion and Future Work

We have presented a tool to compute maps of direct sun radiation in an interdisciplinary project between the Computer Graphics Group and IVL. We find a convincing, state-of-the-art model for representing the interdependencies of terrain and radiation with high face validity and fast exact shadow computation. In cooperation with the biologists and geologists we were able to contribute to the pnv project of the Bundesamt für Naturschutz and our mesurements are the basis for pnv decisions in the final report [19]. Furthermore, our work was presented at ASIM 2005 [122] to researchers and discussed with the simulation community.

In the future, the software can be extended to include different models for terrain other than pure diffuse reflexion as it is used here. A more detailed and exact modelling of indirect radiation is a milestone for the future. Atmospheric effects including frequency dependencies and clouding with scattering of radiation are options for which computer graphics methods are available that can increase the validity of the model and the accuracy for pnv analysis.

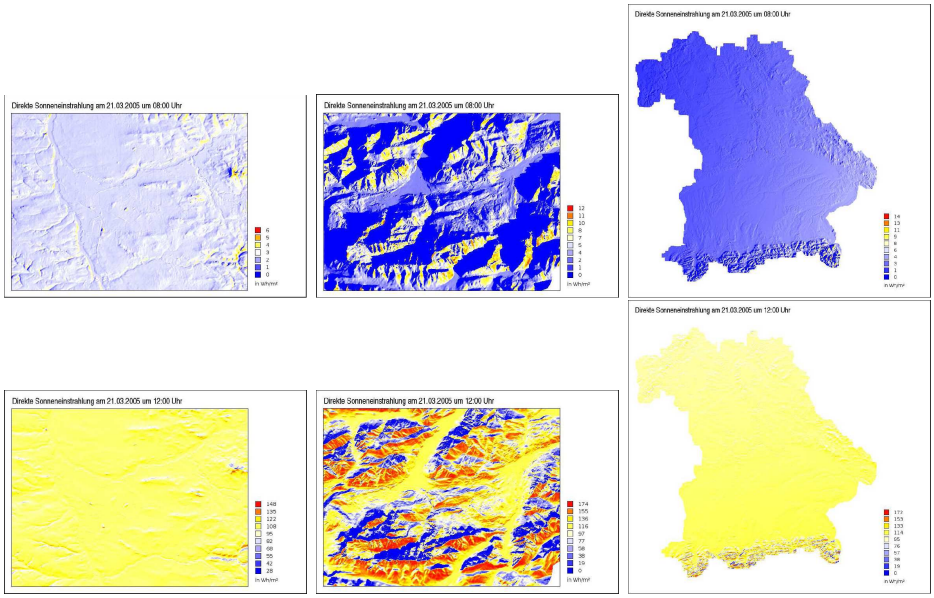

Figure 6.13: 21st March: Energy evaluation for an hour. In each column 8:00 o'clock on the top and 12:00 bottom.

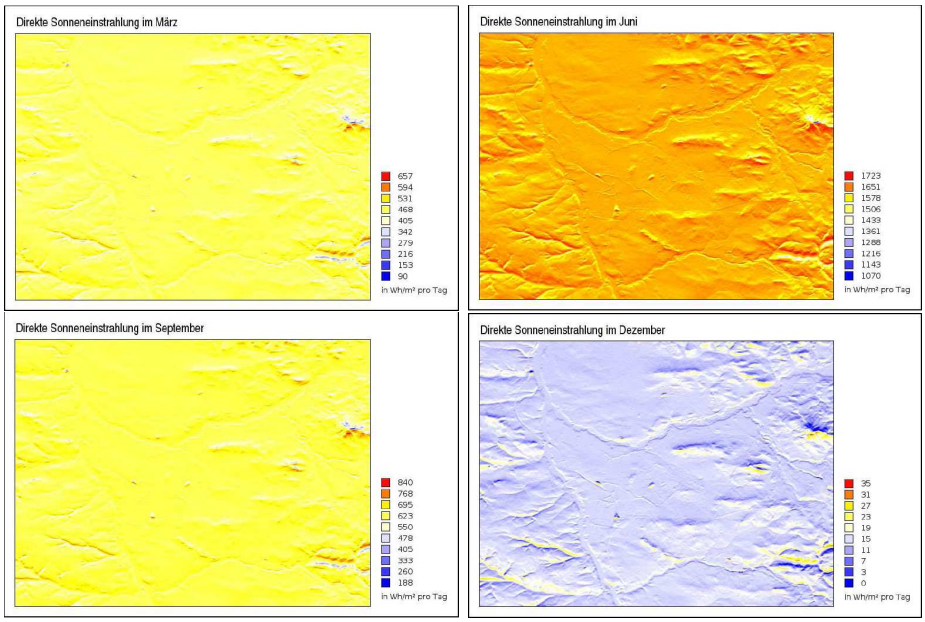

Figure 6.14: Monthly energy evaluation Nürnberg for March, June, September and December.

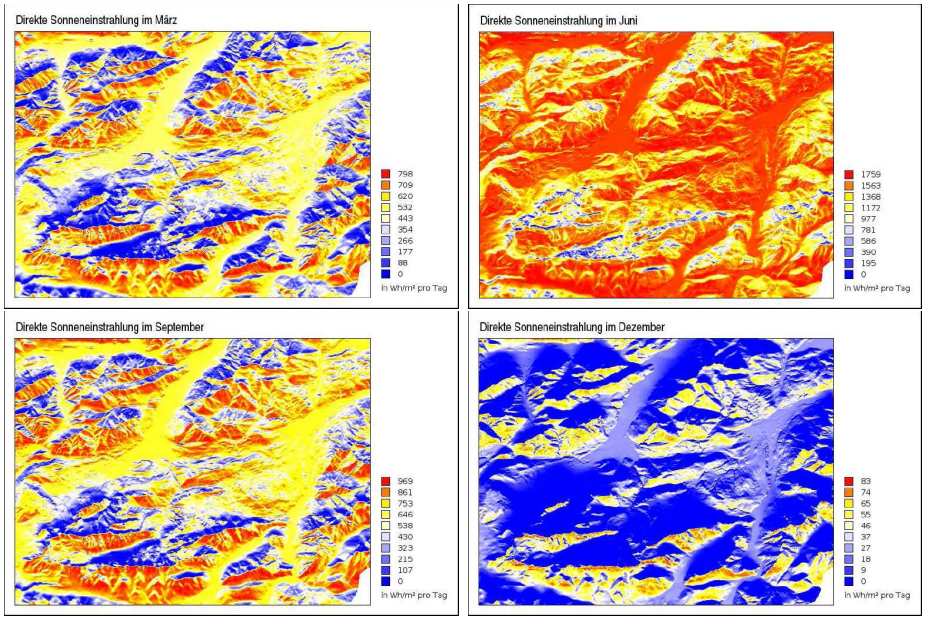

Figure 6.15: Monthly energy evaluation for the Alps for March, June, September and December.

