Rapid Modeling
of
Geology

MASTER DEGREE THESIS

AUTHOR:
MORTEN BENDIKSEN

SUPERVISORS:
ENDRE M. LIDAL
IVAN VIOLA

UNIVERSITY OF BERGEN
DEPARTMENT OF INFORMATICS
SPRING 2013

Abstract

Drawing three dimensional models of geological phenomena is today a process requiring training in specific programs that can be very time consuming. For illustration and communication of geological concepts, geologists therefore often limit themselves to drawing on paper or to two dimensional drawing applications.

In this thesis I propose an approach for making rapid geologic illustrations in 3D. The novel idea for the approach consists of sketch based input on a cube in order to create a layered geological structure. Further details can be added to the layers by sketching geological concepts such as rivers, mountains and valleys. Sedimentary deposits can be created through a procedural modeling approach that employs a volume preserving diffusion algorithm to simulate the flow of depositional material on top of the terrain. Awareness of the geologic domain enables a sparse amount of input strokes to be interpreted into geological structures.

Results show that the proposed approach can be used with success to model geological layers. Compared with a 2D sketch, the creation of a 3D geometry on a computer gives advantages such as perspective control, ease of making changes to the model, etc. The approach shows great promise and can be useful in many situations. A program based on the proposed approach could become a standard way for geologists to draw their illustrations.

Acknowledgements

I want to thank my supervisors, Ph.D Student Endre M. Lidal and Professor Ivan Viola for their inspiration, guidance and help. I also thank my employer Vizrt and especially head of R&D in Bergen, Tommy Nordvik for flexibility in allocation of time between work there and on the thesis. My family and friends also deserve thanks for supporting me through the whole process.

I would also like to thank everyone at the Department of Earth Science who has collaborated with me throughout working with this thesis: Professor William Helland-Hansen for for his early evaluation of the approach and his help in finding further collaborators. Master student Marie Songve for her help in assessment of techniques and advice during the research and development and her help in organizing the user study. Everyone who participated in the user study. Professor Haakon Fossen for providing geological illustrations.

This work is a part of the GeoIllustrator research project, funded by the Statoil Academia Agreement and by the Petromaks program of the Norwegian Research Council (#200512).

Contents

Chapter 1

Introduction

The Earth is constantly changing. We have all experienced this through natural events such as earthquakes, volcanic eruptions, tsunamis, avalanches and floods. When we study such changes in a longer time frame, they are even more dramatic. In geologic time, oceans are formed and then they disappear, mountains are created and different forms of life become extinct, while new life forms develop. Through much of human history big events were explained by appeal to the supernatural. The study of such changes is today the work of a discipline called Earth science. Earth science started in the 19th century, but is founded on scientific fields of inquiry much older than that. Illustration is an important mechanism for earth scientists, in particular geologists and their aspiring students when they are trying to understand these processes that have shaped the Earth and how these processes are still continuing to this day.

1.1 Problem description

It is a common practice to make sketched geological models by hand on either paper or computer. These sketches are used in both professional and educational settings, and facilitate communication and understanding. Geologic phenomena are four dimensional in nature since they occur over time in the three spatial dimensions. There are many techniques and standards for illustrating these phenomena in a two dimensional drawing. One can for example sketch three dimensional phenomena by using perspective drawing techniques, but the model is still confined to the 2D nature of the medium. These techniques and standards can also be limiting as they require significant time and training to master and understand. Before I started working on this thesis a problem was identified; there did not exist any tools aimed at helping geologists sketch 3D models for illustration purposes. On the computer it is already possible to make 3D models in traditional modeling approaches. However, existing tools are often complex, aimed at creating advanced and detailed models, and usually requires training to understand and use. It is from this background that the goal of this thesis was formed.

1.2 Goal

The goal of this thesis is to enable the rapid creation of 3D models of geologic structures by creating an approach that lets geologists quickly specify input in an intuitive way that is easy to learn. The model will be used for illustrative purposes to facilitate communication between geologists by letting them create sketched models quicker, help lecturers explain concepts to students by creating models that can be changed interactively, and reduce the need for artistic skills and long training for students to master illustration techniques.

Earth Science is a big field of study with a lot of different sub-fields. Sedimentology, the study of sedimentary layer structures are perhaps the field of study that has resulted in the most knowledge about the history of the Earth. The study of the processes that deform such layers and other rock structures is called structural geology. These two fields of study are both important for in the search for and recovery of natural resources like oil and gas. The Earth itself also consist of a layered structure of minerals. Because of the importance of layered structures in geology, I will concentrate my effort around the creation of rapid modeling techniques for geological layer structures. The aim is to create an approach for the creation of such structures.

1.3 Approach

To reach the goal of rapid geologic modeling, I propose to employ a sketch-based input for modeling the geological layer structures. The sketched strokes are input via a mouse pointer by the user on the computer screen, and projected onto a transparent cube. Layered geological structures are often sketched in a cube, and I therefore propose to mimic this technique for the sketching interface. The user can rotate around the cube and sketch on the four vertical faces of the cube. On the faces the user sketches the outlines of a surface that will be the top boundary of one of the layer volumes. The surface is then interpolated between the sketched outline. In geology, a surface the is called a horizon. The horizon is also what separates two layers stacked on top of each other. I therefore use the top horizons of previously drawn layers as the bottom boundary of new layers. The user can thus create a stack of layers by adding the layers from bottom to top.

In order to change and model details on the layers, I propose methods for drawing further structure features such as mountains, rivers, valleys and deposits. The user can create ridges, rivers and valleys by sketching on the layers. Separate algorithms for each of the features will then modify the layer surface on which it was drawn. The features the user sketches are positioned on the 2D manifold of the surface it was sketched on, such that a change in the underlying layers representation can be made without having to redraw or manually reposition all the features that exist on that layer. Deposits are created by a procedure that distributes material from the point where the river meets the sea. The material is distributed by a volume preserving diffusion algorithm that considers the topology of the underlying layer surface to create a plausible flow of material from the river.

1.4 Outcome

In the proposed approach it is possible to draw layers in seconds. The sketch in Figure 1.1 shows the cube with a layer structure created inside. On top layer an imagined coastline with fjords an islands have been created by sketching mountains and setting the sea level. For every new layer drawn, it is defined as the volume below its sketched horizon surface minus the volume of previously drawn layers. If the user sketches a horizon surface that intersects with any previously defined layers, the newly created layer will be defined only in the area between the already existing layers and the sketched surface. This way of sketching and defining the layers allows a layer structure such as the one in the figure to be completed in one minute. Other features such as the mountains seen in the figure can also be rapidly created on the layer surface.

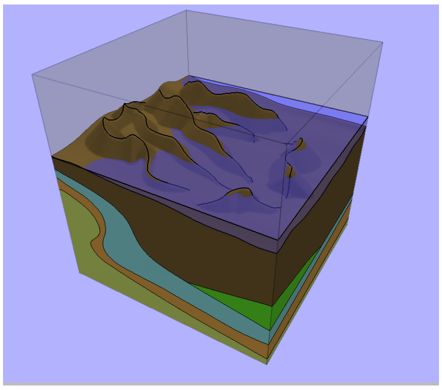


Figure 1.1: Sketched layer structure with an imagined coastline sketched on the top layer.

Ridges, rivers and valleys can be sketched on the layers to modify the layer surface. Ridges can be used to illustrate mountains quickly by letting the user input a sketched curve where the ridge should be placed, and then specify with another curve along the length of the ridge how much the underlying layer surface should elevated. Rivers and valley are also input quickly by sketching a curve that represent the path along which they will run. Their width can then be modified along this path by sketching new outline curves along either side.

Deposits can be created in the sketch by a procedural approach. The deposit is created by indicating the height of the sea level, and then selecting a river where the deposited material flows from. The user can see the deposition level in real time. When a desired amount of material is deposited, the flow is stopped by the user.

1.5 Benefits

The proposed approach enables geologists to create sketches that are useful for communication and educational purposes. They can be created rapidly and with minimal effort. The sketching input is intuitive and quick to learn. Once a sketch has been made, all parts of it can be modified easily. Any layer surface outline can be changed or deleted and the other layers that have been sketched will be redefined to fit. The perspective can also effortlessly be changed. Compared with traditional 2D sketches this lets the geologist and students focus on the geological aspects instead of their sketching technique. The user can thus start sketching without committing to a perspective, the colors she uses, the exact structure of each layer etc. This frees up time to think about the ideas that are to be illustrated instead of how it will look in the end, since the user knows that changes can be made at any point with minimal need for recreating the entire sketch.

Saving to files and loading from files on the computer enables sending of the files over the internet. Sending files to each other lets geologists working at geographically separate locations cooperate on the creation of a sketch by incrementally adding modifications to it. An export feature also enables usage of a finalized sketch in other programs giving the possibility for further modifications by mature 3D modeling techniques once the basic structure has been created. The export feature also makes it possible to use rendering techniques of other programs to give the sketch a different look.

1.6 Work process

This section starts by acknowledging all the help my supervisors have given me through the whole period I have worked on this thesis. Next, an explanation of how the first idea was formed and how the initial research proceeded is presented. Then a description of how collaboration with geologists and geology students helped the progression of the project is given. The section ends by describing the development of the approach itself.

1.6.1 Supervision

Through all my work on this thesis I have received supervision and help from my supervisors Ph.D. student Endre M. Lidal and Professor Ivan Viola, both of whom are affiliated with the Visualization group in the Department of Informatics at the University of Bergen. Endre acted as a daily supervisor, and Ivan as a senior supervisor. My supervisors are the ones who identified the need for an approach to modeling geological structures for illustration purposes and suggested this as a topic for my thesis. They have also given me a lot of advice and guidance regarding research, planning and writing. I have learned a lot regarding these skills during my work, and I owe that to my supervisors. Many of the ideas regarding the approach I have developed have come from discussions I have had together with Endre and Ivan.

As work progressed I frequently presented my results and findings to my supervisors. We discussed what would be the next step and planned for when to meet again. Usually we would meet every other week. Regular meetings motivated me to work, especially in situations when I had difficulties with finding a solution to a problem that needed solving in order to continue. All in all the interaction with Endre and Ivan has been invaluable.

1.6.2 Inspiration and research

As mentioned earlier in the chapter, work started with the need for a tool to rapidly create three dimensional sketches for geological structures. Various existing geological illustrations were investigated and there were some discussions about what would be the goal of the thesis. At the beginning however, there was not a clear idea of how to approach the problem. Research therefore started by reading relevant geologic papers and books, both to get an idea of what kinds of sketches were needed, what was actually to be sketched and to get a little bit of an overview of what geology is.

A large portion of the sketches encountered in literature were drawn inside a cube shaped cutout. The cutout of the geologic structure along a primitive and easy to understand object like a cube, gives the viewer a good way of understanding how the different structures relate to each other. This was how the idea of the sketch cube, which is the starting point of the approach presented in this thesis, began.

To make a final decision to proceed with that approach, a more detailed strategy was needed. Therefore a new research phase began, were techniques dealing specifically with sketch based input, procedural modeling, terrain modeling, geological modeling etc. were researched. A couple of example illustrations that were found in a geological research paper [30], were used as a goal for what was

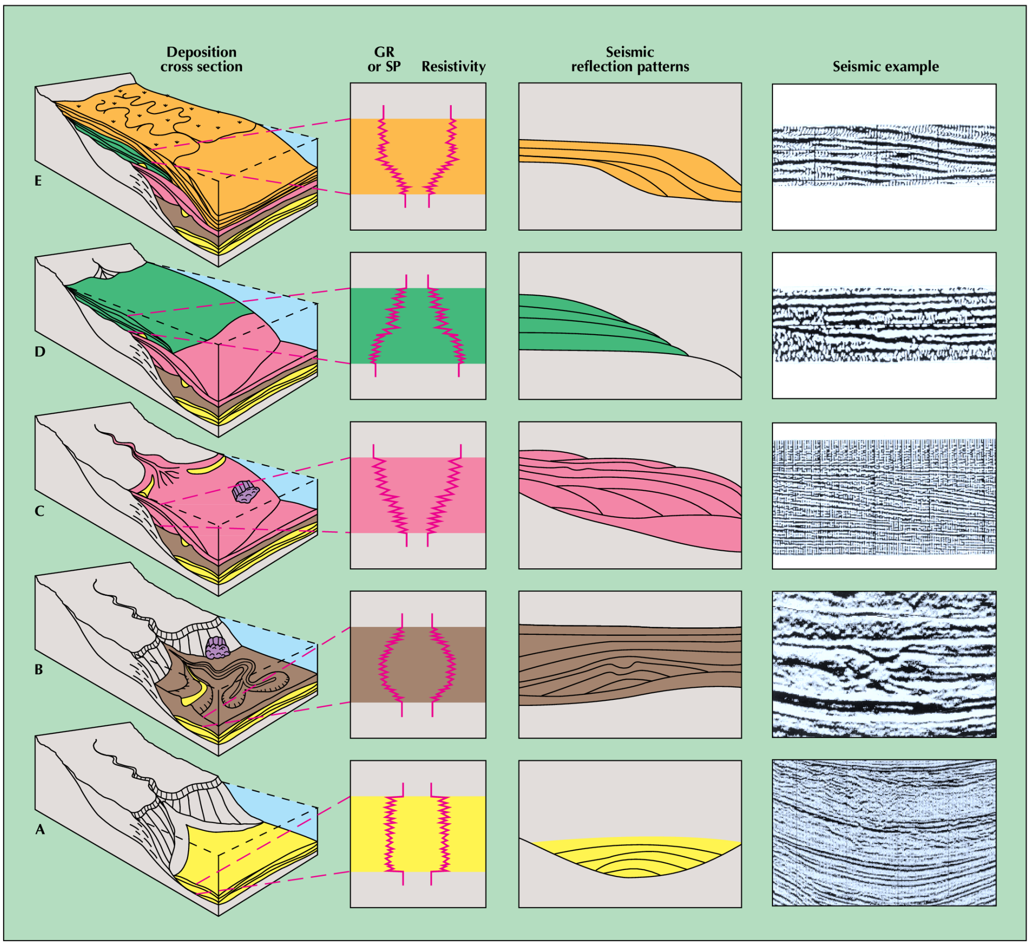


Figure 1.2: Example illustration that served as inspiration in the beginning of research. Image from Sequence Stratigraphy - A Global Theory For Local Success, by Neal et al. [30].

possible to model in the approach. Figure 1.2 shows these illustrations, which depict several depositional rock layers at the different times when they were deposited. The sea level at which they were deposited is indicated by the dotted lines. A set of different geological phenomena could be seen in these examples. Further research of modeling techniques gave some possible ways to specify the input of features such as rock layers and rivers that could be seen in the illustrations. I particular the input techniques from the approach proposed by Gain et al. [16] were inspirational. An idea of how the finished solution could look and function from a user's perspective was forming.

To explain the idea, illustrations and explaining text were put in a project description document. Figure 1.3 is an example illustration from that document

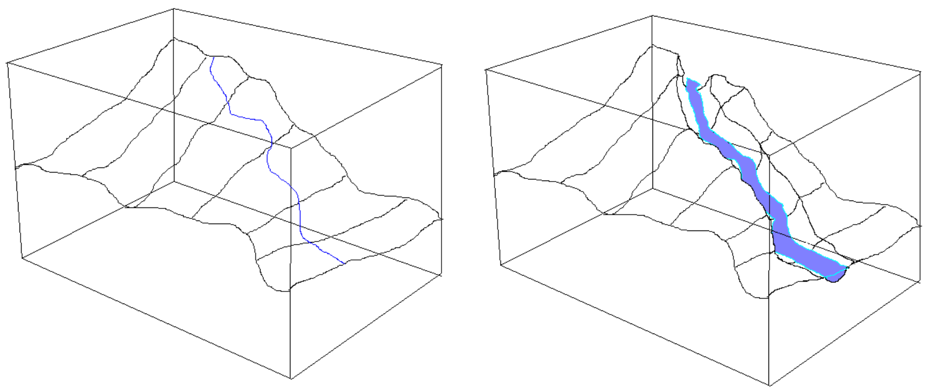


Figure 1.3: Illustration of how the drawing on rivers were planned to work at the beginning of the project, before implementation started. First, drawing a line on a surface where the rivers should go. Then, a river is created by an algorithm.

showing how drawing of river structures was planned. The complete document can be found in Appendix A. This was the basis for the implementation work at the beginning of the project. A lot of further research and clarification of ideas was still needed, but since the general approach had been decided, the only open point remaining before development of a concrete approach could begin was to decide on technology.

1.6.3 Collaboration

After having created a version of the program that could be used as a prototype for demonstration, a Professor from the Department of Earth Sciences was contacted. He also put me in contact with a cooperating geology student. The rest of research and development was guided by the collaboration with them.

The basic idea of the solution was shown to the Professor of Earth Sciences. He showed great interest and believed this could be good approach for illustrations in geology and he had several constructive suggestions for improvements. What we learned from this meeting was that the focus on the ability of creating appealing sketches rapidly was the most important criterion. There were also some other ideas and features suggested for further development. These included illustrating the sea level, illustrating vegetation and many more ideas. Features that I had already thought of, like mountains and rivers, were confirmed to be important also.

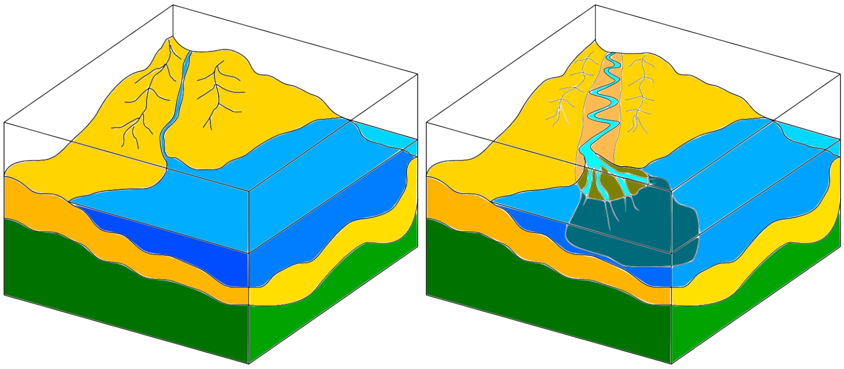


Figure 1.4: Sketches that were made in collaboration with a Master student in Earth Sciences.

To help assessing the development and suggest more improvements in the future, he gave the contact details of one of his Master students in Earth Sciences, Marie Songve.

The Master student of Earth Sciences helped to develop some sketches that would be the main focus of the development (see sketch in Figure 1.4) from that point on. The sketches would serve as a guide for what should be possible to create in the final approach. She also helped to evaluate the direction of the development from this point. In addition to giving concrete advice on the geologic features of the approach, observing her testing the input methods helped to identify some of the user interface problems. After development was concluded, Marie also helped in assessing the success of the project by giving her impressions and trying out the program by creating several sketches on her own. This helped to create some of the results that can be found in Chapter 5 and in evaluating the approach in Chapter 6.

1.6.4 Implementation Work

This section explains the development of the approach without going into much detail about the specific solutions created. The only details included are those that are relevant for understanding the most important choices which are discussed here and which are not explained earlier. For more details about the specific solutions, please refer to Chapter 4.

Implementation work has started by creating the cube and a camera system that would allow rotating around and zooming in and out of the cube. This was achieved relatively quickly. Intersection tests and input on the cube was the next step. At first this was achieved by simply looping through all the triangles and storing each intersection point in a list, and then drawing lines between the points that were drawn. However, the next step was the real challenge, which had not really been considered in depth until now. That challenge was how in the world to create a meaningful layer from lines drawn on the faces of the cube. Different approaches were assessed before ending up with the intuitive and useful solution that is included in the proposed approach. This research challenge ended up taking a lot more time than was initially expected. However, in order to focus on the solution, one has to know the problem, so all the work that might seem like a waste was actually needed in order to realize what kind of solution to look for.

Once a solution to the layer problem was found, there were still a lot of features needed to create a program that could be used for making useful sketches and geological models. Meetings with Marie helped focusing the direction and prioritization of wished functionality implementation. All the different features that were planned in order to reproduce the initial pen and paper sketch, were scheduled to be implemented one after the other. Very often introduction of new capabilities meant going back and modifying existing code. In addition to the geological features there were the general functionality of the program to consider, such as undo functionality, save and load possibilities etc. These also had to work together with the other features in a way that would not conflict with any of the other features. As new features were added, it sometimes seemed as if the complexity was growing exponentially. It quickly became a problem keeping the different functionality working together in a smooth and predictable way. The program code was therefore changed many times attempting to abstract the complexity to make the implementation of new features as easy as possible.

It was while realizing the complexity of the initial approaches that an idea of a parametric and ordered hierarchical structure was introduced. In the start, everything was based on three dimensional coordinates, which made it difficult to calculate the relationship between for example the points on a layer and a river on it. If the layer changed, what point on the new surface would correspond to the beginning of the river? The user would have to draw the river again, which was not a good solution for a rapid modeling tool. The idea was that parametric coordinates would restrict the placement of a feature to two dimensions on its parents surface. Thus, a new point in three dimensions could be calculated based on the parents position and geometry. The inspiration for this approach came from the paper by Schmidt et al. [38]. This would ease the recalculation of coordinates once something changed. The ordered structure meant that each feature would only have to deal with immediate parent and children, and the order in which they had been drawn would be the guiding way to solve conflicts that might arise. The thought behind this was that the user would be thinking of the structures in a hierarchical way already, and after familiarizing herself with the program would know in what order features need to be drawn to create the desired effect. After all, it does not make sense to have a river without first having a surface on which it flows.

The parametric and hierarchical solution could now be used for many problems. For example, how does one specify the height of a ridge and how does this height change if the layer changes, and how and where does it affect the underlying parent? The simple solution was making the ridge baseline a parametric coordinate on the layer, and for each point along the baseline, there was associated a height value, which could then be used to compute the actual resulting top point of the ridge by first finding the position of the baseline point on the surface, and then adding the height. How to do the actual morphing of the underlying surface to create the ridge also became a lot easier, since the lookup of surrounding points now was restricted to two dimensions and the baseline point was already known in this space.

There were also other difficulties remaining to solve to reach the stage the program is in now. However, the rest of the development was more about smaller issues local to each problem. These problems and their solutions that are relevant to understanding choices will be covered in Chapter 4.

1.7 Outline of Thesis

Following the introduction chapter a presentation of the relevant geologic background and terminology is given in Chapter 2. The Geologic Background chapter also includes many examples of sketches made by geologists to illustrate different geological phenomena. A report on the state of the art in the fields of Computer modeling and Geological modeling as they relate to this project follows in Chapter 3.

An explanation of the methodology of the proposed approach is given in Chapter 4. The chapter starts with a conceptual overview of the proposed approach and then gives the specific algorithmic solutions of the proposed approach. Resulting illustrations that were produced in the implementation of the approach can be seen in Chapter 5. Results of the user study that was conducted are also presented in that result chapter. In Chapter 6 I evaluate the expressive power the final solution gives the user for illustration purposes. In Chapter 7 I present my wishes and recommendations for further development of the proposed approach. In Chapter 8 I present a summary of the thesis.

Chapter 2

Geologic Background

This chapter gives a basic introduction of geological concepts. Focus is on those aspects that are needed to understand for the rest of the chapters in this thesis. The information that follows is based on the book "Geologi, Stein, mineraler, fossiler og olje", by Haakon Fossen [15].

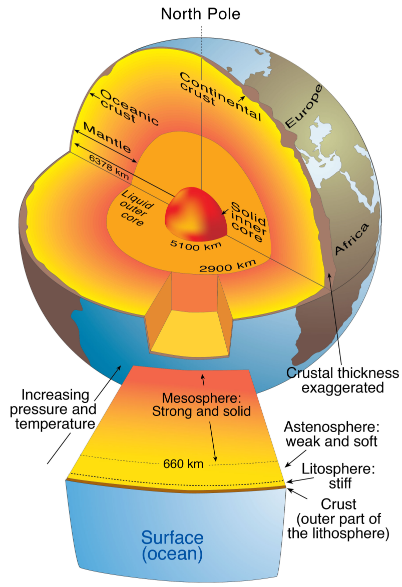


Figure 2.1: The structure of the Earth. Illustration courtesy of Haakon Fossen.

Geology is a complex science, and this explains why there are so many subfields within geology. Just a few of the fields will be mentioned here. Geomorphology is the study of landforms and the processes that shape the surface of the Earth. Sedimentology is about how particles are transported, where they are deposited and how they are compressed into rock. Structural geology is the study of how the rock layers and crust is deformed by various movements. Tectonics is closely related to Structural geology and describes the movement of Earth plates and how that causes the formation of mountain ranges and basins.

