Joni-Petteri Kivistö

REAL-TIME MUSCLE ANIMATION

Opinnäytetyö
Kajaanin ammattikorkeakoulu
Luonnontieteiden ala
Tietojenkäsittelyn koulutusohjelma
Kevät 2012

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ
TIIVISTELMÄ

Koulutusala Luonnontieteiden ala	Koulutusohjelma Tietojenkäsittelyn koulutusohjelma
Tekijä(t) Joni-Petteri Kivistö	
Työn nimi Real-Time Muscle Animation	
Vaihtoehtoiset ammattiopinnot	Ohjaaja(t) Nick Sweetman
	Toimeksiantaja
Aika Kevät 2012	Sivumäärä ja liitteet 39

Lihasten liikkeiden animointi on hiljalleen yleistymässä peleissä. Työvaiheet lihasanimaation toteuttamiseksi eivät ole vielä vakiintuneet ja erilaisia toteutustapoja hyödynnetään nykypäivän peleissä. Elokuvastudiot ovat hyödyntäneet kehittyneitä animaatiotyökaluja kohta yli vuosikymmenen ajan luodakseen lihasanimaatioita. Peleissä ei voi suoraan hyödyntää elokuviin tarkoitettuja työkaluja, mutta niiden tuottamia tuloksia pystytään jäljittelemään. Ilman lihasten animointia pelihahmot voivat vaikuttaa nukkemaisilta ja epäuskottavilta. Kun pelihahmon lihasten liikkeet animoidaan, saadaan hahmoista realistisempia ja todentuntuisempia.

Opinnäytetyön tavoitteena oli pyrkiä selvittämään millaisia teknologioita pelistudiot käyttävät tuottaakseen lihasanimaatiota. Pelistudiot ovat vaitonaisia käyttämistään tekniikoista ja metodeista joilla luodaan lihasanimaatioita. Haastatteluista, blogeista ja tutkimusraporteista voi kuitenkin saada vinkkejä siitä, millaisia teknologisia ratkaisuja pelistudiot mahdollisesti käyttävät. Toisena tavoitteena oli rakentaa pelihahmoille luurakenne joka mahdollistaa lihasten animoinnin mahdollisimman tehokkaasti ja uskottavasti. Luurakenteen suunnittelu toteutettiin Blender 3D -mallinnus- ja animaatio-ohjelmistoa käyttäen.

On vaikea sanoa varmuudella mitä tekniikoita pelistudiot hyödyntävät tuottaakseen lihasanimaatiota eri peleihin. Vaikuttaa kuitenkin siltä että lähestymistavat ovat hyvin erilaisia. Pelistudiot käyttävät todennäköisesti tarkasti tarkoituksiinsa suunniteltuja ja itse tuotettuja työkaluja.

Luurakenteen toteutuksessa hyödynnettiin ainoastaan tavallista, yleisesti käytössä olevaa teknologiaa. Toteutettu luurakenne tukee dynaamista pintakuvioiden sekoitusta perustuen monikulmioiden pinta-alamuutoksiin. Luurakenne mahdollistaa lihasten animoinnin monikulmio- ja pintakuviotasolla. Se on helppo lisätä mukaan pelin kehitysprosessiin, riippumatta pelimoottorista tai käytetyistä ohjelmistoista.

Toteutettu luurakenne on prototyyppi ja lisää tutkimusta tarvitaan siitä, sopisiko se pelikehitykseen.

Kieli	Englanti
Asiasanat	3D, lihasanimaatio, hahmoanimaatio, reaaliaikainen
Säilytyspaikka	☒ Verkkokirjasto Theseus ☒ Kajaanin ammattikorkeakoulun kirjasto

KAJAANIN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

THESIS
ABSTRACT

School Business	Degree Programme Business Information Technology
Author(s) Joni-Petteri Kivistö	
Title Real-Time Muscle Animation	
Optional Professional Studies	Instructor(s) Nick Sweetman
	Commissioned by
Date Spring 2012	Total Number of Pages and Appendices 39

Animating muscle movement in games is becoming more common. The work stages to produce muscle animation are not yet standardized and different methods are applied in games nowadays. Movie industry has used sophisticated tools to create muscle animation soon over a decade. These tools cannot directly be used for game production but their results can be mimicked. Without muscle animation game characters may seem unconvincing and doll like. When the muscles of the game character are animated, a more realistic and convincing feeling is achieved.

The goal of this thesis was to conduct a research on what technologies are used to by game studios to produce muscle animation. Game studios are secretive about the exact technologies and methods they use to create muscle animation. From interviews, blogs and research papers leads can be found on how game studios produce these visual effects. The second goal was to develop a rig structure for game characters that would enable efficient and believable muscle animation. The rig was designed and produced using Blender 3D modeling and animation software.

It is difficult to say for sure what technologies are used by game studios to produce muscle animation. It would seem that there are a few different approaches in use. Most likely game studios use tools that are precisely designed for their use and developed in house.

The rig was built using only standard technology. The finished rig supports dynamic texture blending based on the area variations of the polygons. The rig enables muscle animation on polygon and texture levels. It is easy to add in a game development workflow, despite of game engine or software packages used.

The rig is a proof of concept or a prototype and more research is needed to find out if it is suitable for game development.

Language of Thesis	English
Keywords	3D, muscle animation, character animation, real-time
Deposited at	☒ Electronic library Theseus ☒ Library of Kajaani University of Applied Sciences

PROLOGUE

The idea for the thesis came to my mind when I noticed that some games had started using advanced character rigs that supported muscle animation and wrinkle maps. I got interested and researched the methods used to create such effects. I found out that there is no standard way of doing these effects and that each game handles them differently. I decided to research and develop a method that would in my opinion be the best approach to create muscle animation.

Motion capture is an effective and heavily standardized method for creating animations for games. Because it is so widely used in games the method for muscle animation should be created to be compatible with it. The motion capture equipment used today are unable to record muscle movement. However the muscle movement is closely related to the rotation of the limbs, which are recorded. Since muscles enable the movement of the limbs, muscle movement can be recreated for the character in 3D software from the motion capture data.

For the purposes of this thesis I decided to build a character rig that includes the most basic muscle animation. The character rig is compatible with motion capture data and the rig is easy to apply to other character models. Similar character rigs can be built in most 3D software packages and can be used in all game engines supporting bone animation. Implementing muscle animation and wrinkle maps requires minimal additional coding in the engine. Only one special shader has to be written for the texture animation to work properly.

I limited this thesis to only include muscle animation but at the same time designed it to enable wrinkle maps for clothing using same technique.