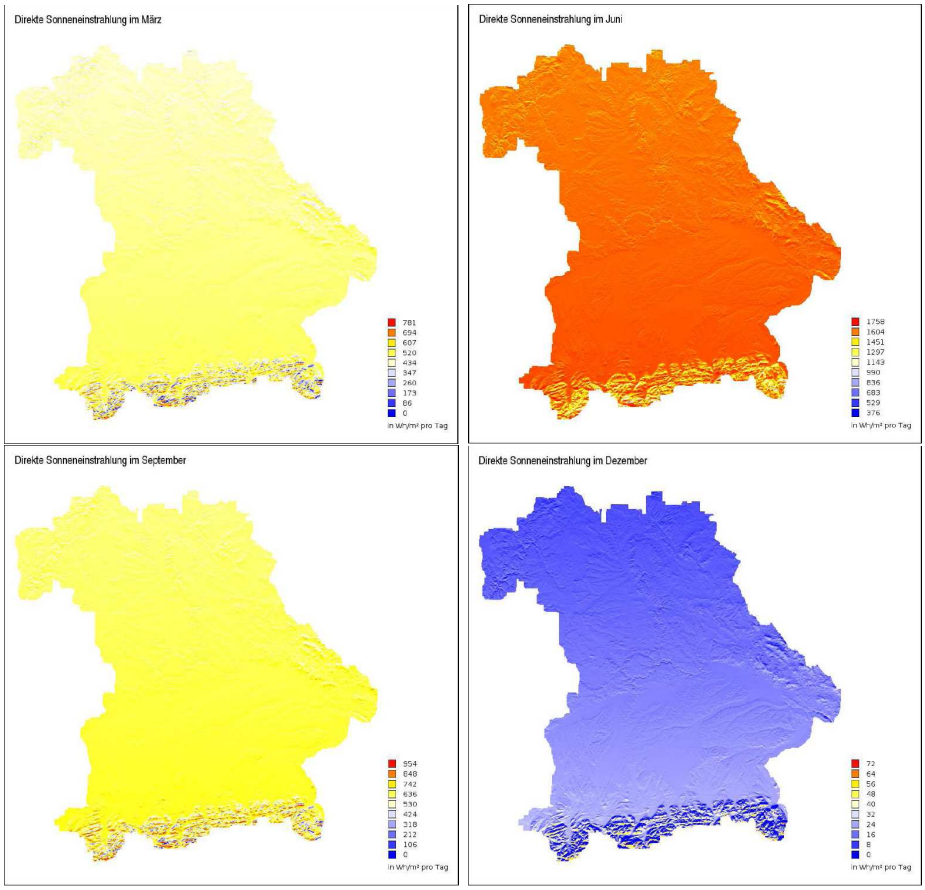

Figure 6.16: Monthly energy evaluation for Bavaria for March, June, September and December.

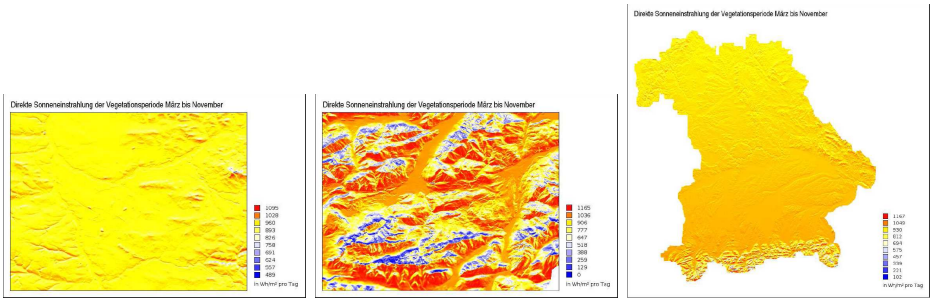

Figure 6.17: Nürnberg, Alps and Bavaria: sun hours for the vegetational period from March to November.

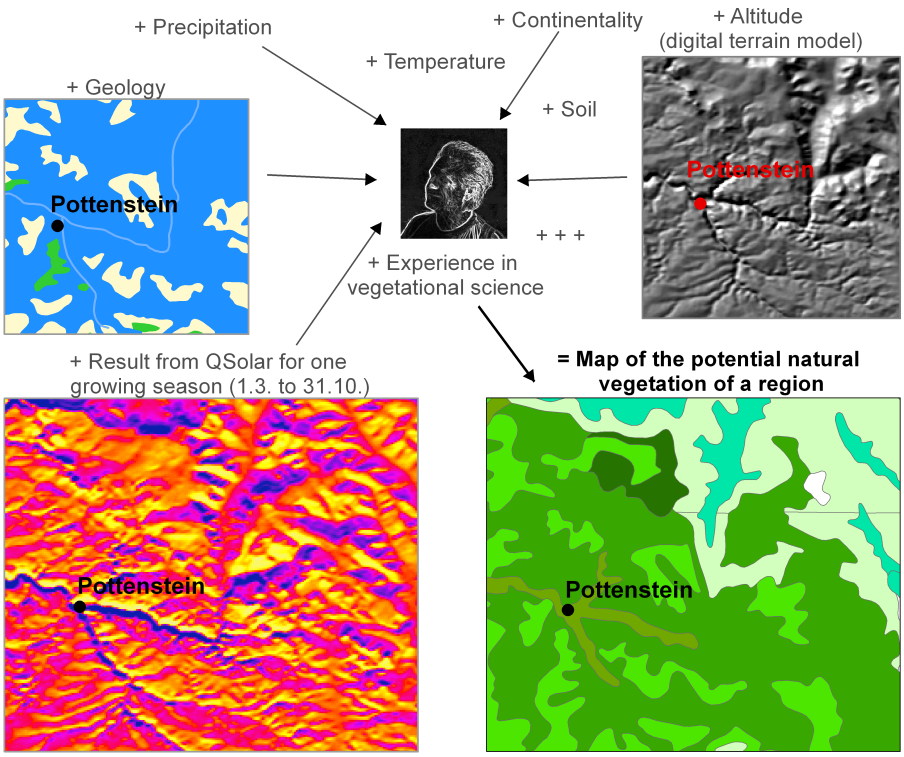

Figure 6.18: The different factors for IVL's pnv decision in the *Pottenstein* region (part of the *Fränkische Schweiz*, a region with an interesting relief. TK 200, CC 6634 Bayreuth). Images courtesy of IVL.

