Helsinki Metropolia University of Applied Sciences
Degree Programme in Media Technology

Lassi Haaranen

3D short animation utilizing distributed computing

Final Year Project. 19 November 2009

Instructor: Julius Tuomisto, Project Manager
Supervisor: Harri Airaksinen, Director

Helsinki Metropolia University of Applied Sciences **Abstract**

Author Title Number of Pages Date	Lassi Haaranen 3D short animation utilizing distributed computing 60 19 November 2009
Degree Programme	Media Technology
Degree	Bachelor of Engineering
Instructor Supervisor	Julius Tuomisto, Project Manager Harri Airaksinen, Director

The purpose of this study was to create a short 3D animation for Open Rendering Environment (ORE) project. The most significant result of ORE project is a web service called www.renderfarm.fi where volunteers can donate their computer's extra processing power to render 3D animations.

As an idea distributed computing is not new but it has been rarely used outside of research and science. ORE project seeks to remedy this by bringing the advantages of distributed computing to the art and culture while creating a community around the service. ORE project is mostly funded by TEKES and it is carried out at Laurea University of Applied Science.

Modeling and animating the short movie took about three months. Models were created for three separate files (overall size of these files was under four megabytes) and the length of the final animation was 15 seconds. All the tools and services used to create the animation were free and open. Inspired by this, all the model and project files were released under an open license. This means that anyone can receive, continue and produce new animations based on these resources.

The animation was rendered with the ORE service when it was technically possible. Only one scene containing physical simulation of the cloth was rendered using one local machine. As a result of this study, a short animation for advertising purposes was made. Also, all the files needed during the process were created and they can be reused later to make new animations.

Keywords	rendering, 3D, modeling, animation, Blender, ORE, BURP, distributed, computing, texturing

Metropolia Ammattikorkeakoulu **Insinöörityön tiivistelmä**

Tekijä	Lassi Haaranen
Otsikko	3D-lyhytanimaatio käyttäen hajautettua laskentaa
Sivumäärä	60 sivua
Aika	19.11.2009
Koulutusohjelma	mediatekniikka
Tutkinto	insinööri (AMK)
Ohjaaja	projektipäällikkö Julius Tuomisto
Ohjaava opettaja	yliopettaja Harri Airaksinen

Insinöörityön tarkoituksena oli 3D-mallintaa lyhytanimaatio Open Rendering Environmentin (ORE) käyttöön. ORE-projektin merkittävin näkyvä tulos on www.renderfarm.fi-palvelu, jossa vapaaehtoiset voivat luovuttaa tietokoneensa ylimääräistä laskentatehoa animaatioiden renderöintiin.

Hajautettu laskeminen ei ole ajatuksena uusi, mutta sitä on aikaisemmin valjastettu lähinnä vain tieteen ja tutkimuksen käyttöön. ORE-palvelu on yritys tuoda hajautetun laskennan antamat edut myös kulttuurin ja taiteen käyttöön sekä luoda yhteisö palvelun ympärille. ORE-projekti on pääosin TEKESin rahoittama ja Laurea Ammattikorkeakoulun toteuttama projekti.

Mallinnus ja animointi kesti yhteensä noin kolme kuukautta. Kolmeen erilliseen tiedostoon luotiin animaatiossa tarvittavat mallit (näiden yhteiskoko jäi alle neljään megatavuun), ja valmista animaatiota syntyi 15 sekuntia. Kaikki animaation luonnissa käytetyt työkalut ja palvelut olivat ilmaisia ja avoimia. Tästä syystä myös lopullinen animaatio ja kaikki siihen liittyvät mallinnus- ja projektitiedostot päädyttiin julkaisemaan avoimen lisenssin alaisuudessa — kenellä tahansa on siis mahdollisuus ottaa, jatkaa ja tuottaa uusia animaatioita näitä resursseja käyttäen.

Animaatio hahmonnettiin (renderöitiin) käyttäen ORE-palvelua silloin, kun se oli teknisesti mahdollista. Tavallisesti yhdellä koneella renderöitiin vain fysiikkasimulaatiota sisältävä kohtaus. Työn lopputuloksena oli mainoskäyttöön tarkoitettu lyhytanimaatio ja sen luontiin käytetyt tiedostot, joiden pohjalta voidaan toteuttaa uusia animaatioita.

Hakusanat	renderöinti, hahmontaminen, 3D, mallinnus, animointi, Blender, ORE, BURP, hajautettu laskenta, teksturointi

Contents

Abbreviations and Terms

Blender	3D modeling software suite, which includes many other features as well. Such as texturing, animating, rigging, and video editing.
.**blend**	File format in which Blender saves it's files
Vertex	Single distinct point in 3D-space Face - A surface between (usually) four vertices.
Mesh	3d object that is formed by vertices and faces.
Rig	Collection of bones in Blender that control a mesh.
IPO	From the word interpolation. IPO-curves are used in blender for animating.
Skinning	Process of applying a rig to a mesh
Texture	Procedurally created or ready made image that is applied to material or used as a material.
Material	A complete set of textures and effects that is applied to the surface of a mesh when rendering
shader	A particular way of shading a certain material. Different shaders give different impressions of the surface
SSS	Subsurface Scattering. A method of scattering light in textures to achieve more natural and softer surfaces.
Rendering	The process of calculating a "ready" image from the 3d models and textures.
Distributed computing	Practice of sharing a heavy computational workload to smaller pieces and using a lot of computers to calculate it, usually facilitated through internet

BOINC	Berkeley Open Infrastructure for Network Computing. A framework for creating distributed computing projects.
BURP	Big and Ugly Rendering Project. BOINC-based rendering service (discontinued)
ORE	Open Rendering Environment. BURP-based rendering service
Work unit	Work package sent to a client computer in BOINC for computing.
GPL	GNU Public License. A software license which allows the modification of source code by all parties, as long as they release the modified source with the same license.
CPU	Central Processing Unit. Where actual computation happens in a computer.

1 Introduction

The goal of the Open Rendering Environment (ORE) project is to create a service for distributed rendering, which means that artists everywhere in the World can get access to a powerful rendering farm without a cost - enabling them to create more complex and challenging animations and images with higher resolutions. The technological background for ORE is formed by three pieces of software: Blender, BOINC, and BURP.

Blender is an open source 3D graphics suite. BURP (Big and Ugly Rendering Project) was designed by Janus Kristensen to render images using Blender's internal renderer. It utilizes distributed computing platform called BOINC, which is designed to enable people in different fields to develop distributed computing projects easily. Later on the technology behind BURP was moved to ORE project in the form of www.renderfarm.fi web service.

This thesis focuses on creation of a short animation for promotion purposes for the ORE project. It approaches the subject from three different points of view.

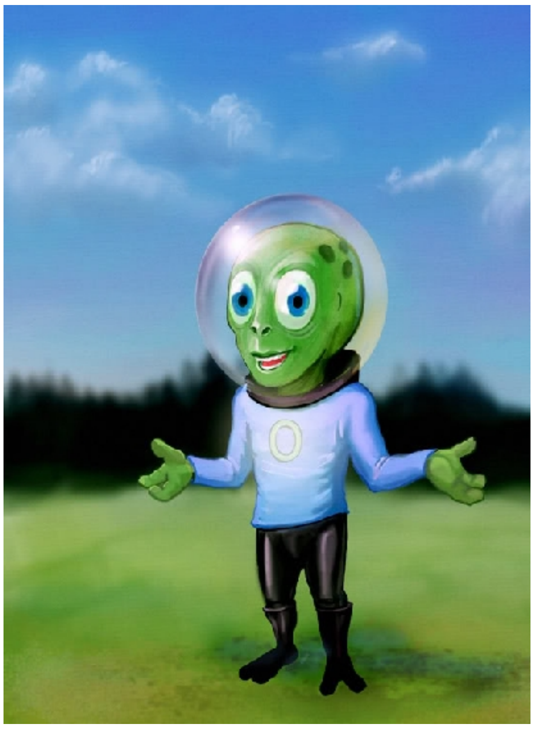
It firstly focuses on the creation of the animation from the traditional modeling and animation point of view (chapters 2 and 3). After that, attention is given to how the project was rendered using this new service and how the service in itself works (chapter 4).

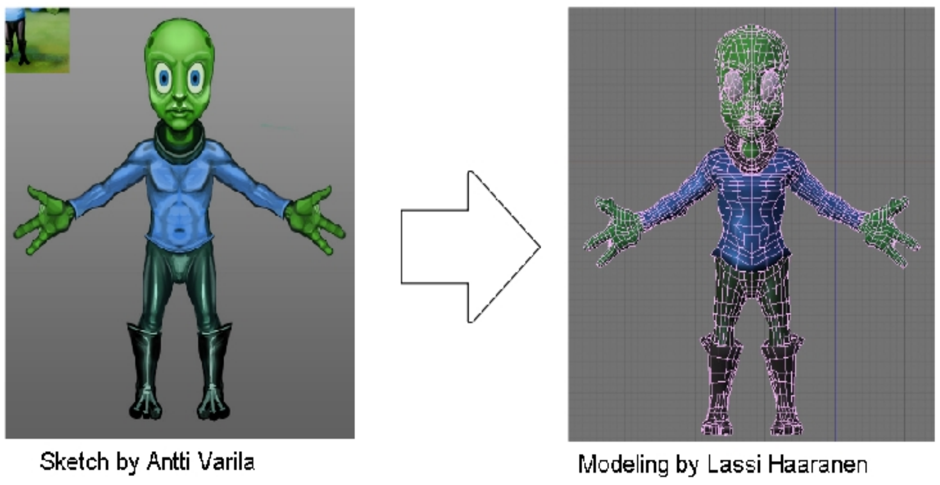
Finally it discusses the social side of this rendering rendering platform. Also included in this part is a small questionnaire about interest in this kind of communal creativity.

2 Designing and Creating the Character

2.1 Initial Sketching and Ideas for Animation

The initial concept of the character was discussed a couple of times with artist Antti Varila and project manager Julius Tuomisto at the beginning of autumn 2008. A humanoid like alien character was agreed upon and initial sketches were made by Varila. Based on the first sketch seen in picture 1, another sketch was made by Varila, seen in picture 2. The second sketch had had a front and a side view and was to be the basis of the modeling. Different ideas were about animations were planned from this point on. The mascot was named Ortho from the word orthogonal.