TABLE OF CONTENTS

LIST OF SYMBOLS

Bone constraints	A data block in 3D software that can be applied to bones. Bone constraints define the bones behavior. For example bone constraints can be used to limit rotation, scaling or location of bones
Bone	A hierarchical 3D sub object. Bones are used to build rigs
Cavity map	A texture map that emulates the way light radiates across the surface
Diffuse map	A texture map that defines the colors of the surface
Digital sculpting	A 3D modeling method that emulates the sculpting in real life
Dummy object	Also called empty or null objects. In 3D software it is an object used to mark a coordinate in 3D space. Dummy objects can be used for different purposes in animation, modelling or texturing
IK/FK	Short for Inverse and forward Kinematics are two different animation methods for bone structures. In inverse kinematics bones lower in the hierarchy are used to move bones that are higher in the hierarchy. In forward kinematics bones higher in the hierarchy are used to move bones that are lower in the hierarchy
Mesh	A collection of vertices in 3D space that are connected to each other in specific way. Mesh forms the basic form of the character

Morph target animation	3D animation method. In morph target animation each key frame created by creating different shapes using the same topology. During the animation vertex positions are interpolated between these shapes
Normal map	Texture map where the surface normal are stored in bitmap
3D Object	Data structure in 3D software that can include different types of data. 3D objects can contain sub objects.
PSD	Pose space deformation is a computer animation technique where skeletal and mesh animation(very similar to morph target animation) are used together to create complex deformation of 3d mesh.
Rig	A 3D object that contains a collection of bones used to animate a character. Rig can also include custom animation controls and scripts
SIGGRAPH	Short for Special Interest Group on GRAPHics and Interactive Techniques. Highly respected gathering of computer professionals.
Specular map	Texture map that defines the amount of specular reflection of the 3D surface
Topology	The organization and flow of the edges of the mesh
Vertex	A data structure in 3D graphics that stores a point in 3D space.
Wrinkle map	Texture map that represent wrinkles. The wrinkle maps are blended in and out the on top of other textures depending of the pose or expression of the character. Wrinkle maps are usually normal, cavity or diffuse maps

1 INTRODUCTION

Today games feature many graphical techniques and tricks to spice up the game visuals. Recent new features include facial motion capture, destructible environment and animation generation in real-time.

One often overlooked feature in games is muscle simulation or animating muscle movement. The reason for this is that this feature is thought to be barely visible or that it is difficult or expensive to implement. In some cases lack of muscle movement can be really visible, for example the lack of bicep movement in muscular characters or static muscles of a huge muscular creature. Things like these can break the immersion in otherwise flawless game graphics and send beautifully crafted characters straight to the uncanny valley. The muscle animation can give more credibility to the characters, by making the characters more organic and less robotic.

The fighting games can benefit greatly from muscle animation. In the fighting games portraying the strength of the fighters can be done by adding muscle animation. In sports games player spends lots of time watching athletes. Clothing used in sports usually does not cover the muscle structures and therefore lack of muscle animation can be distracting to the player. While using muscle animation during the actual gameplay does not necessarily make much sense, during slow motions they are very visible and can add a lot to the visuals experience. Games that do not aim for realistic look can also benefit from muscle animation. Stylized game graphics can exaggerate muscle size greatly. In stylized game graphics muscle animation can be more visible than in photorealistic game.

Applying a basic muscle animation to a character is not difficult and the technology has been around for a long time. Muscle animation can be added quite easily to games. Using already existing technology in 3D software, bones, bone constraints and careful rigging can lead to believable muscle movement. This kind of muscle animation can be automated to considerably reduce the animator's workload. In this thesis I'm going to research common practices to achieve more believable character by including muscle animation for games. I'm also going to develop my own way of optimized workflow for creating muscle animation. The basic principle being that as much as possible is done with already existing tools while keeping as many game engines and modeling software as compatible as possible.

2 THESIS BACKGROUND

Game character creation for next generation games is a fairly standardized process and pipeline is mainly similar. It starts with character concepts, which are tweaked until the team is satisfied with the look and feel of the characters.

A modeler gets to work and creates a high polygon model in digital sculpting software. The most popular digital sculpting software packages nowadays are Zbrush and Mudbox. Digital sculpting is closely related to traditional sculpting.

When the high polygon model is done, a low polygon version is created based on the high polygon model. When creating the low polygon model the artist must make sure that the topology of the character is suitable for animation purposes. The low poly model is then uv-unwrapped and normal data from high polygon model is baked to a normal map based on low poly models texture coordinates. This texture map is saved and used to create illusion that the surface has more detail than there actually is. This technique is called normal mapping.

A texture artist starts to work on texture maps. The normal map is a good starting point of creating other texture maps; optionally the 3D artist can bake ambient occlusion from the high polygon model for the texture artist to use as a starting point. This way 2d artist can better comprehend how the 2d texture will map on the 3D models surface. The most commonly used texture maps, in addition to the normal maps, are diffuse and specular maps. The diffuse map defines the overall colors of the characters surfaces. The specular map defines the amount of specular reflections of the surface. The specular map is usually greyscale. In addition to these the engine might have other special maps that, for example, define specular reflections color or define what effect is played when a bullet hits the surface.

When the texture artist is happy with the texture the rigger builds a rig for the character. The rig is used to animate the 3D mesh of the character. It is connected to the 3D mesh by process called rigging, weighting or weight painting. During this process each vertex of the characters 3D mesh is attached to at least one bone of the rig. Most real-time applications or game engines have a limit that a vertex can be attached to maximum of four bones.

The game character is then animated by an animator. The character is then exported to the game engine using a file format that is compatible with the game engine. The animator then tests the animation in the game engine. Attention must be paid to that transition between different animations happen seamlessly in the game engine. Some tweaking in different areas may take place after this.

In this pipeline there are no standardized methods to produce muscle animation or wrinkle animation for clothing. To create these effects there are a few techniques to choose from. In this thesis these techniques were studied to find the best method in terms of simplicity and reusability. Simplicity was chosen as standards because it allows people with less experience to work on the game. Reusability was chosen as a standard because it cuts down development time and, therefore, costs.

3 HUMAN ANATOMY

Studying anatomy in general is important to a character artist, rigger and animator. Knowing anatomy helps to understand how creatures are built and how they move. This is essential when creating characters or animations that need to look lifelike and believable.

3.1 Skeleton structure

An adult human body has approximately 200 bones. A typical game character rig is similar to a simplified version of a human skeleton. Understanding the human skeleton structure is important if the artist desires to create realistic looking human characters that move realistically. When building a rig that supports muscle animations, it helps to have a basic knowledge on how the simulated muscles are connected to the skeleton. Because the skeleton structure dictates the limbs range of motion, knowing how the bones limit the limb movement helps an artist to avoid creating unnatural poses.

The human's vertebral column or the spine is a special case, because of the 28 moving bones. Using so many bones in the game engine is not reasonable since each bone rotates only a little. The artist has to decide how many bones to use to create the spine.

In 3D software packages bones are represented as straight lines. In reality bones are anything but straight, especially when there is a ball and socket joint bone is L-shaped at the end (Innerbody n/d). When taking the real shapes of the bones into account the rigging process can become much easier when using automatic skinning tools.