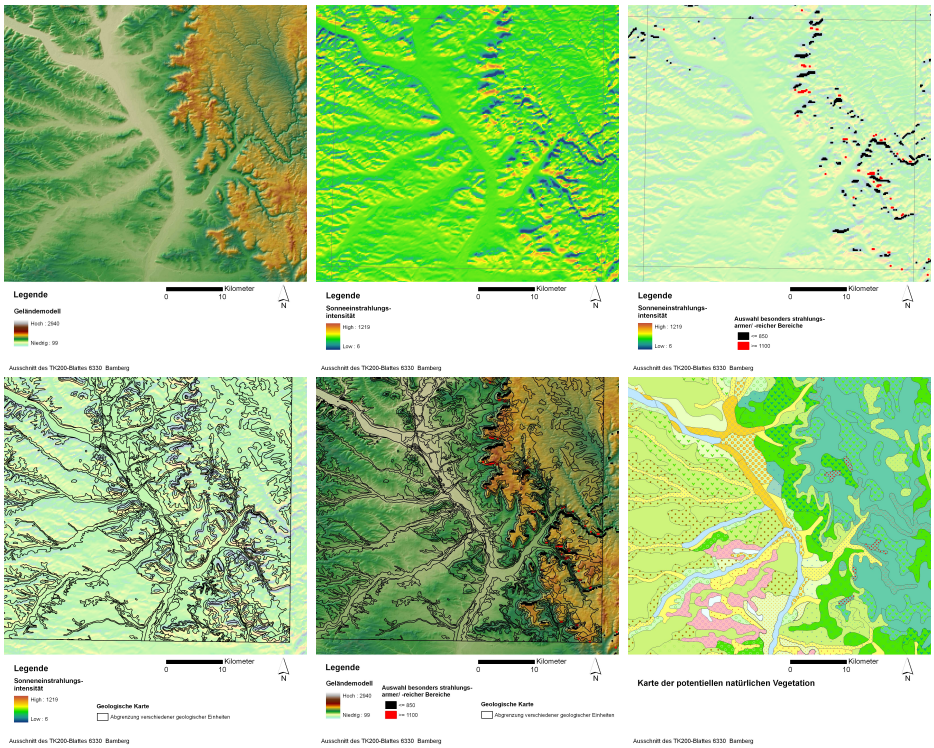

Figure 6.19: Bamberg, TK200, page 6330, from top left: relief, sun radiation, radiation hot- and lowspots, superimposed geology features and radiation hotspots, superimposed radiation hotspots with relief and geology, pnv (without legend). Images courtesy of IVL.

Chapter 7

Conclusion

Visualization has ever-expanding applications in science, education, engineering, medicine and many more fields. In this dissertation interactive visualization was used in interdisciplinary projects with partners from outside engineering. We have shown that the problems encountered all benefit from computer graphic techniques and examples were given in which our work led to scientific advances in the respective fields of science.

The work presented in this thesis can be summarized as follows:

- In archaeology we have presented non-destructive examination methods. With computed tomography scans we have been able to visualize the fragmented nature of ancient pottery. This facilitated a new interpretation of the data and a better understanding of the manufacturing process was gained. In the case of one artefact, the strong absorption contrast made a very clear distinction between original pieces and later additions visible. To enable comparison, we have applied laser sintering methods to manufacture this object including only original parts in the same scale for public display in the Antikensammlung Erlangen.

- Another example in which our techniques led to advances in the field is our work with computer game engines for the rendering of ancient structures. We have shown that game engines can be used efficiently as visualization platforms that require little development or programming skills. Therefore, they are perfectly suited to disciplines without computational background. We have presented examples of our work on various ancient sites of interest and contributed new reconstruction proposals to the ongoing archaeological debate.

- In palaeontology and oceanology we have contributed to the analysis of a deep-sea coral species, *Lophelia pertusa*. We have proposed a methodology to describe the various budding patterns found within the

species. Growth assumptions have been visualized and reviewed with 3D corals obtained from computed tomography scans. A mathematical model was proposed that can be used to incorporate measurements of real corals and describe the differences in growth speed and budding pattern. Virtual reefs of two main archetypes have been constructed and analysed for their reef building and biodiversity potential.

- In biology/geology we have contributed to the creation of maps of potential vegetation. Our method provides a more accurate solution than was previously used. We have shown how large grids of digital terrain data can be efficiently lit by sun radiation incorporating global effects like shadows and atmospheric absorption. We have implemented software for our partner in industry that was successfully applied and integrated in their workflow of related GIS applications. Through our partner we have contributed with our work to the potential natural vegetation mapping project of the Federal Agency of Nature Conservation.

As can be seen in this thesis, cooperation between different fields of science leads to advances from which both fields benefit. On the one hand, visualization provides methods and, on the other hand, the interdisciplinary partner applies these techniques to gain new insights. There is huge potential in these projects as the data and the problems of partners in interdisciplinary cooperations pose new challenges unthought of before. In particular, researchers in non-engineering disciplines are often unaware of the technological possibilites and yet their research poses great challenges, be it in terms of accuracy demands of the employed algorithms, memory requirements of especially large volumes of data or rendering speed. The challenges arising from new spheres of activity in interdisciplinary cooperations undoubtedly provide great stimulus for development in computer graphics and are likely to play an increasingly important role in the future.

Bibliography

[1] Adelson, E. H., and Bergen, J. The plenoptic function and the elements of early vision. In *Computational Models of Visual Processing* (1991), pp. 3–20.

