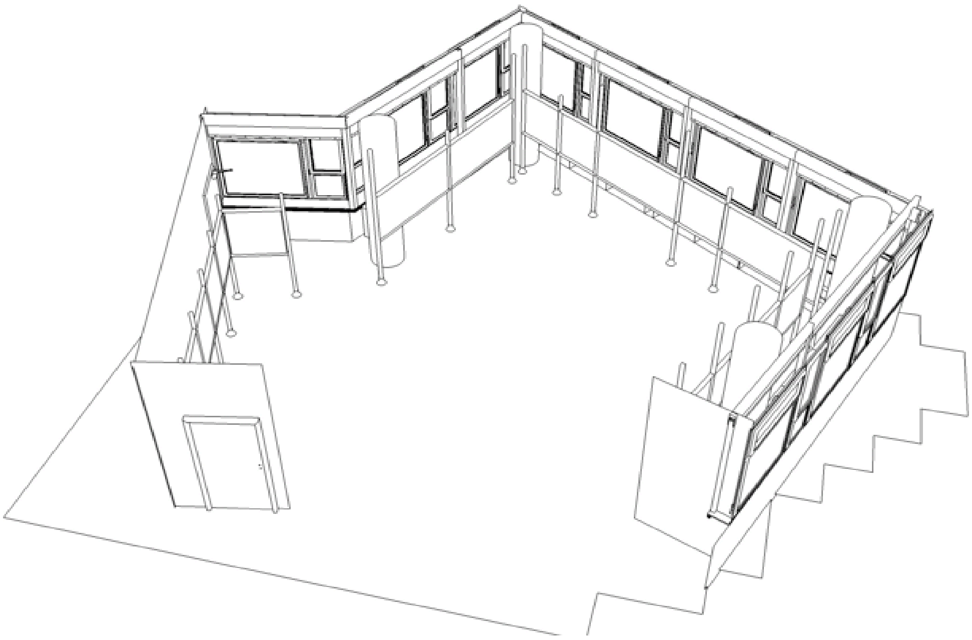
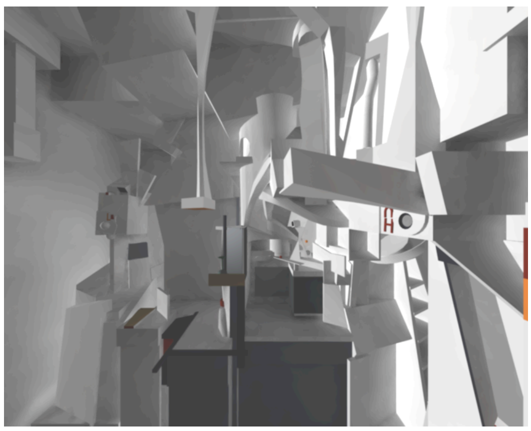
frieder nake, andreas genz – [nake | genzo]@informatik.uni-bremen.de
october 20, 2004 :: outline

Administrative data:
The course is a programming lab in the form of independent study on an advanced level.
VAK 03-708.31. Module 301-2. 2 SWS, 4 ECTS.
Participation is by personal arrangement only.
Listed for the Diplom-Informatik (Hauptstudium) and M.Sc. in Digital Media programs.
Students of the B.Sc. in Digital Media program may be admitted if qualified.
Starts with a preparatory meeting on October 20, 2004, from noon to 1 p.m., in OAS 3000 (Linzer Straße 9a).
There will be no fixed class meetings afterwards. Instead, individual study plans will be arranged for each participant, or pair of such.



Studiengalerie des Studium Generale, TH Stuttgart, Hahnhochhaus 8. St. 1960er Jahre.

Edge line drawing



Two renditions of the virtual Merzbau

The

figures above show views of virtual reconstructions of two sites. The lower two figures, side by side, are computer renditions of the virtual Merzbau. The Merzbau was a sculpture by the German Dadaist artist, Kurt Schwitters. This Dadaistic architecture was digitally reconstructed in 1999 by Zvonimir Bakotin for a web side of Sprengel Museum Hannover. We used it as the initial space of a medium for primary school children, and also as the object of choice for several previous Advanced Topics classes. Students were entrusted with specific tasks to improve on the geometric model, or the rendering of details, of this structure. Some of the studies they did were quite successful (you will find some on our web site).

This term, however, we will switch to a newt basic object. We invite you to work on it with the common goal of improving both: model (geometric background) and rendition (visual appearance).