Geology is the study of the Earth. The main purpose for the geologists work is to gain an understanding of the structure of the Earth and the processes that shape it such as how mountains are built and how the oceans form. Most geologists usually concern themselves with only the surface and the part of the Earth that is called the crust, the top 30 to 40 kilometers. Only rarely do they get into contact with what is deeper down. However, what happens there is still important to understand

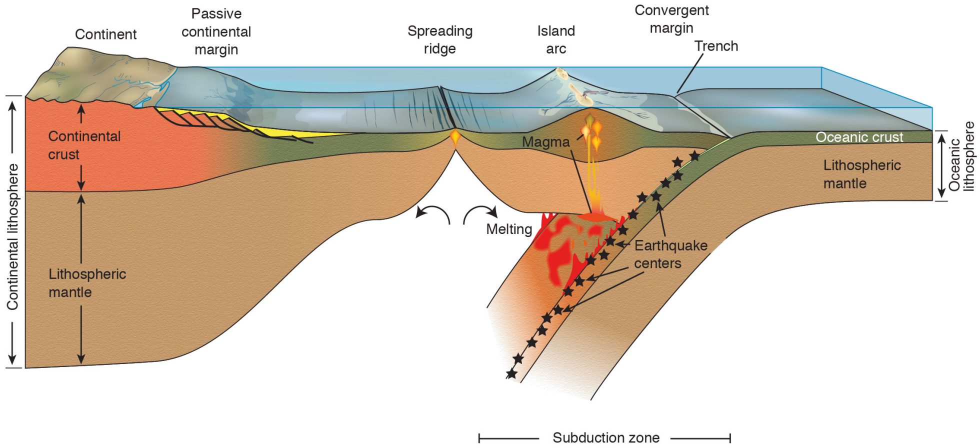


Figure 2.2: Subduction and creation of oceanic plates. Illustration courtesy of Haakon Fossen.

the processes that occur in the crust.

Figure 2.1 depicts the structure of the Earth. The core of the Earth is over 4000 °C. It consists of an iron-nickel alloy. There is an inner core that is solid due to the high pressure, which raises the melting point, while the outer part of the core is liquid. Between the core and the upper parts, there is a thick layer called the mantle. The mantle has a malleable consistency like butter or dough, although it is less malleable the closer it gets to the core. The mantle is thus able to move around without building tension and forming cracks. The upper part of the Earth is called the lithosphere. The lithosphere consists of the upper part of the mantle and the crust and they are is solid. This is why, when the lithosphere is moved by geological forces, large amounts of tension can build, resulting in earthquakes.

The Earths crust consists of several plates that undergo constant change in relation to each other. They collide with each other, glide across each other, or separate from each other. Figure 2.2 shows what happens at the point of contact and divergence. Because of the large forces involved when the plates move around, the crust is constantly being changed by the plates being subdued underneath each other and made anew by magma flowing up at different locations around the Earth at the place of divergence between the plates where spreading ridges form by the molten rock that flows up. The plates are not only made of the continents, but also consist of the oceanic crust and lithosphere. The oceanic plates are made at spreading ridges at the points of divergence and is thinner and heavier than the continental crust. Therefore the oceanic plates float lower in the mantle. At the point of impact between plates, one of the plates is pushed down into the hot and softer parts of the mantle, and starts melting. The soft parts of the mantle lie beneath the lithospheric mantle, and is not draw in the illustration. The molten rock pushes through the existing plate above and creates new land masses.

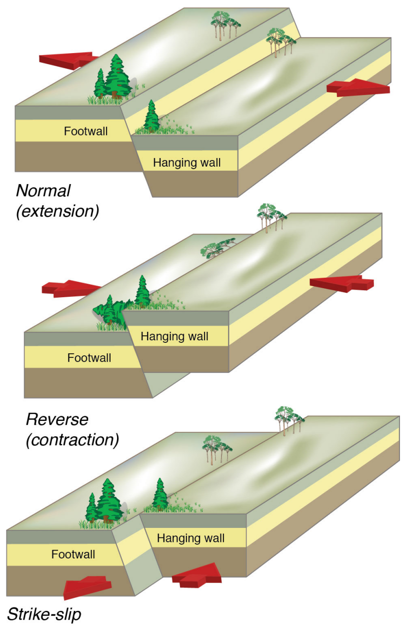


Figure 2.3: Different types of faults. Illustration courtesy of Haakon Fossen.

Because the plates move around, large tensions can build up in them. When the tension gets bigger than what the rock can withstand structures such as folding, stretching and fracturing will appear. Such structures can occur where the plates meet, split apart and glide across each other. Folding and breaking structures can also occur in other places where tension might build. Different types of rock might react differently to tensions according to their mineral composition and the temperature when the deformation occurs. When rock reacts by breaking apart, fractures are created. There are two types of fractures, faults and cracks. Cracks have very little movement along the fracture direction, and occur all over the Earth. They can form when rock is lifted and compacts due to the falling temperatures. Faults are fractures where there has been considerable movement along the fracture direction. Figure 2.3 shows different types of faults that can occur. Instead of breaking apart and creating a fault, the rock might also fold, bend and stretch, creating a shear zone. They look similar to faults, but are wider, and instead of the discontinuity of the broken rock have a zone where the rock is folded and stretched along the movements direction. Such shear zones are created in warm and deep parts of the crust, where rock becomes softer.

In the deeper parts of the crust it is also usual for rock layers to fold. An image of folded rock can be seen in Figure 2.4. Folds are very frequent in metamorphic rock types. One way common folds are created, is by compression of layers, where the forces are working in the direction of the layers. Another way is by shearing movement. However, folds can also occur closer to the surface, especially in sediments that have yet to complete the metamorphosis to firm rock.

The age of the Earth is about 4.6 billion years. Geologic processes that shape it take a very long time and still continue today. Geologists need to understand what processes took place during this time, how geologic phenomena have been and still are interacting with each other over time, and how structures that can be observed, related to each other in the geologic time scale. Often layers can be observed to be stacked on top of each other in which case it is easy to understand that the youngest layers are on top of the older ones and a ordinal time scale can thus be mapped for these structures by simple observation. Even when layers have been distorted, an understanding of the processes involved, makes it possible to establish such an ordinal time scale, often referred to as the relative geologic time scale. By radiometric dating, on the other hand, one can establish an absolute time scale. This is possible by studying the decay time of radioactive materials. This makes it easier to relate different structures around the world in a time scale.

The geologic time scale, as illustrated in Figure 2.5, has been developed by studying layer structures around the Earth and dating them. The scale shows the relative times of different processes, but as radiometric measurements of the different structures have been taken it has become possible to date absolutely the events that created these structures. In a similar way to how a historic time scale can be divided into stone age, bronze age, iron age, middle age and so on, the history of the Earth is also divided into eras and time periods. The closer we get to our own time, the more details are known and included in the geologic time scale. Fossil records also give useful information about the relation and absolute position in time of structures found at different places, without having to perform a radiometric dating every time.

Figure 2.4: Folded rock. Image courtesy of Haakon Fossen.

If we disregard water, organisms and soil, the surface of the Earth is largely made of rock. Rocks are made out of different minerals and come in many different varieties. Minerals are chemical elements bound together in certain ways. For a compound to be called a mineral it must be solid, have a particular chemical composition and a special crystalline structure. The mineral composition of rock is an important aspect that geologists study when examining geologic structures. There are three types of rock. Igneous rock, sedimentary rock and metamorphic rock. Igneous rock forms when magma (molten rock) crystallizes into solid form. When this happens in the deeper parts of the crust, where the temperature is high, crystals have long time to form. When crystal forms slowly, the individual crystal forms become bigger than when they are rapidly cooled down. If the magma reaches the surface before it crystallizes, we get smaller crystal forms, and thus a more fine grained rock.

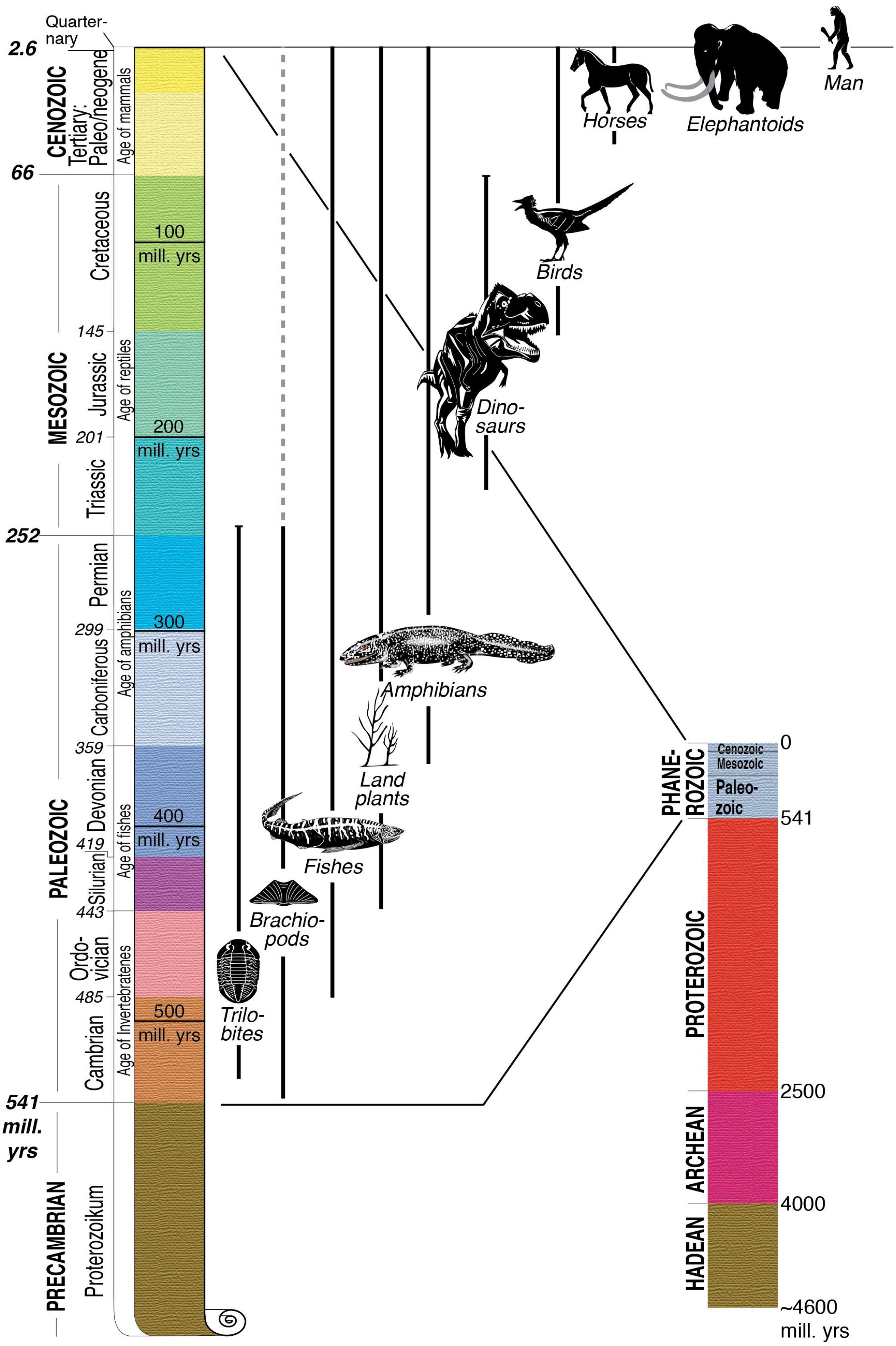
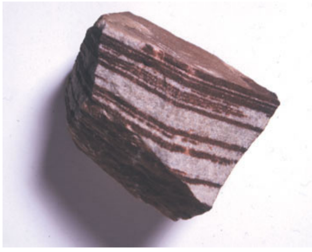


Figure 2.5: The geologic time scale. Illustration courtesy of Haakon Fossen.

(a) Igneous rock (Gabbro).



(b) Depositional rock (Sandstone).

(c) Metamorphic rock (Quartzite).

Figure 2.6: Pictures of rock types. Images from Wikipedia in the public domain.

Sedimentary rock forms when sediments such as clay, sand and gravel are transported and deposited over time and solidify into rock after it is deposited. Sedimentary rock cover about 75% of the continents on Earth. When either igneous or sedimentary rock gets under high pressure and temperature, they start to change their mineral composition and deformation. They then become what is called metamorphic rock.

Many of the processes that can erode rock and deposit material elsewhere can be seen in Figure 2.7. The study of erosion, deposition and how this creates sedimentary rock structures is the most important way geologists have developed scientific knowledge about the history of the Earth. When rock is exposed to the weather, it is eroded away and becomes loose particles of clay, sand or gravel. It is transported by water, wind or ice until it settles at some other location. These forces are constantly trying to flatten the features of the Earth. Mountains are eroded away, while basins and valleys are constantly being filled with material. All the different processes that deposit material will create layers of different kinds of sedimentary rock. These layers are called strata. Figure 2.8 shows such strata once created in a delta system by a river.

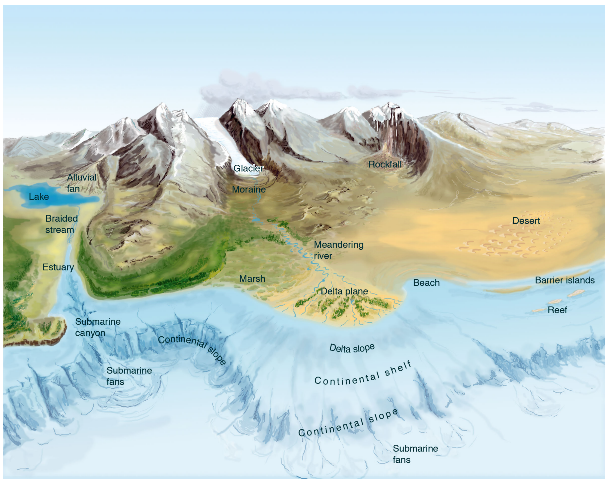


Figure 2.7: Different processes that deposit sediments. Illustration courtesy of Haakon Fossen.

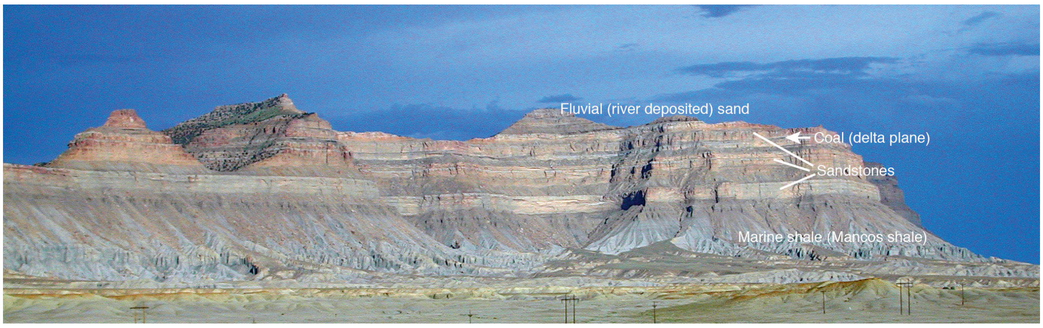


Figure 2.8: Delta deposits in Book Cliffs, Utah. The deposits have formed layers, or strata of different rock types. Image courtesy of Haakon Fossen.

Rivers will form from precipitation and spring water, and as they flow trough, the terrain will be carved out and eroded, while at the same time the river is transporting material downstream. This process creates valleys in the terrain where the river runs in the bottom of the valley. As the river carves out the terrain, it creates a V-shaped valley, more pronounced farther up the river, and considerably wider at the bottom. As the river carves lower, parts of the valley sides will also start falling down, also making the valley wider. In the lower parts, the river will at some point not be carving out any more, but rather only deposit material from the upper parts thus flattening the landscape over a long time. Rivers can flow in different patterns.

Where the terrain is steep, they will flow more straight and where there is enough of water, they can flow in a braided pattern covering a large area. If the terrain is flatter rivers usually follow a meandering pattern, flowing back and forth in meanders. Meandering rivers carve in the outer sides of meanders and deposit in the inner side of meanders. Thus they will move around in the valley where they flow. Sometimes a meander gets cut off, leaving an abandoned meander.

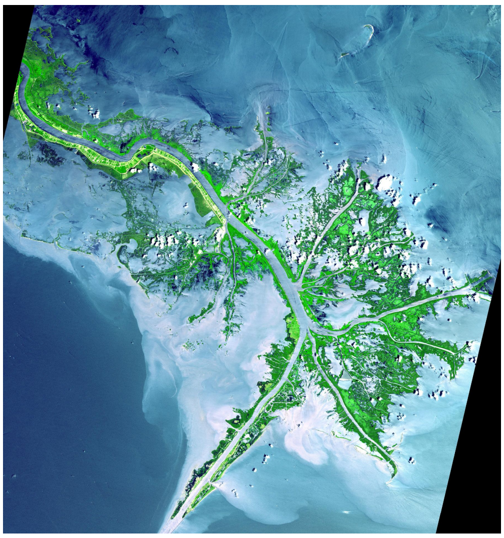


Figure 2.9: The Mississippi river delta. Image from Wikipedia, in public domain.

Where the river meets the sea, it will create a delta deposit system. A delta is formed by the deposition of sediments carried by the river. Since the river is no longer confined to its

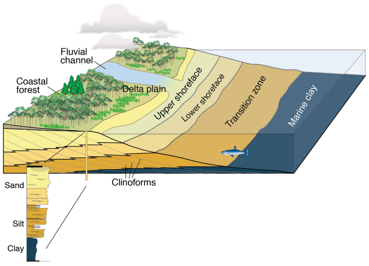


Figure 2.10: A delta depositional system. Illustration courtesy of Haakon Fossen.

valley the flow will spread out and the flow velocity will drop. This means that the flow can no longer carry sediments, and they will deposit. As sediments are carried out into the sea, they tend to create S-shaped deposits called clinoforms which can build outward into the sea.

As the river deposits the material, the slope of the river will therefore decrease as it builds outward in the sea. As the slope decreases the river path will tend to get unstable, and the river finds a new way to reach the sea after a while. In this way the river path changes and distributes the deposits. The delta can take many forms depending on the circumstances. Figure 2.10 shows the structure of a possible delta system. Figure 2.9 shows a picture of the Mississippi river delta.

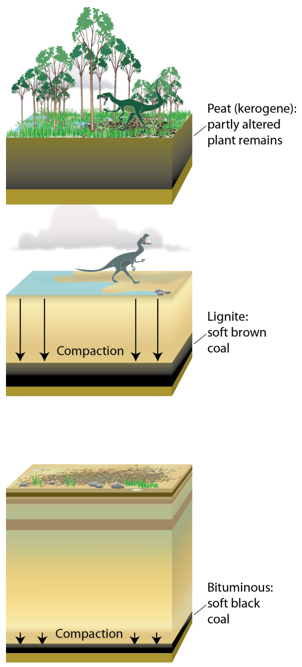


Figure 2.11: Making of coal. Illustration courtesy of Haakon Fossen.

When looking for resources, many different insights from the different fields of geology are important. Coal, oil and gas are examples of very important resources where all necessary knowledge is used to make the search and recovery efficient. Figure 2.11 shows how coal is formed from biological material being compressed and heated. When it is buried, gases and liquids are forced out, and chemical reactions happen. As time passes, the carbon content in the coal rises.

Oil and gas is formed from organic material like plankton and algae that is buried without access to oxygen. It takes millions of years to develop when there is the right temperature. Figure 2.12 shows how, when it is mature, the oil and gas will migrate upwards. If it hits certain structures in the rocks layers it can get trapped there. Such areas are known as oil traps. The study of geologic layers can therefore help in determining where oil and gas can be trapped.

When looking for such resources, very often a seismic study will be made.

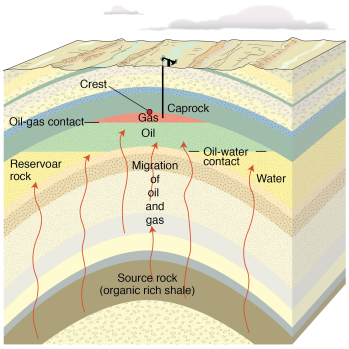


Figure 2.12: Migration and trapping of oil and gas. Illustration courtesy of Haakon Fossen.

Simply explained, this consists of sending sound waves down into the rock and listening for the echoes. At the boundaries of layers some of the sound will be reflected. By recording how long time it takes for the sound waves to reach the boundary and get reflected back to the surface , and by repeating this procedure at different locations, it is possible to create images of the horizons underneath the surface. Another technique for gathering data on the subsurface is by boring holes and taking physical samples or measuring other properties of the rock directly. Interpreting the data to determine where resources might be found is one of the concerns of geologists and geoscientists. Many technologies exist to help the geologists and geoscientists carry out this interpretation to create a representation of the subsurface.

In this chapter we have seen many different illustrations of geological structures. Geologists frequently need to make such illustrations and models of what they are researching. Geological modeling tools is one of the topics that are described in the next chapter, which outlines the state of the art of computer geological modeling.

Chapter 3

Related Work

This chapter explores the field of 3D computer modeling as it applies to geologic modeling with special focus on techniques for rapid modeling. For reference and technical background in the field of computer graphics the book "Real Time Rendering" [26] has been used. The book "Curves and surfaces for CAGD: a practical guide" [14] was used to understand curve and surface theory.

3D models can be made manually, procedurally, or as a combination of both. A traditional way to create 3D models on the computer called computer-aided design or CAD, usually consists of modeling and compositing different kinds of solid objects to create precise models of industrial designs, buildings etc. One commercial program of that type is Autodesk AutoCAD [4]. Another type of modeling consists of editing vertices and control points of curves and surfaces to build geometry meshes of all kinds shapes. This is often used for movies, games and other graphics purposes. A commercial example of that is Autodesk 3D Studio Max [3].

Models can also be represented in various ways. Many modeling programs use a polygonal representation, meaning 3D vertices, connected by edges, forming polygons. These are easy to visualize and manipulate for the computer hardware. Others use a mathematical representation of solid objects. There are well established ways to create detailed and sophisticated 3D models with these kinds of representation. A lot of research has been done on this type of modeling over the years and much progress has been made. However, many of these techniques require a substantial amount of training and the modeling often takes a lot of time. When creating models that do not need perfect precision, such as models for rapid prototyping, illustrative models, models for communication etc. this can be prohibitive. In recent years there has therefore been substantial research effort put into developing alternative methods for user input for enabling more rapid 3D modeling.

3.1 Rapid 3D Modeling

Sketch based input is one way of achieving simpler and more rapid modeling input. With sketch based input the idea is to carry out 3D modeling with a method that is made to feel more like traditional pen and paper sketching as opposed to editing vertices and edges directly. The SKETCH system was one early development in this area from Zeleznik et al. [46]. The tool from Zeleznik et al. was a gesture-based input method for rapid modeling of simple 3D geometric shapes that could be composited using constructive solid geometry (CSG). The Teddy tool from Igarashi et al. [19] later expanded on these ideas by freeform sketching for 3D models where the user draws the silhouette of objects, and they are created automatically. The tool from Igarashi et al. made it easy for even first time users to make simple, expressive figures (see Figure 3.1). An approach by Brazil et al. [8] also show promising results for modeling soft looking objects by sketching variational hermite-RBF implicits. Digital sculpting is an alternative method of input for easing 3D modeling, where the modeling more resembles real world sculpting by applying pressure, pinching and similar methaphors. Z Brush by Pixologic is a commercial modeling program that uses this approach mainly for charachter creation [40].

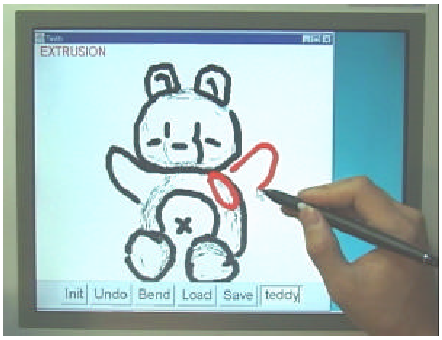


Figure 3.1: The Teddy sketching interface from Igarashi et al. [19].

SketchUp [17], a tool originally made by Last Software, lets the user sketch 2D objects and pushing or pulling their geometries in any direction to extrude 3D objects from the 2D sketches. Many other operations, such as subtraction and addition of objects, texturing, etc. is also possible. By using this approach it is possible to create a large amount of 3D geometries in a way that is quick and intuitive compared to traditional 3D modeling.

Another approach for rapid modeling is procedural modeling. Fractals are one type of rules often applied to create terrains procedurally. However, any algorithm that focuses on creating a model from a rule set, rather than editing the model via user input can be called a procedural method. Parish et al. [32] use a procedural approach for modeling cities while Müller [27] et al. desribe an approach to modeling buildings procedurally.

3.2 Geologic Modeling

In geology, models are used for understanding and communicating about phenomena relating to the structure of the Earth and how it changes over time. At the first international conference for 3-D geoscientific modelling, held in 1989, Dr Brian Kelk defined the requirements for subsurface characterisation and modelling: “The industry requires a system for interactive creation of spatial and spatio-temporal models of the physical nature of portions of the Earth’s crust. i.e. the capability to effectively model and visualise:

- Geometry of rock- and time-stratigraphic units
- Spatial and temporal relationships between geo-objects
- Variation in internal composition of geo-objects
- Displacements or distortions by tectonic forces
- Fluid flow through rock units” [43]

Traditional CAD systems have several problems when used to make geological models (Turner et al. [44]). There have been attempts at using such system to create geological models (Kelk and Challen [21]). Such experiments have shown some problems in using standard CAD systems for geological modeling. The reason for this is the characteristics of geological objects. As Caumron et al. [9] identify these include: complex geometry and topology, scale dependency and hierarchical relationships, indistinct boundaries defined by complex spatial variations and the intrinsic property heterogeneity and anisotropy of most subsurface features. The gOcad tool decscribed by Mallet [24], however, has been developed to make a CAD approach for geomodeling, by basing the modeling on a new interpolation method called “Discrete Smooth Interpolation”. The geometry is defined by bridging together a set of nodes with a location is 3D space and with physical properties attached to these nodes.