[2] Alexa, M., Gross, M., Pauly, M., Pfister, H., Stamminger, M., and Zwicker, M. Point-based computer graphics. In *GRAPH '04: Proceedings of the conference on SIGGRAPH 2004 course notes* (New York, NY, USA, 2004), ACM Press, p. 7.

[3] Aliaga, D. G., and Carlbom, I. Plenoptic stitching: A scalable method for reconstructing 3d interactive walkthroughs. In *SIGGRAPH '01: Computer Graphics Proceedings, Annual Conference Series* (2001), pp. 443–450.

[4] Barcelo, J., Forte, M., and Sanders, D. *Virtual Reality in Archaeology, BAR International Series 843.* Archaeopress, 2000.

[5] Bauermann, I., and Steinbach, E. G. An image-based scene representation and rendering framework incorporating multiple representations. In *Vision, Modeling and Visualization* (2003), pp. 103–110.

[6] Blegen, C., and Rawson, M. *The Palace of Nestor in Pylos in Western Messenia*, vol. 1. Princeton University Press, 1966.

[7] Blinn, J. F. Models of light reflection for computer synthesized pictures. In *SIGGRAPH '77: Proceedings of the 4th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1977), ACM Press, pp. 192–198.

[8] Blum, H. A transformation for extracting new descriptors of shape. *Models for the Perception of Speech and Visual Form* (1967), 362–380.

[9] Boivin, S., and Gagalowicz, A. Image-based rendering of diffuse, specular and glossy surfaces from a single image. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2001), ACM Press, pp. 107–116.

[10] Boss, M. Die Antikensammlung der Universität Erlangen-Nürnberg. In *Die Friedrich-Alexander-Universität Erlangen-Nürnberg 1743-1993. Geschichte einer deutschen Hochschule. Veröffentlichungen des Stadtmuseums Erlangen* (1993), pp. 597–604.

[11] Boss, M. Zum "Palast des Nestor" in Pylos. Internet resource, 1999. `www.aeria.phil.uni-erlangen.de/ausstellung_html/lectures_html/pylos/pyl%os01.html`.

[12] Boss, M. Ein Thron für die Götter: Virtueller Spaziergang durch die Ruinen zum Thron von Amyklai. Tage der Forschung Universität Erlangen-Nürnberg: Auftaktvorlesung, Juli 2001.

[13] Boss, M. AERIA. Antikensammlung Erlangen Internet Archive. Originalsammlung. Internet resource, 2007. `www.aeria.phil.uni-erlangen.de/original_html`.

[14] Boss, M. Antikensammlung. In *Die Sammlungen der Universität Erlangen-Nürnberg. Begleitband zur Ausstellung "Ausgepackt. Die Sammlungen der Universität Erlangen-Nürnberg"* (2007), pp. 83–90.

[15] Boss, M., Kranz, P., and Kreilinger, U. *Wechselwirkungen – Aus der Sammlung Klaus Parlasca.* Gruner Druck GmbH, 2000.

[16] Boss, M., Kranz, P., and Kreilinger, U. *Antikensammlung Erlangen. Auswahlkatalog mit Aufnahmen von Georg Pöhlein.* Jena (Palm und Enke), 2002.

[17] The British Museum. Internet resource, 2007. `www.thebritishmuseum.ac.uk`.

[18] Buehler, C., Bosse, M., McMillan, L., Gortler, S., and Cohen, M. Unstructured lumigraph rendering. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2001), ACM Press, pp. 425–432.

[19] Bushart, M., and Suck, R. Potentielle Natürliche Vegetation von Bayern mit Karte 1:500.000, 2008. Bundesamt für Naturschutz BfN, F+E-Vorhaben 800 85 040: "Erstellung einer Übersichtskarte der Potentiellen Natürlichen Vegetation M 1:500 000 von Deutschland sowie Aufnahme und Beschreibung repräsentativer Bestände der natürlichen Vegetation" (to be pusblished).

[20] Carpenter, D., Jeffreys, J. G., and Thomson, C. W. *The Depths of the Sea. An account of the dredging cruises of H.M.SS. 'Porcupine' and 'Lightning'.* Macmillan and Co., 1873.

[21] Chang, C.-F., Bishop, G., and Lastra, A. Ldi tree: a hierarchical representation for image-based rendering. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1999), ACM Press/Addison-Wesley Publishing Co., pp. 291–298.

[22] Chen, G., Hong, L., Ng, K., McGuinness, P., Hofsetz, C., Liu, Y., and Max, N. Light field duality: concept and applications. In *VRST '02: Proceedings of the ACM symposium on Virtual reality software and technology* (New York, NY, USA, 2002), ACM Press, pp. 9–16.

[23] Chen, W.-C., Bouguet, J.-Y., Chu, M. H., and Grzeszczuk, R. Light field mapping: efficient representation and hardware rendering of surface light fields. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 447–456.

[24] Chen, W.-C., Nyland, L., Lastra, A., and Fuchs, H. Acquisition of large-scale surface light fields. In *GRAPH '03: Proceedings of the SIGGRAPH 2003 conference on Sketches & applications* (New York, NY, USA, 2003), ACM Press, pp. 1–1.

[25] Chomsky, N. Three models for the description of language. *IRE Transactions on Information Theory IT-2 2*, 3 (September 1956), 113–124.

[26] Conway, K., Krautter, M., Barrie, J. V., and Neuweiler, M. Hexactinellid sponge reefs on the Canadian continental shelf: a unique "living fossil". *Geoscience Canada 28*, 2 (2001), 71–78.

[27] Cook, R. L., and Torrance, K. E. A reflectance model for computer graphics. *ACM Transactions on Graphics 1*, 1 (1982), 7–24.

[28] Cuisenaire, O. *Distance Transformations. Fast Algorithms and Applications to Medical Image Processing.* PhD thesis, Université Catholique de Louvain, 1999.

[29] Corpus Vasorum Antiquorum (CVA). Die Antikensammlung der Universität Erlangen-Nürnberg. Band 3, 2008. to appear.

[30] Damez, C., Dimitriev, K., and Myszkowski, K. Global illumination for interactive applications and high quality animations. In *Eurographics 02 STAR* (2002).

[31] Debevec, P., Hawkins, T., Tchou, C., Duiker, H.-P., Sarokin, W., and Sagar, M. Acquiring the reflectance field of a human

face. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 145–156.