You see a drawing of the new object in the upper figure above. It shows a virtual reconstruction of the 8th floor of Stuttgart's Hahnhochhaus, a building in the city, which, for several years in the 1960s and 70s, housed some humanities institutes of the University of Stuttgart. The 8th floor, in particular, was the philosophers' location. The room was used for seminars and other meetings, and also for experimental art exhibitions. Max Bense arranged those.

Here the world's first exhibition of graphic computer art took place, from February 5 to 19, 1965. Three students of our Bachelor programme in Digital Media (Leif Genzmer, Oliver Graf, and Eva-Sophie Katterfeldt) worked on an algorithmic reconstruction of this room. In their Bachelor Report they present the results. They are quite remarkable and promising. With your help, we would like obtain improvements and further refinements of the model, and use it as a proving ground for your advancements into computer graphics.

What

are you expected to do for the **advanced lab in graphics programming**?

We want to reach such students of the Hauptdiplom Informatik and the M.Sc. in Digital Media programmes, who are looking for an extra challenge. If you consider taking this lab, you should have advanced knowledge in the area of computer graphics. That is, you should at least have taken an introductory class on interactive computer graphics, or you should otherwise have gained experience in interactive computer graphics in theory and programming. You should be familiar with a good textbook in the field, but your experience should not be restricted to reading books; you should also have done some graphics programming. You should have a working knowledge of C or C++ to the degree that you independently find your way as you develop an algorithmic solution to a non-trivial problem. You must also be capable of working on your own. Quite frankly, you should love to do that.

We stress this point because the approach of the lab is directed to only those few, who are ready to do serious independent work. After a first introductory meeting, our schedule does not include any further class meetings. We will not teach you by taking you by the hand. We will instead try helping you to do your own work. We will arrange something like a contract with you, under which you will be asked to finish your personal non-trivial assignment within a period of precisely eight weeks. You determine the day when that individual period of eight weeks starts. You may work in groups of two, if you want.

As far as computer graphics is concerned, you must know the fundamentals: raster graphics, ray tracing, radiosity, shading, modeling objects and scenes in 3D space with splines and other techniques, graphic interfaces. And, of course, the mathematics of all that should not cause a nervous breakdown. You must be proficient in such matters to such a degree that you may independently work on a problem that requires understanding important algorithms, concepts, and methods of those areas. You should be interested in a graphics specialization, and feel comfortable with doing individual project work. You may, of course, ask for advice during the eight weeks period. If your general programming skills are restricted to a lower level, you should think twice before taking this class.

We start by explaining the format and introducing our proposals for your choice of individual task. You are expected to select one of them (see the list below). After we have precisely formulated your project in concorde with you, we agree with you on two individual checkpoints and the date of your final presentation.

The lab's features are:

- You gain insight into recent scientific literature on graphics.
- You gain experience in doing thorough and well-planned programming.
- You get an opportunity to solve a problem all by yourself.
- You work on a problem originating in the current research of our group, agis, and you solve it to the point of demonstration.
- You start at almost any point in time.
- You may take the lab as preparation for your thesis work.

The schedule roughly calls for the following. You pick one of the problems we suggest, and you do so individually or in pairs. We take up specifics you may raise concerning the problem definition, and we formulate your personal problem specification after the first warm-up meeting. You work on your problem for no more than eight weeks. At two intermittent checkpoint meetings, you present preliminary results. We are ready to give you advice at almost all times (though not by night), via email, phone, or meeting over a cup of coffee. The lab ends with your demonstration of the software, and a written report.

The report must contain the problem specification, your approach for a solution, important algorithmic steps, data structures, and geometric modeling decisions. You should also draw a conclusion for further work, and comment on your learning experience. The report should contain pictures and diagrams where ever appropriate. It must be written in a clear and precise style, refer to the most important literature, and convince us of your expertise. The paper should have a length of between 8 and 12 pages.

As already mentioned, it is possible to start at any time during the semester, but such that you are finished before its definite end (March 31, 2005). We agree on details of your specific time slot in advance, and to your advantage. Our promise is to maximally support your independent study. You must, however, be aware of the fact that Frieder will be in Aarhus in March 2005, and Andreas' status is pending after the end of February.

The problem specification defines minimal requirements, which your solution must satisfy. We say a word about the hardware and software you are allowed, or expected, to use, and we hint at some literature you may be requested to study. OpenGL (and thus C or C++), Pixar RenderMan, Maxon Cinema 4D on an Apple Macintosh are typical candidates for software support.