3.2 Joints and their range of motion

A joint is the point where two or more bones make contact. Joints can allow movement or provide mechanical support. When the artist knows the locations of the joints, he can place the bones of the rig in the right places. This is important since if the bones are in the wrong places the limbs may not twist and rotate naturally. Especially in the shoulder area the artist needs to be extra careful since even small errors in the joint location are visible to the player.

The shoulder can be difficult to understand at first and studying a 3D model of a skeleton can be an invaluable help. The even better way is to place a model of a skeleton inside the 3D character; this will clearly show where the joints should be located. (Harkins 2003.) Joints do not always need to be at anatomically correct places, better results can be achieved with joints at anatomically incorrect places depending on which joint the artist is placing and to what character mesh. This way the artist can fake deformations that occur in real life trough complex interactions muscle, fat and bones by using fever bones in 3D software.

The range of motion of the characters limbs can be limited automatically by using bone constraints called rotation constraints. Setting up rotation constraints can be tedious and take time. They can be generated from motion capture data to save time. Rotation constraints can be cumbersome depending from software used. They can also limit artistic freedom. Although they can be useful for some purposes, they are not necessary since a capable artist can notice and easily avoid the unnatural rotations.

3.3 Main muscle groups

When creating a rig that supports muscle animation for a game character it is not reasonable to try to animate each and every muscle in the human body. Instead the artist should decide which muscles are the most visible to the player and therefore the most essential. The visibility is influenced, for example, by object size relative to screen space, camera perspective, game resolution, size of display and speed of the presentation. Using muscle animation is not reasonable in games that are played in below high definition resolution or using small screens like mobile devices.

The camera system that the game uses defines what the player will see. Most attention in the game characters and game graphics in general should be paid to areas which player spends the most time looking at. In 3D games the first person, orthographic, over the shoulder and side view camera angles are the most common. In addition to these common camera types used during the actual gameplay games often include close ups of the characters during the cut scenes.

In first person perspective player is looking at characters face and upper body, mostly from the front. In this perspective the player character is not visible but player can look at the other characters very closely but this applies only to friendly characters or enemy characters

during melee attacks. During firefights and other long to medium range engagements muscle animation is not needed as it is not noticeable. When a game is played from over the shoulder -perspective the back muscles of the main character are most visible to the player. In over the shoulder -perspective the other characters cannot be viewed closely during the game play. In side view camera angles, common to the fighting games, characters are viewed from the either side. When a game uses the side view perspective the silhouette of legs and arms is more pronounced. In this perspective animating biceps and triceps can be a good idea. Games that have the perspective far away from the player character, such as orthographic perspective, do not benefit from muscle animation.

Most of the time game characters are not running around naked and muscles are covered by clothing. If the clothing the character uses is skin tight, the muscle movement should still be very visible despite of the clothing.

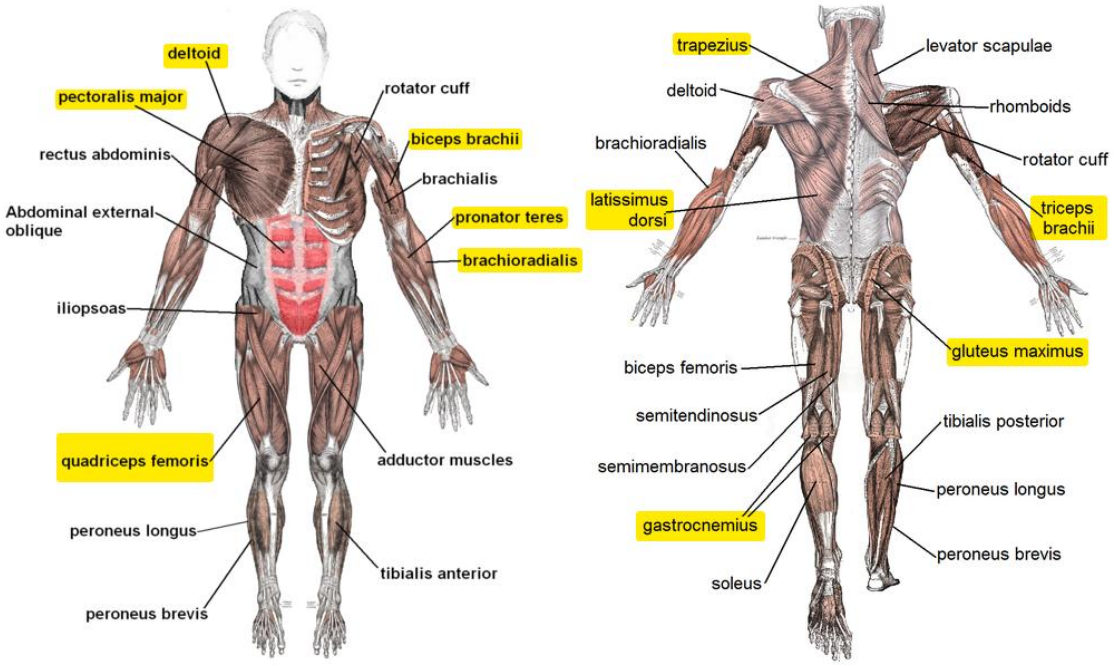

Figure 1. Table of muscles. (Physical & Sports Therapy n/d).

The most visible muscles are naturally the largest ones and only they should be simulated. The muscles I decided to animate are: pectoralis major, deltoideus, biceps, triceps, pronator teres, brachioradialis, trapezius, latissimus dorcii, gluteus maximus, gastrocnemius and quadriceps femoris which are highlighted in Figure 1. (Physical & Sports Therapy n/d).

4 RIGGING

Rigging is the process of building a rig and then attaching it to a 3D mesh in 3D software. A rig consists of bones and dummy objects and can include custom tools and code written specially to add certain properties or to tackle some specific problems. (Nieminen 2009.)

4.1 Bones

Bone objects are similar in all 3D software packages; they are hierarchal structures that can be customized to behave as the artist wishes. 3D software packages feature extensive tools to create and edit the bone structures. (3dkingdoms 2006.)

In Blender 3D modeling software a rig consists of two types of bones: deformer bones and control bones. These two are the same building blocks but what differentiates them from each other is how they are used. In other 3D software the rig building process may be little different for example dummy objects can be used instead of control bones. Regardless of the differences how objects are used or represented the process is very similar in all 3D software.

The control bones are used to move the deformer bones. They are located or represented outside the 3D mesh so the animator can see and select them easily. Character animation is done mainly by animating the control bones. (Williamson 2011 a.)

Deformer bones are used to deform the 3D mesh of a character and are located inside the 3D mesh. Deformer bones are usually hidden during the animation process to keep the viewport clear of clutter. (Williamson 2011 b.) After the animation process some of the control bones serve no purpose in the game engine, so they are left behind when the character is moved to the game engine.