[32] Debevec, P., Reinhard, E., Ward, G., and Pattanaik, S. High dynamic range imaging. In *GRAPH '04: Proceedings of the conference on SIGGRAPH 2004 course notes* (New York, NY, USA, 2004), ACM Press, p. 14.

[33] Debevec, P. E. Facade: modeling and rendering architecture from photographs and the campanile model. In *ACM SIGGRAPH 97 Visual Proceedings: The art and interdisciplinary programs of SIGGRAPH '97* (New York, NY, USA, 1997), ACM Press, p. 254.

[34] Debevec, P. E. Image-based modeling and lighting. *Computer Graphics 33*, 4 (1999), 46–50.

[35] Debevec, P. E., and Malik, J. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1997), ACM Press/Addison-Wesley Publishing Co., pp. 369–378.

[36] Dräger, O. *Corpus Vasorum Antiquorum (CVA). Die Antikensammlung der Universität Erlangen-Nürnberg. Band 1.* Verlag C. H. Beck, 1996.

[37] Dräger, O. *Corpus Vasorum Antiquorum (CVA). Die Antikensammlung der Universität Erlangen-Nürnberg. Band 2.* Verlag C. H. Beck, 2007.

[38] Interpretation manual of European Union habitats. In *Council Directive on the Conservation of Natural Habitats and of Wild Fauna and Flora, Version EUR 15.* European Commision Directorate General XI Environment, Nuclear Safety and Civil Protection, Coastal Zones and Tourism, 1996.

[39] Everitt, C. Interactive Order-Independent Transparency. White paper. NVIDIA Corporation, 1999.

[40] Evers-Senne, J.-F., and Koch, R. Interactive rendering with view-dependent geometry and texture. In *GRAPH '03: Proceedings of the SIGGRAPH 2003 conference on Sketches & applications* (New York, NY, USA, 2003), ACM Press, pp. 1–1.

[41] Fecker, U., Guenegues, A., Scholz, I., and Kaup, A. Depth map compression for unstructured lumigraph rendering. In *Proceedings Visual Communications and Image Processing* (2006).

[42] Fichter, E. Amyklai. *Jahrbuch des Deutschen Archäologischen Instituts 33* (1918), 166 et seqq.

[43] Foley, J. D., van Dam, A., Feiner, S. K., and Hughes, J. F. *Computer Graphics: Principles and Practice*, 2nd ed. Addison-Wesley, 1990.

[44] Forte, M. About virtual archaeology: Disorders, cognitive interactions and virtuality. *BAR International Series 843* (2000), 247–259.

[45] Fossa, J. H., Mortensen, P. B., and Furevik, D. M. The deep-water coral *lophelia* pertusa in norwegian waters: distribution and fishery impacts. *Hydrobiologia – Biology of Cold Water Corals 471* (2002), 1–12.

[46] Freiwald, A., Fossa, J., Grehan, A., Koslow, T., and Roberts, J. *Cold-water Coral Reefs*. UNEP-WCMC, 2004.

[47] Freiwald, A., and Roberts, J. *Cold-water Corals and Ecosystems*. Springer, 2005.

[48] Frings, M. *Der Modelle Tugend: CAD und die neuen Räume der Kunstgeschichte*. VDG Verlag, Weimar, 2001.

[49] Georghiades, A. S. Recovering 3-d shape and reflectance from a small number of photographs. In *EGRW '03: Proceedings of the 14th Eurographics workshop on Rendering* (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 230–240.

[50] Gernot, S. Image-based object representation by layered impostors. In *VRST '98: Proceedings of the ACM symposium on Virtual reality software and technology* (New York, NY, USA, 1998), ACM Press, pp. 99–104.

[51] Gianni, M. High seas bottom trawl fisheries and their impact on the biodiversity of vulnerable deep-sea ecosystems. *Report for ICUN, NRDC, WWF Conservation International* (June 2004).

[52] Gortler, S. J., Grzeszczuk, R., Szeliski, R., and Cohen, M. F. The lumigraph. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1996), ACM Press, pp. 43–54.

[53] Gunnerus, J. E. Om nogle norske coraller. *Det Kongelige Norske Videnskabers Selskab Skrifter 4* (1768), 38–73.

[54] Hakura, Z. S., Snyder, J. M., and Lengyel, J. E. Parameterized environment maps. In *SI3D '01: Proceedings of the 2001 symposium*

on Interactive 3D graphics (New York, NY, USA, 2001), ACM Press, pp. 203–208.

[55] Härdtle, W. Potentielle natürliche Vegetation. Ein Beitrag zur Kartierungsmethode am Beispiel der topographischen Karte 1623 Owschlag. *Mitteilungen der Arbeitsgemeinschaft Geobotanik in Schleswig-Holstein und Hamburg*, 40 (1989).

[56] Hargreaves, S., and Harris, M. Deferred shading. NVIDIA Presentation 2004 and Game Developers Conference, 2004.

[57] Hawkins, T., Cohen, J., and Debevec, P. A photometric approach to digitizing cultural artifacts. In *VAST '01: Proceedings of the 2001 conference on Virtual reality, archeology, and cultural heritage* (New York, NY, USA, 2001), ACM Press, pp. 333–342.

[58] He, X. D., Heynen, P. O., Phillips, R. L., Torrance, K. E., Salesin, D. H., and Greenberg, D. P. A fast and accurate light reflection model. *ACM Transactions on Graphics 26*, 2 (1992), 253–254.

[59] Hoffman-Wellenhof, B. *GPS in der Praxis*. Springer, Wien, 1994.

[60] Hoffmann, H. *Sotades. Symbols of Immortality on Greek Vases*. Clarendon Press, 1997.

[61] Hormann, K., Labsik, U., Meister, M., and Greiner, G. Hierarchical extraction of iso-surfaces with semi-regular meshes. In *Proceedings of the Seventh ACM Symposium on Solid Modeling and Applications* (2002), pp. 55–58.

[62] International Color Consortium (ICC). Internet resource, 2006. `www.color.org`.

[63] Isaksen, A., McMillan, L., and Gortler, S. J. Dynamically reparameterized light fields. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 297–306.