Picture 1: Original sketch for the mascot[1]



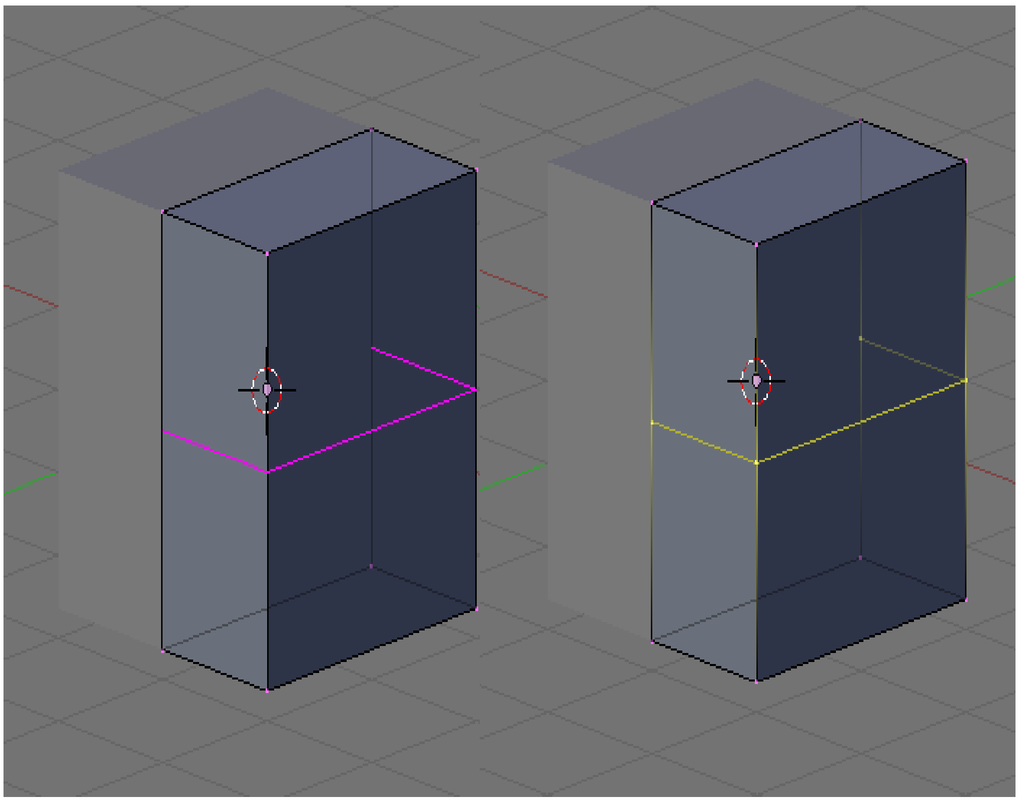
Picture 2: From Sketch to Model [2]

The finished product would be a short animation to be used for promotion of the service. Part of this promotion is also the openness of this animation: anyone can get the animation and the production files and start creating their own projects based on those. The animation also acts as a showcase for www.renderfarm.fi.

2.2 Modeling Ortho

While modeling the character Tony Mullen's Introducing Character Animation With Blender was co,nsulted frequently. In his book Mullen shows how he models, textures and animates a superhero character step by step. Workflow and basic modeling principles of this project were mainly taken from Mullen's book and from personal experience from previous projects. [3]

Rough modeling of the shape was primarily done using a technique called box modeling. In box modeling the work is begun with a simple cube and geometry is gradually added to it to achieve the desired shape. It helps to keep the geometry of the object being modeled relatively simple and malleable which is important in character modeling. This is especially true when there will be complex animation (such as bipedal walking).


Picture 3: Subdividing the cube

Starting the modeling from a cube can be done in two different ways. One way of doing this is by extruding more geometry to the object and the other one is dividing the cube to smaller pieces and shaping those like in picture 3. Both methods were used in this production.

Blender has a wide variety of modifiers to help the modeling process which were also frequently used [4]. The most important of these was a mirror modifier, which enables mirrored modeling essential when modeling symmetrical shapes, such as most creatures [4]. Mirror modifier was set to reflect z-axis, so that Ortho's left side was created automatically while the right side was modeled.

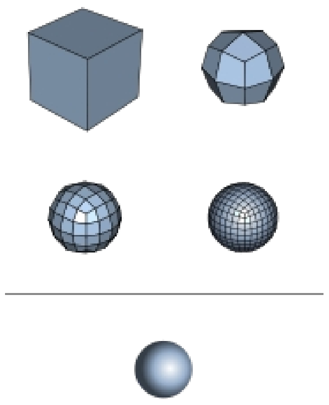
Modeling was based on Varila's second sketch [2], which included front and side view. The sketch was placed on the background in Blender and used as a reference when modeling. Mullen [3] starts his modeling from the legs; in the case of Ortho it was started from the most defining characteristic, his head. Geometry was added to the cube by loop cuts and the shape was gradually defined to resemble the original sketch. During the construction of face, special focus was given to the complexity of the mesh as the goal was to keep it as simple as possible to ease the animation of the facial expressions.

After finishing the head torso, hands, legs, boots and collar were modeled in a similar manner. Starting from a box and refining the shape until it matched the original sketch or was otherwise deemed as ready. For usability reasons all these meshes were kept separate until they were ready and only then joined. Having different pieces separate allowed keeping them in separate layers, which meant that it was possible to hide and show them as need be, allowing easy access and visibility from all directions to the relevant part that was being modeled.

After the shape of the head was done, focus was turned to the eyes. Because of the planned close up scene, they needed to be as expressive as possible. A lot of the details and ideas for textures were taken from Building a Better Eyeball Tutorial by Jon McKay [5]. Since the current animation didn't call for animating the pupils, the vertex keys to handle pupil dilation from McKay's tutorial were not made. However, they could be easily implemented later on if necessary.

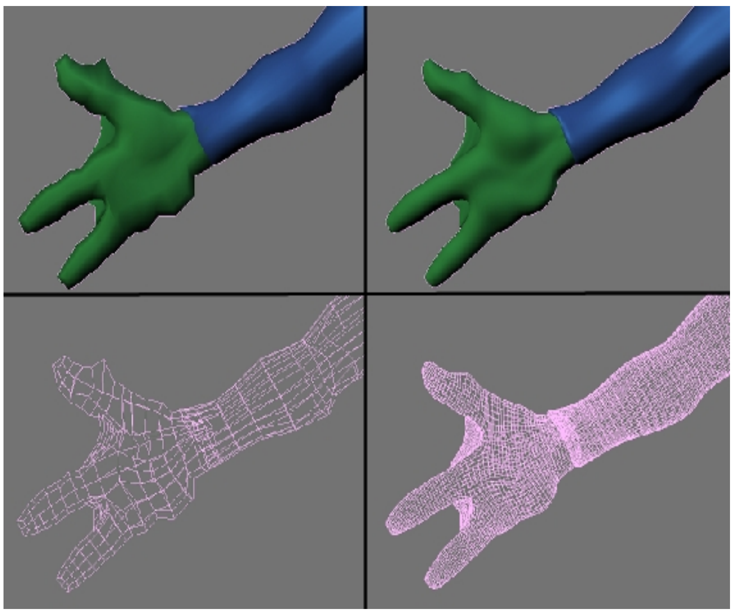
While the eyes received a lot of detail, the eyelids were not modeled due to time constraints. In retrospect, this might have been a poor decision since the eyelids give a lot of expressive detail to the face. And adding anything to finished model afterwards can be cumbersome.

In addition to mirroring subdivision, a surface modifier was also extensively used throughout the process. Sub surface modifier divides to surface in such a manner that it gives it a more rounder and natural shape. This can be seen in picture 4 where cube (in the upper left corner) is given five levels of subdivision.



Sub surface modifier divides the current faces into two or more pieces giving more detail and smoother result. While modeling two levels of subdivision were used to give a general idea how the modifier would affect the final work. In the final rendered version three levels of subdivision were used to ensure that no jagged edges or rough shapes were present in the models. The subdividing method utilized was Catmull-Clark [7] which gives a natural and smooth result suited for organic modeling.

Without the sub surface modifier the character of Ortho uses little less than 4,000 vertices (see left side of picture 5) and with two levels of sub surfing about 60,000 vertices (right side of picture 5). Vertex count rises up to to 240,000, with three levels of sub surfing, which is used while rendering. While the increase in time to render is considerable (it doubles or triples the time in a simple scene with only monkey primitive found in Blender in the scene with two lamps) when so much more geometry is added to the model.

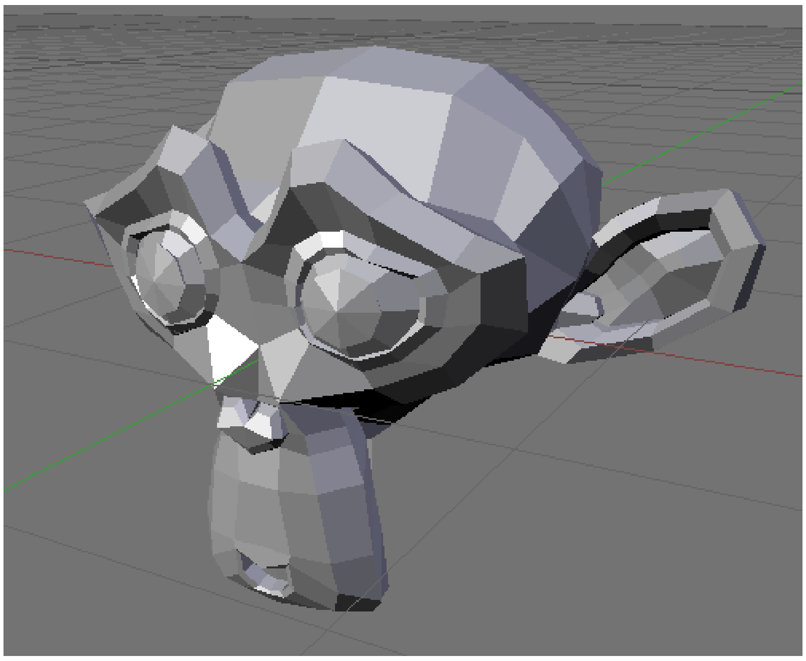


However, the amount of geometry does make a noticeable difference in the outcome in terms of smoothness, as can be seen on the picture 5. On the left side of the picture is the textured and wireframe versions of the character without any subsurfing which look more jagged and edgy than the subsurfaced versions on the right side.

2.3 Modeling Everything Else

Even though Ortho was definitely the most important part of the animation it also had other elements (see chapter 3.1 for script details to see how these models where used). A visually important part of the animation was of course the ORE logo, which was done before the animation was started. It is just a simple text converted into meshes.

Another element in the animation was the opening scene. The animation starts with theater curtains being down and a bunch of monkey, seen in picture 6, silhouettes as an audience.


Picture 6: Suzanne

The audience in front of the curtains were originally supposed to be drawn by hand, acting as a metaphorical link between 2D and 3D world. This would have been done by scanning the original drawing (the 2D part of the metaphor) and placing it in front of the 3D animated scene. Due to time constraints of the artist that was meant to do them this idea was scrapped and a new idea was chosen.

The audience in the final product consists of Blender's mascot monkey Suzanne, featured in picture 6. Blender has a built in monkey primitive of Suzanne, so the heads of the audience did not require any actual modeling at all. The tails were simply extruded cylinders, with narrowing at one end.

Accessories were originally planned for Ortho but these were skipped due to lack of time. Among some of the things that were planned were cloak, laser gun, some sort of spaceship, and a space helmet. In the end none of these were required for the actual animation, even though different ideas of using them in future short animations were bounced around.