A realistic geological scenario will follow certain constraints. Caumron et al. gives rules for modeling that define boundaries between layers. For example, geological objects have a spacial continuity such that abrupt changes of normal orientation are not common. This is relevant for the creation of layers in the approach I propose, as it allows the use of smooth curves to represent layers. They also describe the typical process of creating a structural model. The modeling usually starts with fault modeling, then the connection between fault surfaces is defined. Finally horizons are modeled. If the fault structure is very complex, it is normal to start with horizon definition and introduce the faults after.

Natali et al. explores different modeling techniques in their recent survey paper [28]. They describe a data-oriented taxonomy where modeling is divided into three different scenarios, one data-free, one sparse-data, and one dense-data. Natali et al. also show how geological modeling trends are approaching modeling methods that have been developed in computer graphics and give an in-depth description of selected methods that can be applied for geological modeling. The sparse-data and dense-data scenario occurs when there is geologically measured data available and used as input for a modeling approach. In this scenario there exists many approaches aimed at geologic use. The data-free scenario on the contrary has no ground truth information and relies entirely on procedural and/or geometric modeling. Not many approaches for input aimed specifically at geology exists in this scenario, but relevant techniques from other fields exist. In this thesis the approach described is a data-free scenario, since the idea is to create a sketching tool. However, some approaches from the sparse and dense-data scenarios are also relevant as the user sometimes will guide the interpretation of the input data to varying degrees.

The data sparse scenario is the most common in the geosciences and is most often the result of borehole data collection. In such situations the data points are spread around and need to be interpolated, which is relevant for the approach I propose. The main interpolation methods are the B-Spline method, inverse distance method, Kriging method, and discrete smooth interpolation method [23,25]. Interpolation methods are interesting in regards to the approach I propose as they might be used to interpolate horizons from the sketched curves for horizon creation.

The data dense scenario is usually the case when data from seismic surveys is available. Here the problem is how to display the huge amounts of data, and how to interpret them to make a model of the structures present. Approaches that aim at making the interpretation process easier and quicker have emerged in recent years. Patel et al. describe techniques for rapid horizon extraction from seismic data in both 2D [34] and 3D [33]. Amorim et al. [1] have an interesting approach that allows sketching directly over the raw seismic reflection volume and its derived data to help build the structural model of the subsurface. This helps the expert in interpreting and and building a structural framework for a reservoir by using a sketch-based input for helping in the interpretation process.

Many existing tools are based on the assumption that there is extensive data available and that geologists will have a lot of time developing a model of the area of interest. Several tools exist for modeling, displaying, editing and automatically calculate parameters for geological modeling. Petrel [37] is an example of a commercial program for geologic modeling that is in use. Most of such existing tools rely on an intensive work flow, and are based on interpreting data gathered from seismic surveys or bore hole data. The modeling is done either automatically, or

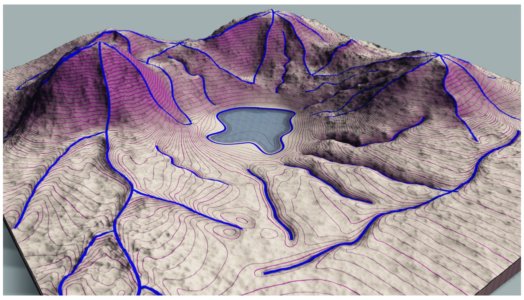


Figure 3.2: Terrain generation by Hnaidi et al. [18].

semi-automatically by letting the user indicate various parameters and alter the suggested model manually. Sometimes up to a year is spent on developing such models. Recently a need for rapid developments of geologic prospects have been identified. A lot of techniques have therefore been taken from other fields that model terrain, such as the video games industry and movie industry [28].

Procedural generation is a standard way to generate terrains. This usually happens in one of three ways: fractal landscape modeling, physical erosion simulation and synthesis of terrain from images or sample terrain. Before Olsen [31] fractal noise was mostly used to create terrain surfaces, because of computer limitations on simulating erosion processes. Olsen proposed a synthesized fractal terrain and applies an erosion algorithm on that. The representation is a 2D heightmap. Hnaidi et al. [18] generate terrain that is constrained by a set of curves that characterize the features of the landscape. The ability to make realistic looking landscapes could be interesting in a rapid sketching tool, as the generated landscape is constrained by curves that could be input by sketching.

A method for eroding terrain is described by Benes et al. [5] where a concise voxel representation is created and then eroded by thermal weathering simulation. The representation allows for caves and hole structures. The same authors also propose a method for procedural modeling of terrain by hydraulic erosion [6]. Stava et al. [41] employ an interactive physics based hydraulic erosion. The user interacts during the generation of the terrain. These erosion techniques could be applied in the approach I propose as a way to make more realistic horizon surfaces. Erosion could also be used to simulate material transportation and deposits creation, rivers, etc.

Peytavie et al. [36] propose a way to model and render rock piles and stones which are found in most landscapes without any computationally demanding physically-based simulation. Peytavie et al. also have proposed a framework for representing complex terrains with such features as overhangs, arches and caves and including different materials such as sand and rocks [35]. This is done by a discrete volumetric representation with different kinds of material and an implicit representation

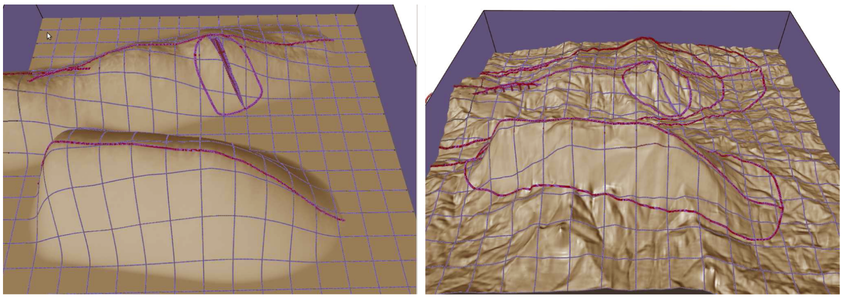


Figure 3.3: The terrain syntesis proposed by Tasse et al. [42]. Left: user sketched curves. Right: final result.

for the modelling and reconstruction of the model. To allow more complex structures in sketches, such a representation could be useful, while rock piles might be useful to illustrate avalanches or other geological phenomena involving loose rocks.

Tasse et al. [42] propose a texture-based terrain synthesis framework controllable by a terrain sketching interface. They enhance the realism of the generated landscapes by using a novel patch merging method that reduces boundary artifacts caused by overlapping terrain patches. The high computational cost of texture synthesis is reduced with a parallel implementation on graphics hardware. This approach could also prove useful to create more realistic surfaces.

Natali et al. [29] describe an approach where the user sketches the boundaries of geological layers. Then the user can sketch folding and faulting operations, and thus create many different scenarios. The input in this approach is restricted to making conceptually 2D sketches, allthough the visualization is in 3D. Projecting drawings on the 3D structure can however give some more information and context to the 3D geometry. The techniques for texturing that they describe makes their sketches carry a lot of expressive power of the internal structure of specific layers. The painting of details on surfaces also opens for many illustration purposes. This approach has been developed in parallel to the approach I propose. As far as I know, this is the only sketch based approach to modeling subsurface geological layers in 3D without measured data other than the one described in this thesis. However, Lidal et al. [22] present Geological Storytelling, a novel graphical approach for performing and presenting rapid and expressive geomodeling of a multitude of model variations in 2D that handles sketching processed over time. This approach allows the user to sketch, play back, and compare geological events as he believes they might have occurred over time.

Cockett's Visual Geology [11] and Jessell's Noddy [20] are two geologic modeling tools designed for educational purposes that allow rapid building of geological layer structures by user input of parameters. Visual Geology lets the user

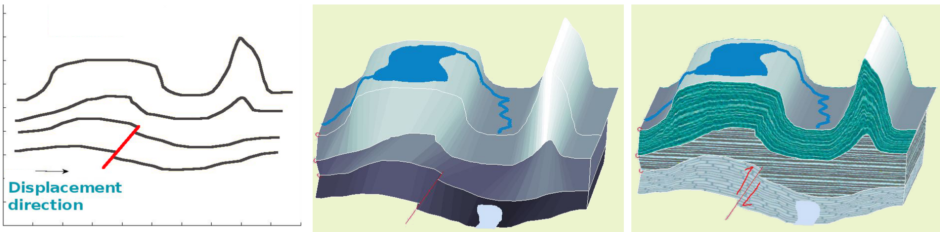


Figure 3.4: The proposed interface by Natali et al. [29].

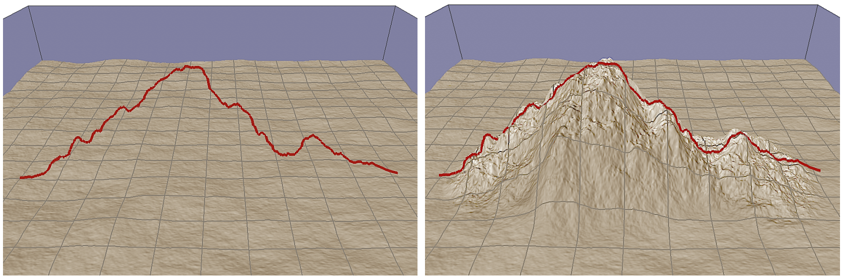


Figure 3.5: The terrain sketching approach from Gain et al. [16].

specify how many layers she wants and their thickness, and then apply various functions such as a fault line and its angle, a folding phenomenon and its angle, wave frequency etc. The basis for Noddy is the ability to construct a complex geological history as a succession of relatively simple structural, sedimentary and igneous events similarly to Visual Geometry, which allows the user to rapidly create models and then calculate resulting gravity and magnetic fields. Both of these are based on parameterized procedural methodologies.

There exists several approaches for sketching terrain, which are applicable to geology. Harold is an early example of a sketch based system that incorporates methods for sketching terrain, made by Cohen et al. [12]. In Harold, the user can sketch hills on the terrain by simple strokes that start and end on the terrain. The terrain is then warped to try and match the stroke. Watanabe et al. [45] made a further development of this, where the shape of the stroke also influences the width of hills that are generated, making for more natural looking hills. They also incorporated noise on top of the generated terrain to make the visualization more realistic. Gain et al. [16] later improved further on this by allowing the user to sketch the width of the hill and change the baseline along wich this hill runs. The approach by Gain et al. can be seen in Figure 3.5. To achieve real-time terrain creation Bernhardt et al. combine CPU and GPU processing in their sketch-based approach for generating and displaying complex and high-resolution terrains. The user can see the terrain changing as she is sketching. De Carpentier combine brushing and procedural terrain creation [13]. These terrain sketching techniques

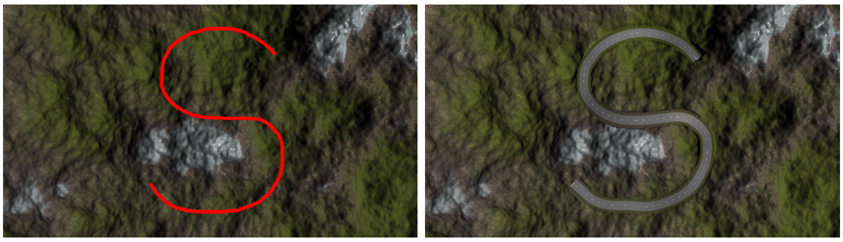


Figure 3.6: The road sketching approach from Applegate et al. [2].

are all interesting in enabling sketching of terrain features like mountains. The ridge sketching in particular is inspired by all these papers on terrain sketching.

Applegate et al. [2] have a sketch based system for highway design which is illustrated in Figure 3.6. Their tool is guided by input sketches and a combination of prioritized constraints, including the curvature of roads, their inclination, and the volume of underlying terrain that is displaced. The rivers in my proposed solution are sketched in a similar way to this highway sketching method by projecting the sketches onto the terrain and then creating geometry and making terrain modifications along the path the user has sketched. In developing my proposed approach I have taken inspiration from many of the techniques that were presented in this chapter. In the next chapter I present all the techniques which I have employed to create my proposed approach.

Chapter 4

Methodology

This chapter starts by giving an introduction and conceptual overview of the solution in Section 4.1. Section 4.2 provides an explanation of the design. A more detailed view of the specific features of the program is given in Section 4.3. The chapter concludes with Section 4.4 by describing technology choices, and supporting functionality of the program.

4.1 Basic Concept

A conceptual overview of the approach is illustrated in Figure 4.1. The arrows represent processes, either in the computer or performed by the user. The rectangles represent a form of data. The user starts with an idea in her mind of what she wants to model. She indicates what she wants to create through input using the mouse. The raw input date goes trough an initial interpretation resulting in the conceptual data. The program interprets the conceptual data, and for each feature recognized, creates a representation of it in the scene graph. The representation is then used by the geometry synthesis code, to create new geometry and alter the shape of existing geometry. This procedure is executed at interactive frame rates. Once the scene geometry is ready, it is used by the visualization code for creating an image that is given back to the user on the computer display. The user then compares what she sees with what she had in mind. She can then perform further refinement of the model by either changing some of what she already drew, or adding new features by drawing on the existing geometry.

The basic idea of the modeling metaphor, is based on a cube on which the user will draw curves that represent an outline of a surface. The cube can be viewed from all directions by means of a virtual camera that rotates around the scene. The surfaces that the user sketches represent the horizons of geological layers that can represent the strata of depositional rocks or other layered structures. By

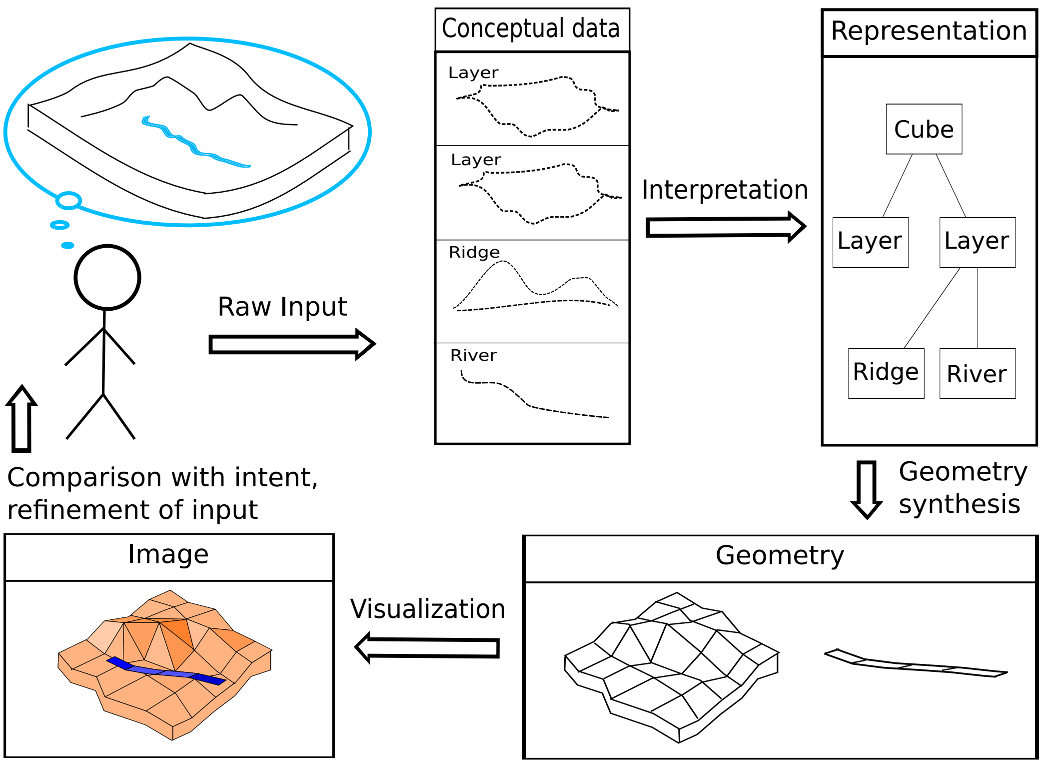


Figure 4.1: Conceptual overview.

further sketching or user input it is then possible to modify these layers to create other structures such as rivers and ridges. For the rest of the text, such structures that can be modeled by the user are called features. Most of the input consists of drawing simple curves on surfaces, and then indicate what kind of feature is wanted. However, it was an idea from early on in the project to see if it would be possible to make a combination of a sketching approach to modeling with a procedural approach. By procedural I mean that the geometry is calculated by some algorithm instead of the user sketching it or in other ways specifying its final geometric properties. A feature for creating depositional delta structures has been implemented procedurally. A deposit in the current approach is a layer of material that has been deposited by a river where it enters the sea. The geological background chapter explained how all sedimentary rock layers are created by such deposited material. All features are created based on the users input, meaning that there is no data representing real world conditions given as input to the algorithms. The users input has to be interpreted in some way. The most significant part of the input is the selecting of geometric structures in the scene, and drawing of curves on these structures. The first part of this process is to transform the screen space coordinates of the mouse into a ray pointing into the scene from the position of the camera. This vector is then used to check for intersections with the geometrical structures in the scene. Depending on which mouse button is pressed, the feature that the geometric structure belongs to is either marked as selected, or the point of intersection is added to a structure representing a curve that is visualized to the user. Some special interpretation rules are in place for different features which will be covered later. By clicking on various buttons in the user interface, the user provides information of what her input was intended to represent.

Once this basic input has been gathered and the intent of the user understood, each of the features is further interpreted. The output is an internal representation of the model. The internal representation is contained in a tree structure, where the cube is always the starting point. Every feature added will add a new node in the tree. Schmidt and Singh 2008 [38] was an inspiration for this parametric and hierachical way to represent the sketches. This internal representation does not store the 3D points where the user drew in the scene, but rather uses a 2D parametric representation. Each 3D point can be found by looking up its placement by a function defined in the parent feature. This internal representation is used later in the process to compute a 3D geometry based on triangles, which can then be visualized on the computer screen via the graphics card. The user will utilize this generated geometry as visual feedback, and decide whether it is close to what she intended, or whether she needs to give additional input or change her input to achieve what she intended to. Thus the process continues until a satisfactory result has been reached. Now, the user can store the work for later, send it to someone else, or take screenshots for use in a paper, lecture etc.

The purpose of the internal representation is to capture the modeler's intent as well as possible. From a technical viewpoint, it is the the meaning of the input that is interesting and stored for the representation. How the geometry is presented in the end is interesting for the user, but capturing the intent enables a modeling approach that lets the user to go back and change earlier features in the model without having to redraw everything, since all the geometry can always be recomputed from the internal representation.

The available features are layers, rivers, ridges, valleys, deposits and a sea level indication. All of the different features can be combined to create a scene inside the cube. Figure 4.2 depicts an example of a scene with all the features used. There are three layers, where the bottom two are intersecting. On the top layer there are several ridges drawn. A river runs down a valley into the sea. Where the river meets the sea, several deposits have been created. On both the layers and deposits surfaces, the colors have been set manually. The layers are created by drawing on the cube, while rivers, ridges and valleys are created by sketching a path along which they will run on top of the layers surface. The sea level is simply indicated by specifying its height on the cube with the mouse. The deposits are created procedurally by selecting a river, and then indicating that a

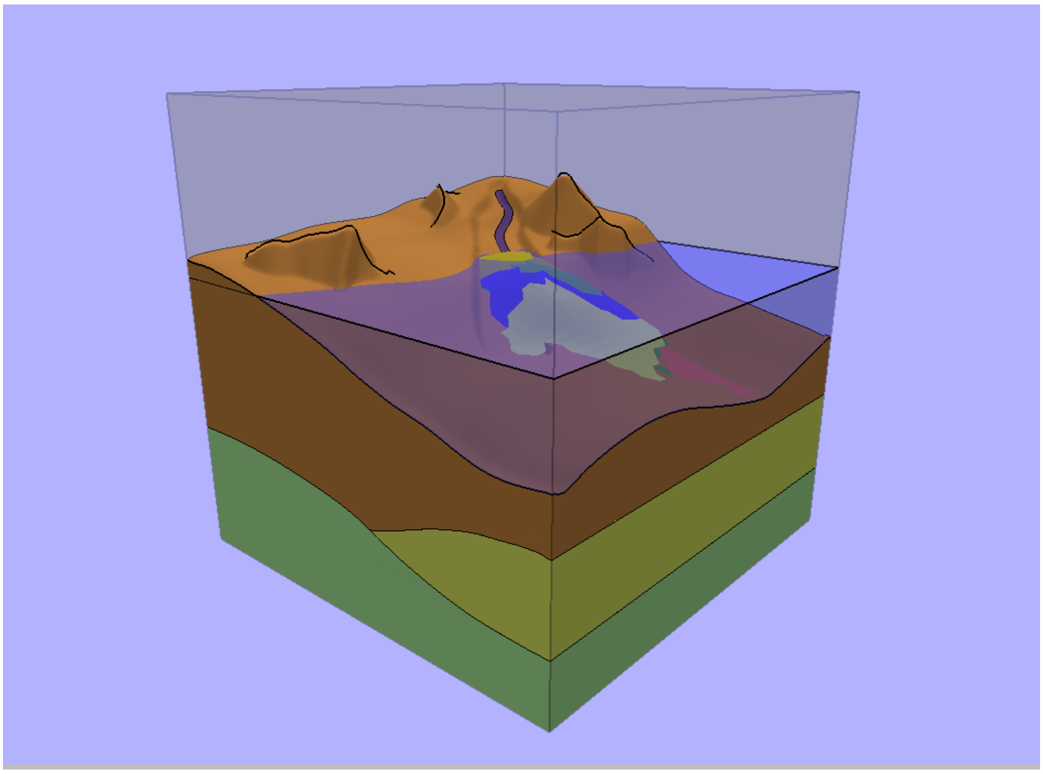


Figure 4.2: Example of a scene showcasing all supported features.

deposit is to be made. The deposit will be made at the intersection of the river and the sea level.

Other features of the solution are there to make the program more useful but does not directly result in a change in the scene. These are things like undo functionality, save to file and load from file. An export functionality enables the scenes geometry at any point to be exported to allow opening it in other modeling programs.

4.2 General Approach

In this section we will explore the concepts that all of or most of the features have in common. After follows a more in-depth explanation of each specific feature and how they work algorithmically.

The user interaction starts with input parameters that need to be interpreted to create a representation of the geological structures the user intends. Thus the first step is to interpret the users input.

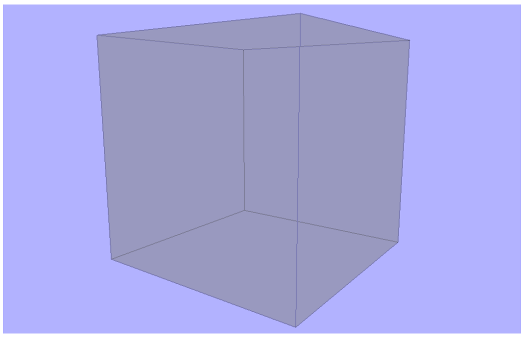


Figure 4.3: The initial state is the empty cube.

4.2.1 Sketch input and interpretation

The initial state for input is the empty cube (Figure 4.3). At this stage the input consists of the user rotating the camera around the cube and drawing on the cube to create layers.

To capture the users input it is first necessary to map the location of where the user draws on screen to the cube. This is done by projecting the screen space coordinates onto the geometrical model. In Figure 4.4 an illustration of this process is given. There is a structure for each object in the scene that contains all the triangles that it consists of. Each of the vertices of the triangles are stored together with a two points that serve as the parameters that uniquely represent the point on the 2D space of the surface of the object. This is called a parametric coordinate in the rest of the thesis. The drawing of points on surfaces is achieved in the same manner for all objects in the scene.

While drawing curves on a surface the 3D intersection points in the scenery space is stored in a list in order to immediately visualize it to give the user feedback. The parametric coordinates are stored in a separate list and is what is actually used later for the representation.

When drawing on a screen you are limited to the resolution of the screen. This means that the input points that are gathered will also be limited to this resolution However, because the actual surface where you are interested in drawing exists in a point in space farther away and not on screen, moving from one pixel to the next, means you will move a much greater distance on that surface than on screen, creating jaggedness. Also, depending on the angle and distance of the camera, the jaggedness you get will be uneven. For this reason we need to smooth the input points that have been gathered.

The smoothing is achieved by regarding the n points of the input as the control points of a n-dimensional Bezier curve. This procedure runs on the list of parametric coordinates of the points of the curve. The points that are drawn on

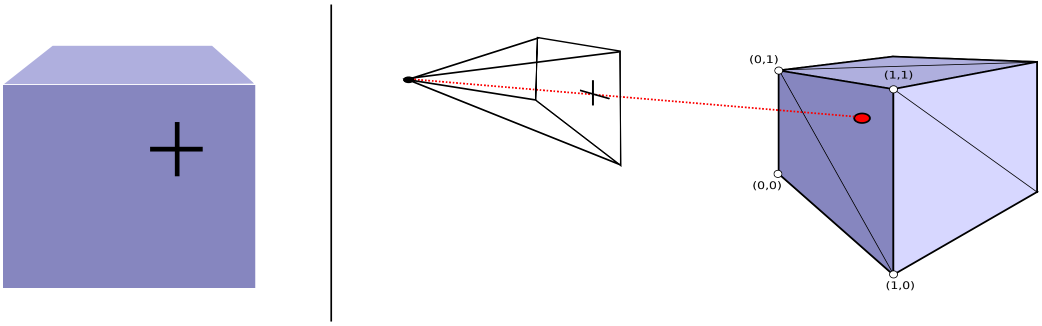


Figure 4.4: Illustration of intersection. Left: the view of the user with a black cross representing the mouse cursor. Right: The point of the mouse cursor is projected onto the objects by creating a ray from the camera through the cursor and checking for intersections with objects in the scene. The numbers indicate the parametric space values of the vertices of the surface.

the screen follow a zig-zag pattern, and this pattern is exaggerated by the projection into the scene space. The further away the camera is from the surface being drawn on, the more pronounced this will be. The Bezier curve will approximate the control points, but will lie somewhere between them as illustrated in Figure 4.5. Because most of the points lie on either side of the intended line we want the actual line that is drawn to lie somewhere between, approximating the intended line of the user, while keeping it smooth.