In many cases a bone can be a hybrid of the two types, serving both functions. There are no strict rules on how to do the rigging. However, the animator's preferences define how the rig should be built since the rig's main function is to enable the animator to work efficiently.

4.2 Dummy objects

Dummy objects, also called null objects or empties, are basically just points in space. They are used for variety of purposes in 3d graphics. In rigging they are used as targets for the bone constraints or same way as the control bones. Using the dummy objects instead of control bones might be a good idea since they look different from bones and because of that may help keeping the interface clearer. Depending on software they might not be a part of the bone structure, therefore, animating dummy objects is done outside the bone animation interface and this may complicate the animators work.

4.3 Bone constraints

The bone constraints are tools that can be linked to bones. Then values can be inserted to the constraint to for example limit the bone rotation and location or to tell the bone to point the certain direction. By using bone constraints artist can automate repetitive tasks or functions. When the rig is built properly using bone constraints, such as IK constraints, the animation process becomes more efficient.

4.4 Custom animation controls

When the rig is so complicated that animating it by traditional means becomes difficult, the rigger should consider building custom animation controls. Building custom animation controls is common when doing facial animations or rigging complex mechanical structures.

The custom animation controls can be thought as complicated control bones. They consist of custom shapes, either polygonal or curve based, and bone constraints. With these tools the rigger builds a series of sliders and buttons that can be easily selected and animated. These sliders and buttons move the corresponding deformer bones. They can exist in 3D space floating near the character, be situated in the custom control panel somewhere in the interface or as a separate window.

4.5 Tools for muscle animation

Tools and plugins for creating muscle animation are available for most 3D modeling software. These are often created to use in pre-rendered movies and cannot be used in real-time applications such as games. The reason for this is that these tools are complex and require lot of computational power. These tools do more than just animate muscle tissue. They also take into account the underlying bone structure, skin and fat tissue. Although animation on high level like this is very demanding from hardware, it can create fantastic results. (Autodesk d/n.) When computational power increases tools like these might become available for a real-time application instead of just pre-rendered movies.

4.6 Common rigging practices for games

When the character and animations are moved to the game engine, all the animations created in 3D software have to be stored in format compatible with the game engine. In this final format all the animations are stored as key frame animations. The key frame animations cannot change dynamically when they are used in the game engine. A dynamic animation, or a code generated animation, is the term used for the animations that are created or modified in real-time in the game engine.

The tools that are available when rigging for games are rather limited compared to the pre-rendered movies. The tools that are available for pre-rendered movies are not computationally light enough to be used in real-time. The basic bone structures and animations created by using bone constraints are well supported, but only in the key frame animations.

Morph target animation is supported in many game engines. The morph target animation is an alternative animation method for games, and it is commonly used for the facial or environmental animation. It can be used for character animation but for this purpose it is very inaccurate and can be only used if characters are small on screen. In this method each vertex location is stored in the key frames. Because of this, creating animation using morph targets is slow and takes more storage space than bone animation. The morph target based animation is tied to the 3D mesh it was created for, so unlike bone animation, it cannot be reused for 3D meshes with different topology (Murdock 2008, 650).

Developing animation tools customized specifically for the game engine in use can speed up the development considerably. These tools can give technological advantages and features that cannot be created by key frame animation.

Game companies often build custom animation tools for their specific needs. In most cases the use of these tools is not visible to the player. These tools often automate repetitive tasks or improve compatibility between game engine and animation software used in development.

Sometimes new tools are developed to add a unique animation feature to the game. The new tools like these improve visuals and may give the game a technological edge. New technological features can boost sales; the games using advanced technology often get more visibility in the media.

Developing custom animation tools can be expensive and time consuming. Lots of skill and knowhow is required from the programmers and artists, not to mention the expensive hardware that may be required for motion capture or other purposes. These tools often add new problems to the workflow which must be solved. New tools can be very beneficial to company but the possible costs in development time and money must be carefully reviewed.

4.6.1 3Ds max biped

The biped is a character animation tool in 3Ds Max (Murdock 2008, 1000). Many game companies that use the 3Ds max for animation also use the biped toolset in some form. One of the biped's strengths lies at using motion capture data. The motion capture data is usually very compatible with biped.

The 3Ds Max is not designed specifically for creating animation for games; instead it is designed to be tool for all things related to 3D graphics. For example, Maya is designed more as an animation tool and, therefore, has more extensive animation toolset.

4.6.2 Motion capture rigs

Motion capture is a valuable tool for creating animations for games and movies. The motion capture tools used in movies and games are fundamentally the same.

The motion capture produces a predefined rig structure that has captured animation stored in key frames. To be used in games, motion capture rigs animation must be copied to the game character rig. Often the motion capture rig and game character rig are not identical and animation cannot be directly transferred between rigs. This process is called animation mapping or retargeting and it is done in 3D software or in software specialized for animation and motion capture. The Motion builder is popular software for this purpose and is widely used in the industry. (Radoff 2008.)

4.7 Existing methods for creating real-time muscle animation

Recently games have started utilizing muscle animation in situations where the lack of muscle movement would be the most noticeable. The most notable cases are the Red Dead Redemptions horses and boxers in the Fight Night series. While some might argue that the muscle deformation is quite unnoticeable graphical feature, the human eye seems to be good at picking up the subtle motions of the muscles. (Barreby 2009.)

The muscle animation used in games today can be divided in two groups, the muscle simulation by modifying texture maps in real-time and the muscle simulation by modifying geometry in real-time.

The muscle animation by modifying texture maps in real-time is the most common technique. Modifying the texture maps in real-time is not a new trick in game graphics; it has been used successfully in facial animations. The same techniques used in the facial animation can be applied to the muscle animation. Techniques like the texture blending and the wrinkle maps used to create wrinkles in faces, can be used also to create wrinkles and creases of the muscles. This technique involves blending two or more texture maps. These textures mimic the forms the skin takes when the underlying muscles contract or relax. When these two textures are blended gradually between each other when limbs move it creates an illusion that muscles are moving below the skin. The texture blending is easy to implement on

any modern game engine. The texture blending requires a control logic which defines when to blend the textures and how much. Changes in polygon area or the bone positions with additional texture masks can be used to control the blending of the textures. Depending on the technique used, the control logic can be stored in 3D format in use or calculated in real-time by the game engine. Using the texture blending is effective way to do muscle animation as it is really visible and easy to implement. The artist has total control over this method. As a downside modifying the textures maps has no influence on the silhouette of the character.

The muscle animation by modifying the geometry in real-time is less used, less noticeable and more time consuming method than modifying the texture maps. It is a rarely seen feature in games because it is thought that cost in artist time used does not contribute significantly to the visuals quality. It has one huge advantage however, by modifying the geometry the characters silhouette can be altered. Using this method the muscles really move, contract and expand. In many games this would be overkill, but for some special cases it can give really good boost to the visuals, as seen in the Fight Night series. Many problems exist using this method. First problem is that there is no standard way to produce this effect, and unseen problems may arise when implementing it. Producing good looking results that would work in almost any case with minimal additional work for the animator is complicated. This method relies heavily on talents of the rigger.