Only space helmet was actually modeled but it was not used in the final animation. The helmet used a simple cube mesh which was split up five times to ensure enough vertices so that a smooth and round shape would be possible. After that a lattice modifier was added to it to get the desired shape (which would resemble shape of Otho's head) as easily as possible.

Even though modeling space ships were discussed, they were not designed or modeled. However, a space ship from an unrelated project was used in a promotional poster for a related flash game project in which Ortho was starring as well, see chapter 5.4 for details.

2.4 Texturing

The visual look planned for Ortho was a mixture of looking friendly and yet 'edgy'. Also, most of the desired visual look was achieved during the modeling phase which meant that most of the used textures were fairly simple and contained no special tricks. Texturing is the general process of applying different surfaces on top of the modeled geometry. [8]

Simplicity holds true for the cloth textures used in Ortho's pants and shirt. They have no ray tracing features or other special techniques, they are just Blender's basic textures with fairly high specular value to give shiny 'space age' look.

Because the large role of eyes and eye movement in the closeup scene they were formed with four different objects with each of them having their own material. The objects (and materials) are cornea, veins, pupil, and iris. Most of these are just normal materials, but iris uses procedurally created textures as a basis for the uneven look. Iris-like look is achieved by creating a texture with musgrave algorithm shown in picture 7. This was then mapped on to the eye with blend-type texture to get a smooth fading effect to the iris.

Ortho's skin in head and hands also utilizes more complicated materials. They use advanced light scattering method called subsurface scattering which tries to mimic the behavior of soft and smooth surfaces typically found in organic objects (such as skins, fruits, etc). These surfaces typically let some light through them which then bounces inside the object and finally scatters out at some angle. Subsurface scattering gives the material soft and natural look, an effect which can be hard to achieve using traditional shading methods. Blender's Subsurface scattering [9] is based on Henrik Wann Jensen's and Juan Buhler's paper "A Rapid Hierarchical Rendering Technique for Translucent Materials" [10] which was presented at ACM SIGGRAPH.

This method is based on two separate passes. First, the irradiance is calculated in certain points (points are found by Turk's point repulsion algorithm [10]). On the second pass the SSS shader is used instead of the normal light one - brightness of a point is based on its neighbors brightness.

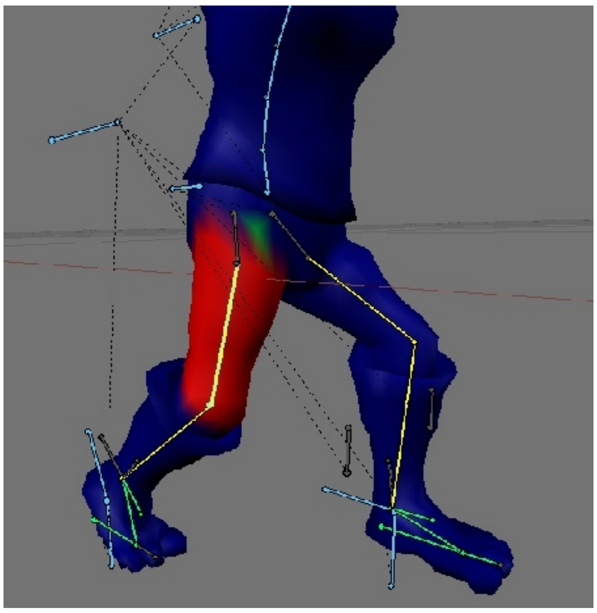
Because the curtain was a late addition and not originally planned readymade material was used for the curtain's rope and curtain in itself uses a basic material created for it. Material for the rope was taken from Blender Open Material Repository [11] and it is a fairly simple gold material without textures and reflection with ray tracing.

2.5 Rigging and Skinning

In order to animate something with more complexity and fluidity than simply having object's location, rotation or size differ more advanced techniques are called for. Making a skeleton to a 3D-character that allows movement of different limbs and appendixes is called rigging. Most of Ortho's rig is based on Mullen's [3] book. The only exception to this is the foot rig which is based on a different tutorial [12]. The created rig proved to be very handy and easy to work with when creating the walk cycle and rest of the animations.

The process of actually attaching bones of a rig to specific regions of model is called skinning. In Blender this can be done in a couple of ways. The simplest one is to select vertices by hand and create vertex groups from them and attach them to the desired bone. This technique was used mainly on the head where all of the head were require to move using a single bone (for instance to turn the head).

A more advanced technique of weight painting [13] was used for the rest of the bones. In weight paint mode different bones were selected and after that by painting with mouse cursor the desired areas of movement where chosen. The advantage of weight painting is that there can be vertices of differing weights near joints or other areas where complex deformations happen. This means that near joints vertices move a little when the bone connected to them is moved, but not as much as the vertices which are further away from joints. This is a crude way to mimic muscle movement in the character but compared to creating vertex groups by hand it is very efficient. No actual muscle or fat deformations were used in anything in the animation.

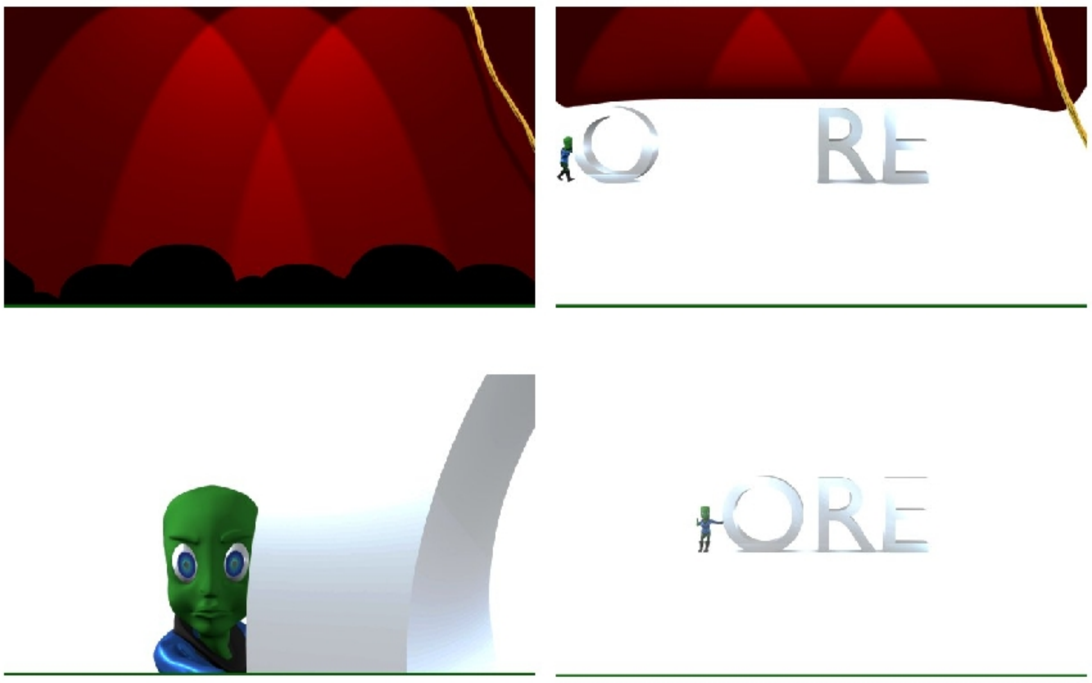


In picture 8 Ortho's leg and related weight map is created by painting. Blue areas mean that no deformation will happen when current bone is moved. Green areas deform slightly when the bone in question is moved and red areas move completely with the bone (unless there are other constraints and bones controlling those specific vertices). The weights of the vertices range from 0 to 1.000 and the colored weight paint map is simply a representation of that.

3 Creating Animation

3.1 Basic Script

Due to the relative complexity of the rig and the walk cycle of bipedals the actions chosen for the animation were fairly simple and straightforward to implement. The actual script and timeline developed itself fairly organically once the original of the animation idea was agreed upon.



Picture 9: Screenshots from The Final Animation

Basic script, depicted by screenshots on picture 9, is that the animation opens with theater curtains with silhouette of audience in front of them. Curtains are pulled up and the main scene is revealed.

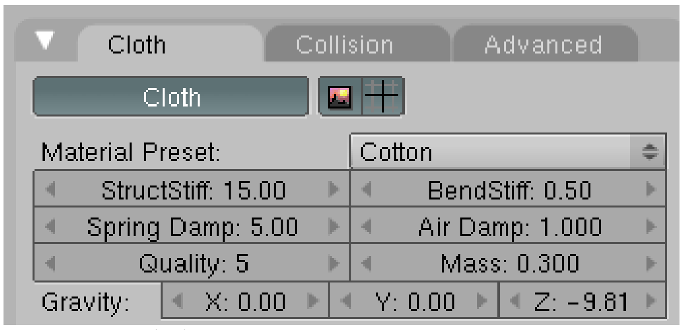
In the scene there is Ortho pushing O from ORE-logo to its place. He notices that curtains are already up and he is late so he glances at the audience and then hurries to put the O in its place. After that he shows the thumbs up and the show can begin. Finally the small text explaining the meaning of ORE drops down from the sky in a smooth manner.

Several different techniques for animating were used. The simplest ones were transformations of location and rotation. More advanced techniques involved shape keys for the facial expressions and a full bipedal rig for movement of the main character.

3.2 Curtains

The ability to simulate clothes and cloth-like movement is a hard problem to solve and several years of hard work was put into development of cloth simulation in Blender. It seems that Blender is the first free 3D software suite to add cloth simulation to its tools, which happened in version 2.46. [14]

One way of simulating cloth is to create it out of of masses (points) and then connect them to each other via springs. Blender's cloth algorithm is based on Xavier Provot's [15] method of calculating the deformations. In this method, additional calculation is performed in order to ensure that the cloth doesn't become super-elastic, creating a fabric that behaves like a sheet of rubber. This is done by adding additional stiffness to the springs, as shown in picture 10.



Material Preset: Cotton
StructStiff: 15.00 | BendStiff: 0.50
Spring Damp: 5.00 | Air Damp: 1.000
Quality: 5 | Mass: 0.300
Gravity: X: 0.00 | Y: 0.00 | Z: -9.81

Picture 10: Cloth parameters

Lengths of all the springs are noted down in the beginning of simulation and then recalculated in each frame. Values are given how much springs are allowed to stretch and contract. If the springs exceed the given stretching values (which would lead to rubber-like behavior), they are recalculated with their neighbors so that offending lengths are corrected. In relatively stiff cloths such as cotton, the stretching is usually lower than 10 per cent of the original length [15]. The curtains in the animation were simulated with Blender's cotton preset which allows 15 per cent of stretching, rest of the parameters were as seen in picture 10.