By the end of your assignment, you should feel like having attained some extraordinary result after a demanding effort. You should be proud of what you have achieved, and we would want to share your pride. (Perhaps, your demonstration is worth to be presented to people from the outside.)

At the preparatory meeting, we present the assignments we invite you to. We fully specify your problem only after that first encounter so that we may take into consideration your preferences and conditions.

A general reference to the graphics literature besides the well-known standard of J. Foley, A. van Dam, St. Feiner, and J. Hughes: *Computer Graphics. Principles and Practice*. Reading, MA: Addison Wesley, is Alan Watt and Fabio Policarpo: *The Computer Image*. Reading, MA: Addison Wesley 1998.

Preliminary Remark

The three B.Sc. students, who were mentioned above, turned the Studiengalerie into a geometric model in order to demonstrate how the location may have looked. They also used it to display works of art by Georg Nees. These, furthermore, became the starting points of a hypermedium on Nees' art.

The model grew into considerable complexity. In Maxon Cinema 4d, it consists of, roughly, 75,000 vertices, and 93,000 triangles or quadrilaterals. When transformed into VRML-2 format, the file needs 14 MB, containing 215,000 vertices and 280,000 polygons (of three or four vertices). The VRML export has one or two minor visible bugs.

Because of the hypermedia requirement, the current version runs only under Mac OSX. We want to first create a starting point (one group) for a number of tasks, which other groups would be working on. Their goal will be to generate a correct and efficient model, which allows convincing (photo)realistic rendering as well as illustrative sketch work (non-photorealistic rendering). We favor a combination of the two as a special aesthetics that may be more appropriate for the situation. This is our hypothesis. We invite you to work with us on verifying or falsifying it.

Our first issue to discuss will be: how create that new starting point? Should it be from scratch, or from the current implementation?

Proposal A.
Generate a basic geometric model of the Studiengalerie and an adapted version of the Merzbau viewer

Problem. The geometric complexity of the current model of the Studiengalerie is extremely high compared to its visual complexity. A more efficient but still correct polygonal model is in need. Also, for various former Advanced Topics projects, students used a viewer that was specifically geared towards the Merzbau architecture in a VRML format. They completed some interesting implementations that would be nice to see applied to the Studiengalerie model.

Objective. Adapt the current viewer such that it may deal both with the Merzbau and the Studiengalerie in VRML format. Implement an algorithm for simplification of the model, and a routine for again storing the model in VRML. Also, improve parts of the model interactively (for example in Cinema 4d) if the outcome of your algorithm is unsatisfying. – This proposal needs some more detail of specification.

References

LEIF ARNE GENZMER, OLIVER GRAF, EVA-SOPHIE KATTERFELDT: Künstliche Kunst im virtuellen Raum – ein digitales Museum für Georg Nees, Bachelorreport, FB3, Universität Bremen, 2004 (we will make parts of it available)

RICK CAREY, GAVIN BELL, CHRIS MARRIN: The Virtual Reality Modeling Language (VRML 97). ISO/IEC 14772-1. http://www.vrml.org/Specifications/VRML97

Proposal B.
Create a „mixed aesthetics" rendering of the Studiengalerie

Problem. A photorealistic rendering of the gallery may be considered a nice exercise in graphics. But from an aesthetic point of view, and also in the context of developing a style more appropriate for digital media, a mixture of minimalistic illustration and realistic rendering of selected details appears appropriate.

Objective. The Studiengalerie should be rendered in such a way that details like door-knobs or screws on display panels, or similar (i.e. detail that has nothing to do with the main function of a gallery) are rendered in perfect photorealistic manner, whereas large areas of importance to the gallery function of putting pictures on display, should be left largely untouched, or illustrated by sparse strokes and similar features of non-photorealistic kind. We still keep this proposal rather open because it needs some reading, discussion, and inspiration, which we hope to develop jointly with a group of students.

References

LEIF ARNE GENZMER, OLIVER GRAF, EVA-SOPHIE KATTERFELDT: Künstliche Kunst im virtuellen Raum – ein digitales Museum für Georg Nees, Bachelorreport, FB3, Universität Bremen, 2004 (we will make parts of it available)

THOMAS STROTHOTTE; BOOCH & BOOCH (will be provided)

Proposal C.
Allowing for lighting experiments, typical for a gallery

Problem. Galleries have walls and mobile display panels, or other possibilities, to hang pictures. Pictures are often accentuated by pointing spotlights on them. The Studiengalerie should possess an unlimited supply of lights that may interactively be placed anywhere in the room and adjusted to will.