The best visual quality can be achieved by implementing both of the methods, the modification of textures and the mesh in real-time. The Fight Night series is one of the few games that currently do this, but as technology develops and computing power increases I believe these features become more common.

There are other possible ways to produce muscle movement in real-time, which are not currently used in games. These methods may involve soft body physics or other code generated texture or geometry manipulation. These methods are complicated, heavy and take the control away from the artist.

The game companies are secretive about the latest techniques and software they use. Muscle animation is very new and rare feature. For this reason it is hard to find information on exactly what technique was used in various titles. However from interviews and blogs you can build a good idea on how they implement graphical features such as muscle animation.

In Red Dead Redemption, developed by Rockstar Games, the horses have very noticeable muscle animation. Developer from the Rockstar Games told in an interview that they used a technique quite common in the facial animation to produce this effect (Sony 2011). It is safe to say that in this technique texture maps are blended depending on texture masks and bone rotation or location. It is difficult to say whether muscle animation on 3D mesh level was done. It is possible that developers used polygons stretching values as a mask to blend the textures. If this was the case, the muscle animation is done in the mesh level also.

Fight Night series is known for its photorealistic 3D boxers. Boxers in the Fight Night series look impressive. The boxers muscles flex, fat tissue jiggles and muscle grains show trough skin. The techniques used in this game are by principle very similar, but more sophisticated than used by Rockstar Games. In a blog developer talks about about complex bone structure which also blends textures and physics driven fat tissue simulation. (Ben Ross 2011.)

4.8 Muscle animation in theory and practice

Muscle simulation or animation has been researched from the early days of computer graphics. One of the earliest papers on muscle simulation was presented at SIGGRAPH in 87 (Chadwick, Haumann & Parent 1989). Although this method is not directly used today, it was ground breaking to its time.

In facial animation muscle simulation can help to create more realistic results. Movements of facial muscles can be determined from motion capture data. Accurate jaw motion can also be extracted from the marker data. Capturing muscle movement doesn't necessarily require complicated hardware. (Sifakis, Neverov & Fedkiw 2005.) The deformations achieved using this kind of method would have to be stored as morph target animation to be used in game engine. Similar method could probably be applicable to character animation as a whole but it seems to be unnecessarily complicated approach.

NURBS-Based models to create muscle animation have been researched. NURBS have advantages over pure polygonal modeling when simulating muscles. This particular method was not designed for animation. (Zhou & Lu 2005) Although NURBS approach is tempting considering the qualities of NURBS modeling, NURBS are arguably not well supported in most modeling software and even less in game engines when it comes to character animation. If animator would have this kind of NURBS based tool and wishes to use it in game

animation the logical approach would somehow convert the deformation data to morph target animation. This kind of method seems overly complicated and would at least require expensive custom tools. In pre rendered movies NURBS are used to simulate muscle movement (Stinson Thuriot 2005). Simulations tools used in movies are often produced by studio and not available commercially (Ritchie, Callery & Biri 2007 a).

References for muscle animation and especially real-time muscle animation are hard to come by. In pre rendered movies muscle animations are done using muscle simulation plugins, PSD tools or combination of both. Muscle simulation is often too heavy computationally to achieve in real-time, however PSD and similar techniques can be used. PSD techniques or similar techniques using morph targets offer a tempting solution for creating muscle animation. Morph target animation is supported by many game engines starting from games like halo 2 and half-life 2 (Ritchie 2007 b). When morph target animation is driven by bone rotation or location complex deformation can be created. This technique can be used to correct bad deformations at problem areas caused by skeletal animation or to do facial animation (Hess 2009). This is also one of the most viable techniques to create muscle animation for games. Disney used PSD in the movie bolt to create more advanced deformations (Lee & Hanner n/d).

Problem with PSD or similar technique is that it requires deformation calculations from skeletal animation and morph target animation to work together. This can be computationally more expensive and requires additionally support from game engine. Setting up and testing a rig that uses both methods is time consuming and for each character with different topology this process has to done individually. One character can easily require more than 15 different morph targets and more than 30 if the character is asymmetrical. Compatibly of these morph targets can present problems when 2 morph targets area of influence overlaps each other. Because these reason different methods was used in the project part.