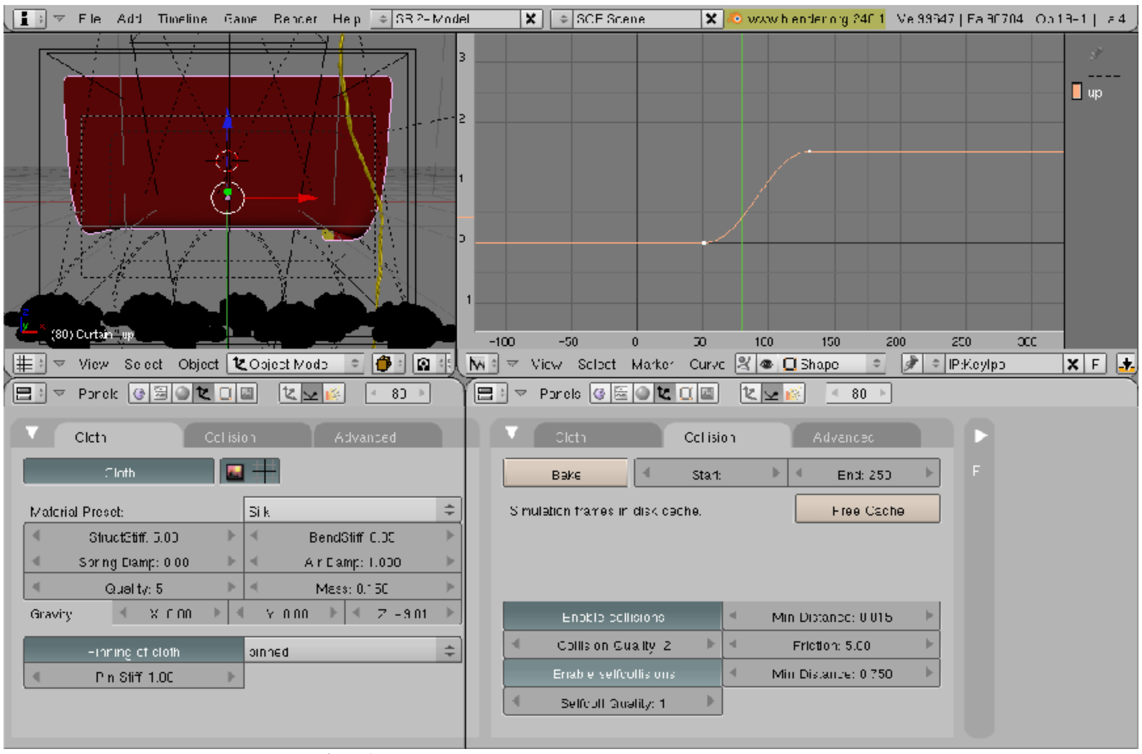
The animation starts with theater curtains being down and pulled up. Cloth-like movement of the curtains was done with a cloth-modifier. Some additional movement was added with a wave-modifier. Thea ctual lifting of the curtains was done with just two simple shape keys, in the first one the curtains are in a rest position (starting position) and in the other half a dozen of vertices in the bottom of the curtain were moved up above the rest of the curtain. Keys were added for the two shapes to the start and the end of the 100 frame animation. With the start and end frames being in place the only needed user interaction for the animation is to just start the calculation process.

There are many different tools in Blender which require computation before the actual result of these is visible. Usually these tools have something to do with physical simulation or particle systems. This process of calculating some animation with a given algorithm (be it simulation for cloth or water movement, or how particles behave in a particle system) is called baking in Blender [16].

Baking process means that Blender will calculate the different steps for the cloth to get from the starting position to the end position, taking all the modifiers into account. Baking creates number of small files on the computer it is run on that contain shapes of vertices or similar information that changes in the animation of the simulation.

Using modifiers and shape keys in this manner really simplifies the animation process and takes the workload from human animator to the computer. From the user side the "animating" only consisted of setting the parameters for the start of the animation as seen on picture 11.

The curtains were done in a separate .blend-file to keep the scenes as simple as possible. Everything in that is completely separate from the main scene, most notably including the lighting which was specifically to the curtains. The lighting of the Ortho and logo was done in the main file. Once the curtains were rendered to stack of images they were put on top of the animation with the sequence editor in blender.


File Add Timeline Game Render Help
SR:2-Model
SCE:Scene
View Select Object Object Mode
View Select Marker Curve Shape IP:Keylpo
Panels 80
up
(80) Curtain_up

Cloth | Collision | Advanced

Cloth
Material Preset: Silk
StructStiff: 5.00 | BendStiff: 0.05
Spring Damp: 0.00 | Air Damp: 1.000
Quality: 5 | Mass: 0.150
Gravity X: 0.00 Y: 0.00 Z: -9.81
Pinning of cloth: pinned
Pin Stiff: 1.00

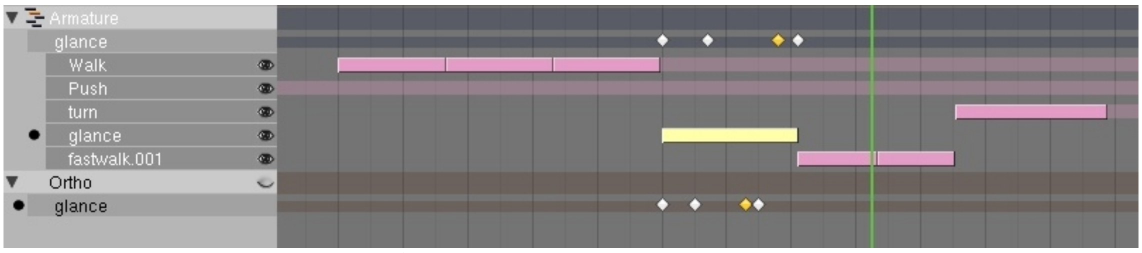
Bake | Start: | End: 250
Simulation frames in disk cache. | Free Cache
Enable collisions | Min Distance: 0.015
Collision Quality: 2 | Friction: 5.00
Enable selfcollisions | Min Distance: 0.750
Selfcoll Quality: 1

Picture 11: Start parameters for the curtains

The bobbing and tilting monkey head (seen in picture 11) animations at the start were done with simple location and rotation changes. The tails were animated by very simple armatures which consisted of chain of similar bones all linked to previous one and one inverse kinematic (IK) constraint at the tip of the tail [3, 130]. The IK constraint was used to achieve natural looking curving of the tail in a way that was not time consuming.

3.3 Actions of Ortho

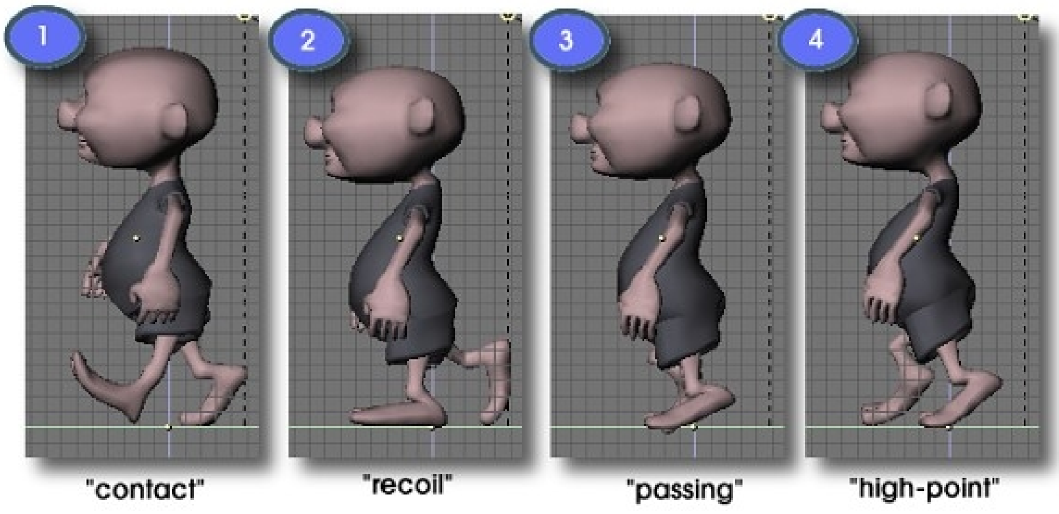
Movements of Ortho were divided into five separate actions: walk, faster walk, turning at the end, pushing and glance in the middle of the animation. These actions were edited in Blender's Action Editor and put together with Non-Linear-Animation tool (NLA), which was added in Blender version 2.40 [17]. Interface and the different actions used are seen in picture 12. NLA editor allows easy editing for different actions and their timings. It also makes it possible to blend together two actions that are happening at the same time.



Picture 12: Non-Linear Aniation tool

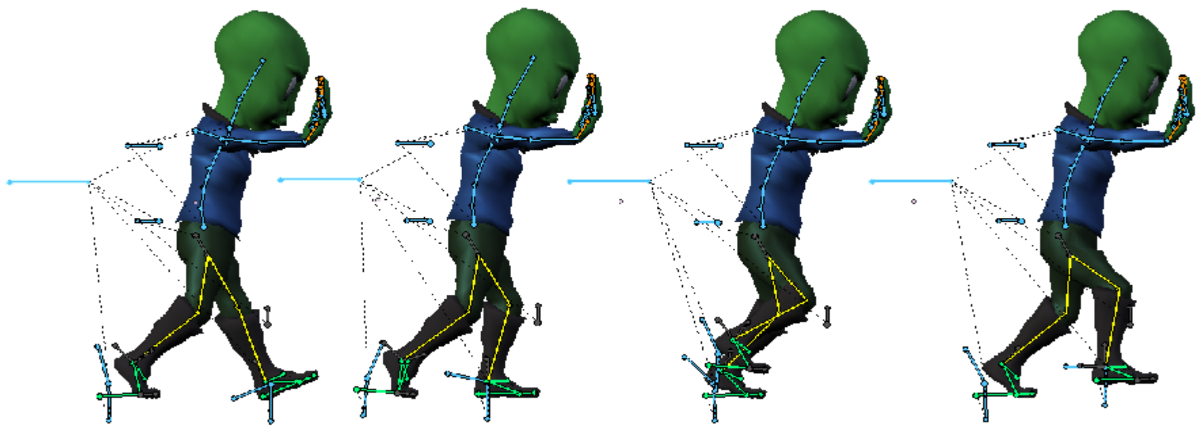
The most visible and important action is the walking (depicted by first three pink bars in picture 12), which was made using a technique of creating the movements of the legs and feet while the character is actually standing still and only afterwards adding actual movement of the character. This method is generally used with walk animations, regardless of the used software suite. The motion forward in Blender is controlled by a separate bone called the stride bone [18], which in Ortho's case is his root bone. Stride bone defines how much the character moves forward in one step of one foot.

Animating a walk cycle is usually done by finding four key poses for the walk, as seen in picture 13, and then adjusting the motion paths between them. This leads to animation where the character is walking in one place, like on a conveyor belt. After these poses are done the actual movement in the 3D world is added to the character. The position can be controlled by a curve that the character follows, or as was done in this case a stride bone. Blender's stride bone feature calculates the distance traveled by measuring how much a certain bone is moving. In other words, how long is one stride or step of the character. [19]



Picture 13: Four key poses of walking [19]

The four basic poses are called contact, recoil, passing, and high-point as seen in picture 13. Ortho was first positioned to the contact pose and the rotations and locations of the bones were saved. After this, the bones were positioned in the recoil pose and the rotations and locations were once again saved. This resulted in a very crude walking animation which was further refined by adding more detail to the movement between passing pose (number 3 in picture 13) and high-point pose (number 4 in picture 13). Ortho can be seen in the poses in picture 14.