An alternative considered for the input, was to use a Hermite spline such as the Catmull-Rom spline. However, this approach did not give satisfactory results, as the control points were too noisy. Since the Hermite spline interpolates all the control points, this gave too many unwanted artifacts as the line would have to wiggle around to achieve that.

A similar but more advanced approach can be found in the stroke capture section of Cherlin, 2005 [10], and that is where I got the idea for using bezier curves from. In this approach, the Bezier curves order is reduced before using it by reverse Chaikin subdivision, which separates high frequencies. By discarding them it lowers the number of control points and makes the evaluation faster. I found it sufficient and fast enough to perform a simple evaluation of the input points directly. I did not spend more time on improving the input interpretation.

It is also possible to oversketch the lines that are already drawn. The oversketching principle was mentioned in Chapter 3. The oversketching methods utilized are different according to the context and what the current task is. When drawing the layers on the cube, the points of the new line are inserted into the original line. This happens by finding the points in the original line that are closest to the first and last point of the new line, removing the points between and inserting the new points. While changing a ridge height, for each point that is

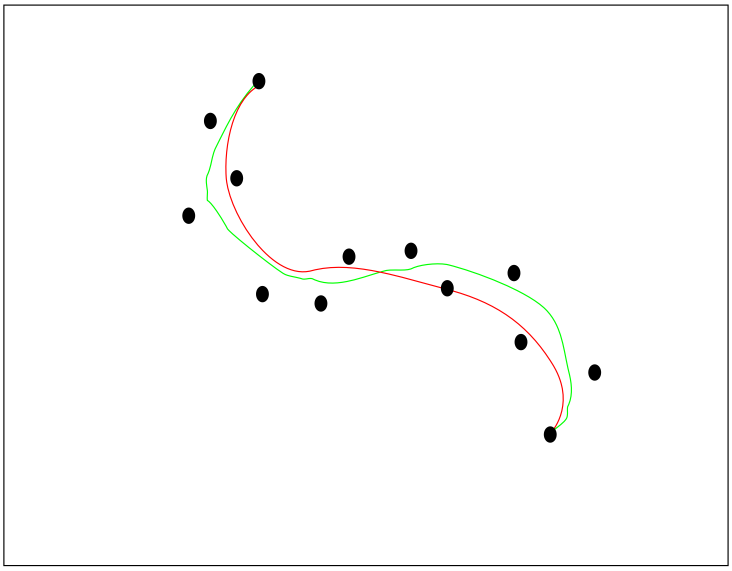


Figure 4.5: The green line indicates the mouse movement and how the user would expect the line to appear. The black dots are the actual points gethered at the surface from the intersection tests. When using these points as input for a Bezier curve, we end up with the red line.

input the old height at that location will be set to the height of the new point immediately. When changing the sides of rivers and valleys, the procedure is similar to the layers, only the lines are also smoothed after insertion. The details of each of these will be explained better in Section 4.3 under the relevant feature explanation.

Once the input has been interpreted, the new data (or changed data) will be stored in memory in a representation specific to the kind of feature the user has indicated she wants.

4.2.2 Representation

The different features that can be drawn are represented in an internal representation before creating the structure that can be visualized. Relevant parts of this representation is also visualized to the user in a way that she can understand it. This makes it easier to make changes, and reason about what can be changed and what effects that will have on the final result.

The curves are stored as a list of points. On a higher level of abstraction, most of the features that can be drawn are built by using such curves in different combinations and different interpretations. In many cases the curves are augmented by some additional information, such as the height of ridges. With the deposits

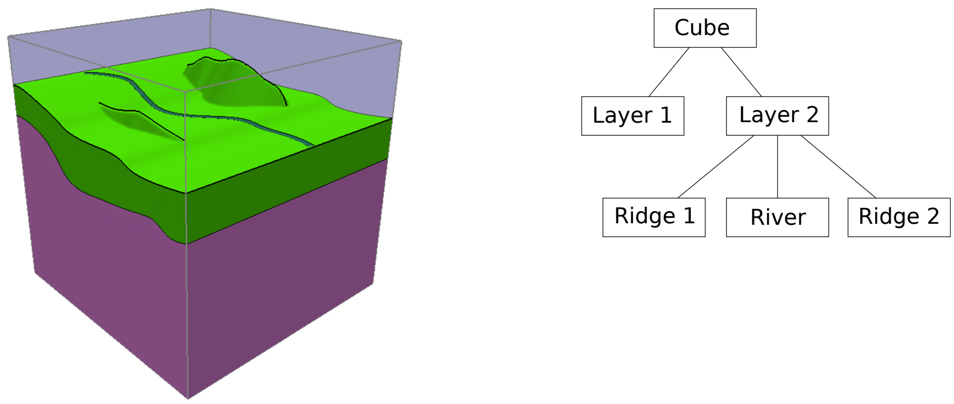


Figure 4.6: The scene is represented by a tree of nodes, where each nodes geometry and position is calculated according to the relationship with its parent. On the left we see a scene where there are two layers drawn. On one of the layers there are two ridges and a river drawn. This results in a tree structure like the one on the right.

however, the representation does not include lines at all but the shape is rather defined by a procedural method.

All the features relate to each other in a child-parent relationship creating a tree structure. The cube is the top node in this tree. All layers are children of the cube. All the other features are then the children of a layer, as can bee seen in Figure 4.6. This structure together with the parametric representation is useful to enable incremental refinement of features, meaning that any part of the whole structure can be modified at any time, without having to redraw every part that relates to that change. When a node changes, it knows whether it needs to tell the parent and a parent knows whether it needs to trigger some recalculation in a child. Such notifications are however only sent along and used for telling the geometry synthesis to do necessary recalculations, since the parametric representation itself does not actually need to change.

4.2.3 Geometry synthesis and Visualization

Before a scene configuration can be visualized, a geometry needs to be synthesized. For each type of feature in the scene, there is a corresponding algorithm for creating the relevant geometry. When the layers geometry is being constructed, it needs to take into account the children. This is because some of them can actually change the surface of the layer. Each node in the scene will store its geometry so it does not need to be generated over and over for each frame. When there are changes made to the representation of the node or any other that will affect its appearance, the geometry needs to update. It will be notified through a dirty-flag mechanism, which triggers the relevant algorithm for generating new geometry.

The visualization is achieved by painting the geometry object associated with each feature to the screen. The geometry object consists of a data structure that contains the vertices and for each vertex an associated normal. This class is also responsible for intersection testing, and thus each vertex also has an associated parametric coordinate. The parametric coordinate of the intersection point is interpolated from the three vertices of the triangle where an intersection has been found. Simple OpengGL functions are used for drawing the triangles. There is also color vectors associated with the geometry object to enable different material colors for the different objects. To achieve a transparency effect, care must be taken to draw the transparent objects last. The cube will render all its children before it draws itself. The layers is rendered before its children thus the rivers transparent water will be drawn after. The geometry object also has a list of points that it uses to draw polylines if present, intended for illustrating the sketched lines of features in the scene such as the outline around horizons of the layers, a line tracing the top of the ridges, etc.

4.3 Specific solutions

4.3.1 Cube

There is no input to change the representation of the cube itself. The cube is represented by the size of its three dimensions, height, width, and depth. The cube also has a position, although this is always the origin in the current approach. The geometry of the cube is constructed by creating and positioning six square faces with sizes that correspond to the height, width, and depth of the cube and oriented in such a way so as to build a representation of the cube. Each cube face is constructed by two triangles and the rectangle outlined by a line strip. When visualizing the cube geometry, only the triangle of the three faces furthest away from the camera are rendered, but all lines are shown. This effect is achieved by front face culling. It preserves the transparent appearance of the cube while avoiding any occlusion of the geometry inside. The cube can also be made invisible like other objects, which can be useful for making screenshots where the cube is not necessary.

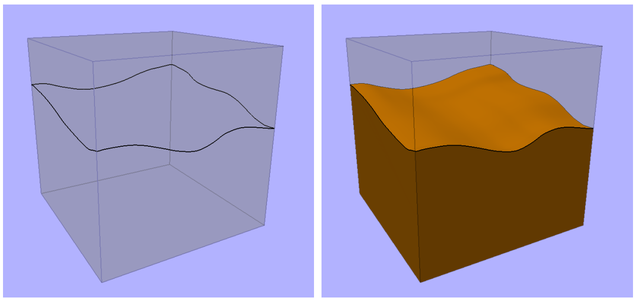


Figure 4.7: The input and creation of a layer.

4.3.2 Layers

Layers represent geological strata. A horizon is the boundary that separates two layers or the surface of the top layer. A layer can be created fast and easy by sketching the outlines of the horizons on the cube. An algorithm creates the layer by using the horizons of the layers below combined with the new horizon, and filling in the area in between. The newly sketched horizon becomes the top of the surface of the new layer. In Figure 4.7 we see how a layer is made by first drawing the four curves that outline a horizon on the side of the cube, and then letting the algorithms create the layer surface. A surface like this can be created by the user in seconds.

The layer can also be edited. When editing, the old curves are shown with a red stippled line while the new line that is the result of oversketching is drawn with a black solid line, as can be seen in Figure 4.8. Once done editing, an algorithm will compute new geometry for the layer based on the new curves.

New layers may be added on top of existing layers as seen in Figure 4.9. If the new layer intersects with previously drawn layers, only the part that is above will be drawn, as seen in Figure 4.10. The horizon surface is simply drawn with depth buffer checks enabled. The geometry of the other surfaces of the layer needs to be synthesized by another algorithm.

A Layer is a volume represented by the four curves the user has input on the cubes faces to create a horizon and implicitly the horizons of the previously created layers. The order in which the layers are drawn when multiple layers are created is important in defining their relationship. A new layer is defined to be that part of it that sticks above the previously drawn horizons. To create geometry from this representation, the four curves from user input on the cube is used to create a surface representing the top horizon of the layer. Then, for each of the sides, the users input on this side and a precomputed line that represents the bottom of the layer (that is, where it meets the layers below) are used to create a surface for the side of the layer. This bottom curve is actually given by the previous layers

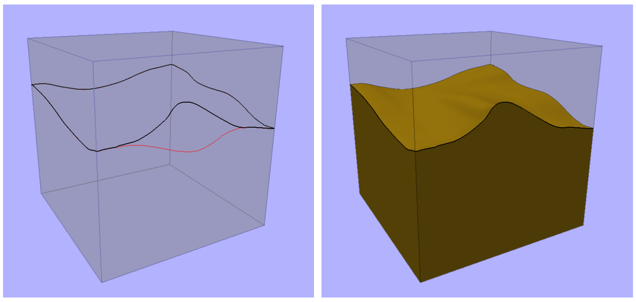


Figure 4.8: Editing of a layer. The left hand picture shows the old curves in red, and the new curves in black. On the right we see the resulting layer after editing.

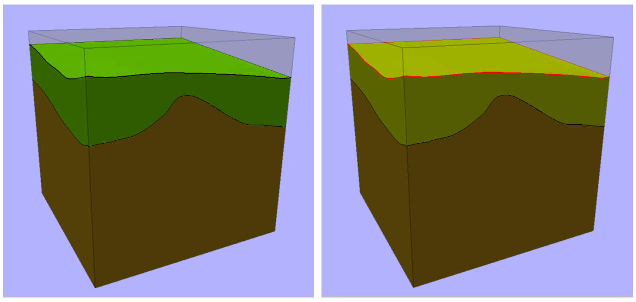


Figure 4.9: It is possible to create multiple layers as can be seen here. When one has multiple layers, it is also possible to select one of the layers by right clicking it, as seen on the right. When a layer is selected, it is highlighted by a slightly brighter color, and its outline is drawn in red.

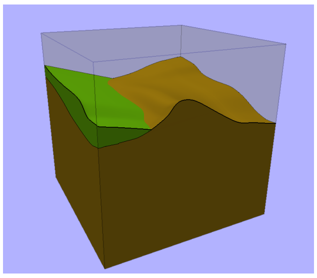


Figure 4.10: Layers might also intersect with each other. In that case what is above of the last drawn layer will be what is visible, while what is below is not part of the new layer.

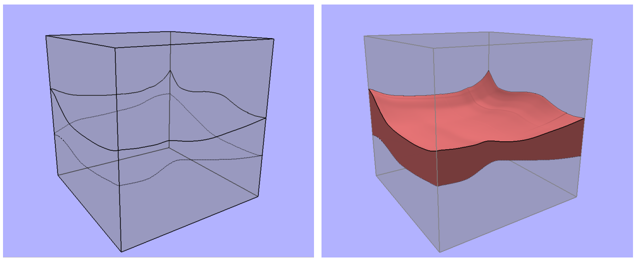


Figure 4.11: The layer is represented by the four curves the user draws, plus the lower line that represents the top of all previously drawn layers. The layer geometry is generated between these two outlines.

that already exist, and therfore all previous layers need to be taken into account to be able to compute the layers geometry. It is kept temporarily once it has been computed though, in order to avoid unneccessarily recomputing it. In Figure 4.11 you can see the curves of the layer representation and the computed curves of all the layers below.

A considerable amount of time was spent on figuring out how the horizon surfaces would be created. The first attempt was to simply loop through all the points and draw triangle strips beginning with the first and last point, then second and second to last, and so on. This created some structure that could resemble a layer, but the connection of points was very arbitrary and therefore it was difficult to predict the results of what was drawn. It was also difficult to modify it further. The second idea was to perform a simple interpolation of the points on the cube faces for each point, by going from left to right, back to front of the cube along the curves drawn on the cubes faces. Before this could be achieved, it was then clear that the four sides of the cubes needed to have separate structures and detection of which face was being drawn on. This was achieved by creating a separate structure out of each face of the cube. Each face also keeps its own structure of the input points that the user inputs.

Input for layers is drawn on the cube faces. On faces where the user draws, there is an auto-complete function while on the left, right and opposite side there is a suggestion function. The simple auto-complete will automatically complete a line the user draws by extending it towards the left and right side of the current face, by simply adding points at the beginning the and end horizontally until the edge is reached.

The suggestion algorithm is different according to the state of the other faces. If there has been no user input on the opposite side of the cube, it will automatically mirror that input to that opposite side of the cube from where she is drawing, and then extends lines between the first and last point of the curve on the current face to the first and last point of the curve on the opposite face (see Figure 4.12).

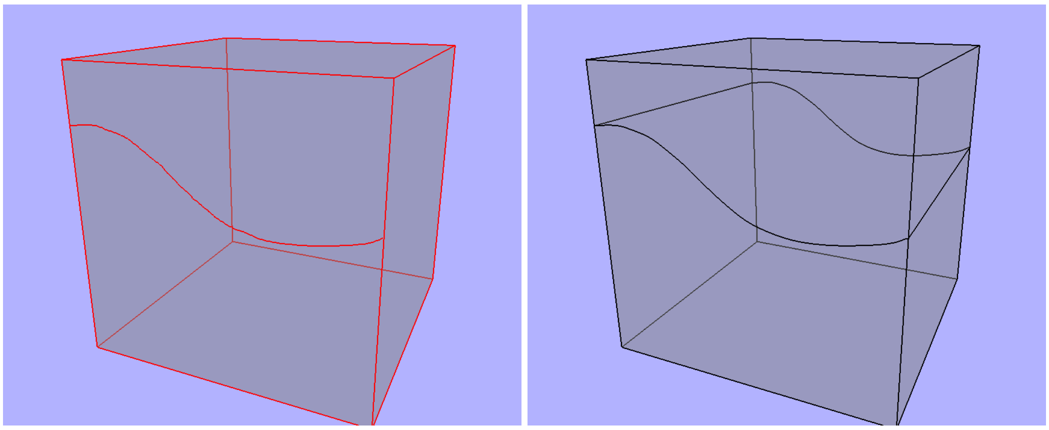


Figure 4.12: When the user draws the first curve on a face of the cube, it is replicated on the opposite side, and then lines extended on both left and right hand sides.

These straight lines will end up on the faces to the left and to the right of the face the user is currently drawing.

When further changes to this initial suggestion are made, what will happen depends which side has been drawn on directly by the user already (see Figure 4.14). If the opposite face has already been drawn on by the user, it will not be changed. If the left and right side has been drawn on, they will be modified so that the leftmost point of the new curve aligns with the rightmost point of the left hand face, and the rightmost point of the new curve aligns with the leftmost point of the right hand face. If nothing has been sketched there will simply be drawn a straight line between the curves on the adjacent faces.

The alignment of an existing curve is illustrated in Figure 4.13. The alignment of the curve is achieved by constructing one line from the first and last existing points of the curve and a second between the new position where the first and last point should be. For each point along the curve, the distance to the first constructed line is added to the corresponding point on the second constructed line, yielding the points of the new aligned curve. The suggestion algorithms ensures that the lines on all the faces are at all times connected at the edges of the cube such that they are always ready to create a layer from.

The direct interpolation of a layer from the curves did work to create a surface structure, and it was a robust solution for creating surfaces where further modification like adding ridges would work. The problem was that it did not approximate each point as it had been drawn on the cube faces, and thus it was difficult to predict what was needed to input in order to get the desired result (see Figure 4.15). A solution where the user could expect the layer surface to pass through the actual points she would draw was desired to achieve the goals of being a tool for rapid

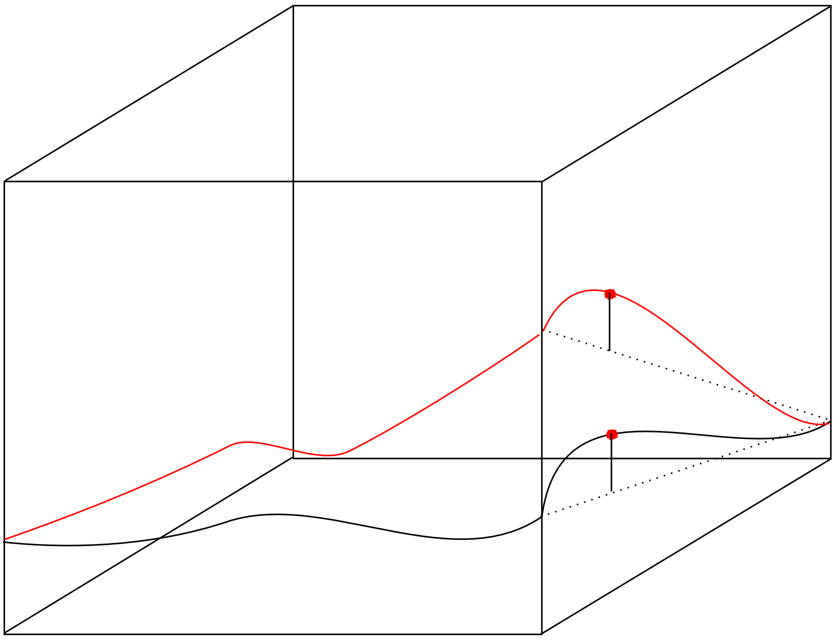


Figure 4.13: The modification of a preexisting curve on adjacent side. The black lines are the preexisting curves. The red are the new curves. When the user changes the curve on the front face of the cube, the curve on the side is modified to align with it. The dotted lines are constructed help lines. The distance from the constructed lines are used to move the points, as illustrated by the red dots.

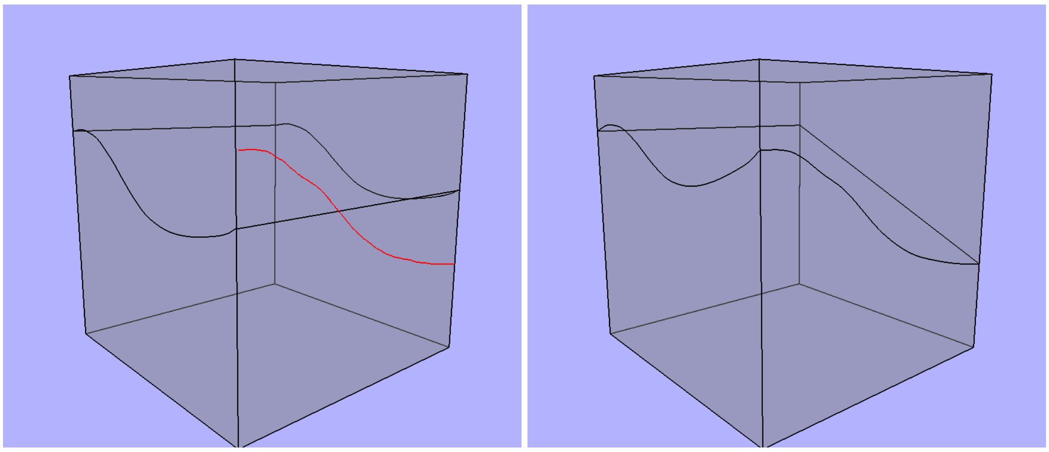


Figure 4.14: When the user modifies a preexisting face of the cube different modifications can happen. On the left hand side we see that the curve that was drawn earlier is modified to line up with the new endpoint. On the right hand side, which was a generated line, we simply generate a new line like earlier. On the opposite side nothing changes.

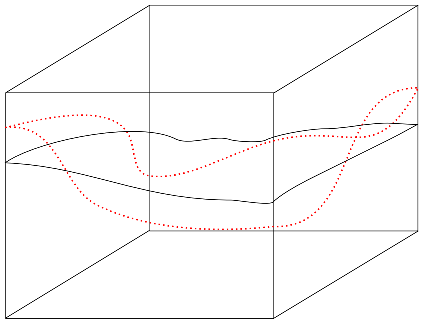


Figure 4.15: The simple first interpolation scheme to layer creation. The red dotted lines represent the user input, while the black lines are the outline of the layer. This approach was discarded, since it was difficult to make correct input for the desired result.

sketching that did not require advanced training and experience.

The solution came from the geologic field itself in the form of a technique that geologists use to model surfaces in sand. This old technique involves using two wooden profiles on each side of a sand box, and then slide a third profile across to create a surface. The technique is depicted in Figure 4.16. I read about this technique in the book Curves and Surfaces for CAGD, by Farin [14]. In this book it is simply used as an example of precomputed surfaces, so it was somewhat of a coincidence that I found this technique there. The idea is that the user can model a surface in sand by carving two profiles on the left and right side of a wooden box. Then, by dragging a third free hand wooden profile along the two sides you modify the sand surface accordingly (see Figure 4.16). The algorithm I present here can be thought of as very similar, only it will allow you to create two versions of the free hand profile, between which the actual profile will be interpolated at each point as you drag it across the sides.

The layers horizon surface is computed in a irregular height field. Construction of the layer surface geometry is achieved by looping through the length of the curves drawn on the cube, from left to right for the front and back curves. For each of the points, again looping through from the front to the back of the left and right curve. The points of the curves are stored as 2D parametric coordinates in the surface of the cube, and are translated into 3D points by a conversion function on the cube. They are also always ensured to be stored from left to right of the cube faces by seeing if the angle between the vectors from the origin to the two points is positive or negative, and reversing all the points if needed. If we call the 3D points of the left, right, front and back curves C_l, C_r, C_f, and C_b respectively, and denote the point along them as $C_l(x)$, $C_r(x)$ and so on, where $0.0 <= x <= 1.0$, we get a grid of new points that represent the heights of the surface according to the pseudo-code in Algorithm 1. These points are stored in a 2D array where the

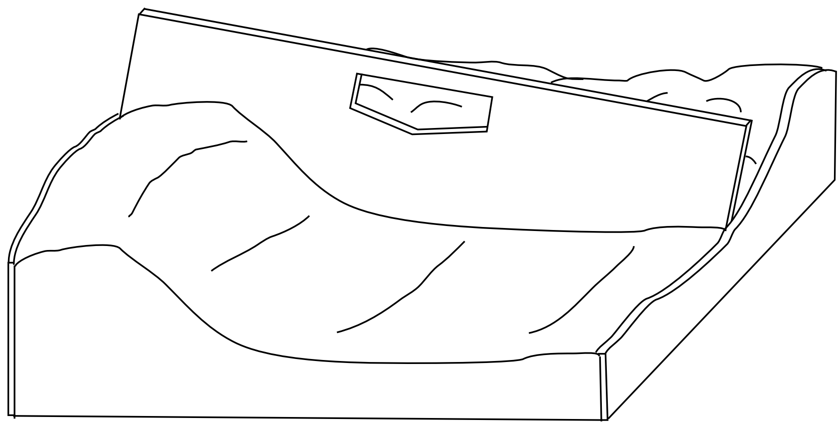


Figure 4.16: Modeling a surface in sand, by dragging one free curved profile across two different fixed curved profiles on either side of a box filled with sand.

index divided by the arrays size corresponds to the parametric coordinate over the horizon surface of the layer.