5 PROJECT DOCUMENTATION

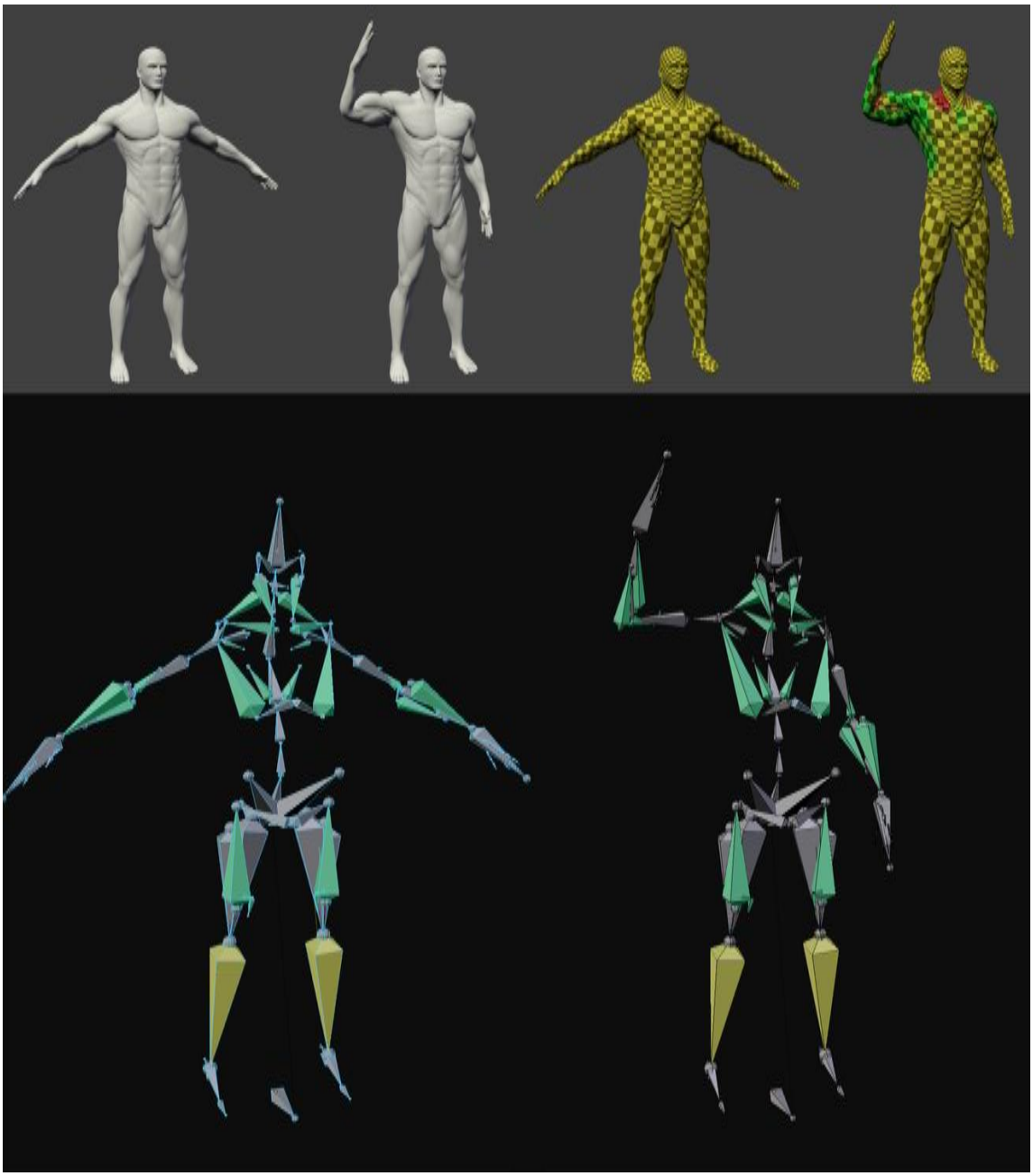

Figure 2. Dynamic texture blending and muscle bone structure

Believable muscle animation requires that the silhouette and fine details change as the muscles move. In game graphics, the character mesh defines the silhouette and texture is used for the details.

Texture blending is an easy and lightweight method to fake the skin wrinkles and creases that appear near the joints or around the muscles when they are flexed. As mentioned earlier

the texture blending needs controls to work properly. Something must define when to blend textures and how much. In the facial animations common practice is to use a separate texture masks to define where to blend the texture. The texture mask has color coded areas which each are linked to the bones moving the corresponding areas. These bones rotations or locations define how much the textures are blended. For muscle simulation this technique is not as suitable as for facial animation. Creating the texture mask for each muscle croup can be tedious and due to limitations in the method could require several texture masks.

In real life creases and wrinkles happen because the skin stretches or contracts. When using a standard rig structure the 3D mesh stretches and contracts the similar way as the skin at the joint areas but not at the area around the muscles. If rig is built so that it deforms also the muscle areas, the 3D mesh will stretch very similar way as real skin would (Figure 2). The stretching values can be calculated by measuring changes in the polygon area; using these values a dynamic texture mask can be calculated. This dynamical texture mask can be used to blend between multiple different texture layers each with multiple different stages. In the project three different stages were used; normal, squashed and stretched. The technique I decided to apply to the rig relays heavily on ideas presented by Christopher Oat in the paper Real-Time Wrinkles (Oat 2007 a). These texture layers can be any texture map type desired, the most useful textures for purpose of faking changes in skin would be normal and cavity maps. For the purposes of this thesis I used three different checkered diffuse maps, yellow for no change in polygon area, red for decrease in polygon area and green for increase in polygon area (Figure 2.). These three textures are used to clearly demonstrate how much and where textures are blended.

5.1 Building the rig

There is lot of information on regarding creating the muscle animation for 3D animated movies. Problem is that all the rigging techniques used in pre-rendered movies cannot be applied to the game character rigs. Unlike pre-rendered movies, the games run in real-time and the rig must be computationally lightweight and compatible with the current game engines. (Sanders n/d.) The rigging method I'm using is similar to ones used in the movies, but modified to be compatible for real-time applications such as games.

The game character animation process relays heavily on motion capture. For this reason my rig is built so that it is usable with the motion capture data. This means basically that control bones and bone constraints are designed so that attaching motion capture data stays simple and straightforward.

Organic matter has layers and complex structures that are very difficult to simulate accurately. However accurate simulation is not needed to give the player an illusion that the game character is made of organic matter.

5.2 Muscle bones

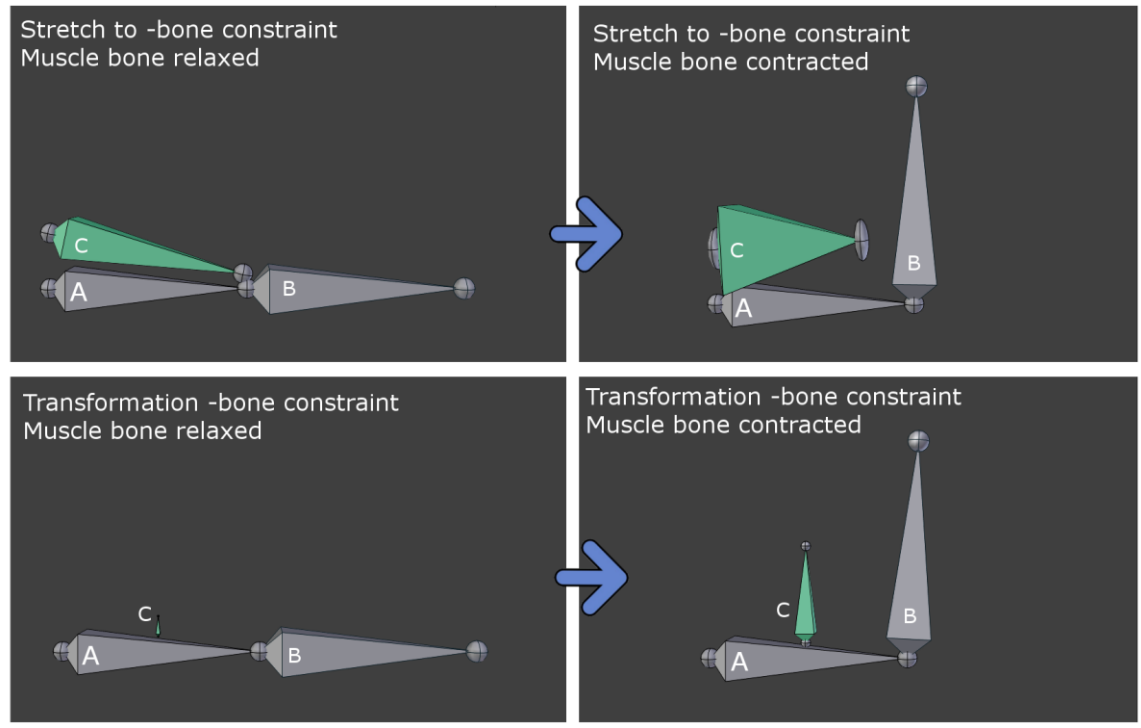

Figure 3. The logic behind the two bone constraints used to create muscle bones.