[64] IVL – Institut für Vegetationskunde und Landschaftsökologie. Internet resource, 2007. `www.ivl-web.com`.

[65] Jacob, C. Genetic l-system programming. In *PPSN III: Proceedings of Parallel Problem Solving from Nature* (Berlin, 1994), Springer-Verlag, pp. 334–343.

[66] Johnston, S. F. Lumo: illumination for cel animation. In *NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering* (New York, NY, USA, 2002), ACM Press, pp. 45–ff.

[67] Kaandorp, J. A., and Kübler, J. E. *The Algorithmic Beauty of Seaweeds, Sponges and Corals*. Springer-Verlag, Berlin, 2001.

[68] Kajiya, J. T. The rendering equation. In *Computer Graphics* (1986), vol. 20, ACM Press, pp. 143–150.

[69] Katz, S., and Tal, A. Hierarchical mesh decomposition using fuzzy clusering and cuts. *ACM Transactions on Graphics 22*, 3 (2003), 984–961.

[70] Kautz, J., Sattler, M., Sarlette, R., Klein, R., and Seidel, H.-P. Decoupling BRDFs from surface mesostructures. In *GI '04: Proceedings of the 2004 conference on Graphics interface* (2004), Canadian Human-Computer Communications Society, pp. 177–182.

[71] Koller, D., Turitzin, M., Levoy, M., Tarini, M., Croccia, G., Cignoni, P., and Scopigno, R. Protected interactive 3d graphics via remote rendering. *ACM Transactions on Graphics 23*, 3 (2004), 695–703.

[72] Kontkanen, J., and Laine, S. Ambient occlusion fields. In *I3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games* (New York, NY, USA, 2005), ACM, pp. 41–48.

[73] Kowarik, I. Kritische Anmerkungen zum theoretischen Konzept der potentiellen natürlichen Vegetation (PNV): Schwachstellen und Entwicklungserspektiven. *Tuexenia*, 7 (1987), 53–67.

[74] Kutulakos, K. N., and Seitz, S. M. A Theory of Shape by Space Carving. Tech. Rep. TR692, 1998.

[75] Kókai, G. *Erfolge und Probleme evolutionäre Algorithmen, induktiver logischer Programmierung und ihrer Kombination*. PhD thesis, Friedrich-Alexander-Universität, Erlangen-Nürnberg, 2003.

[76] Kókai, G., Ványi, R., and Toth, Z. Parametric L-System Description of the Retina with Combined Evolutionnary Operators. In *GECCO: Genetic and Evolutionnary Computation COnference* (1999), vol. 2, pp. 1588–1596.

[77] Labsik, U., Hormann, K., Meister, M., and Greiner, G. Hierarchical extraction of iso-surfaces. *Journal of Computing and Information Science in Engineering 2* (2002), 323–329.

[78] Latta, L., and Kolb, A. Homomorphic factorization of brdf-based lighting computation. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 509–516.

[79] Lawrence, J., Rusinkiewicz, S., and Ramamoorthi, R. Efficient BRDF importance sampling using a factored representation. *ACM Transactions on Graphics 23*, 3 (2004), 496–505.

[80] Lensch, H. P. A., Kautz, J., Goesele, M., Heidrich, W., and Seidel, H.-P. Image-based reconstruction of spatial appearance and geometric detail. *ACM Transactions on Graphics 22*, 2 (2003), 234–257.

[81] Levoy, M., and Hanrahan, P. Light field rendering. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1996), ACM Press, pp. 31–42.

[82] Li, H., and Wu, E. A representation for composition of virtual indoor environment. In *VRST '98: Proceedings of the ACM symposium on Virtual reality software and technology* (New York, NY, USA, 1998), ACM Press, pp. 171–178.

[83] Lindenmayer, A. Mathematical models for cellular interaction in development, parts i and ii. *Journal of Theoretical Biology 18* (1968), 180–315.

[84] Lindenmayer, A., and Rosenberg, G. *Automata, Languages, Development*. North Holland Publishing Co., 1976.

[85] Linne, C. *Systema naturae per Regna tria naturae, secundum classes, ordines, genera, species, cum characteribus differentiis, synonymis, locis*. 1758.

[86] Loop, C. *Smooth Subdivision Surfaces Based on Triangles*. PhD thesis, University of Utah, Seattle, 1987.

[87] Lophelia.org. Internet Resource, 2007. `www.lophelia.org`.

[88] Lorenzen, W., and Cline, H. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics 21*, 4 (1987), 351–358.

[89] The Louvre. Internet resource, 2007. `www.louvre.fr`.

[90] Masselus, V., Peers, P., Dutre, P., and Willems, Y. D. Relighting with 4d incident light fields. *ACM Transactions on Graphics 22*, 3 (2003), 613–620.

[91] Matusik, W., Pfister, H., Ngan, A., Beardsley, P., Ziegler, R., and McMillan, L. Image-based 3d photography using opacity hulls. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 427–437.

[92] Matusik, W., Pfister, H., Ziegler, R., Ngan, A., and McMillan, L. Acquisition and rendering of transparent and refractive objects. In *EGRW '02: Proceedings of the 13th Eurographics workshop on Rendering* (Aire-la-Ville, Switzerland, Switzerland, 2002), Eurographics Association, pp. 267–278.

[93] McMillan, L., and Bishop, G. Plenoptic modeling: An image-based rendering system. In *SIGGRAPH '95: Computer Graphics Proceedings, Annual Conference Series* (1995), pp. 39–46.

[94] McMillan, L., and Gortler, S. Image-based rendering: A new interface between computer vision and computer graphics. *Computer Graphics 33*, 4 (2000), 61–64.

[95] Meeus, J. *Astronomical Algorithms*. Willman-Bell, 1998.

[96] Meister, M. Adaptive Triangulierung und Profilerstellung antiker Vasen. Studienarbeit am Institut für Computergrafik, FAU Erlangen-Nürnberg, Dezember 2001.

[97] Meister, M., and Boss, M. On using State of the Art Computer Game Engines to Visualize Archaeological Structures in Interactive Teaching and Research. In *CAA '03: Enter the Past – The E-way into the Four Dimensions of Cultural Heritage, Computer Applications and Quantitative Methods in Archaeology* (2003), vol. 31, pp. 505–509.