Algorithm 1 An algorithm for creating a surface from the four curves on the faces of the cube. The 3D points for the front, back, left and right curves are accessed as Cf(x), Cb(x), Cl(x) and Cr(x) respectively, where x is a parameter for the length of the curve from 0 to 1.

for all points in grid(i,j) **do**
  $left \leftarrow Cl(1) * (1 - j) + Cl(0) * j$
  $right \leftarrow Cr(0) * (1 - j) + Cr(1) * j$
  $start \leftarrow left * (1 - i) + right * i$
  $frontBack \leftarrow Cf(i) * (1 - j) + Cb * (1 - j) * i$
  $leftRight \leftarrow Cl(1 - j) * (1 - i) + Cr(i) * i$
  $difference \leftarrow frontBack - start$
  $grid(i, j) \leftarrow leftRight + difference$
end for

Algorithm 1 explained; for all points in the grid denoted by grid(i,j), first interpolate the left hand side corners, $C_l(0.0)$ and $C_l(1.0)$ of the cube according to the coordinate i. Do the same for the right hand side corners $C_r(0.0)$ and $C_r(1.0)$, with coordinate $1 - i$. Interpolate these two points using the coordinate j, yielding a starting point $start$. Interpolate the front and back curve points $C_f(i)$ and $C_b(1 - i)$ using the j coordinate and calculate the difference from the starting point. Interpolate the left and right curve points $C_l(j)$ and $C_r(1.0 - j)$ using the i coordinate. Add the difference to this point, yielding the final point.

The algorithm thus starts by doing a bilinear interpolation of the four corner values at the point of consideration. Then a linear interpolation of the two points of the front and back curves currently being considered. The difference between the points of these first two interpolations is then calculated. Another linear interpolation is done between the two points of the left and right curves at the point of consideration. To this last point the previous difference is added. The effect of this algorithm is analogous to the wooden profile explained earlier being dragged across the left and right curves while the actual wooden profile is being interpolated between the front and back curves. The result is the same no matter if viewed as if dragging the front and back interpolated curves across the left and right curve or vice versa.

I also examined another technique, inspired by Inverse Distance Weighing (IDW) interpolation from Shepard [39]. The idea is to loop in a similar fashion, but directly interpolate the four points $C_l(i)$, $C_r(1-i)$, $C_f(j)$, and $C_b(1-j)$ by IDW. The difference from IDW is that only these four points are used for the height at that point, instead of every point having influence on all points in the final height field. The points were weighted by the function $1/distance$ if $distance$ was more than 0. If distance was 0, that point was wighted 100 percent. The results looked OK, but I did not end up using it for a couple of reasons. First, I find the wooden box metaphor easier to understand, and the results easier to predict. I was unsuccessful in explaining IDW as a metaphor to my geology associate, and she agreed that the wooden box was easier to understand. Second, this method needed a power parameter for how the distance is weighted. This would have had to be exposed to the user, since I could not find a single parameter that yielded good results for every situation. Finally, the points in the grid tended to get distorted according to how close to the edges a point on the grid was. This created further problems with the parametric space later on as it would be distributed unevenly across the surface. For these reasons that approach was not included in the final solution.

The 3D points of the surface are not stored in a regular height grid, but are rather stored in a 2D array according to the parametric space over the horizon surface. This fits well with the parametric representation of features, and makes it easy to associate the parametric coordinates with each point in the grid. It also enables the user to draw curved surfaces without loosing resolution on steep slopes. It also enables the user to draw surfaces that loop back over themselves, that is they do not go monotonously/strictly from left to right or from right to left. This does mean that some features to be drawn on the surface of the layer later, might not have a well defined behavior in such areas where the surface does loop back over itself, but it also enables the user to model certain geological phenomena such as folding of layer structures.

Once the surface has been created, each of the features that might exist on the layer and needs to make modifications get the chance to do so. All features that

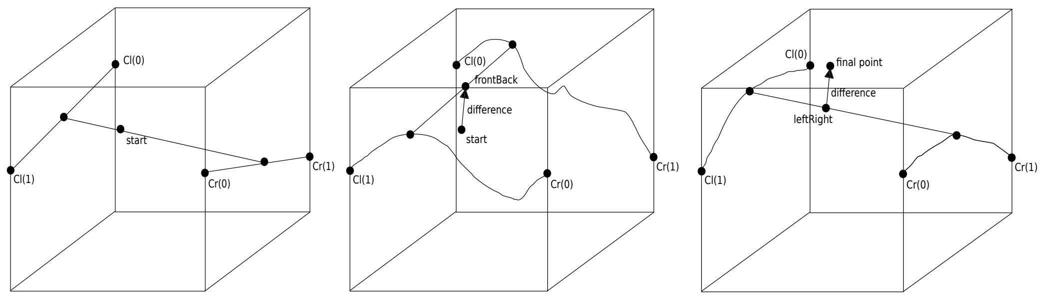


Figure 4.17: The calculation of a point in the layer grid. First, find the starting point by interpolating the four corners for the current position in the grid. Second, interpolate the front and back curves at the current position, and calculate the difference from the starting point. Third, interpolate the left and right curve at the current position, and finally add the previously calculated difference to this point, yielding the final point.

are drawn on the surface implement an interface that enables the layer object to tell them to do their modifications at the correct time. Each feature does this in the order they were created.

For the geometry synthesis, triangles are created between the points of the grid. Normals for each vertex are created by averaging the normal of the triangles surrounding the point. Then for each of the sides of the cube another surface is created between the curves of this layer and the layers below.

The sides of the layer are created for each face of the cube by filling the area between the new horizon and the horizons already made. If the current layers curve goes below the layers below, then no fillings are created in the area demarked by this section of the curve. This is achieved by finding each intersection of the new curve with the below curve, and creating a polygon of the points between the intersections wherever the new curve is above the below curve. The polygon is then triangulated by the ear clipping method, and the triangles added to the geometry object. The cube will maintain a curve that represents the top of the layers below, and this will be updated as new layers are added. This updated curve is also used to visualize the outline of the new layer. Therefore the layer creation code needs only to take into account this curve, and not all of the previous layers curves.

For updating the curve representing the top of all layers, the following procedure is used.

1. Create an empty curve newPoints

2. For each intersection from left to right, a point is created at this intersection. Then check which of the points from each curve preceding the intersection

is the topmost. Add all the points from the topmost curve since the last intersection, or if this is the first intersection, since the beginning to newPoints.

3. At the end add the rest of the points since the last intersection, or if no intersections since the beginning to newPoints, from the topmost curve.

For detecting which areas of a face of the cube to cover with the layers surface polygons are created and then geometry is created from there. This procedure is used:

1. Create a list of lists of points, “polygons”

2. Create a point between the first point of the layer curve and the curve demarking the top of all layers, call it “previousIntersection”

3. For each intersection from left to right, create a point “intersection”. Then add “previousIntersection” to a new list of points “polygon”. Check which of the two curves are the tomost at the point before the intersection. Add all points between the previous intersection and the current intersection from the topmost curve. Add the point “intersection”. Add, in reverse order, all the points between the current intersection and the previous intersection from the lowermost curve. Add “polygon” to the list “polygons”.

4. For each polygon in the list “polygons”, create a triangulation suitable for visualization, and add all triangles to the geometry.

The intersection testing on the layer is not achieved the same way as all other objects because of the large amount of triangles. Only the surface, and not the sides are used. The height grid is kept from the geometry synthesis step. A skip parameter is used in the code when doing intersection tests. It is used to skip certain number of cells to get a rough intersection area. This works very well as long as the skip is not too large (depending on processing speed) and it speeds up the intersection testing. The parametric coordinates are the same as the indices into the height grid divided by the size of the grid.

The visualization of a layer is straightforward and achieved by simply drawing all the triangles and the outline. If the layer is made invisible or marked for editing, the outline is drawn in a dotted line and no triangles are drawn.

4.3.3 Rivers

Rivers can be drawn on top of the surfaces by indicating its path with a curve. An algorithm then creates a river as shown in Figure 4.18. The input capture of the initial curve of the river is achieved by drawing on the surface as explained earlier.

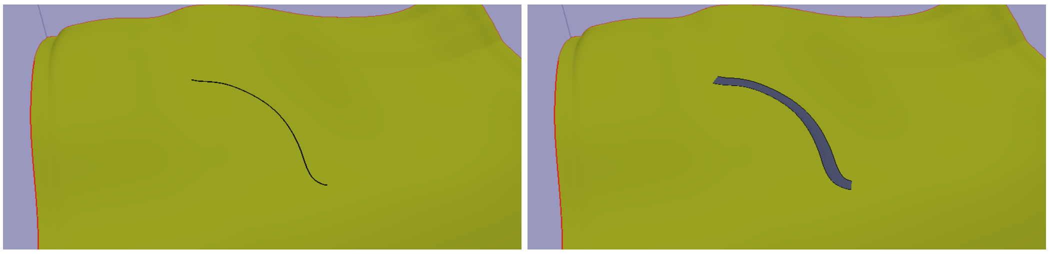


Figure 4.18: Sketching of rivers by indicating where it should run, and letting the program create a river along this path.

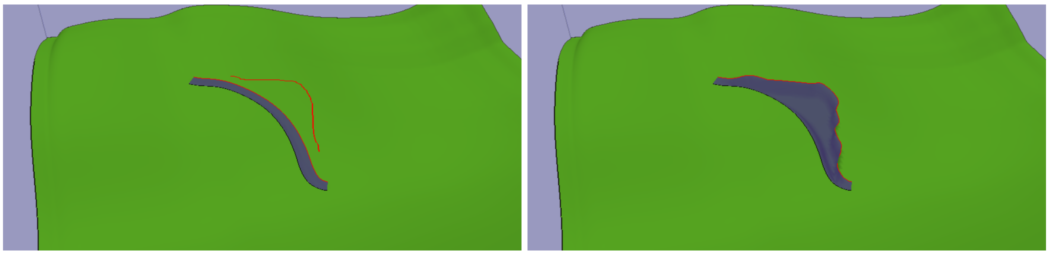


Figure 4.19: Changing a river by oversketching.

The initial curve is interpreted as the center point of the river. Each of the two sides is then computed by constructing a vector in the parametric space perpendicular to the river direction, and extending a new point in both directions from the center point. At the ends of the river a logarithmic function is used to create a smooth falloff towards zero, to make the two sides meet. These two sides of the river become the representation of the river, and they are the lines that can now be further modified by the user. The initial line is discarded.

It can also be changed by oversketching as shown in Figure 4.19. The oversketching is done on one side of the river at a time. The user also has the choice of replacing the entire side of the river if that will let her more easily make the changes she wants. When oversketching or changing the sides of the river, the layer geometry as it was before the river made any changes is used for the intersection tests. This is because it gets difficult to draw a new side of the river outline on the surface, if that side goes inside the river itself, as the terrain in that area is deformed by the river.

The geometry is made by simply creating a triangle strip between the two sides of the river as can be seen in Figure 4.20. The three dimensional points are found by calling a lookup function on the parent (the layer). This way of creating the geometry can put a limitation on the shape of rivers that can be drawn without artifacts. If the river sides are such that they loop back over themselves or bend in such a way that a triangle will cut a corner, then artifacts will appear in the scene.

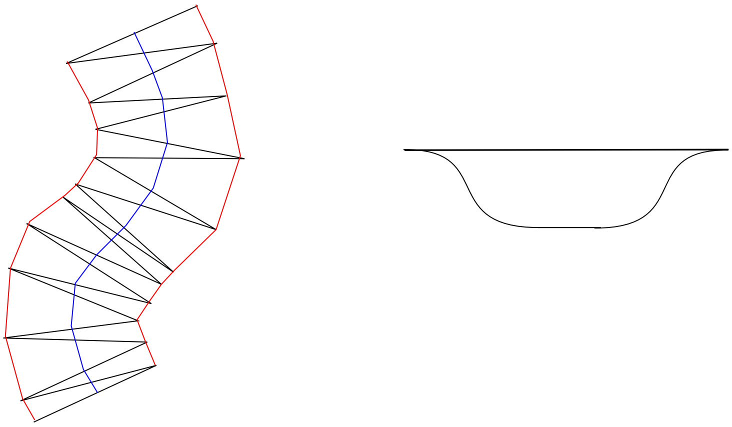


Figure 4.20: The triangles in a river are created by first extending the initial sketch line (blue) out to either side (red). Then triangles are simply drawn between the points at either side. The river also changes the terrain by using the same triangles. Every point on the surface that lies inbetween the triangles is changed in height according to a sinusoidal function of how far from the sides of the river it is, ensuring a smooth falldown into the river.

Another approach that was tried was performing a simple triangulation. However this only created a bumpy and uneven river, so it was not a satisfactory solution.

To change the terrain of the layer, the same triangles as in the geometry synthesis step is used, but only in the parametric space. For each triangle, all the points on the layer surface that fall within it is modified in height. The new height is calculated by a sinusoidal function which takes as parameter the distance of the point from the edge of the river. The function is $sin(\pi * x - \pi/2)/2 + 0.5) * depth$ where x is the distance from the edge, and $depth$ is the maximum depth. This function is only used where the distance from the edge is below a threshold value. This means that the river bank will have a smooth transition from edge to slope, and from slope to the bottom of the river where it will be flat.

When visualizing the rivers the geometry is drawn similar to the other objects. Figure 4.21 shows three overlapping rivers. To account for overlapping rivers, the layer object utilizes the OpenGL stencil buffer as it draws the rivers one by one. First it draws the river geometry for each of them, and then the edge lines. Where a pixel is drawn from a river, the value of the stencil buffer is incremented. When drawing a river, nothing will be drawn where the stencil buffer has been written previously. This ensures the rivers geometry is not overlapping, which is important because they are transparent, and because z-fighting could occur. Also the edge lines do not run over a crossing river because the geometry drawing has ensured the stencil buffer has been written to in that position.

Figure 4.21: Three overlapping rivers.

Figure 4.22: A valley and the same valley after a change to both of its sides.

4.3.4 Valleys

Valleys were created in order to be able to recreate the sketch made with the collaborating Master student of Earth Sciences. In that sketch a meandering river has created a valley with a plain. A valley functions almost identical to a river, only it does not create a geometry for any water and is initially wider and deeper than the river. It is, like the river, made by first drawing a line and then it can be changed by the same mechanism as the river. I will therefore not go into more details about the valleys other than illustrate how they look like in Figure 4.22.

4.3.5 Ridges

Ridges are also drawn by a line on the layer surface. Once a line has been drawn and the user indicates that she wants a ridge, a generic shape of a ridge is created automatically as seen in Figure 4.23. The user then has the choice to change the height profile along the ridge's baseline. This is done by sketching on a temporary sketching surface that is constructed along the ridges baseline, as can be seen in

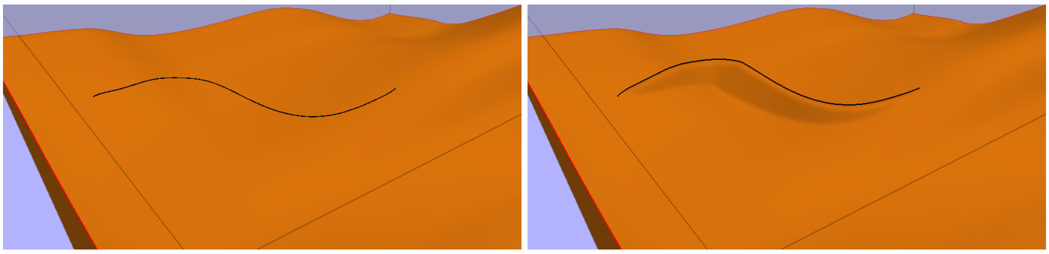


Figure 4.23: Sketching of a ridge by indicating where its base goes, and letting the program create a ridge along this path.

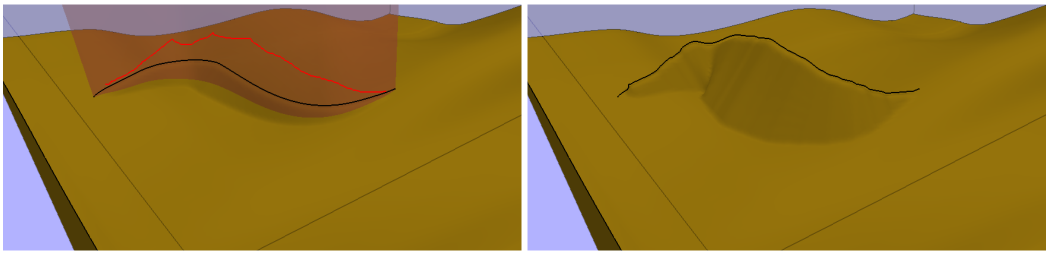


Figure 4.24: Changing a ridge is done by drawing on a wall that is constructed along the ridges baseline.

Figure 4.24.

Input for the ridges is first drawn on a layer as a curve. A ridge is represented by this curve and a height associated with each point in the curve. The curve is the base line that the user drew on the layer where she wants the ridge to follow along. The heights are the height of the ridge at each point of the curve. Initially the height list is just a smooth function from side to side of the ridge, with a peak in the middle. The height can be changed if the user indicates so. This new height line is input on a temporary sketch wall constructed for this purpose. The input procedure is similar to other lines, but in the end it is not actually stored as regular curve. When the user is done inputing the height line, the height along the entire wall is stored in a list, one for each point on the base line. The wall is constructed by vertices with parametric coordinates that make it easy to read the height from each intersection, as well as where along the curve each point is. If the user specifies a line that does not go strictly from left to right or right to left, but loops back over itself, only the last input for that position will be relevant as it will overwrite the former height stored there.

The geometry for the ridge is only the sketch wall, which is made by simply performing a lookup of the corresponding 3D point of each point along the base line, then making a triangle strip between these points and points that are a certain height above.

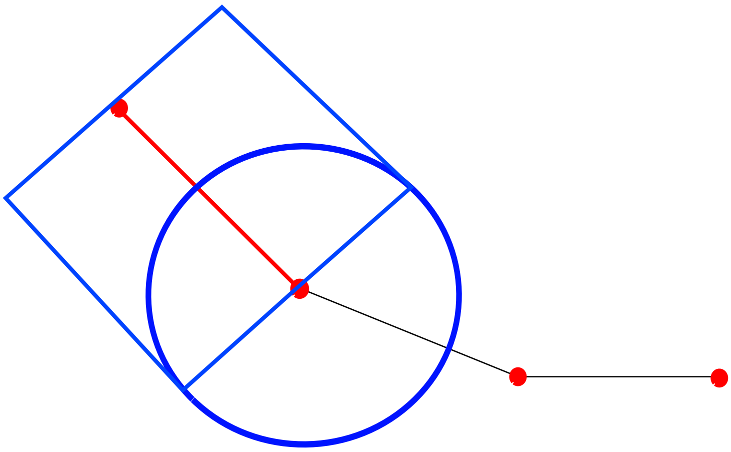


Figure 4.25: The changing of terrain for ridges. The points that fall into the square are modified in height according to how far they are from the line segment in question (red segment). Then the points that do not fall into the square, but do fall into the circle are adjusted according to their distance to the second point of the segment.

The ridge object itself is visualized only by a contour along the top of the ridge. This is constructed by iterating along the points of the base line. For each point in the base line, the corresponding 3D point is found by looking up this point on the layer the ridge belongs to, that is its parent. Afterwards, the height of this point is simply increased based on the relevant height in the list. This yields a new list of points which can then be used to draw a line on screen. When the height of the ridge is being changed, the sketch wall is also shown. The sketch wall is transparent to let the user see other structures that lie behind, so that it is easier to judge how high to sketch.

Figure 4.25 illustrates the area that is changed by the ridge on the terrain.The changing of the terrain from the ridges is done by finding each point that lies within a rectangle extended to each side of each segment of the line. The size of the square is given by the height at each of the two points of the segment. The points are found by checking each point for whether it lies withing the rectangle. Then the points that do not fit into this square, but that do lie within a circle with the center in the second point of the segment, and with a radius of that point's height are found. All these points are set to have a height according to how far they are from the center line of the ridge for the square and the second point for the circle. This way of changing the heights assures that the ridge side looks uniform. The circle will assure that each consecutive segment fits nicely together.

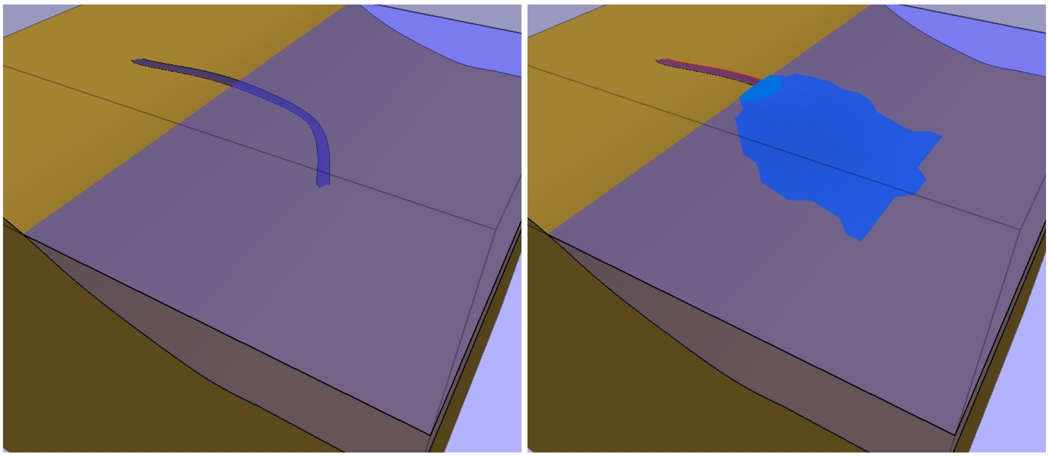


Figure 4.26: Creation of a deposit is done procedurally from the point where the river meets the sea.

4.3.6 Sedimentary Depositions

Sedimentary depositions can be modeled where a river meets the sea. The user indicates which river is to start depositing, and the rest is done procedurally by a simple simulation. The procedure continues until the user stops it. The user can indicate more than one deposit to be made for a single river. This will make the deposits build outward on top of each other in the direction of the river while also following the terrain.

The input for a deposit is implied by the river that the deposit is flowing from and where that river goes below the sea level at the time when the user requests a deposit be made and how long she wishes to let it continue depositing. Deposits are represented by the position where they start, directly below the point in the river where the sea starts, and the amount of matter that is deposited from this point over time. The intention was to also take into account the direction and speed of flow in the river, but this has not been completed yet. When creating multiple deposits at the same time they do however build outwards according to this direction.

In order to visualize the creation of a deposit as it builds over time, it needs an additional step to generate an intermediate representation of the deposit before generating the geometry. This step consists of simulating the flow of matter across the surface underneath. For the simulation a simple volume preserving diffusion algorithm is used, that is a modified version of the one by Boeschs [7]. An illustration of the approach by Boesch is given in 4.28. The algorithm works by considering one cell at a time, and comparing the heights of the neighboring cells height from the previous iteration. Half of the difference, clamped by the avail-

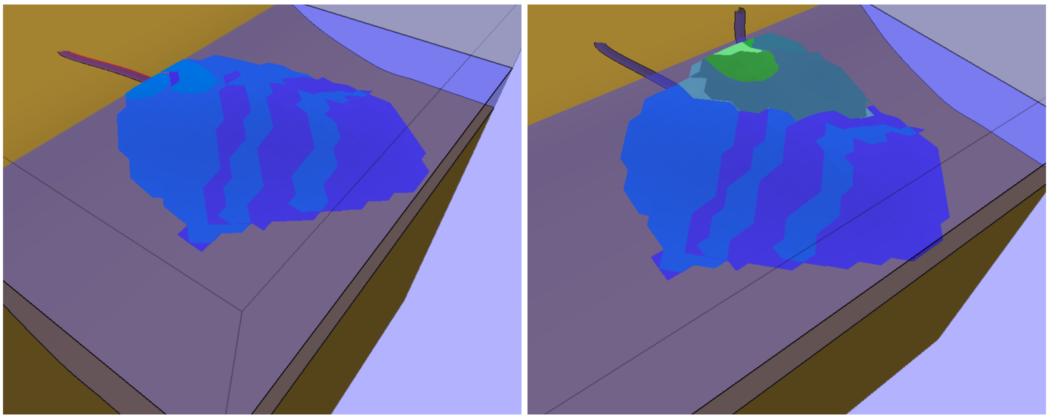


Figure 4.27: The user can indicate when she wants a new layer of deposits, or when she want's to stop depositing. She can also add new rivers with new deposits.

able amount of water, will be added to the current cell. This is first done for each cell considering its neighbors along the x-axis. Then the process is repeated, but this time considering the cells neighbors along the y-axis. In my version of the algorithm, even less than half of the difference is added according to how far from the rivers mouth the cell is. This algorithm assumes a regular height grid, and all the underlying layers must be taken into account. The layers are represented as a irregular grid and thus a sampling must be performed to create a regular grid. First a grid overlay over the cube is created at the desired resolution and each point in the grid is set to a value low enough that no intersection could occur at this height. Then at regular intervals, a ray is cast directly down into the cube, doing intersection tests for each layer, updating the grid value to the height of any intersection when that intersection is higher than the current value.

When the height grid is ready, the simulation can begin. The simulation uses an additional grid, to keep track of the amount of deposits at each point. Here follows a description of the simulation algorithm (The algorithm is given as pseudocode in Algorithm 2);

1. Matter is deposited at the point where the deposit starts (where the river meets the sea), such that is does not go above the sea level. A variable keeps track of how much has been deposited over time.