Because the rigs used to animate the game character is basically a simplified human skeleton, I decided to add a layer of bones on top of it, which I call muscle bones. These so called muscle bones behave the same way as the deformer bones with the exception that they have bone constraints applied to them. These bone constraints that can be found on most 3D software. Most of the muscle bones have a stretch to -constraint applied to them. The stretch to -constraint tells the bone to stretch to a specified target, while keeping original volume. This means that if the bone length decreases it will scale up to keep the original

volume and vice versa. The target specified for the constraint can be any 3D object or sub object. For my purposes I'm using other bones as the target objects. In few cases the stretch to -constraint was not accurate enough, so instead of the stretch to -constraint I used the transform -constraint. Transform -constraint gives the two bones customized relationship. For example when the deformer bone rotates a certain amount the muscle bone expand in relation to it a certain amount.

Muscle bones are colored green. When using stretch to –constraint muscle bone C is parented to bone A but its rotation is tracked to a point close to tail of bone B. When bone B rotates, muscle bone C gets shorter because its other end is parented to bone A and other end tracks to bone B. Muscle bone C expands to keep the original volume. In the example the muscle bone C volume is multiplied to better illustrate the transformation. (Figure 3.)

When using transformation -constraint the muscle bone C is parented to bone A. Muscle bone C is scaled up by a factor of five if bone B rotates around its x-axis 90 degrees. Smaller rotation values of bone B result in smaller scaling values, for example, rotation of 45 degrees scales the muscle bone C by a factor of two and a half. (Figure 3.)

The muscle bones are attached to the deformer bones and they follow the animations of the rig. Normally the 3D mesh would be attached to only to the deformer bones, but in my method they are attached to the muscle bones also. The motion capture data can be used without additional work since muscle bones are moved automatically by the rig. When the rig moves the muscle bones expand and contract and the 3D mesh will follow accordingly. Using rig with muscle bones is computationally little heavier than traditional rig structures because adding muscle bones to the rig increases bone count.

5.3 Applying rig to character

The rig must be attached to the 3D mesh by process called weighting, also called weight painting or rigging. In the weighting process vertices are attached to the bones by a weight value. The weight values for single vertex are normalized and always add up to one. The bigger the value by which vertex is attached to the bone the more the vertex is moved by it. Weighting is done inside of the 3D program used.

Most of the 3D software has automatic weighting tool. Automatic weighting tools tend to perform the better the more bones the rig has. This tool gives a good starting point to the weighting. The vertex weights need to be fine-tuned manually since automatic weighting tool alone rarely gives perfect results. Before manually editing the vertex weights it is a good idea to slightly modify the positions of the bones and run automatic weighting again. Moving the bones and using the tool again can often be enough to correct deformations that are slightly off target. This is often much faster than weighting manually.

When using the automatic weighting tool, vertices may get attached to bones with really small values. These small values should be cleaned, in other words changed to zero, since the difference which they make to animation is not visible to player but require more computational resources. This rarely causes problems unless the game has many animated characters simultaneously on the screen. Cleaning unnecessary values is a good practice and since it takes only little time it should not be overlooked. Most 3D software have tool that allows rigger to delete values below wanted accuracy. Depending on the software or game engine used this can be done automatically by exporter or importer.

Using the automatic weighting tool can lead to situation where vertex is attached to more than 4 bones. These vertices should be reassigned to maximum 4 bones. This is because most game engines limit the number to four, and the deformations can otherwise look different in 3D software than the game engine.

5.4 Blending the texture maps

Using masks created from polygon area is a rarely used method for blending texture maps. These masks have many advantages, they save time because masks do not need to be created manually or stored in the memory instead they are created dynamically in real-time. Because dynamic masks are side product of the rig, they work automatically when the rig is applied to different characters.

As always there are also disadvantages. Getting the blending values and areas right is difficult since none of the 3D software have tools designed for previewing the blending in real-time. Different animation poses need to be previewed to ensure that the 3D mesh deforms and textures blend accordingly. Previewing texture blending is not possible in real-time without a custom real-time shader. Functionality of this kind of real-time shader can be emulated by

using pre-rendered materials. Texture blending cannot be previewed in real-time when using pre-rendered materials; instead each animation frame has to be rendered out. The need to render each frame makes previewing texture blending slow. Pre-rendered materials are sufficient when used purely for demonstration purposes. If similar methods would be used in a game project, writing a custom real-time shader for previewing in the 3D software viewport would be beneficial.

The texture blending uses texture masks calculated from polygon area. The different textures start to blend together when polygon area decreases or increases over a threshold or dead zone defined in the pre-rendered material. The thresholds or dead zones of the texture blending need to be adjusted for each different character. Estimating the right values can be tricky but can be found through trial and error in a few minutes.

5.5 The rig in game engine

The file containing the 3D mesh, rig and animation need to be moved from the 3D software to the game engine. Before this can be done the file need to be converted to a suitable format. When the 3D software creates the final format used in game engine this is called exporting. When game engine creates the final format this is called importing. Game engines may have their own custom file formats. A few standard formats are also used such as the fbx and collada. The rig was tested in the unity game engine using the fbx file format. The rig worked as expected and no problems were found.

To get the texture blending working in the game engine a special shader is required. This kind of shader has been demonstrated in Ati's real-time demo called White out. (Oat 2007 b.) Writing identical shader for 3D software used would make previewing texture blending faster and more accurate. The shader was not tested in this thesis project.

6 COMPARING THE RESULT

In addition to the rig with muscle animation a standard rig without muscle animation was build. These two rigs were attached to identical character meshes. The two rigs were then given identical animations and the differences between deformations and texture blending were compared between the two. The character mesh used for demonstration was created by Nick Zuccarello (Zuccarello 2012).

From the beginning it was clear that texture blending by polygon area mask would not work properly with standard rigs except for the area around the joints. However I wanted to see how much better the muscle rig was for this purpose than the standard one.

Several frames were rendered to images to compare the differences between the two rigs. The differences are more noticeable in animation but even from the still images the differences can be observed.