[98] Meister, M., and Freiwald, A. Modelling Lophelia Colonies. In *Erlanger Geologische Abhandlungen: 2nd International Symposium on Deep-Sea Corals* (September 2003), vol. 4, pp. 61–62.

[99] Merhof, D., Enders, F., Vega, F., Hastreiter, P., Nimsky, C., and Stamminger, M. Integrated visualization of diffusion tensor fiber tracts and anatomical data. In *Proceedings of Simulation and Visualisierung '05* (2005), pp. 153–164.

[100] Merks, R., Hoekstra, A. G., Kaandorp, J. A., and Sloot, P. M. A. Models of coral growth: spontaneous branching, compactification and the laplacian growth assumption. *Journal of Theoretical biology*, 224 (2003), 153–166.

[101] Merks, R., Hoekstra, A. G., Kaandorp, J. A., and Sloot, P. M. A. Polyp oriented modelling of coral growth. *Journal of Theoretical Biology* (2004).

[102] Montenbruck, O., and Pfleger, T. *Astronomy on the Personal Computer*. Springer, Berlin, 2004.

[103] Moore, P. *Lexikon der Astronomie*, 2nd ed. Herder Freiburg im Breisgau, 1989.

[104] Neubauer, A., Wolfsberger, S., Forster, M.-T., Mroz, L., Wegenkittl, R., and Buhler, K. Steps - an application for simulation of transsphenoidal endonasal pituitary surgery. In *VIS '04: Proceedings of the conference on Visualization '04* (2004), pp. 513–520.

[105] Niemeyer, H. G. *Einführung in die Archäologie*, 4th ed. Wissenschaftliche Buchgesellschaft, 1995.

[106] United Nations Atlas of the Oceans. Internet Resource, 2007. `www.oceanatlas.org`.

[107] Oh, B. M., Chen, M., Dorsey, J., and Durand, F. Image-based modeling and photo editing. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2001), ACM Press, pp. 433–442.

[108] Oliveira, M. M., and Bishop, G. Image-based objects. In *SI3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics* (New York, NY, USA, 1999), ACM Press, pp. 191–198.

[109] Pausanias, P. *Beschreibung Griechenlands (Übersetzung von Ernst Meyer)*. Artemis Verlag, 1967.

[110] Peitgen, H.-O., Jürgen, H., and Saupe, D. Chaos and fractals. *New Frontiers of Science* (1993), 363.

[111] Pfister, H., Zwicker, M., van Baar, J., and Gross, M. Surfels: surface elements as rendering primitives. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 335–342.

[112] Pietsch, M., Timpe, D., and Wamser, L. Das augusteische Truppenlager Marktbreit. *Berichte der Römisch-Germanischen Kommission (BerRGK) 72* (1991), 264–324.

[113] Prueckner, C. Ein Thron für Apollon. *Kotinos. Festschrift für Erika Simon* (1992), 123–130.

[114] Prunsikiewicz, P. Visual models of morphogenesis. *Artificial Life* (1994), 67–74.

[115] Prusinkiewicz, P., and Lindenmayer, A. *The Algorithmic Beauty of Plants*. Springer-Verlag, New York, 1990.

[116] Pyl, T. Der amykläische Apoll. *Archäologische Zeitung* (1852), 465 pp.

[117] Qt Programming library - Trolltech. Internet resource, 2007. `www.trolltech.org`.

[118] Ramamoorthi, R., and Hanrahan, P. Frequency space environment map rendering. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 517–526.

[119] Ramamoorthi, R., and Hanrahan, P. A signal-processing framework for reflection. *ACM Transactions on Graphics 23*, 4 (2004), 1004–1042.

[120] Raskar, R., Tumblin, J., Mohan, A., Agrawal, A., and Li, Y. Computational photography. In *Eurographics 06 STAR* (New York, NY, USA, 2006), ACM Press.

[121] Rauh, J. Virtuelle Sonnenstunden über Bayern. Studienarbeit am Institut für Computergrafik, FAU Erlangen-Nürnberg, August 2005.

[122] Rauh, J., Meister, M., Greiner, G., and Suck, R. QSolar - A Tool to generate Maps of direct Sun Radiation. In *ASIM '05: Frontiers in Simulation, Simulationstechnique 18th Symposium* (September 2005), pp. 670–675.

[123] Reche-Martinez, A., Martin, I., and Drettakis, G. Volumetric reconstruction and interactive rendering of trees from photographs. *ACM Transactions on Graphics 23*, 3 (2004), 720–727.

[124] Regan, M. J. P., Miller, G. S. P., Rubin, S. M., and Kogelnik, C. A real-time low-latency hardware light-field renderer. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1999), ACM Press/Addison-Wesley Publishing Co., pp. 287–290.

[125] Rezk-Salama, C., Engel, K., Bauer, M., Greiner, G., and Ertl, T. Interactive Volume Rendering on Standard PC Graphics Hardware using Multi-Textures and Multi-Stage Rasterization. In *Proceedings Eurographics/SIGGRAPH Workshop on Graphics Hardware* (2000), pp. 109–118.

[126] Rieche, A. *Archäologie virtuell: Projekte, Entwicklungen, Tendenzen seit 1995*. Habelt Verlag, Bonn, 2002.

[127] Sato, Y., Wheeler, M. D., and Ikeuchi, K. Object shape and reflectance modeling from observation. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1997), ACM Press/Addison-Wesley Publishing Co., pp. 379–387.

[128] Schirmacher, H., Zöckler, M., Stalling, D., and Hege, H.-C. Boundary surface shrinking - a continuous approach to 3d center line extraction. In *Proceedings of IMDSP '98* (1998), pp. 25–28.

[129] Scholz, I., Denzler, J., and Niemann, H. Calibration of real scenes for the reconstruction of dynamic light fields. In *IEICE Transactions on Information & Systems* (2004), pp. 42–49.

[130] Schöning, U. *Theoretische Informatik kurz gefasst*. BI Wissenschaftsverlag, Mannheim, 1992.

[131] Schroeder, W. J., Zarge, J. A., and Lorensen, W. E. Decimation of triangle meshes. *Computer Graphics 26*, 2 (1992), 65–70.

[132] Segal, M., Korobkin, C., van Widenfelt, R., Foran, J., and Haeberli, P. Fast shadows and lighting effects using texture mapping. In *Computer Graphics* (1992), vol. 26, ACM Press, pp. 249–252.