Objective. Enhance the model of the Studiengalerie such that it allows for complex lighting possibilities. Make use of photon mapping to accomplish this task.

References

HENRIK WANN JENSEN: Realistic Image Sythesis Using Photon Mapping, A K Peters, 2001

Proposal D.
Implement a path tracer for rendering both models, MERZbau & Studiengalerie

Problem. Given is a geometric model as a sequence of triangles or polygon meshes, or as a VRML scenegraph. OpenGL uses the PHONG local reflection and shading model, so surfaces are not always shaded satisfactorily and global light transport is not modeled at all. The radiosity reflection approach, as implemented by Christian Lauterbach, consumes too much memory and produces some nasty lighting artifacts.

Objective. Implement the following path tracer, which should better be capable of generating a realisticly looking light transport. For every light ray shot from a light source, follow exactly one possible light path through the scene and store resulting surface colors in an appropriate datastructure. For every ray/surface intersection in this model, exactly one new ray is generated from the set of possible reflected rays. The probability of the chosen ray direction depends on the bi-directional reflection distribution function (BRDF) of the surface. We restrict calculations to perfectly diffuse light reflections only, because they are view independent. When the whole path tracing process is finished, store resulting surface colors in light maps attached to polygons, and store the entire scene into a new VRML file containing the light maps. – Time permitting, extend the GLUT viewer to show the produced VRML file in real-time.

References

ANDREW GLASSNER: An Introduction to Ray Tracing. Academic Press, San Diego, 1989.

JAMES KAJIYA: The Rendering Equation. Proceedings of SIGGRAPH'86, 143–150

RICK CAREY, GAVIN BELL, CHRIS MARRIN: The Virtual Reality Modeling Language (VRML 97). ISO/IEC 14772-1. http://www.vrml.org/Specifications/VRML97

Proposal E.
Rendering the polygonal model with reduced aliasing artifacts

Problem. We again use the geometric model as a list of triangles. Rendering generates some annoying artifacts due to rasterization and unmotivated popping colors.

Objective. The geometric model is to be cleaned, inserted into a BSP-tree, and rendered with polygon antialiasing. Cleaning may be done by collapsing neighboring vertices into one, but also by insertion into a BSP-tree. This data structure allows for efficient depth sorting of faces, so that they can be processed by the painter's algorithm. So it becomes possible to draw polygon edges in an antialiased manner.

Possible add-on. Compute a balanced BSP-tree.

References
RICK CAREY, GAVIN BELL, CHRIS MARRIN: The Virtual Reality Modeling Language (VRML 97). ISO/IEC 14772-1. http://www.vrml.org/Specifications/VRML97
MASON WOO, JACKIE NEIDER, TOM DAVID, DAVE SHRINER: OpenGL Programming Guide. Reading, MA: Addison-Wesley 1999
JAMES FOLEY, ANDRIES VAN DAM, STEVEN K. FEINER, JOHN F. HUGHES: Computer Graphics. Principles and Practice. Reading, MA: Addison-Wesley 1990 (BSP-tree!)

Proposal F.
Implementation of the REYES architecure for rendering

Problem. Given is the geometric model as a sequence of triangles in world coordinates. Former implementations render this model without any cinematic qualities like antialiasing, motion blur, depth of field, or transparency. The REYES architecture is capable of efficiently allowing for such effects. This is the reason why the REYES architecture became the basis of the Renderman product used in most professional movie productions.

Objective. Implement a renderer on the basis of the REYES architecture. The rendering process can be subdivided into stages: For every triangle: If it is outside the view volume, drop it. Otherwise, project the triangle onto screen space. If it is too large (i.e. the longest edge is longer than pixel size), subdivide it to more than one triangle. Otherwise, compute a color for the triangle by a simple local coloring model. (So far, recent work has already generated results.) Next displace the triangle in space for motion blur and depth of field effects. After that, rasterize it with stochastic supersampling, and record the computed color in the framebuffer if the sample passes the Z-buffer test. – Time permitting, additionally implement transparencies. For that, perform depthsorting and over-compositing in the last stage. It may be necessary to subdivide the image plane into buckets due to frame memory consumption.

References
ROBERT COOK, LOREN CARPENTER, ED CATMULL: The Reyes Image Rendering Architecture, Proceedings of SIGGRAPH'87, 95–102
ANTONY APODACA, LARRY GRITZ, BRIAN BARSKY: Advanced RenderMan: Creating CGI for Motion Picture, Morgan Kaufmann Publishers Inc., 1999