2. For each axis in the grid (x-direction and y-direction), for each point in the grid; Set the deposit height of the current cell to its previous value plus half the difference as compared to the previous cell (according to the axis currently considered and the next cell. Both these values are clamped by the available amount of matter.

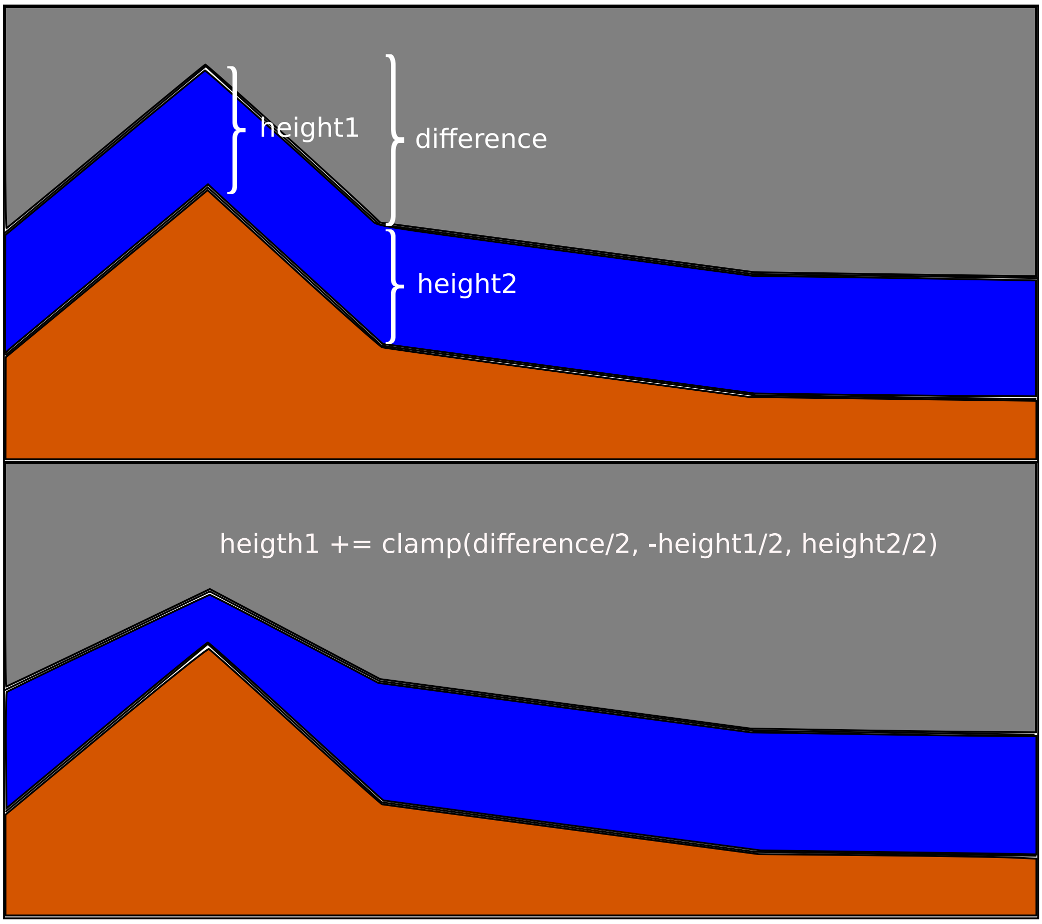


Figure 4.28: Recreation of figure from approach by Boesch [7].

Algorithm 2 Pseudocode of the deposit simulation algorithm. Let the current cell be denoted as C(i,j) the previous value of the cell $P(i, j)$, the terrain height $T(i, j)$ and the axis vector $A(i)$ is either $x = (1, 0)$ or $y = (0, 1)$, and a flow rate function $F(i, j) = 2 + M(i - A(0, j - A(1))^{flowRate}$ where flowrate is a number between 1 and 2 as chosen by the user. The new value for each cell is then computed as follows:

---

for $A = x, y$ **do**
    $left \leftarrow (i, j) - A$
    $right \leftarrow (i, j) + A$
    $tl \leftarrow T(left)$
    $dl \leftarrow P(left)$
    $tr \leftarrow T(right)$
    $dr \leftarrow P(right)$
    $diffLeft \leftarrow (tl + dl) - (T(i, j) + P(i, j))$
    $diffRight \leftarrow (tr + dr) - (T(i, j) + P(i, j))$
    $flowLeft \leftarrow clamp(diffLeft/F(left), -d/2, dl/2)$
    $flowRight \leftarrow clamp(diffRight/F(right), -d/2, dr/2)$
    $C(i, j) \leftarrow P(i, j) + flowLeft + flowRight$
    if $flowLeft > 0$ **and** $M(left) + 1 < M(i, j)$ **then**
        $M(i, j) \leftarrow M(left) + 1$
    end if
    if $flowRight > 0$ **and** $M(right) + 1 < M(i, j)$ **then**
        $M(i, j) \leftarrow M(right) + 1$
    end if
end for

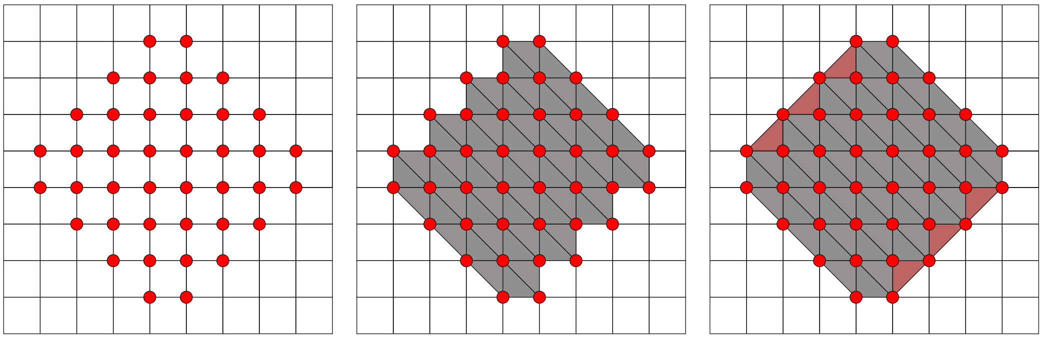


Figure 4.29: Triangle orientation. Left; the grid points that have matter deposits above the threshold. Middle; the first, naíve approach where all triangles are oriented the same direction. Right; the more sophisticated approach, where the triangle orientation depends on which surrounding points have deposited matter. As seen on this illustration, the second approach gives a more uniform look on each side of the structure, while the first approach gives more jagged edges and non-uniform look.

The simulation runs until the user is satisfied and stops it or if the deposit is a preexisting one, until the target deposit amount has been reached. The total amount of deposited matter is stored in the target variable when the user stops the simulation.

Geometry is generated based on the height of the deposits and underlying terrain. When generating geometry special care needs to be given to the orientation of the triangles to give a uniform and smooth look to the visualization. For each column i and row j in the deposit grid, triangles are generated according to Algorithm 3.

To decide where to add triangles, and in which orientation, for the space in between each of the grid cells, the four surrounding points are considered. If both the lower right and upper left cell has deposited material, then triangles will be created between these two cells and one triangle for the two other cells if they have material deposited to. Otherwise, if both the upper right and lower left cell has material deposited, then triangles are created using these two points and creating one triangle extending to each of the other two points.

This algorithm in gives triangles that follow the entire outline of the deposit smoothly as illustrated in Figure 4.29, where a naive approach always gives jagged edges at one side and smooth edges at the other side. It works by examining which of the surrounding points have matter deposited, and orienting the triangles accordingly.

When creating a deposit, the layer object will check for previous deposits, and if such exists, the grid data will be reused for the next deposit. It is reused by

Algorithm 3 Triangle orientation decision. D(i,j) is the deposit amount. A threshold t is used to decide where to draw triangles and where not.

```
for j = 0 → gridsize − 2 do
    for i = 0 → gridsize − 2 do
        a ← (i, j)
        b ← (i, j + 1)
        c ← (i + 1, j)
        d ← (i + 1, j + 1)
        if D(c) > t and D(b) > t then
            if D(a) > t then
                create triangle between a, b, and c
            end if
            if D(d) > t then
                create triangle between b, d, and c
            end if
        else if D(a) > t and D(d) > t then
            if D(b) > t then
                create triangle between a, b, and d
            end if
            if D(c) > t then
                create triangle between a, d, and c
            end if
        end if
    end for
end for
```

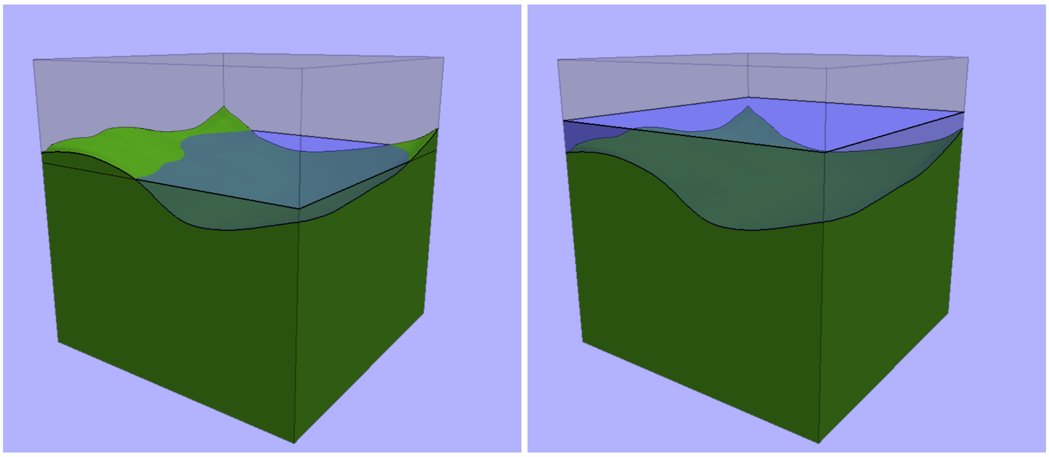


Figure 4.30: Indicating where the sea level goes.

copying the terrain height grid and then adding the height of deposits at the points of the grid. This gives speed improvement, and also enables deposits to stack on top of each other.

4.3.7 Sea level

The sea level can be enabled at any time and moved up or down as the user specifies. In addition to being used in scenes to illustrate the sea, the sea level is used as an input parameter for the deposit feature. The sea level is implemented simply by creating a layer with straight outline curves. Each time any layer changes the sea level layer is recomputed. The input is made on the cube by clicking the mouse after having indicated by a button that one wishes to change the sea level. It is visualized with a transparent blue color.

4.4 Implementation details

This chapter explains the choice of technology such as the programming language and libraries. Then we explain how the supporting features such as save, load, undo and menus work.

The choice of technology to base the work on was made by a bit of trial and error. Different scene graphs in c++ and Java were tested to see what would work the best. In the end it was decided that the project would be based on c++ as programming language with Qt for user interface and OpenGL for graphics. Custom visualization and scene graph code would be created, as some specific functionality would be needed that might not fit well into a premade solution.

Qt was chosen as the user interface library because of previous experience. In addition to having a good set of user interface classes it also has numerous helpful classes with things like collections and event handling that make the process of programming in c++ comfortable.

Some tests were initially conducted in java by porting the code from c++ as it was made. Although the java runtime itself is theoretically fast enough, naively using its methods of abstraction slows down data access a lot. This forces the programmer to write complicated and unreadable code that defeats one of the main purposes of using java in the first place. The abstractions in c++ can be a bit more complicated to use than in standard Java, but also allow much faster data access and better control over memory allocation. The main problem here was that Java would constantly allocate objects on the heap even though they would only be needed temporarily. The Java compiler does have optimizations that removes such allocations where possible, but rather frequently it cannot know what the life expectancy of an object is. This was possible to overcome by writing specially designed code to enable the compiler to see where a temporary object is not being reused and by passing primitive types instead of pointers to objects in many places. Another problem is that arrays are restricted to contain primitive types or pointers. Thus an array of objects cannot be allocated such that its objects lie in contiguous memory, and this slows down access substantially. This was also possible to overcome by making arrays of primitive types instead of objects. Both these approaches put together made the code run comparably fast to the c++ code, but difficult to modify. It would probably have been possible to solve all such problems with Java by relying on performing all heavy calculations on the GPU by using OpenGL Shading Language or a similar GPU programming language. This was decided against for lack experience with GPU programming and a lack of time for learning and experimenting with development tools designed for such programming. None of the algorithms explained utilize this possibility, although that would probably increase the running speed noticeably.

The save feature is implemented by a function in each node in the scene tree that will convert that nodes data and a string representing its type into QVariant objects. A QVariant is an object defined in the QT library that can hold any type of primitive value and also collections. Thus it is possible to build a tree structure with arbitrary data. When this conversion is done the data is sent to a library that will convert such a tree of QVariants to a text string in the Json format. The Json format is a way of structuring hierarchical data similar to XML. It comes from JavaScript where objects can be declared with object literals. The Json format uses the same format as these object literals. The text string is stored in a file the user specifies. At loading time the entire process goes in reverse. With the aid of a function that instantiates an object from the type string the correct type of object is instantiated. Then a function that populates that objects data is executed. Each object knows what kind of special procedures it must run to become valid. To restore the state exactly as it was at save time is the most complicated and crucial part of the save and load code.

The undo function works by simply copying the whole scene tree whenever something is changed. Each node type in the tree has a copy method that is used to create a copy of it. Only the representational data is copied. The most complicated part is like with the save and load feature to keep the state of the copies exactly the same as the original. The new copy is then pushed onto a stack. When the undo button is pressed the stack is popped and the scene tree is replaced for the one that was at the top in the stack. The copying code does only copy the reference to the geometry objects, thus saving time on not recomputing all the geometry if that is unnecessary.

The menus in the program are dynamic. That means that they change according to context. At first all the different options were always displayed, but this was a bit confusing for the user. Therefore a method was made that is run each time the user makes a change. This method checks what state everything is in, and populates the menu accordingly.

Pictures of these supporting features together with all the other features can be found in the next chapter, where we will see examples of what can be produced with the solution at this time.

Chapter 5

Results

This chapter starts by showcasing geological models of selected scenes that have been created using the developed modeling environment. Then it presents the results of a user study that was conducted. An evaluation of the results follows in the next chapter.

5.1 Example Scenes

Figure 5.1 present some examples made by two geology students after a 15 minutes introduction to the program. The sketches represent attempts at reproducing the sketches from Figure 1.4, which were made early on as the goal of what should be possible. Figure 5.2 shows the same scene as reproduced by the author of the program along with a ray-traced image made in the program Blender. As can be seen, transparency is not properly exported, so the deposits can barely be seen where it is above the water.

Figure 5.3 shows the process of how a glacier erodes the landscape. The first sketch is made by drawing the outline of the rock with valley from the first illustration, then simply adding some ridges and rivers. The second sketch is then made by editing the layer to create the valley that is carved by the glacier and deleting the rivers. Then the glacier is created by drawing the end lines on the front and back of the glacier as the contours of a new layer. On the other two sides, the slope of the glacier is indicated. The third sketch is then made by deleting the glacier layer, adding some new rivers and valleys, setting the sea level, and creating a small deposit. A limitation of the program can be seen in that it would be difficult to recreate the glaciers that do not lie “straight” in the layer, because by editing the underlying layer it is only possible to make a valley that follows the layer interpolation direction. The glaciers that are positioned diagonally in the cube, would have to be created by the valley feature. However, the valley feature

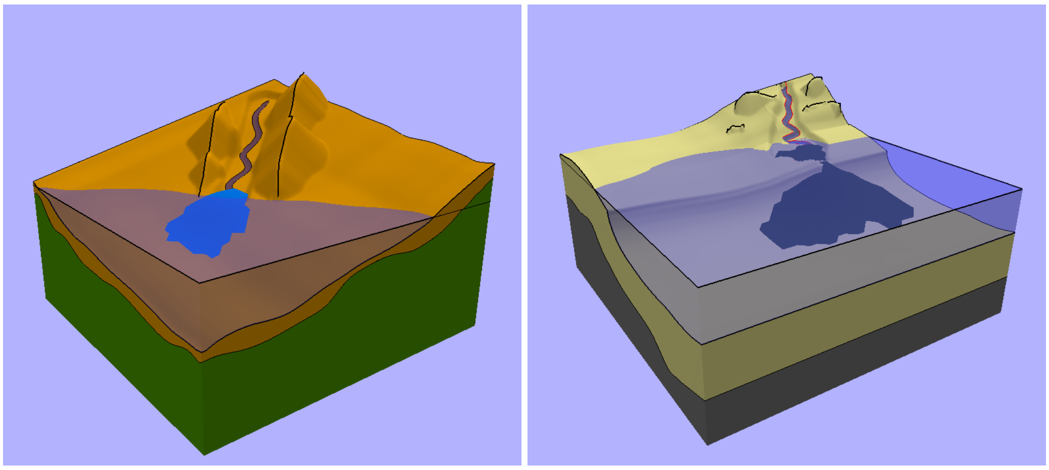


Figure 5.1: Reproductions by two geology students made after a short introduction to the program.

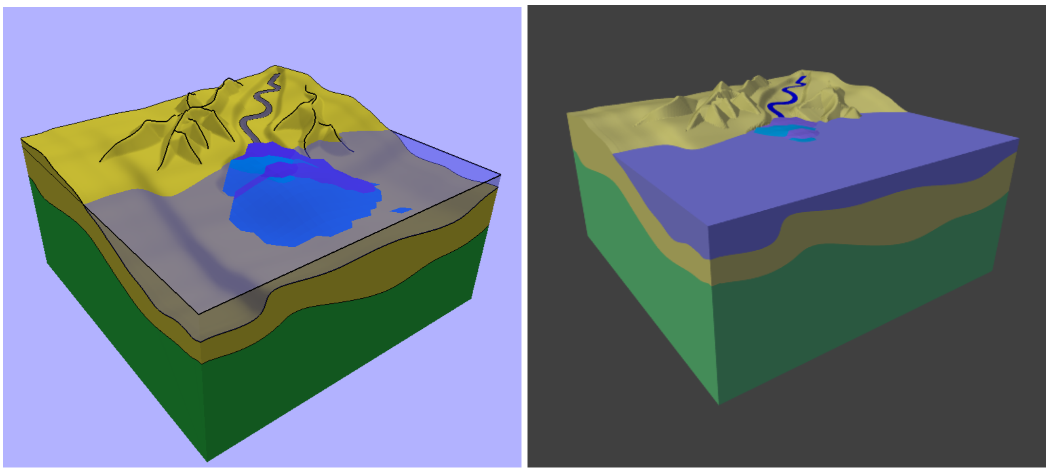


Figure 5.2: Reproductions by the author of the program, and a rendering made by ray-tracing the exported geometry in an external program.

does not currently allow adjustment of the depth. Additionally, it could be difficult to create the glacier layers in such a way that they fill the desired volume, when the glacier does have borders at the cube faces.

Figure 5.4 shows the process of how the oceanic part of a plate is submerged underneath a continent on another plate where they collide. The structures have to be build from bottom to top, or rather any layer that intersects another must be drawn last. The mantle must be drawn even though it does not appear in the sketch. It is shown in gray here for illustration, but it could be made invisible. Then the oceanic lithosphere is drawn, the oceanic crust follows. Then the continental lithosphere and the continental crust is drawn such that the sketched curves intersect the oceanic crust at the subduction point. The last layer is the accretionary prism, which represents sediments scraped of from the subducting oceanic plate and gathered at the wedge between the plates. Finally, the sea level is indicated, some ridges of mountains are added and an inland sea is created by drawing a river and widening it. The melting rock, rising magma and other volcanic features can not be recreated at this point.

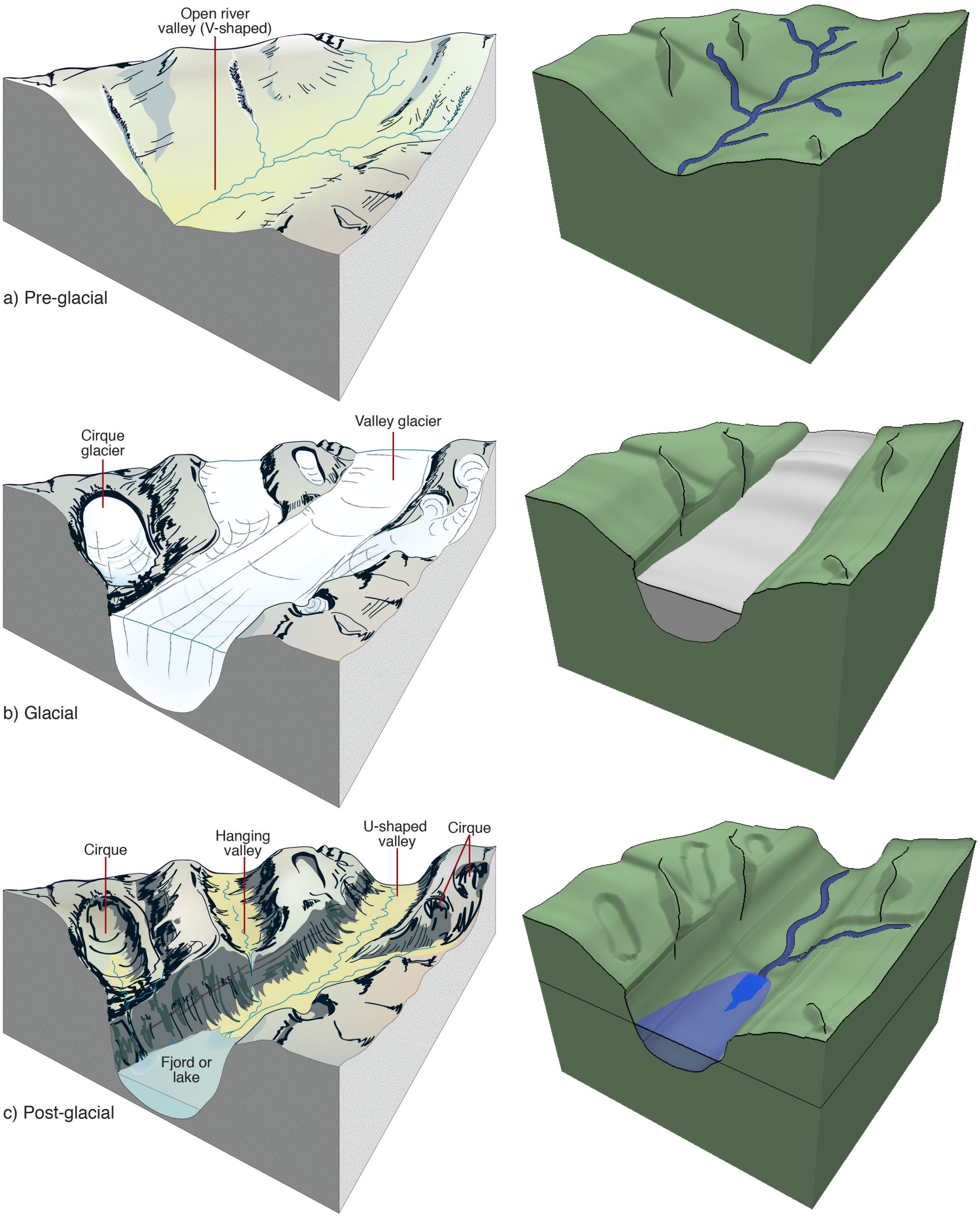
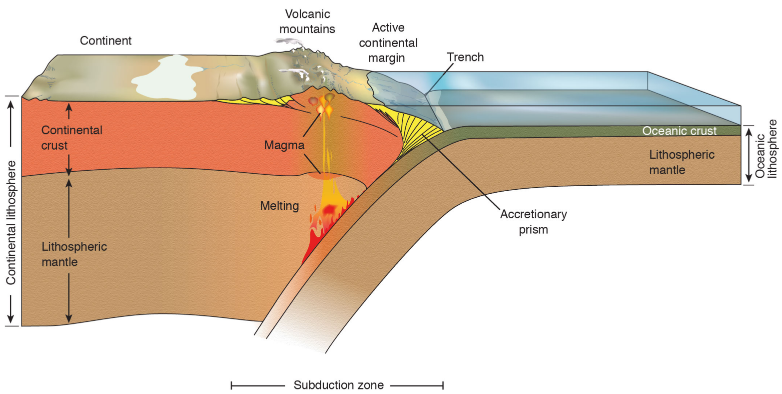


Figure 5.3: Glacier illustration and attempt at reproduction.



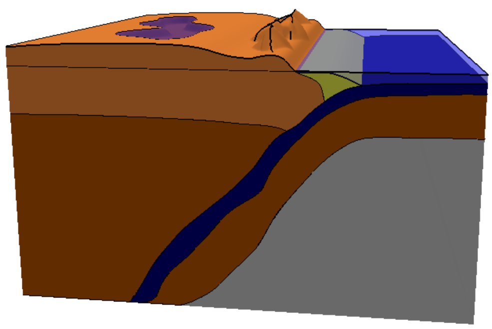


Figure 5.4: Illustration of subduction of oceanic lithosphere underneath continental lithosphere and attempt at reproduction.

5.2 User study

Evaluating the usability of a modeling approach is not easy. The study was conducted to gather feedback from a group of geology students. The study consisted of several questions relating to the different aspects and features of the approach. The answers had to be given by indicating on a scale from 1 (worst) to 10 (best) according to how much the user agreed to a question, liked or disliked a feature, etc. In addition the subjects were given the opportunity to explain their choices and give comments for each question.