Picture 14: Ortho in walking poses

At this point Ortho was walking on a conveyor belt, not moving forward at all. The poses in picture 14 differ from the poses in picture 13 because the hands do not follow the traditional walk poses because they are already positioned in the pushing movement required in the animation

To get Ortho actually moving forward in the 3D space, a stride bone was added which moves between contact and recoil pose the distance that Ortho travels. When a bone has been designed as a stride bone (or stride root) it actually moves the character at the same time. Walk cycle was then further refined by varying the speed of the stride bone, since the speed of a bipedal doesn't stay the same during one step. Motion is faster near and during the contact pose and then slower on recoil. This gives the walking animation a more lifelike feel as opposed to robotic walk in which the character moves in a constant speed and creates the impression of flowing forward.

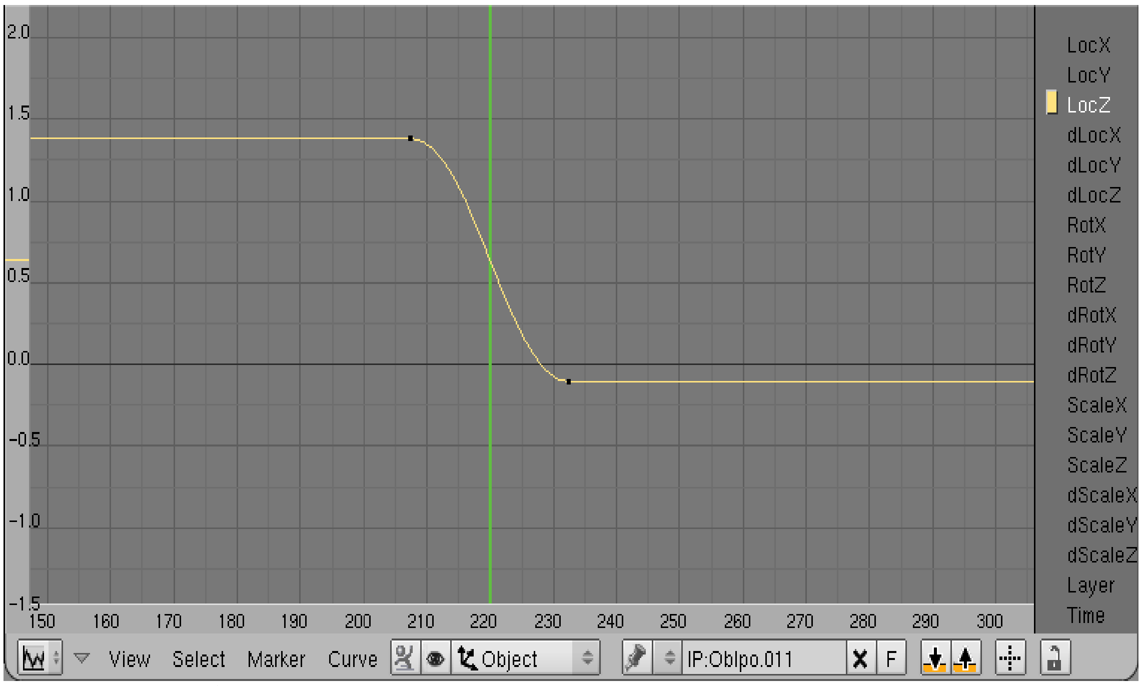
In total, five different actions were used to achieve all that was desired. Two of these were walks, the first walk is used in the beginning when Ortho pushes the O slowly forward. Latter walk is used after glance-action, this walk is modified from the original one. It was made faster by shortening its duration and adjusting the spine so that it would be leaning more forwards, creating the impression of speed and urgency.

Glance in between these two walks is simply a turn of the head to look towards the audience shown in the close up part of the animation. It does not involve forward moving, so only the neck and head bones needed animating. In addition to those, shape keys (see chapter 3.5) were used at this point to create different facial expressions.

Last action in the animation is the turning when Ortho and the O reach their destination. One more action was used during the two different walks. It was an action for pushing and in that only the hands were positioned forward and a slight looping pushing motion was added.

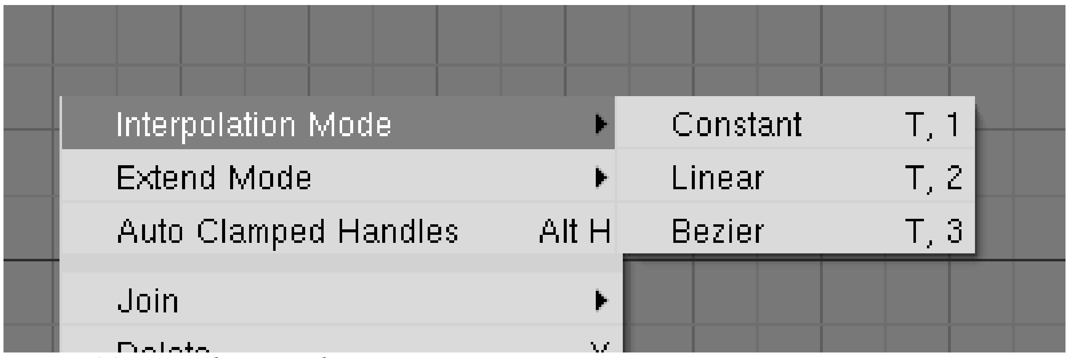
3.4 IPO-Based Animations

Word IPO comes from interpolation which means interpolating the movement in the animation between two or more key frames. Key frames are frames where the objects and parameters of everything are set by the animators. They can occur every frame or every hundred or thousand frame. The movement of the objects and the changes in parameters in other frames than key frames are interpolated from the key frames. So the animated parameters will gradually change from previous key frame to the next one to fulfill the start and end parameters. This is illustrated in picture 15 below. [3, 242-243]



Picture 15: Interpolation curve

Simple IPO-curve can be seen in picture 15 where the yellow line is the actual IPO-curve. Y-axis describes the amount of movement, which is about 1.5 blender units. X-axis describes the time of the animation in frames and the green line in the center is the play head currently in frame 220. Only the LocZ IPO was used in this, which describes the location of the object in z-dimension. The value of the location is shrinking as the time progresses forward, which means that the object is falling. The following picture (picture 16) illustrates the way the change happens in time.



Picture 16: Interpolation modes

There are three different interpolation methods for curves in Blender: constant, linear, and bezier seen in picture 16 [3, 252-253]. Bezier interpolation was used in most of the IPO-animations, which means that the objects accelerate and decelerate faster in the start and end of the sequence. This gives them smooth movement, as opposed to linear interpolation which would mean that the speed of the object remains the same during the whole duration. This technique gives an especially clunky result when the letter is at the end of the animation.

Rolling of the big O in the logo was keyed by hand after Ortho's actions were done. The earlier method of trying to copy the IPO curve of the root bone in Ortho's walk action didn not work well and the movement looked choppy and was hard to sync. Keying all the movements by hand means that there are a lot of key frames and the further adjustments in the animation were cumbersome to do. In retrospective, some sort of parenting of O to Ortho's hand bones would have been more ideal.

3.5 Shape Keys

At the final stages when an additional camera was added close to Ortho's face it created a need for facial expressions. These were carried out by using shape keys [3, 188]. Different expressions for the face were created and changes in these were then animated by keying different expressions (shapes) to certain times. Three different expressions were used, one which is the basis for the other two, in other words, the normal expression. Actual facial expressions were the surprised look when he glances at the audience and the smile Ortho gives once he gets the O in place.

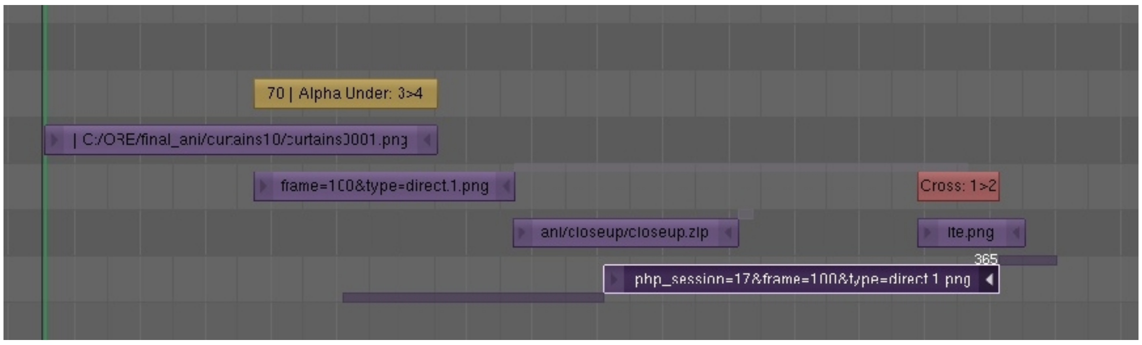
Shape keys are based on vertex transformation. In the basis key vertices are in the same position as they were modeled in. Creating a new shape key means moving the vertices into a new position and saving that as the new expression. One limitation of using vertex keys is that once they are added, you can't add or remove vertices or some unexpected results may happen. Since the need for vertex keys developed so late in the production process, the pitfall of doing them too early and having to do them again after some remodeling was avoided. [3, 188-198]

Ready expressions are animated by adding values on certain frames. This produces regular blender IPO-curves which can then be refined to suit the animation. One particular advantage of shape keys is that they can be mixed freely. So a character can have one expression, for instance, raising eyebrows with value of 0.5 which means half way between not having that expression at all and having the full expression. And at the same time there can be another expression either controlling the eyelids or some other part of the face altogether. [3, 192]

An alternative method of animating the facial features would have been to use actual bones to control the expressions directly or to have the bones control the shapes. [3, 195-197] While this would allow more detailed expressions and easier handling during lengthy animations it was not used because the longer setup time and the questionable advantages in short animations.

3.6 Compositing in Blender

Contrary to most 3D modeling softwares, Blender contains a sequence editor which has all the necessary tools for basic video editing. It also has more advanced features with which more complex effects can be achieved. Due to the sequence editors similar fast and simple workflow, the final editing of the animation also happened in Blender. Picture 17 presents the sequence view in Blender.


70 | Alpha Under: 3>4
| C:/ORE/final_ani/curtains10/curtains0001.png
frame=100&type=direct.1.png
Cross: 1>2
ani/closeup/closeup.zip
ite.png
365
php_session=17&frame=100&type=direct.1.png
Picture 17: Different animation strips

Different stacks of images which were different parts of the animation, were put together in different video channels and mixed using alpha under meta strip, which stacks images on top of each other using the alpha channel [20]. This allows combining, for instance, of the curtains and the main scene, which were rendered separately.