[133] Sen, P., Chen, B., Garg, G., Marschner, S. R., Horowitz, M., Levoy, M., and Lensch, H. P. A. Dual photography. *ACM Transactions on Graphics 24*, 3 (2005), 745–755.

[134] Shade, J., Gortler, S., wei He, L., and Szeliski, R. Layered depth images. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1998), ACM Press, pp. 231–242.

[135] Sillion, F., and Puech, C. *Radiosity and Global Illumination*. Morgan Kaufmann Publishers, 1994.

[136] Sims, D. Archaeological models: Pretty pictures or research tools? In *IEEE Computer Graphics and Applications* (1997), vol. 17, pp. 13–15.

[137] Siu, A. M. K., Wan, A. S. K., and Lau, R. W. H. Modeling and rendering of walkthrough environments with panoramic images. In *VRST '04: Proceedings of the ACM symposium on Virtual reality software and technology* (New York, NY, USA, 2004), ACM Press, pp. 114–121.

[138] Sloan, P.-P., Kautz, J., and Snyder, J. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 527–536.

[139] Stein, M., and Westheide, W. *Spezielle Zoologie: Teil 1.* Spektrum Akademischer Verlag, New York, 2006.

[140] te Pas, S. F., and Pont, S. C. A comparison of material and illumination discrimination performance for real rough, real smooth and computer generated smooth spheres. In *APGV '05: Proceedings of the 2nd symposium on Appied perception in graphics and visualization* (New York, NY, USA, 2005), ACM Press, pp. 75–81.

[141] Torrance, K. E., and Sparrow, E. M. Theory for off-specular reflection from roughened surfaces. *Journal of the American Optical Society 57* (September 1967), 1105–1114.

[142] Trautmann, W. Erläuterungen zur Karte der potentiellen natürlichen Vegetation der Bundesrepublik Deutschland. *Schriftenreihe für Vegetationskunde*, 1 (1966).

[143] Troja - Traum und Wirklichkeit. Internet resource, 2001. `www.troia.de`.

[144] Tuexen, R. Die heutige potentielle natürliche Vegetation als Gegenstand der Vegetationskartierung. *Angewandte Pflanzensoziologie*, 13 (1956), 5–42.

[145] Unger, J., Wenger, A., Hawkins, T., Gardner, A., and Debevec, P. Capturing and rendering with incident light fields. In *EGRW '03: Proceedings of the 14th Eurographics workshop on Rendering* (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 141–149.

[146] van der Linden, J. Multiple light field rendering. In *GRAPHITE '03: Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia* (New York, NY, USA, 2003), ACM Press, pp. 197–202.

[147] Vlasic, D., Pfister, H., Molinov, S., Grzeszczuk, R., and Matusik, W. Opacity light fields: interactive rendering of surface light fields with view-dependent opacity. In *SI3D '03: Proceedings of the 2003 symposium on Interactive 3D graphics* (New York, NY, USA, 2003), ACM Press, pp. 65–74.

[148] Vogelgsang, C. *The lgf3 Project: A Versatile Implementation Framework for Image-Based Modeling and Rendering.* PhD thesis, Institut für Informatik, Universität Erlangen-Nürnberg, 2005.

[149] Vogelgsang, C., and Greiner, G. lgf3 - a versatile framework for image-based modeling and rendering. In *SIGGRAPH 2004 Sketches and Applications* (2004).

[150] Wald, I., and Slusallek, P. State-of-the-art in interactive ray-tracing. In *Eurographics 01 STAR* (2001), pp. 21–42.

[151] Wamser, L. *Die Römer zwischen Alpen und Nordmeer. Katalog-Handbuch zur Landesausstellung des Freistaates Bayern.* L. Wamser, Rosenheim, 2000.

[152] Wamser, L. Von der Ausgrabung zur Rekonstruktion. *Die Römer zwischen Alpen und Nordmeer 1* (2000), 435–438.

[153] Wikipedia. Internet resource, 2007. `www.wikipedia.org`.

[154] Willems, G., Verbiest, F., Vergauwen, M., and Gool, L. V. Real-time image based rendering from uncalibrated images. In *3DIM '05: Fifth International Conference on 3-D Digital Imaging and Modeling* (2005), pp. 221–228.

[155] Wilson, J. B. The distribution of the coral Lophelia pertusa (L.) in the North East Atlantic. *Journal of Marine Biological Association of the UK 59* (June 1979), 149–164.

[156] Winter, M., Greiner, M., Vogt, F., Niemann, H., and Krüger, S. Visualizing distances between light field and geometry using projective texture mapping. In *VMV '05: Vision, Modeling and Visualization* (2005), pp. 257–274.

[157] Winter, M., Meister, M., and Greiner, G. Vmv '06: Multiple Unstructured Lumigraphs in a Rendering Framework. In *Vision, Modeling and Visualization* (2006), pp. 121–128.

[158] Wood, D. N., Azuma, D. I., Aldinger, K., Curless, B., Duchamp, T., Salesin, D. H., and Stuetzle, W. Surface light fields for 3d photography. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 287–296.

[159] Xu, Z., Schwarte, R., Heinol, H., Buxbaum, B., and Ringbeck, T. Smart pixel - photonic mixer device (pmd). In *M2VIP '98: International Conference on Mechatronics and Machine Vision in Practice* (1998), pp. 259–264.

Lebenslauf

Martin Meister

Geburtsdatum:	01. März 1976
Geburtsort:	Weiden
Staatsangehörigkeit:	deutsch
Familienstand:	verheiratet

1982 – 1986	Grundschule Krummennaab
1986 – 1995	Stiftland-Gymnasium Tirschenreuth
1995	Abitur
1995 – 1996	Studium der Ur- Frühgeschichte, Klassischer Archäologie und Paläontologie (FAU)
1996 – 2002	FAU Erlangen: Studium der Informatik
Feb. 2003	Diplomierung (Arbeit zum Thema "3D-Puzzle: Rekonstruktion rotationssymmetrischer Gefäße")
Feb. 2003 – Dez. 2003	Techniker am Lehrstuhl für Computergrafik (FAU)
seit 2004	Wissenschaftlicher Mitarbeiter am Lehrstuhl für Computergrafik (Promotion)