5.2.1 Responses

A summary of what the subjects responded is given here. The complete results can be found in Appendix B (in Norwegian).

For the total user experience, the users indicated the average scale values of 7. The subjects found the tool useful for making illustrations and found the look pleasing. However, the menu items were confusing for some. The ease of use was praised as the most significant advantage of the approach. The approach was described to give the ability to play with the different ideas and thoughts around geological scenarios. One user said he had never seen an approach giving the possibility to create simple illustrations as quickly and easily as this. The approach was described as very self-explanatory, and in particular one user was impressed with the ability to make changes to the illustration after having made a basic version of the scene.

On the difficulty of learning and using the program the subjects indicated an average score of 6.5. Some users thought the approach was very easy to use, while others needed some more effort to learn it. They commented that having the menus visible at all time would be less confusing. There was also a suggestion to make a list for selecting the different objects in the scene, as that could sometimes pose difficulties.

When assessing the usefulness of the tool in its current form the subjects gave an average score of 6.25. Comments mentioned that the tool can in its current form be useful for illustrating some very simple geological scenes, but that it would require more development for it to be useful for depicting more complex scenes. Suggestions for improvements was a feature for creating faults, the ability to alter the width of mountains and the depth of valleys. One subject indicated that it was difficult to suggest, because he didn't know what is possible to make.

For the potential of the approach the subjects gave an average score of 8.25. One subject indicated a belief that if the features suggested could be implemented, the program could become very useful for them. Another said that the way they make illustrations today is usually by hand, which is time consuming. This approach helps creating quick illustrations.

The ease of changing features in the illustrations were rated at an average of 6.25. One user found this difficult, again because of the confusing menus. Another suggested a feature for clearing the cube. One user remarked that it was very easy to make changes to layers, and that the methods of changing all the terrain features and rivers was excellent. This user also said it was important to not complicate this too much, since the strength of the approach lies in its simplicity.

The rest of the questions asked the users to evaluate the different features of the program and comment them. Evaluating the cube as a starting point, the subjects gave an average score of 9.5, while the drawing of layers got a score of 8.5. Together they were described a super idea and with a good input method. It was easy and understandable. One subject enjoyed the fact that one could draw and change the horizons on a vertical face and from any angle of choice.

The ridges feature was given a score of 6.75. It should be possible to modify the width of the ridges. One subject thought this was a good solution, but nevertheless suggested that it would be useful if more vertical sketching surfaces could be put in at demand in different directions, which the subject imagined would enable modeling of almost everything desirable.

The rivers and valleys were also given a score of 7.5. The river solution was described as good. For the valleys the subjects again wished for further modification opportunities like depth and profile control. The deposits feature was given a score of 7. One commented that deposits do not really gather above sea level. Another wished it would be possible to view the individual layers deposited. One user said this was the easiest to use geological process in any program he had seen, but commented that the approach was very sensitive to the order in which they were created, and the program would crash if not being careful. One user commented that valleys are actually the wedge between two mountains, and that this feature looked more like a canyon. He therefore suggested to merge the ridges and valleys feature one and the same, which would make a more natural valley form.

The saving, exporting and undo functionality received scores of 8.25, 5 and 8.5 respectively. They were all described as functional. One subject said the undo feature would sometimes skip too many steps back.

At the end subjects were asked to give any further remarks. They mostly reiterated what was said in the earlier comment fields, but gave general praise for the idea and ease of the approach.

The next chapter will give my evaluation of these results and the entire project.

Chapter 6

Evaluation

The results in the previous section indicate that the approach suggested has a potential use in illustrating geological phenomena. As Figure 5.1 shows, even after a short introduction of about 10 minutes, test subjects from the geologic field were able to draw figures resembling the sketch they were trying to reproduce. Figure 5.2 shows that a person familiar with the program can create more detailed sketches, and the ray-traced image indicates the possibility of using the created models in further applications. Exporting could thus allow more detailed features to be added to the models at a later stage by people with such expertize.

Figure 5.3 and 5.4 shows that reproduction of some illustrations of geological educational material is already possible with this approach. Some details are missing, but the most important processes are captured. With a little more development of the approach, these examples should be perfectly reproducible to the extent that all the important features are captured in a reproduction.

When the approach gains more maturity, I expect it or some similar approach could become the standard way to illustrate geological phenomena by students, teachers and researchers. I base this expectation on the fact that all these drawings were made in a matter of minutes. Subjects from the user study indicated that they use considerable time creating such sketches by 2D methods and that they believed it would save them time. Subjects in this study were all students, but I would expect also experienced geologists sometimes use considerable time when creating conceptual geological illustrations and that they could all benefit from the new modeling approach.

The user study also sheds some more light on what aspects of the approach work and in what direction any further work might consider going. The subjects responded very positively to the approach, although they indicated that further development is needed to make it useful in many scenarios they need to illustrate. According to the study, the chosen input method for creating a surface makes it possible to draw simple surfaces quickly compared to what subjects draw on paper. However, as betrayed by the glacier reproduction attempt (Figure 5.3), the algorithm that interpolates the lines drawn works in such a way that it will favor structures that manifest themselves along the interpolation direction. Similar structures could be made in that instance by the valley feature, but it can be frustrating for a user to draw the same type of features in one case and another in other cases. Other features like rivers, ridges and sketches were all described as easy to use. However, the valleys and ridges could benefit from more control over depth and width respectively.

Deposits are a feature that shows how a procedural approach can be combined with the sketching approach for creating surfaces. It is the last feature that was included, and will need more development to show its real potential. Subjects remarked on some bugs, and requested more features regarding the deposits however, so in their eyes it presumably can also be useful to illustrate deposits in the proposed way.

During the development of the approach, work was conducted in an experimental fashion. I would try out different approaches, and decide which one worked best based on trying it, often together with Marie. After a while I decided I needed some discussion and collaboration with people from the geologic field to continue. This was very helpful during the development phase in order to get direction and focus where my own experience and knowledge about geology was not sufficient. I suspect that involving geologists from an even earlier point could have helped the process. Even in questions not pertaining directly to geology, it was helpful to get input from people not from the computer science field. They were not familiar with the internals of my particular implementation and way of thinking, and would therefore mention problems that I would not notice. Several usability issues were ironed out this way.

All in all the project has been a success. I could definitely have used more time to add numerous features and refining the ones that are there today, but the results show that the idea of drawing in a box was a good one and that the approach could be used, with further development, by geologists to make many of their illustrations.

Chapter 7

Further work

Of the difficulties I found with modeling geology in a rapid way is to make all the different features interact correctly, while keeping the input on an abstract level to keep the modeling simple. There are many different processes that geologists need to model. There is a trade off between the amount of time needed to model, and the possibilities the modeling approach offers. The real challenge is to minimize the negative effects of this trade off while incorporating more features.

I would really like to develop an intuitive approach for making fault structures in the layers. Faults are definitely a needed feature, and by my own understanding of geologist's needs, how often faults are encountered and based on test subjects feedback, a feature for sketching faults would be the natural next step for further research. I think an easy way to input such faults could be to let the user sketch curves on each horizon where the intersection of faults and layers should be. An interpolation of these curves would then create a surface representing the fault. The user could then indicate by sketching on this surface how far the layers involved have faulted. Finally an algorithm would be needed that could morph the layers and any features already drawn on them.

Another feature I would like to see, is the ability to combine several cubes into a bigger scene. This would require some means of using what has already been sketched on one cube as a starting point for adjacent cubes. In order to not impose too much work on the user, the surface creation algorithm would need modification to ensure continuity of surfaces across cubes. I think that conducting further research into using the Inverse Distance Weighing interpolation approach, could help with this since it could take into account the points on both adjacent cubes in the weighting. The Discrete Smooth Interpolation developed by Mallet [24] would also be interesting to explore for allowing smooth transition between the cubes. The ability to change a cubes height, width and depth is another possible feature that could be useful. To make this cube change work together with the multiple cubes all cubes could be restricted to the same size, so they will align properly.

There is also a number of possible improvements on the features that are already included. The river width control can today be a bit difficult to use if the user wants to create narrow rivers. This difficulty could perhaps be removed if there was a way to change the width across the entire river in one go, instead of having to carefully sketch along the entire rivers length. Ridges could benefit from some method of sketching the width along the length of the ridge. I think both depth and width control could be achieved by additional sketching surfaces. If the user could sketch the profile in of a river, valley or ridge orthogonal to the direction of the initially sketched base line, I think this would be sufficient to create many desirable versions of such features. To make this feature even more powerful, it could be possible to input several such profiles along the length of the feature, and the profile could be interpolated between them.

To make deposits a more powerful feature, I would make them into fully fledged layers. By this I mean that it should be possible to modify them in all the same ways as the sketched layers in the cube. The procedure that creates the deposits could also have more parameters, like how far it will flow, how much material is transported, how fast the river flows, etc. It could take into account the possibility of the river stream carrying different sized particles at the same time, thus depositing them in different regions. The flow speed of the river could be calculated based on the terrain slope. The amount of deposited material would be dependent on several factors possibly outside the sketch, such as terrain material and length of river. Deposits could also benefit from more control over the shape.

Many of the illustrations could benefit from a feature that allowed painting on the surfaces of the sketch like Natali et al. [29], and by creating billboards such as Cohen [12] suggests. Painting on the surfaces would for example allow the illustration of subduction of oceanic plates sketch in Figure 5.4 to be completely reproduced. A billboard feature would also allow geologist to input context giving features such as vegetation and animal life, plumes of smoke from volcanoes, etc. Such details can be seen in many of the illustrations in Chapter 2. With a painting and billboards feature artistic users could input details that would improve the visual appearance of the illustration. I would also like to see a texturing feature similar to what Natali et al. proposes where textures by default follow the layer structure, but the user can sketch to override its direction and deformation to illustrate details of a layers internal structure. Another way to add a more realistic look for the horizon surfaces could be achieved through a fractal noise method.

If implementing any of these features it is important to focus on the usability and ease of use. In that respect I would continue to involve users from the geologic field. One of the characteristics test users have commented was the ease of use of the features and of creating sketches in this approach. If it would become more difficult to use, they might prefer to draw illustrations by hand. It should preferably not become more complicated to model the scenes that are already possible to create. New features should simply add possibilities that are as easy to use as the rest.

Chapter 8

Conclusion

This thesis started by explaining how geologists have a use for an application that lets them create simple models that lets them illustrate their thoughts amongst themselves. In education and literature such illustrative tools are also useful. A goal was stated: to create an approach for rapid and easy sketching of geologic structures in 3D. The relevant geological background was explained and the state of the art for rapid sketching of geology was described. An approach for a tool for rapid modeling of geologic structures was given and how it was developed was explained in detail. This approach explained was mostly based on sketch input, but also incorporated a procedural method to explore the possibility of combination of these two rapid modeling metaphors.

Screenshots of geologic scenes that can be modeled using the currently implemented algorithms were given. A user study showed that such a tool is indeed highly interesting for the target group of people. The approach developed has some merit to it according to this user study, although further development is still needed for it to be useful in more than a few simple cases. However, the subjects of the study indicated their belief that the approach has potential. The input methods were described as easy to use, although most of the features need further development to enable sketching of desired geological scenarios. Even in the state the implemented solution is in now though, some users expressed a possibility for applying the solution in real world situations.

Finally some thoughts for further research were discussed what I think would be the natural next steps to improve this approach. Particularly, fault structures are a feature that occurs in many geological illustrations. When developing new features, focus should be given to what potential users are comfortable with and to preserving the ease of use that the approach already has.

I expect that when this approach gains more maturity, or similar approaches are developed to maturity, they could become the standard way to illustrate geological phenomena by students, teachers and researchers, based on the illustrated results and user study. Although further research and development is still needed to enable more geological scenarios to be illustrated, the user feedback indicated that the approach is intuitive to use and enables rapid illustration of certain geological scenarios. The goal of the thesis, to create an approach that can be used for making rapid 3D illustrations for geologic uses, has thus been reached.

Bibliography

[1] R. Amorim, E.V. Brazil, D. Patel, and M.C. Sousa. Sketch modeling of seismic horizons from uncertainty. In *Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling*, pages 1–10. Eurographics Association, 2012.

[2] C.S. Applegate, S.D. Laycock, and A.M. Day. A sketch-based system for highway design. In *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling*, pages 55–62. ACM, 2011.

[3] Autodesk. Autodesk 3ds max. In *http://usa.autodesk.com/3ds-max/*.

[4] Autodesk. Autodesk autocad. In *http://usa.autodesk.com/autocad/*.

[5] B. Benes and R. Forsbach. Layered data representation for visual simulation of terrain erosion. In *Computer Graphics, Spring Conference on, 2001.*, pages 80–86. IEEE, 2001.

[6] B. Beneš and R. Forsbach. Visual simulation of hydraulic erosion. *Journal of WSCG*, 10(1):79–86, 2002.

[7] F. Boesch. Webgl gpu landscaping and erosion. In *http://codeflow.org/entries/2011/nov/10/webgl-gpu-landscaping-and-erosion/*, November 2011.

[8] E. Brazil, I. Macedo, M.C. Sousa, L.H. de Figueiredo, and L. Velho. Sketching variational hermite-rbf implicits. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium*, pages 1–8. Eurographics Association, 2010.

[9] G. Caumon, P. Collon-Drouaillet, C. Le Carlier de Veslud, S. Viseur, and J. Sausse. Surface-based 3d modeling of geological structures. *Mathematical Geosciences*, 41(8):927–945, 2009.

[10] J.J. Cherlin, F. Samavati, M.C. Sousa, and J.A. Jorge. Sketch-based modeling with few strokes. In *Proceedings of the 21st spring conference on Computer graphics*, SCCG '05, pages 137–145, New York, NY, USA, 2005. ACM.

[11] R. Cockett. Visible geology. In *http://app.visiblegeology.com/*.

[12] J.M. Cohen, J.F. Hughes, and R.C. Zeleznik. Harold: A world made of drawings. In *Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*, pages 83–90. ACM, 2000.

[13] G.J.P. de Carpentier and R. Bidarra. Interactive gpu-based procedural heightfield brushes. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, pages 55–62. ACM, 2009.

[14] G. Farin. *Curves and surfaces for CAGD: a practical guide*. Morgan Kaufmann, 2001.

[15] H. Fossen. *Geologi, Stein, mineraler, fossiler og olje*. Fagbokforlaget, 2008.

[16] J. Gain, P. Marais, and W. Strasser. Terrain sketching. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*, I3D '09, pages 31–38, New York, NY, USA, 2009. ACM.

[17] Google. Sketchup. In *http://www.sketchup.com/*.

[18] H. Hnaidi, E. Guérin, S. Akkouche, A. Peytavie, and E. Galin. Feature based terrain generation using diffusion equation. In *Computer Graphics Forum*, volume 29, pages 2179–2186. Wiley Online Library, 2010.

[19] T. Igarashi, S. Matsuoka, and H. Tanaka. Teddy: a sketching interface for 3d freeform design. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 409–416, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[20] M.W. Jessell. Noddy: an interactive map creation package. *Unpublished MSc Thesis, University of London*, 1981.

[21] B. Kelk and K. Challen. *Experiments with a CAD system for spatial modelling of geoscientific data*. 1992.

[22] E.M. Lidal, H. Hauser, and I. Viola. Geological storytelling: graphically exploring and communicating geological sketches. In *Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling*, pages 11–20. Eurographics Association, 2012.

[23] J.L. Mallet. Discrete smooth interpolation in geometric modelling. *Computer-aided design*, 24(4):178–191, 1992.

[24] JL Mallet. Gocad: A computer aided design program for geological applications. *Three-dimensional modeling with geoscientific information systems*, 354:123–142, 1992.

[25] J.L. Mallet. Discrete modeling for natural objects. *Mathematical Geology*, 29(2):199–219, 1997.

[26] T. Möller, E. Haines, and N. Hoffman. *Real-time rendering*. AK Peters Limited, 2008.

[27] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool. *Procedural modeling of buildings*, volume 25. ACM, 2006.

[28] M. Natali, Lidal E.M., I. Viola, and D. Patel. Modeling terrains and subsurface geology. In *Eurographics 2012 (approved)*. Eurographics Association, 2013.

[29] M. Natali, I. Viola, and D. Patel. Rapid visualization of geological concepts.

[30] J. Neal, D. Risch, and P. Vail. Sequence stratigraphy—a global theory for local success. *Oilfield review*, (2):51–62, 1993.

[31] J. Olsen. Realtime procedural terrain generation. *Department of Mathematics And Computer Science (IMADA)*, 2004.

[32] Y.I.H. Parish and P. Müller. Procedural modeling of cities. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 301–308. ACM, 2001.

[33] D. Patel, S. Bruckner, I. Viola, and EM Groller. Seismic volume visualization for horizon extraction. In *Pacific Visualization Symposium (PacificVis), 2010 IEEE*, pages 73–80. IEEE, 2010.

[34] D. Patel, C. Giertsen, J. Thurmond, J. Gjelberg, and E. Grller. The seismic analyzer: Interpreting and illustrating 2d seismic data. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1571–1578, 2008.

[35] A. Peytavie, E. Galin, J. Grosjean, and S. Merillou. Arches: a framework for modeling complex terrains. In *Computer Graphics Forum*, volume 28, pages 457–467. Wiley Online Library, 2009.

[36] A. Peytavie, E. Galin, J. Grosjean, and S. Merillou. Procedural generation of rock piles using aperiodic tiling. In *Computer Graphics Forum*, volume 28, pages 1801–1809. Wiley Online Library, 2009.

[37] Schlumberger. Petrel seismic interpretation software. schlumberger information solutions (sis).

[38] R. Schmidt and K. Singh. Sketch-based procedural surface modeling and compositing using surface trees. *Computer Graphics Forum*, 27(2):321–330, 2008.

[39] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pages 517–524. ACM, 1968.

[40] S. Spencer. *ZBrush Character Creation: Advanced Digital Sculpting*. Sybex, 2008.

[41] O. Šťava, B. Beneš, M. Brisbin, and J. Křivánek. Interactive terrain modeling using hydraulic erosion. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 201–210. Eurographics Association, 2008.

[42] F.P. Tasse, J. Gain, and P. Marais. Enhanced texture-based terrain synthesis on graphics hardware. In *Computer Graphics Forum*. Wiley Online Library, 2012.

[43] A.K. Turner. *Three-dimensional modeling with geoscientific information systems*, volume 354. Springer, 1992.

[44] A.K. Turner. Challenges and trends for geological modelling and visualisation. *Bulletin of Engineering Geology and the Environment*, 65(2):109–127, 2006.

[45] N. Watanabe and T. Igarashi. A sketching interface for terrain modeling. In *ACM SIGGRAPH 2004 Posters*, SIGGRAPH '04, pages 73–, New York, NY, USA, 2004. ACM.

[46] R.C. Zeleznik, K.P. Herndon, and J.F. Hughes. Sketch: an interface for sketching 3d scenes. In *ACM SIGGRAPH 2007 courses*, page 19. ACM, 2007.

Appendix A

Project Description

In the following pages, the project description document that was made at the beginning of the project is included. As can be observed, some changes to the planned approach were made, but the main idea of sketching inside a cube is basically the same.

Description of master thesis

Morten Bendiksen

Mars 2011

Rapid modeling of geological structures

Geologist often make sketches of geological structures, both in order to communicate ideas amongst themselves, and to other interested people. We propose to develop a computer program to aid in this sketching.

In developing this program, the following techniques will be explored:

- Having an initial empty sandbox from which the structures can be "carved"
- Drawing layers by turning and sketching on the sides of the box
  - Constraints on drawing. No layer-/self-intersection
  - Layers will be interpolated from this
  - Modifying layers by sketching on them and pushing or pulling
- Drawing rivers by sketching on the surface of horizons
  - Will carve out a plausible river following this path
  - Allows adjustments of size and depth
  - Adjusting of sealeves determines deposits
- Drawing ridges and valleys
  - Drawing the path of the new feature on existing terrain
  - Adjust look of contour along this path
  - Possibly change the height by similar approach as in Teddy
- Picking layers with mouse pointer
  - Further editing of layer is then possible

- Changing color
- Setting tranparency

- Expanding sketch with new cubes
  - Drawing layers in the new cube might use the edges of layers in adjacent cube
  - While drawing lines will snap to existing lines
- Representations
  - The terrain features will be modeled and stored by some sort of implisit representation
  - Might be possible to go back and forth in the history of this structure to make changes withouth loosing later work
- Procedural geologic modeling
  - Generate deposits by specifying terrain and sea level and such
  - Changes to the generated structures might change the initial conditions needed to make such an end result
- Visualization options
  - Generating simulations of seismic data

Illustrations of use case

Here we show a possible sequence of manipulations to quickly create some geological structures in a scene.

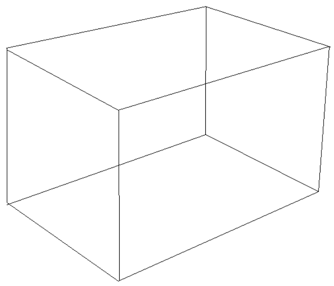


Figure 1: We start with the empty box

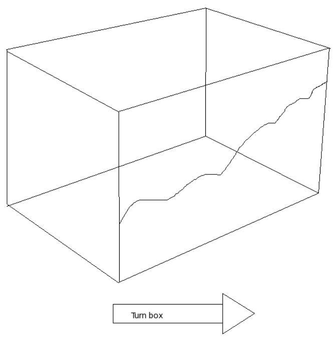


Figure 2: We draw the imagined layer in the box by turning it and drawing on the sides

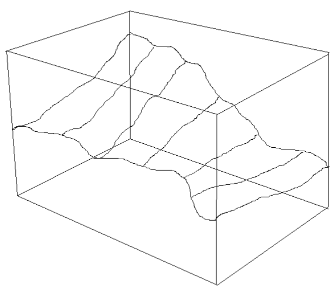


Figure 3: A layer is interpolated from the four sides we draw

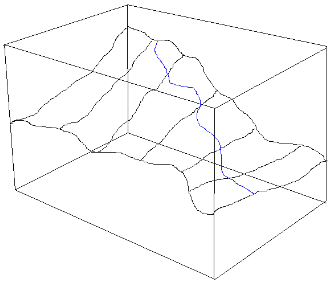


Figure 4: We draw a river path on this layer

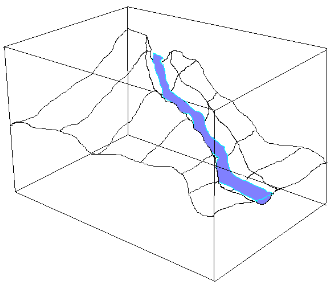


Figure 5: The computer will carve out from this layer as needed to make a river follow this path in a plausible way

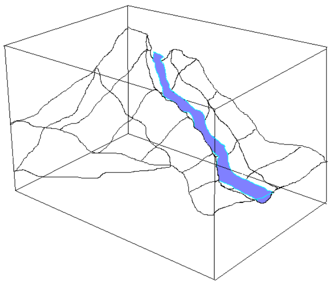


Figure 6: Now we draw a new layer. This can use the previous layer as a drawing surface in stead of only the sides of the box

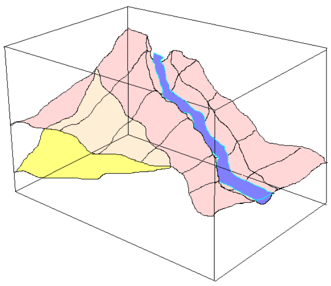


Figure 7: We add some color to the layers. In this figure the layers are partially transparent.

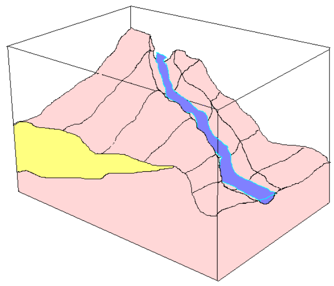


Figure 8: Here we have turned of tranparency and the sides become opaque

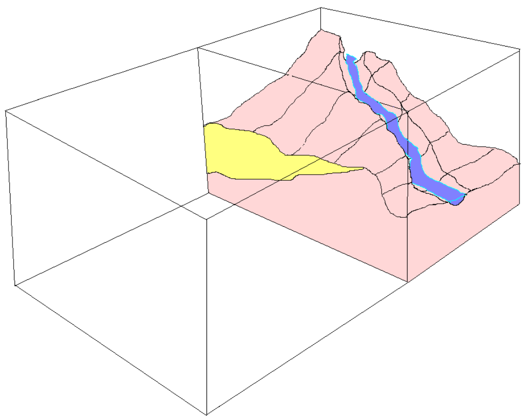


Figure 9: Now we might add a new cube to expand our drawing

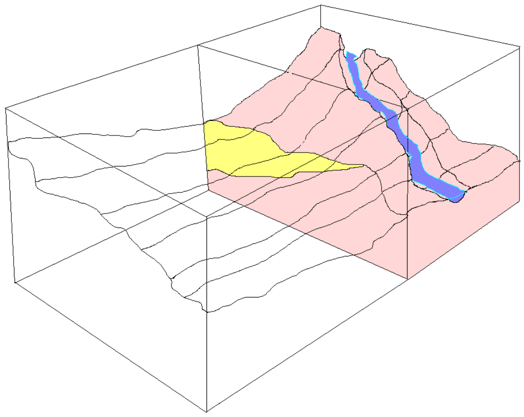


Figure 10: Drawing a new layer. Lines snap towards existing lines in adjacent cube.