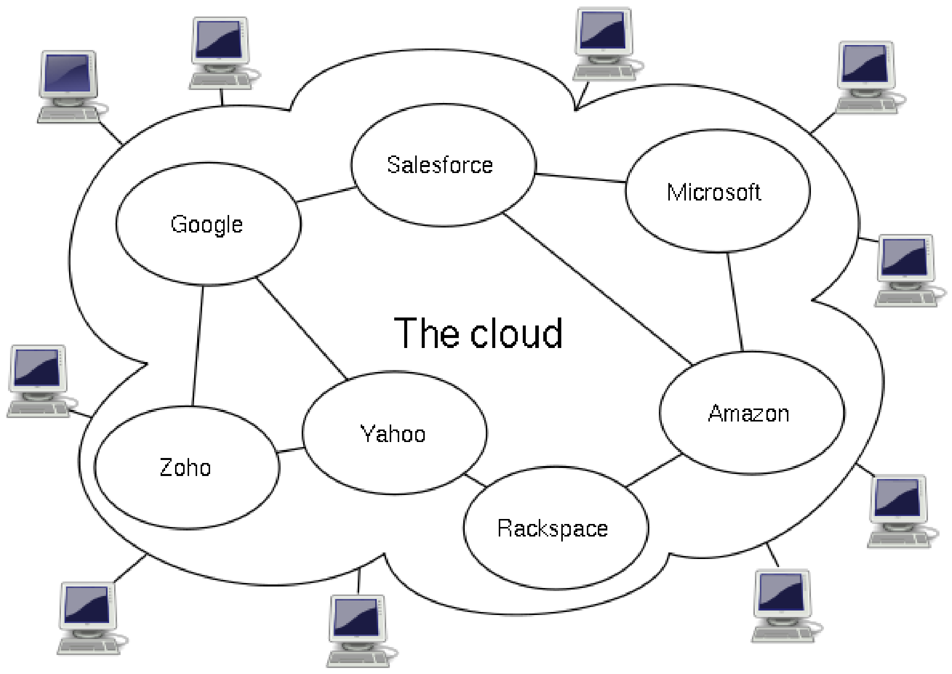
The final animation uses three stacks of images (image sequences, purple in picture 17) and 2 meta strips (red and yellow in picture 17). Another meta strip that was used was cross, which simply crosses the the ending frame of the animation to white, giving the fade to white effect.

4 ORE, BOINC, and BURP — The Technical Side

4.1 Idea of Distributed Computing

Distributed computing means splitting tasks which require a lot of time to compute on one computer (even if that computer happens to be a super computer) to smaller tasks which are then carried out by a large number of "normal" computers. This is a variation of an older computation method called parallel computing. Because Internet was readily available by the time first distributed computing project started, it was a natural way to transfer the work units. It is worth noting that distributed computing is only suitable when high latency is allowed, in other words, when a task can easily be split up in such a way that the computations are not dependent on each other. [21]

A concept related to distributed computing is cloud computing (the cloud in picture 18), but they do not mean exactly the same thing. As a term cloud computing, where the cloud refers to the general metaphor for the Internet as a cloud, still lacks a clear and concise definition, but many define it loosely as some kind of utility or service computing that happens *outside* of your firewall. In contrast with distributed computing where the computing happens in your own computer. But since there is no clear definition, sometimes distributed computing is included in the cloud computing. [22]



Picture 18: Cloud computing [23]

Distributed computing was first popularized by SETI@home project, which analyzes radio signals from space in hopes of finding extraterrestrial life or something else of interest. Their public website was launched in 1999, and they showcased that distributed computing works and caught the imagination of the general public. Sharing the computational work load in this manner opens up new possibilities for all kinds of tasks and problems which require such power to compute. [24]

Traditionally distributed computing has only been about science and ORE seeks to remedy to this. Rendering images for science and entertainment has long been bound by the computing power available and time constraints. ORE-project aims to solve this by offering the world of distributed computing to the field of culture (see chapter 5).

4.2 History of BURP and Blender

Blender is a free 3D modeling software that is licensed under GPL. It is used by artists to create 3D images and animations for all intents and purposes ranging from simple hobbyists doing projects for their own entertainment to professionals in different fields, such as movies or architecture. Blender started its life as a closed source modeling tool for a Dutch animation studio NeoGeo, in 1998 Blender's main creator Ton Roosendaal found a company called NaN to develop Blender further. However, the company failed and after it went bankrupt "Open Source Community" collectively gathered money bought the rights for the source code with 100 000 Euros and released the source under GPL [25].

BURP started when Danish Janus Kristensen got the idea of using distributed computing to calculate animations which were slow to render. It was originally suggested to him that this was impossible to do but the first adaptations to BOINC to render images were done in less than a day. From here the development really started and soon burp.boinc.dk domain was reserved for the project [26]. The development effort from burp.boinc.dk has been transferred to www.renderfarm.fi but BURP still lives a separate life in http://burp.rederfarming.net [27].

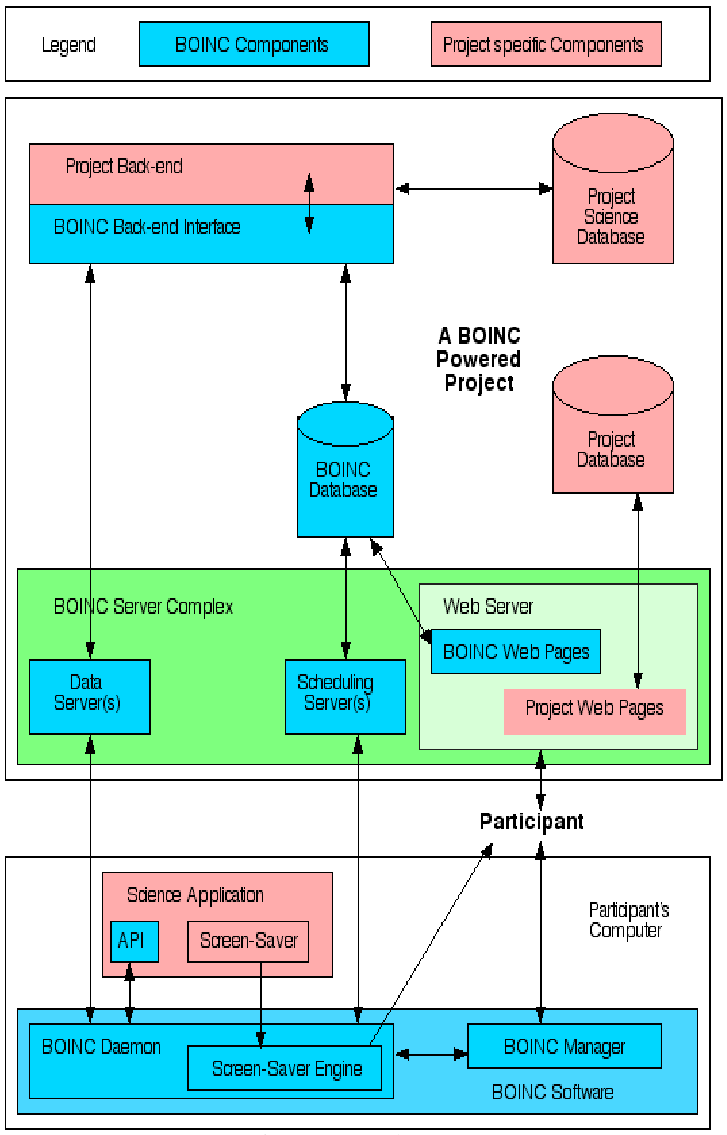
Other software suites also support some kind of shared rendering, though these are usually renderfarms that one has to create for this particular purpose, usually called network rendering. One example of network rendering would be Autodesk's Maya software which supports network rendering in three different ways, either through rendering with Maya or rendering with mental ray (a separate rendering software). [28]

4.3 Evolution to BOINC and How It Works

BOINC is divided into two separate programs – server and client. Client-side of BOINC is fairly simple. Installing constitutes from downloading a program (BOINC client), installing it, and choosing to which projects to contribute your spare CPU cycles. When the client is installed and connected to a project, it downloads work units (commonly referred to as WUs) and starts computing results from them when the computer is idle. When work units are complete, the client sends them back to the server to be verified and requests for more work. [29]

Running a BOINC project means running a BOINC server which requires a little more technical knowledge. Theoretically, any application that requires computing power can be adapted to use BOINC. If the source code of the application is available it eases the integration with BOINC, as was the case with Blender. If the source code is not accessible, there is a BOINC-wrapper which can be used, and which then handles the splitting the jobs to work units. [29]

In the case of BURP, Blender is used to render the animations on the client machines. The blender in question has been modified a little to accommodate some special needs of BURP. The modified Blender is sent automatically via the BOINC client and users do not need to concern themselves of keeping it updated. Modifications to Blender are fairly simple, most importantly there are added hooks in Blender so BURP can ask how the rendering is progressing. In addition, some of the Blender's error reporting features have been modified. In case of Blender crashing it does not show anything to user, but logs it in the BURP files and the logs are sent to the BURP server where an administrator can take a look what is happening. The process is illustrated in the following picture. [30]



Picture 19: Overview of BOINC architecture[31]

Server side of BOINC is divided into four different parts: the feeder, the scheduler, the validator, and the assimilator. These parts are inside the green box in picture 19. Feeder takes work from the database and distributes it in memory. Scheduler takes work units provided by the feeder and sends them to clients that are requesting more work. It also tries to match the work units to clients that best suit them (more powerful clients get work units that take longer to calculate). Feeder and scheduler are part of the scheduling servers on picture 19. [31]

The validator, BOINC Back-end Interface in picture 19, takes results for a single work unit which are validated. Each work unit is calculated multiple times and then they are validated against each other to see if they match. On a normal BOINC project they are easily checked for binary compatibility. On BURP, however, results of the work units can differ a bit and this is allowed. Validator checks color of each pixel in the image and small differences are allowed. These differences, in theory at least, are too small for human eye to notice, they are caused by differences in the machines and operating systems that are participating in the project.

Last part of the BOINC-chain is the assimilator which takes results that have been validated by the validator and moves them out of BOINC and into the BURP storage (Moving from BOINC Back-end Interface to Project Back-end and to Project Science Database in picture 19). After they have been moved there, additional video encoding may happen and then they are moved to their final destination, which in the case of ORE is the web server.

Downloaded work units are further divided in to parts by Blender and rendered in the minimum part size (100x100 parts), because only time Blender can update the progress is just after it finishes a part. [30]

4.4 The Problem of Parallelization

Animation itself was rendered with the ORE service. The process was the same as with any other project that would be rendered with ORE. Firstly you need to register your account at www.renderfarm.fi after that it is just a matter of submitting the file to the site. Once the site administrator has accepted your work (for instance, that it conforms to the laws and doesn't contain copyrighted material), it moves to the rendering queue. Once every single work before that has been completed, it will be rendered.

Due to restrictions of the ORE the opening scene with curtains could not be rendered using it. This is because of the cloth simulation in the curtains. ORE does not work with any of the physic simulations found in Blender. The reason for this is that Blender calculates cache files that are not contained in the .blend file in it self. If these simulations were supported, every client machine participating in the rendering would need to calculate the physic simulations again. Calculating these simulations can be time consuming and it is unnecessary to calculate them again with every client, especially since they are always calculated in a single thread so multi-core processors are of no help.

In normal animation the order in which the frames are rendered does not depend from each other. In other words, it does not matter in which order they are rendered so they can be rendered simultaneously by different machines, which is the basic idea of ORE.

However, parallelization of physics simulations is a particularly difficult problem because the result of the next frame is dependent of the current one and can not be predicted. So in order to know the shape of the mesh in cloth simulation in frame 50 you first need to know the shape of the mesh in frame 49, and before that you need to know it in frame 48 and so on.

Methods for parallel simulation of cloth have been presented, such as Bender & Bayer (2008) on their paper about Parallel Simulation of Inextensible Cloth [32]. They use a method of impulses instead of mass spring systems. In this the mesh is split into strips that behave independently and, thus, can be calculated in parallel [32]. This method is suited for realtime cloth calculation, and it remains unclear whether it can be calculated in high latency environment such as distributed computing.

If complicated and time consuming cloth simulations need to be rendered by using ORE, there are workarounds. These involve calculating the simulation on one machine (and still only on one core) and then baking the mesh. After this the animation of the cloth is stored in the mesh and is the same as normal vertex key animation and can then be rendered using distributed computing. This, however, means that the simulation aspect is lost and the animation is "locked" to the baked mesh and can not be further refined in the future, except by going back to the original simulation and repeating the process.

4.5 Rendering the Animation

The resolution of the final animation is 1920x1080 pixels. Discounting all of the test renders and experiments, the total number of frames that were rendered for the animation was 513. Out of these 513 frames 375 were used in the animation with the frame rate of 25 frames per seconds (FPS) giving 15 seconds of animation. The difference between number of frames rendered and used can be explained by two things. Firstly, more frames were rendered to give more creative freedom in the final cutting and compositing state. Secondly, some of the separate animation strips blend into each other which means that it takes double the amount of rendered frames in those parts.

Due the problem described in the previous chapter, not all of the animation was rendered using the www.renderfarm.fi-service. Since the first scene uses cloth-modifier to achieve the animation of the curtains, it had to be rendered locally on a single computer. This part of the animation contains 150 frames and it took approximately 5 minutes per frame to render. So, this sequence took a little over 12 hours to render.

Unfortunately, www.renderfarm.fi in its current state does not list how long it took to render a single animation or how long it would have taken to render on a single machine. Of course a single machine can be considered as anything, but the example times provided here were taken from test renders with a few years old desktop computer. The two parts that were rendered with this service were the logo part and the close up part. However, some idea of the timescales can be achieved by examining the close up part of the animation.

A close up contains 85 frames in the final animation and 91 frames were rendered for it. It took a little over a day to render these using the service and on a single machine it takes over two hours average to render a frame which means it would have taken over a week to render this part. The single factor that raises rendering time is the subsurface scattering used, which means that a separate pass has to be done for each frame to calculate the scattering.

5 The Social Side — www.renderfarm.fi

5.1 Earlier Applications & Cultural Significance

David Anderson [33, p. 69] aptly wrote about the social side of distributed computing: "*For scientific computing, it could contribute to a democratization of science: a research project that needs massive supercomputing will have to explain its research to the public and argue the merit of the research. This, I believe, is a worthwhile goal and will be a significant accomplishment for SETI@home even if no extraterrestrial signal is found.*"

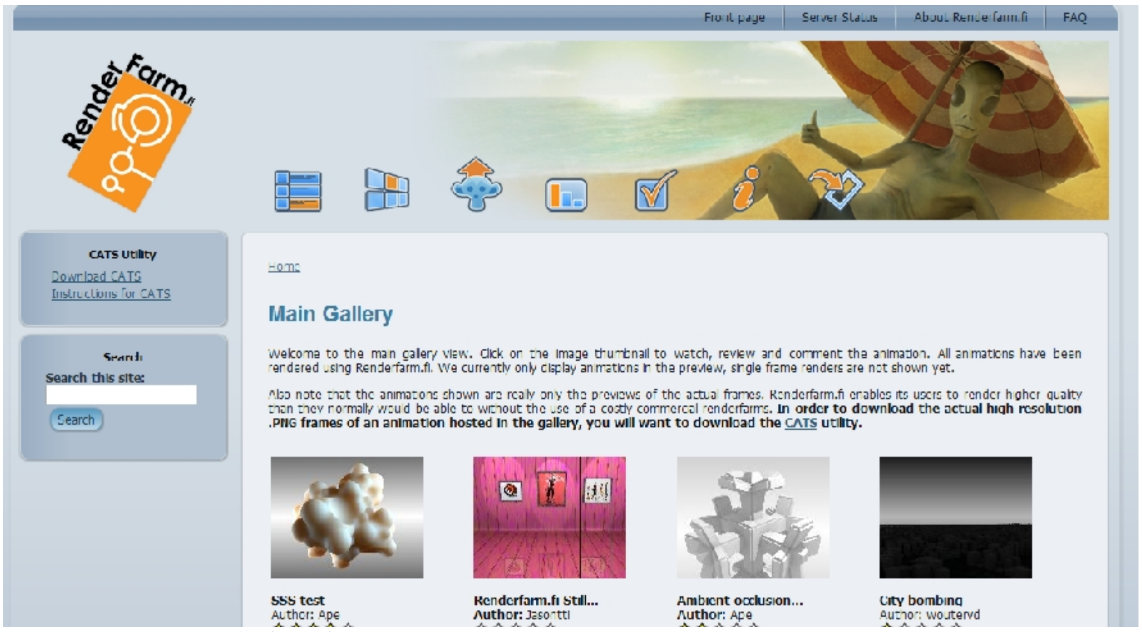
Most of the distributed computing efforts focus in the field of science and research. Ranging from researching cures for diseases, protein folding, finding prime numbers, or analyzing noise from space for signs of extraterrestrial intelligence [34]. ORE differs from all of these projects because its main goal is to produce something of cultural - not scientific - value, something which has no direct application in the realm science. Of course, scientific image rendering can also be done with ORE but it is mainly targeted for digital artists and enthusiasts to enable them to create something which would not otherwise be possible or feasible for them.

Some mentions of previous efforts to create distributed rendering farm can be found but none of them seems to be active. Distributedcomputing.info mentions two similar projects [34]. The first one is RenderFarm@Home but it seems to be completely abandoned in 2007. Any information related to project is hard to find and most likely outdated. The official website of the project just informs that "*Renderfarm@Home indefinitely suspended*". [35]. The other one is Internet Movie Project that seemed active at some point, but its home page currently only show the text: "*hello again POVers.*" [36].

More than being just a technological achievement, ORE also tries to build a community for artists, people interested in 3D animation, and distributed computing. An important part of this is the website – www.renderfarm.fi – which is the main outcome of ORE project, the website is explored more in chapter 5.2 and communal side of the project is examined in chapter 5.3 alongside with some results from a small questionnaire regarding electronic communities, distributed computing, and content creation.

5.2 Home of ORE

A very important aspect of ORE is its website www.renderfarm.fi. Of course one of the most important functions of the website is submitting work to be rendered with ORE but it also serves many other functions. And the significance of the website for submitting works will diminish, when Blender script is released and one can submit works straight from Blender [37]. Already completed and rendered animations can be seen in the gallery and it is possible to rate them and discuss the works. Users can also participate in discussion in the more general message forums. It is the general meeting ground for both Blender and BOINC enthusiasts where discussion about ORE, www.renderfarm.fi, 3D-graphics, distributed computing, culture, and life in general happen. Picture 20 shows the main gallery of the website.



Main Gallery

Welcome to the main gallery view. Click on the image thumbnail to watch, review and comment the animation. All animations have been rendered using Renderfarm.fi. We currently only display animations in the preview, single frame renders are not shown yet.

Also note that the animations shown are really only the previews of the actual frames. Renderfarm.fi enables its users to render higher quality than they normally would be able to without the use of a costly commercial renderfarms. **In order to download the actual high resolution .PNG frames of an animation hosted in the gallery, you will want to download the CATS utility.**

SSS test
Author: Ape

Renderfarm.fi Still...
Author: Jasoritti

Ambient occlusion...
Author: Ape

City bombing
Author: woutervd

Picture 20: www.renderfarm.fi [38]

The website is running with Drupal 6 content management system and customized theme seen in the picture 20. Drupal can be downloaded for free from http://www.drupal.org. BOINC itself has a simple content management system in itself, but it lacks features by modern standards. For instance, changes to user pages (or new users etc.) do not happen instantaneously. Instead they are updated when a specific task is run by the scheduler. As part of the ORE team, the author's task was to check and accept animations to be rendered that complied with the rules of the service.

Other important aspect was also being active in the forums and in the community in general with writing blog posts and helping users install ORE-clients via Internet Relay Chat (IRC). Two guides about how to get ORE-client running on one's computer were written: one for individuals and the other one for instance for schools with the instructions to mass installations.

Integrating BOINC and Drupal has meant easier development of the social side of the site which, one could argue, is as important as the technical side. The social side of the website and integration with Drupal is examined in more detail in Lauri Viitala's (2009) thesis [39].

In what appears to be a pure coincidence, the main BOINC project also seeks to integrate with Drupal and leave the old content management system behind. As of September 2009, there is a conversion project going on in BOINC to integrate Drupal with BOINC. This project is till on planning stage and no clear schedule is given. [40]

5.3 Interest in Distributed Computing

A small questionnaire was given out at the end of one week Blender course held at the Metropolia University of Applied Science. Among the regular feedback and teacher evaluation, there were a couple of questions about electronic communities, Creative commons and distributed rendering. More specifically students were asked:

Each question had three answer options: Yes, No, or Maybe. In addition to those questions, interested parties were asked to give their email address to get an invite to www.renderfarm.fi. Relevant questions and results are summarized in table 1.

	Yes	Maybe	No
Do you plan to join an electronic Blender community such as Blender.org, Blenderartists.org or Renderfarm.fi?	5	10	3
Would you be willing to license your creative works under a Creative Commons license?	8	10	0
Would you be willing to donate you computer's processing power to do publicly distributed rendering?	4	13	1

Table 1. Questionnaire results

18 students filled the questionnaire and the results were quite interesting. Five answered yes to the first question about joining electronic community and ten were maybe interested. However, ten people left their email address to get their invite to www.renderfarm.fi. Only three persons were not interested in electronic communities at all. Similarly eight students were willing to license their works under Creative Commons license and the rest were maybe willing to do so.

On the subject of donating one's own computing power to distributed rendering, only four answered yes and the majority of thirteen answered maybe, leaving only one person not interested in sharing his or her computing power. There seemed to be uncertainty o issues such as how easy this sharing would be to the participant, which could be a major factor in willingness to share resources and help in distributed rendering. Even the simplest of setup or program installation can be too much effort, if one is not particularly interested in the topic.