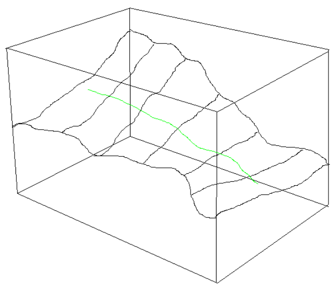


Figure 11: To create a new ridge or valley, start by drawing the ridge or valley path on the existing terrain.

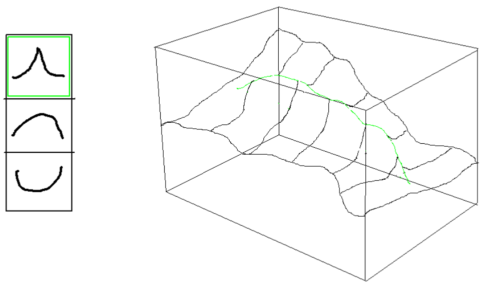


Figure 12: Now chose what the contour will look like and the new feature will be created. It should be possible to create different contours along different positions on the path drawn.

Appendix B

User Study Results

In the following pages a summary of the user study is included. The study form was created using the Form feature of Google Drive. The complete results in a spreadsheet can also be found at the following link:

User study spreadsheet: http://tinyurl.com/geoSketchUserStudy

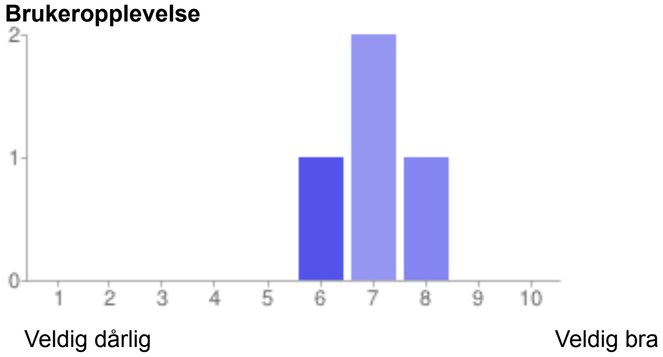
4 responses

Summary See complete responses

Ditt navn

Daniel   Anne Stensland   Marie   Jørn Morten Aadneram

Brukeropplevelse

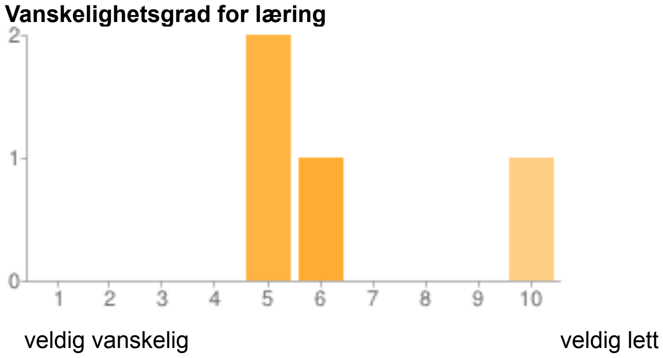


1 - Veldig dårlig	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**1**	25%
7	**2**	50%
8	**1**	25%
9	**0**	0%
10 - Veldig bra	**0**	0%

Hva var de gode og dårlige egenskaper i din opplevelse av programmet.

Vanskelig oppsett av meny. Selve tegneverktøyet var funksjonelt og fint.   Programmet gjør illustrering av geologiske fenomener lettere. Nyttig verktøy å ha i forelesninger.   Lett å framstille prinsipielle romlege, 3D-illustrasjonar.   Genial i sin enkelhet, men det er dog dens enkelhet som setter begrensninger for mer innfløkte oppgaver. En kan raskt sette opp / lage / tegne en enkel og fin geologisk modell av både morphologi, havnivå og teste ut hvordan systemet ditt endres over tid. Problemet er at en ikke kan legge til "dypereliggende" lag eller tegne elver/fjellformasjoner etc på tvers av   ...

Vanskelighetsgrad for læring

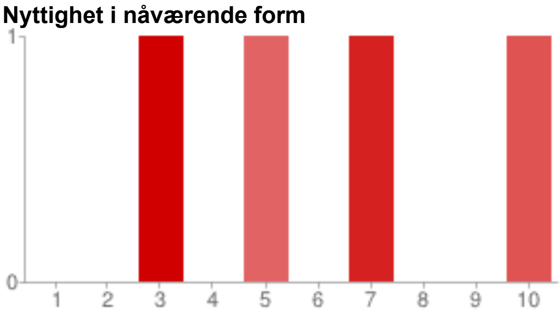


1 - veldig vanskelig	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**2**	50%
6	**1**	25%
7	**0**	0%
8	**0**	0%
9	**0**	0%
10 - veldig lett	**1**	25%

Hva gjorde det lett å bruke programmet og hva gjorde det vanskelig?

Kunne vært mye lettere å bruke programmet dersom de ulike verktøyene hadde vært synlige i en meny til en hver tid. Det samme gjelder de ulike objektene man lager (horisonter, lag, elver, fjell, daler, osv). Disse burde dukke opp etterhvert som man lager dem. Programmet er meget brukervennlig og det er lett å finne frem på siden. Det er litt vanskelig å orientere seg i begynnelsen, men med litt øvelse blir det mye bedre. Aldri benytta 3D-illustrasjonsprogram før, så lite erfaring på området. Det vanskelege var difor i hovudsak å venne seg til framgangsmåten, noko som i og for seg gjekk ganske ...

Nyttighet i nåværende form



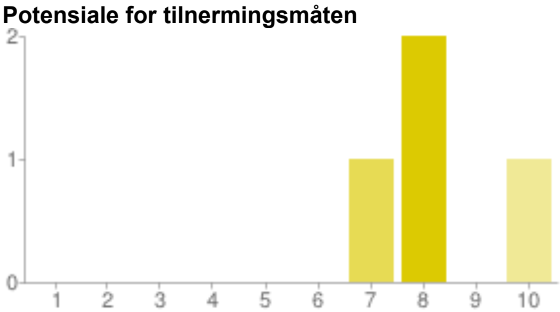
helt unyttig veldig nyttig

1 - helt unyttig	**0**	0%
2	**0**	0%
3	**1**	25%
4	**0**	0%
5	**1**	25%
6	**0**	0%
7	**1**	25%
8	**0**	0%
9	**0**	0%
10 - veldig nyttig	**1**	25%

Hva gjorde programmet mer/mindre nyttig?

Nå kan programmet muligens være nyttig til helt enkle skisser av den enkleste geologi, men videre forbedring tror jeg trengs for at dette skal være nyttig nok. Hva gjelder forbedring, ville muligheter for å illustrere forkastninger vært en veldig nyttig funksjon. Når dette er sagt, så vet jeg ikke hvilke alternativer som finnes i akkurat denne kategorien av 3D-illustrasjonsprogrammer. Programmet er nyttig for illustrasjoner av enkle systemer, men enkelte faktorer burde forbedres. F.eks burde det gått an å endre bredden på fjell og dybden på daler. Veldig nyttig for hurtig visuell framstilling ...

Potensiale for tilnermingsmåten



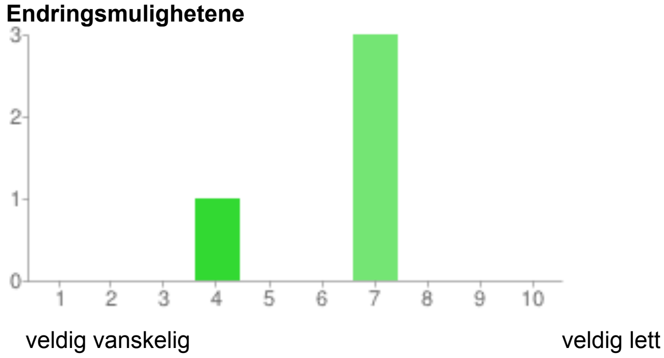
helt unyttig veldig nyttig

1 - helt unyttig	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**0**	0%
7	**1**	25%
8	**2**	50%
9	**0**	0%
10 - veldig nyttig	**1**	25%

Hva kan være fordeler og ulemper med dene tilnermingsmåten til å lage modeller/skisser?

Ved grundig videreutvikling av programmet tror jeg det kan være meget nyttig. Men da må brukervennligheten økes som nevnt over. Metoder som blir brukt i dag er ofte manuell tegning, det er tidskrevende og krever visse tegne ferdigheter. Dette programmet skaper hurtige illustrasjoner. Detaljnivå kan fungere som både fordel og ulempe. Tilnærmingsmåten begrensar seg til skissering av ein viss type samansatt miljø. vil ikkje vere tilstrekkelig til å skissere all verda av scenario. Fordel med tilnærmingsmåten er at programmet fungerer som eit veldig effektivt verktøy for ei hurtig framstilling, ...

Endringsmulighetene



1 - veldig vanskelig	**0**	0%
2	**0**	0%
3	**0**	0%
4	**1**	25%
5	**0**	0%
6	**0**	0%
7	**3**	75%
8	**0**	0%
9	**0**	0%
10 -veldig lett	**0**	0%

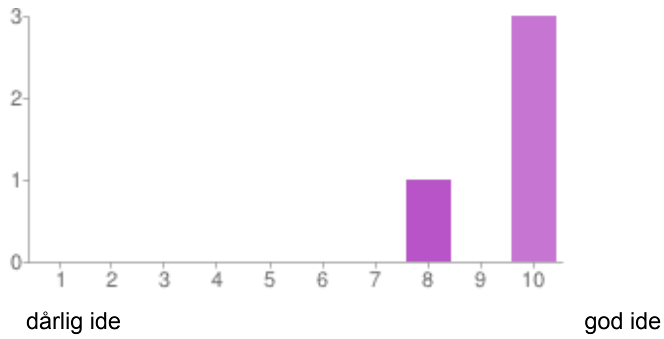
Hvordan kunne endringsmulighetene være lettere?

Syns dette var vanskelig, og dette kommer nok av at brukermenyen og verktøylinjene var ganske forvirrende. Hadde disse vært slik som foreslått, ville dette automatisk gått mye enklere. Det kunne vært en knapp som gjorde kuben het blankt igjen. Endringsmoglegheitane kunne ha vore lettare dersom ein hadde ein viskelærfunksjon tilgjengeleg, som gjorde det mogleg med mindre justeringar. At det går an å justere i breidda og høgda på fjell er bra, det same gjeld for yttergrensene til elveløp. Ein burde i tillegg kanskje hatt moglegheita til å justere djubda på dalar og elveløp. Veldig lett å gjøre e ...

Spesifikke deler av programmet

Vennligst gi en evaluering av hver funksjon i programmet, og kommenter hva som er begrunnelsen og hva som kunne være bedre/annerledes.

Kuben som utgangspunkt

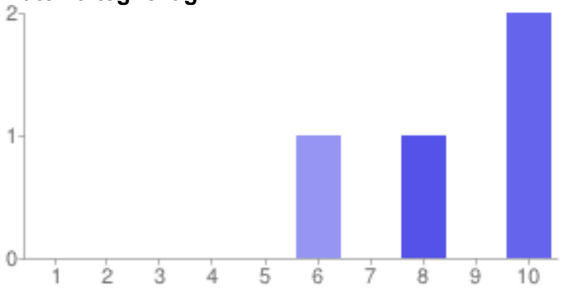


1 - dårlig ide	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**0**	0%
7	**0**	0%
8	**1**	25%
9	**0**	0%
10 - god ide	**3**	75%

Kommmentarer til kuben som utgangspunkt

Super ide. Selve tegnemåten var også bra. Veldig enkel og forståelig. Veldig greit at man kan tegne horisonten/flaten fra en vertikal vegg i kuben, og i tillegg kunne endre denne som man vil og fra hver vinkel man vil. Kuben fungerer godt som utgangspunkt, muligheita til å rotere på denne undervegs i teikneprosessen bidrog til å gi eit godt overblikk. 3D - En har et fast lukket system som en kan se på ifra alle vinkler

Måten å tegne lag



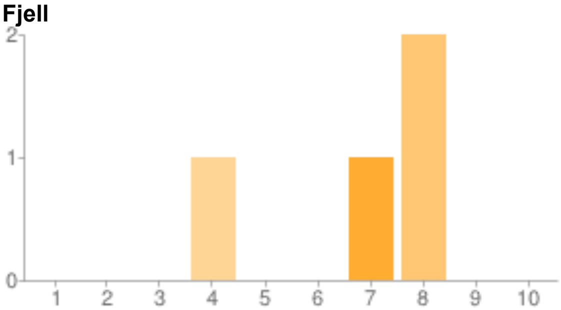
dårlig løsning god løsning

1 - dårlig løsning	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**1**	25%
7	**0**	0%
8	**1**	25%
9	**0**	0%
10 -god løsning	**2**	50%

Kommmentar til måten å tegne lag

Se over. Effektiv måte å få framstilt "lagkake-modellar". Lett å gløyme i byrjinga at ein måtte trykke "make layer" for å få framstillt lag. Fint at ein sjølv fekk moglegheita til å velje farge på dei ulike laga. Setter opp hovedhelningen til laget for så å justere om en vil ha forhøyninger/fordypnigner.

Fjell



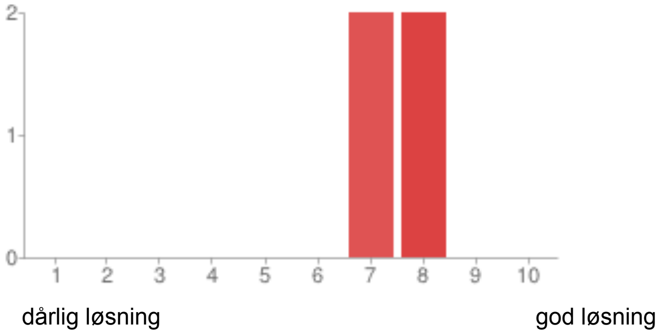
dårlig løsning god løsning

1 - dårlig løsning	**0**	0%
2	**0**	0%
3	**0**	0%
4	**1**	25%
5	**0**	0%
6	**0**	0%
7	**1**	25%
8	**2**	50%
9	**0**	0%
10 -god løsning	**0**	0%

Kommmentar til fjell

God løsning i grunn. Kunne kanskje lagt til muligheter for å modifisere fjellene ved å legge inn ytterligere vertikale plan i forskjellige retninger, slik at man kan modellere "nesten alt man vil". Bør kunne lage fjellene bredere. Litt tungvint å få til detaljert utforming av fjell, innarbeiding av fleire parameterar kunne ha betra dette. Bra at ein har moglegheita til å skalere høgd og breidd etter ein har angitt fjellområde. Veldig virkelighetstro, og med mulighetene til å justere høyden langs hele aksen er bra. Dens begrensning kommer når en vil prøve å sette sammen flere "fjell", resultat ...

Elver

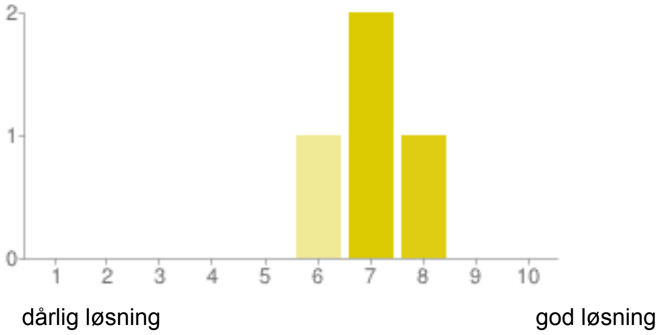


1 - dårlig løsning	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**0**	0%
7	**2**	50%
8	**2**	50%
9	**0**	0%
10 -god løsning	**0**	0%

Kommmentar til elver

Grei løsning. Ser ikke helt hvordan man skal kunne forbedre denne funksjonen. Kanskje det skulle gått an å legge inn sideelver osv. Elvene fungerer greit. Lett å skissere ynskja form på elveløp, samt justere breidda på ytterkantar som angir område for elv. Samme som ved fjell, evnen til å justere sinusiteten er bra

Daler

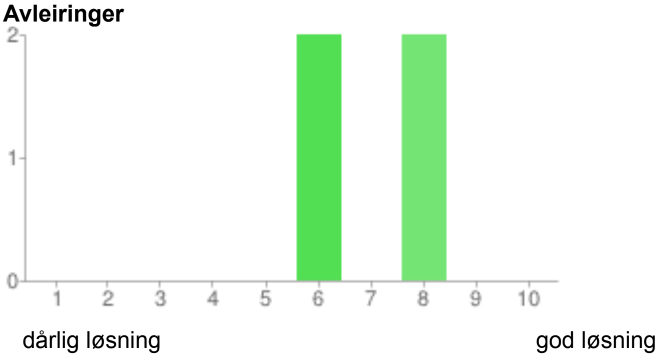


1 - dårlig løsning	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**1**	25%
7	**2**	50%
8	**1**	25%
9	**0**	0%
10 -god løsning	**0**	0%

Kommmentar til daler

Samme som for fjell.   Burde kunne lage dalene dypere.   - enkelt å angi område for dal. - kunne blitt bedre dersom ein ved å teikne ei u/v-form fekk angitt form og djubde på dalen i tillegg til lengde.   veldig bra i og for seg, men føles mer som en tegner canyoner. Mitt forslag hadde vært å fått til en felles "fjell/daler" funksjon. Har en to fjell ved siden av hverandre blir nødvendigvis fordypningen imellom de en dal, og mer virkelighetstro

Avleiringer

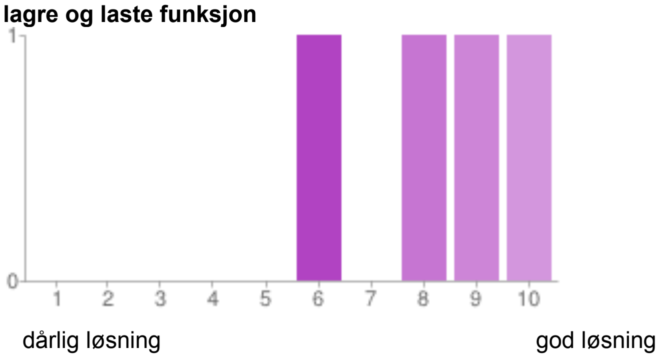


1 - dårlig løsning	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**2**	50%
7	**0**	0%
8	**2**	50%
9	**0**	0%
10 -god løsning	**0**	0%

Kommmentar til avleiringer

Usikker her. La merke til en bug i programmet der elven fortsatte å avsette sediment OVER havnivå. Elvene frakter sedimentene til der hvor havnivået ligger og begynner å avsette sediment der som følge av at elvens bevegelse/kompetanse oppheves.   Burde vært mulig se lagdeling i avleiringer.   - Funksjon for avleiringar fungerer fint som ein peikepinn på korleis transport i omr. vil fungere. Tar dog ikkje hensyn til parameterar som lithologi, transportlengd, forvitringsratar, etc. så angir berre eit omtrentleg bilete på kvar ein vil få avsetjingar, samt i kva mengd dei vil opptre.
Enkleste geologis   ...

lagre og laste funksjon

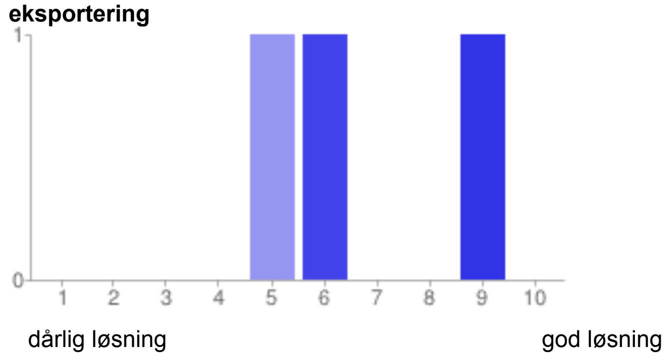


1 - dårlig løsning	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**1**	25%
7	**0**	0%
8	**1**	25%
9	**1**	25%
10 -god løsning	**1**	25%

Kommmentar til lagre og laste funksjon

Helt grei. Virker som funksjonen tilsvarer andre program. Fungerer bra. Ingenting å si, den lagrer og laster det det skal

eksportering

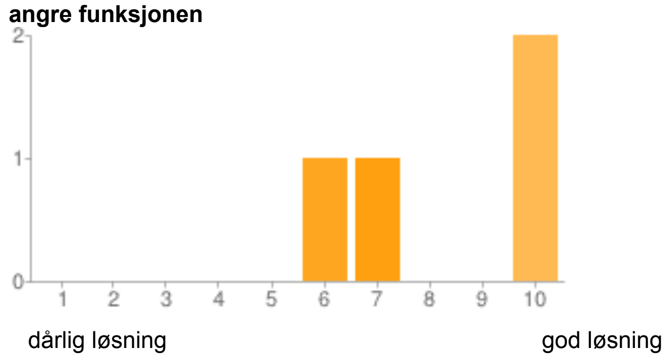


1 - dårlig løsning	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**1**	25%
6	**1**	25%
7	**0**	0%
8	**0**	0%
9	**1**	25%
10 -god løsning	**0**	0%

Kommmentar til eksportering

Virker også helt grei. fikk ikke muligheten til å prøve, men ble vist at det lett kan eksporteres til andre tegneprogram

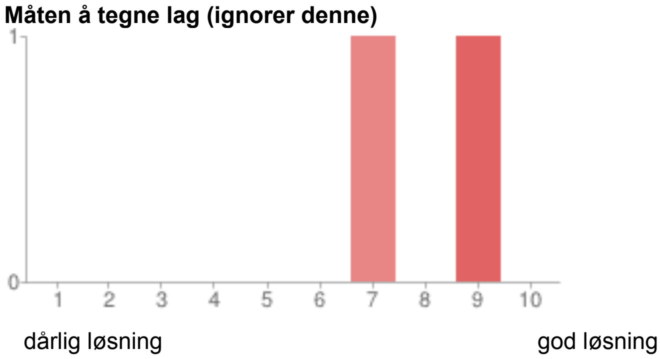
angre funksjonen



1 - dårlig løsning	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**1**	25%
7	**1**	25%
8	**0**	0%
9	**0**	0%
10 -god løsning	**2**	50%

Kommmentar til angre funksjon

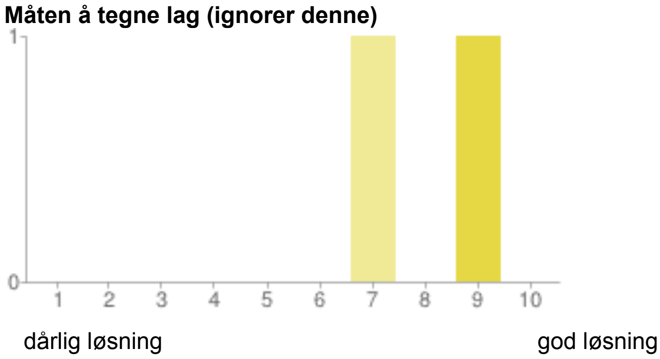
Her virket det som om at ved bruk av funksjonen så kunne programmet hoppe flere enn ett steg tilbake. Dette burde selvsagt endres på. Opplevde at programmet "låste seg" litt under bruk av denne. Ellers heilt ok. Fint at ein ved eit tastetrykk kan angre heile forrige innteikning. Som den skal



1 - dårlig løsning	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**0**	0%
7	**1**	25%
8	**0**	0%
9	**1**	25%
10 -god løsning	**0**	0%

Kommmentar til måten å tegne lag (ignorer denne)

God løsning å generere lag utifra horisontene man tegner. Kanskje det burde være en mulighet også å generere nye lag lavere enn tidligere tegnede lag.



1 - dårlig løsning	**0**	0%
2	**0**	0%
3	**0**	0%
4	**0**	0%
5	**0**	0%
6	**0**	0%
7	**1**	25%
8	**0**	0%
9	**1**	25%
10 -god løsning	**0**	0%

Kommmentar til måten å tegne lag (ignorer denne)

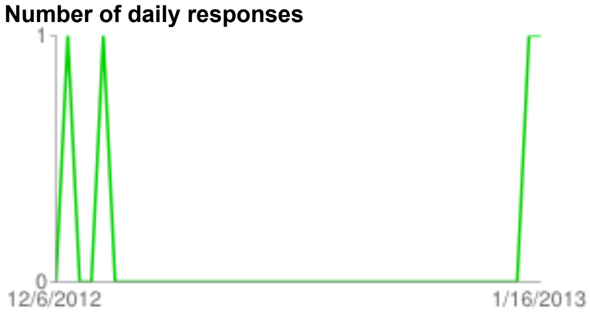
Samme som over.

Sluttkommentarer

Om du har noen flere kommentarer, forslag og forbedringsforslag, gi dem her

Er imponert over programmet du har klart å få til. Det er garantert mye å tenke på for å få til så mye. Jeg har hatt INF109 tidligere, og selv det enkle faget kunne ha omfattende programmering, så skjønner veldig godt at dette må være kraaaazy!! Når det er sagt så kunne det vært enklere og mer brukervennlig. Har prøvd å komme med så gode forslag så jeg kan. Er også vanskelig å vite hva som lar seg gjøre og ikke sett fra et programmerings-ståsted. Her er noe jeg skrev ned i word da jeg jobbet med programmet (mye det samme som jeg har nevnt allerede): "Burde ha en meny for alle muligheter ma ...

Number of daily responses



Appendix C

Contact details

Please contact me if you have any questions regarding this thesis.

Email: morten.bendiksen@gmail.com