Even though the sample was quite small and participants were interested in 3D-modeling, it still shows promising amount of interest in services like www.renderfarm.fi. Whether this will actually increase the amount of amateur and small production 3D-animations in both quality and quantity will remain to be seen.

5.4 Other Uses for Ortho

Since the model of the character Ortho was to be released under Creative Commons license (even though it wasn't yet publicly available), it was used for couple of other projects which were related to ORE. The fist one of these happened before even the rig of the character was ready and the pose of the character was quickly made by just moving vertices to suitable places by hand.

Firstly the character was featured in the promotional material for the ”Digital / communal creativity” -seminar held in Laurea Leppävaara in the Autumn of 2008. It was featured with the original ORE-logo in posters and a bigger print of the face was also handed out to seminar guests as well. Picture 21 shows another poster featuring Ortho.

THE
ORTHO
WARS
RENDERFARM.FI presents
in association with LAUREA-AMMATTTIKORKEAKOULU a SKYLAMB production
HEIKKI SEPPÄLÄ NIKO SUOMALAINEN JUKKA SÖDERSTRÖM JUHA VÄÄRÄNEN "THE ORTHO WARS" with LASSI HAARANEN and JULIUS TUOMISTO
DECEMBER 5
www.renderfarm.fi
Picture 21: Poster for Ortho wars

Later on it was featured in a mini game Ortho Wars poster in picture 21. It was also created to promote ORE-service and www.renderfarm.fi. The poster also featured spaceships and moonlike landscape which were taken from writer's previous projects and are also licensed under Creative Commons.

6 Conclusions

The animation is the actual outcome of this thesis and a part of writer's contribution to the ORE project. However, it may be that the actual process of creating the animation and seeing and using ORE for some real project might be more valuable than the actual end result in itself. An important part of this project was this written documentation, which gives detailed account of the process of creating animation, and more specifically, an animation that contains humanoids. In addition how to use ORE to enable higher resolution rendering with higher amount of detail has been reported in detail in this study.

The content of the animation was not completely set in stone from the beginning, so naturally it evolved during the process. Even at the very late stages, one camera was added and the actions refined. Also the audience changed from drawn 2D idea to 3D animation. However, the end result is close to what was originally planned.

As a field, distributed computing is not a new idea but its application to the realm of art instead of science is fairly new. Even though people are not that familiar with the concept, there seems to be interest in it. This can be seen in the results of the questionnaire presented in chapter 5.

Final animation contains 375 frames which give 15 seconds of animation with 25 frames per second. The total number of rendered frames (excluding all of the test renders), was 513. Rendering more frames than necessary allowed more fluidity in cutting and compositing phase and was necessary due to some parts fading into each other.

With the spirit of the free and open tools and services used to create this animation, the production files and the final animation is released to be distributed freely according to Creative Commons License. It can be used to create more promotional material for www.renderfarm.fi or, alternatively in some completely unrelated project. The animation and the related production files can be found at: http://users.metropolia.fi/~lassiha/ore/

References

1 Varila A. Sketch for Ortho [digital drawing]. Leppävaara: Laurea University of Applied Sciences; August 2008.

2 Varila A. Second sketch for Ortho [digital drawing]. Leppävaara: Laurea University of Applied Sciences; August 2008.

3 Mullen T. Introducing character animation with Blender. Indianapolis, In: Wiley Publishing, inc; 2007.

4 Blender Foundation. Modifier System. Release Logs / Blender 2.40 [online]. Amsterdam, The Netherlands: Blender Foundation
URL: http://www.blender.org/development/release-logs/blender-240/modifier-system/. Accessed 10 November 2008.

5 Mckay J. Building a Better Eyeball [online].
URL:http://members.optusnet.com.au/%7Ejmckay001/blender/Eyeball_Tutorial.pdf. Accessed 8 November 2008.

6 Wikipedia Foundation. Catmull-Clark subdivision surface [digital image]. San Fransisco, CA: Wikipedia Foundation; December 2006.
URL: http://en.wikipedia.org/wiki/File:Catmull-Clark_subdivision_of_a_cube.svg. Accessed 20 January 2009.

7 Catmull E & Clark J. Recursively generated B-spline surfaces on arbitrary topological meshes. Computer-Aided Design 1978;10(6):350-355.

8 Hess R, editor. The Essential Blender- Guide to 3D Creation with the Open Source Suite Blender. Amsterdam, The Netherlands: Ton Roosendaal / Blender Foundation Netherlands; 2007.

9 Blender Foundation. Subsurface Scattering. Release Logs / Blender 244 [online]. Amsterdam, The Netherlands: Blender Foundation
URL:http://www.blender.org/development/release-logs/blender-244/subsurface-scattering/. Accessed 26 February 2009.

10 Jensen H W & Buhler J. A Rapid Hierarchical Rendering Technique for Translucent Materials [online]. Proceedings of SIGGRAPH 2002
URL: http://graphics.ucsd.edu/~henrik/papers/fast_bssrdf/fast_bssrdf.pdf. Accessed 26 February 2009.

11 Gold material [Blender material file].
URL: http://www.blender-materials.org/index.php?action=view&material=44-gold. Accessed 20 November 2008.

12 Tutorial: Foot Rig (without action constraint) [online].
URL: http://calvin.sdlfk.org/calvin/FootRig2/index.html. Accessed 7 October 2008.

13 Blender Foundation. Weight Paint (Blender Manual). Amsterdam, The Netherlands: Blender Foundation
URL: http://wiki.blender.org/index.php/Doc:Manual/Modelling/Meshes/Weight_Paint. Accessed 13 October 2009.

14 Blender Foundation. Cloth Simulation (Blender Manual). [online]. Amsterdam, The Netherlands: Blender Foundation
URL: http://wiki.blender.org/index.php/Doc:Manual/Physics/Clothes. Accessed 13 October 2009.

15 Provot X. Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior [online]. Institut National de Recherche en Informatique et Automatique (INRIA)
URL: http://www.-rocq.inria.fr/syntim/research/provot. Accessed 20 January 2009.

16 Blender Foundation. Physics Caching and Baking. Release Logs / Blender 2.46 [online]. Amsterdam, The Netherlands: Blender Foundation.
URL: http://www.blender.rog/development/release-logs/blender-246/physics-caching-and-baking/. Accessed 23 October 2009.

17 Blender Foundation. Action and NLA editor. Release Logs / Blender 2.40 [online]. Amsterdam, The Netherlands: Blender Foundation.
URL:<http://www.blender.org/development/release-logs/blender-240/action-and-nla-editor/. Accessed 13 October 2009.

18 Blender Foundation. Advanced Stride support. Release Logs / Blender 2.40 [online]. Amsterdam, The Netherlands: Blender Foundation.
URL: http://www.blender.org/development/release-logs/blender-240/advanced-stride-support/. Accessed 14 October 2009.

19 Blender Foundation. How to setup a walkcycle using NLA. Blender Documentation Volume I – User Guide [online]. Amsterdam, The Netherlands: Blender Foundation.
URL: http://www.blender.org/documentation/htmlI/x8053.html. Accessed 13 October 2008.

20 Crewind Technologies. How Does Alpha Channel Work? [online]. Chennai, India: Crewind Technologies; 2009.
URL: http://www.icongalore.com/xp-icon-articles/alpha-channel-explained.htm. Accessed 23 October 2009.

21 Godfrey B. A primer on distributed computing [online].
URL: http://www.bacchae.co.uk/docs/dist.html. Accessed 14 October 2009.

22 Knor E & Gruman G. What Cloud Computing Really Means [online]. San Fransisco, CA: Infoworld; 7 April 2008.
URL: http://www.infoworld.com/d/cloud-computing/what-cloud-computing-really-means-031. Accessed 12 October 2009.

23 Johnston S. Cloud Computing [digital image]. San Fransisco, CA: Wikipedia Foundation; October 2009.
URL: http://en.wikipedia.org/wiki/File:Cloud_computing.svg. Accessed 13 October 2009.

24 SETI@home Classic: In Memoriam [online]. California, CA: University of California.
URL: http://setiathome.berkeley.edu/classic.php. Accessed 13 October 2009.

25 Blender Foundation. History [online]. Amsterdam, The Netherlands: Blender Foundation.
URL: http://www.blender.org/blenderorg/blender-foundation/history/. Accessed 14 December 2008.

26 Kristensen J. Internet distributed 3D rendering with Blender [online]. 17 June 2004.
URL: http://blenderartists.org/forum/showthread.php?t=25702. Accessed 10 January 2009.

27 BURP | RenderFarming.net. Homepage [online].
URL: http://burp.renderfarming.net. Accessed 14 October 2009.

28 Autodesk. Overview of retwork rendering [online]. Document). San Rafael, CA: Autodesk, Inc.
URL: http://download.autodesk.com/us/maya/2009help/index.html?url=Network_rendering_Overview_of_network_rendering_.htm,topicNumber=d0e583742. Acessed 18 October 2009.

29 BOINC. How BOINC Works [online]. University of California; October 2009.
URL: http://boinc.berkeley.edu/wiki/How_BOINC_works. Accessed 15 Octobe 2009.

30 Kirstensen J [interview]. 19 February 2009.

31 Unofficial BOINC wiki. System Architecture [online]. October 2009.
URL: http://www.boinc-wiki.info/BOINC_System_Architecture. Accessed 15 Octobe 2009.

32 Bender J & Bayer D. Parallel simulation of inextensible cloth [online]. Karlsruhe, Germany: Institut für Betriebs- und Dialogsysteme, Universität Karlsruhe; 2008. URL: http://i31www.ira.uka.de/docs/VRIPhys08.pdf. Accessed 27 January 2008.

33 Anderson D. SETI@home. In: Oram A, editor. Peer-to-Peer: Harnessing the Power of Disruptive Technologies. Sebastopol, CA: O'Reilly; 2001.

34 Pearson Kirk. Active Projects [online]. Distributed Computing.
URL: http://distributedcomputing.info/projects.html. Accessed 14 October 2009.

35 RenderFarm@Home. Homepage [online].
URL: http://www.renderfarmathome.com.au. Accessed 14 October 2009.

36 Internet Movie Project. Home pag [online].
URL: http://www.imp.org. Accessed 14 October 2009.

37 Letwory N. ORE Uploader – r 67 [program]. Leppävaara: Lauea University of Applied Science; 2009
URL: http://www.renderfarm.fi/page/uploader-beta. Accessed 14 October 2009.

38 www.renderfarm.fi. Interface [digital image]. Leppävaara: Laurea University of Applied Sciences; 2009. Captured on 11 October 2009.

39 Viitala, L. A more intriguing volunteer computing experience through Drupal and social technology. Leppävaara: Laurea Univesity of Applied Sciences; 2009.

40 BOINC. DrupalConversion [online]. University of California; October 2009.
URL: http://boinc.berkeley.edu/trac/wiki/DrupalConversion. Accessed 15 October 2009.