6.1 The upper body

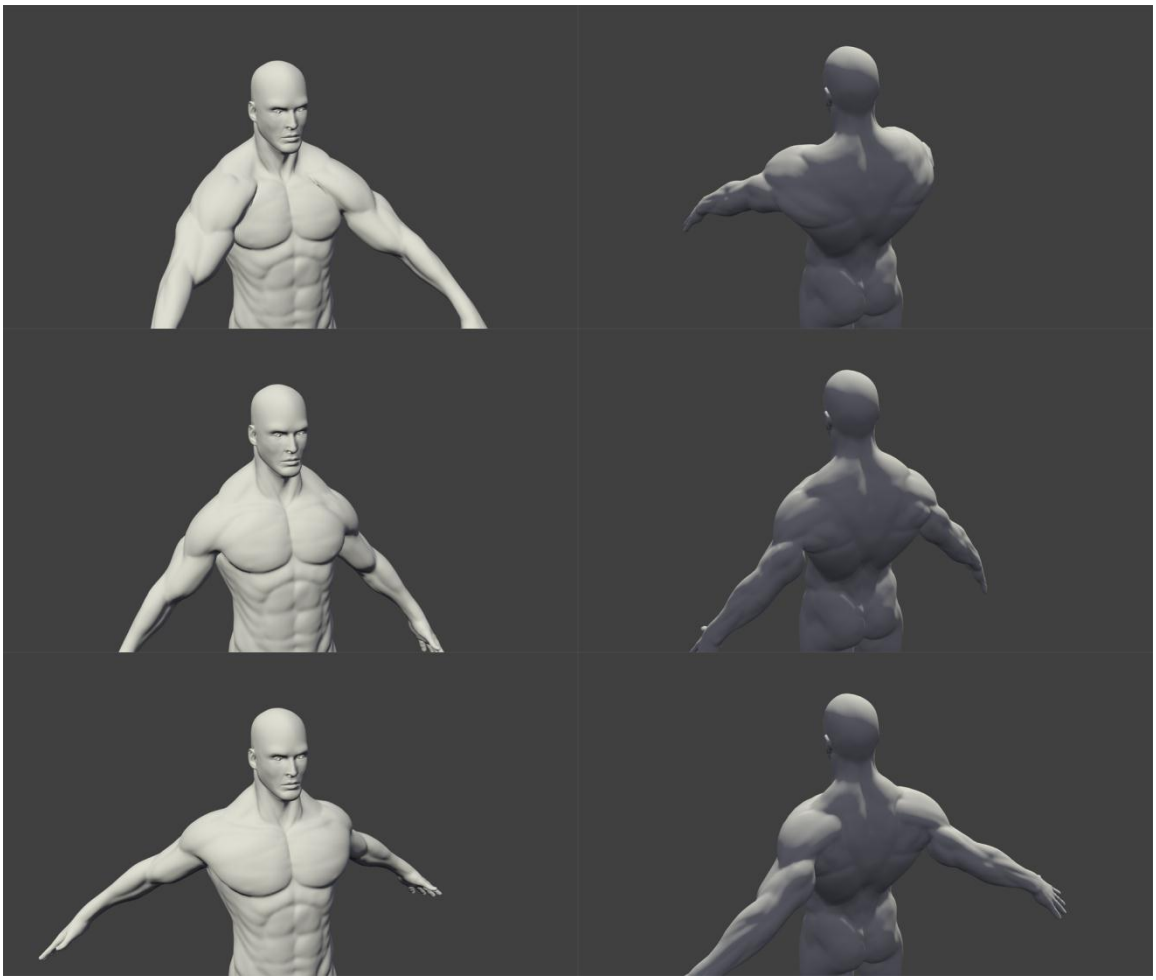

Figure 4. Previewing the mesh deformation on the rig without muscle animation.

The chest and the back muscles stay stationary in figure and because of that look unnatural. When the shoulders are moved forward, the mesh around the scapula stretches too much, when instead the middle part of the back should stretch. (Figure 4.)

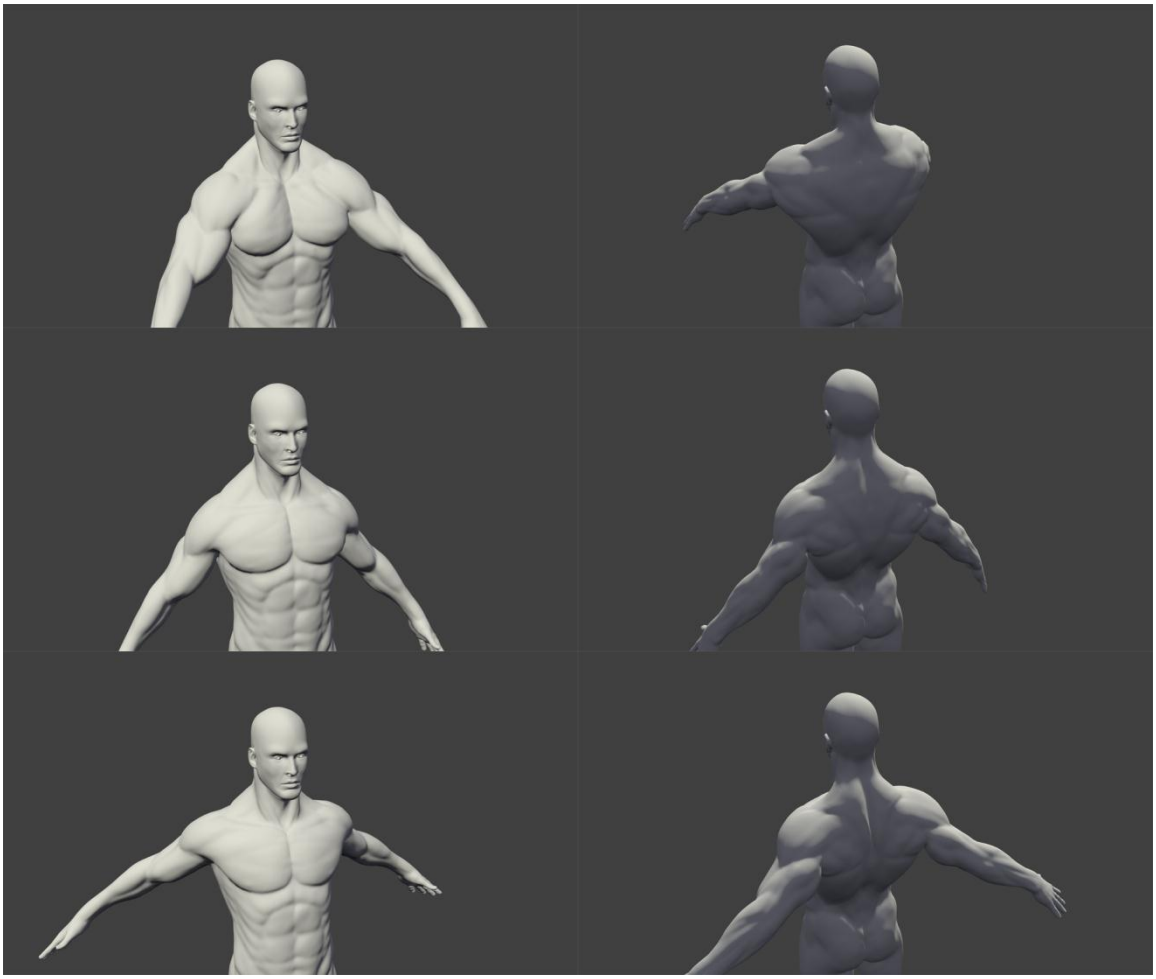

Figure 5. Previewing a mesh deformation on the rig with a muscle animation.

The chest muscles have now rounder shape when the shoulders are moved forward. The back muscles stretch accordingly from the center when moving the shoulders forward. The chest muscles stretch and flatten nicely when the shoulders are moved backwards. The back area looks more organic and mesh around the scapula no longer stretches unnaturally. Deformations are now smoother in general. (Figure 5.)

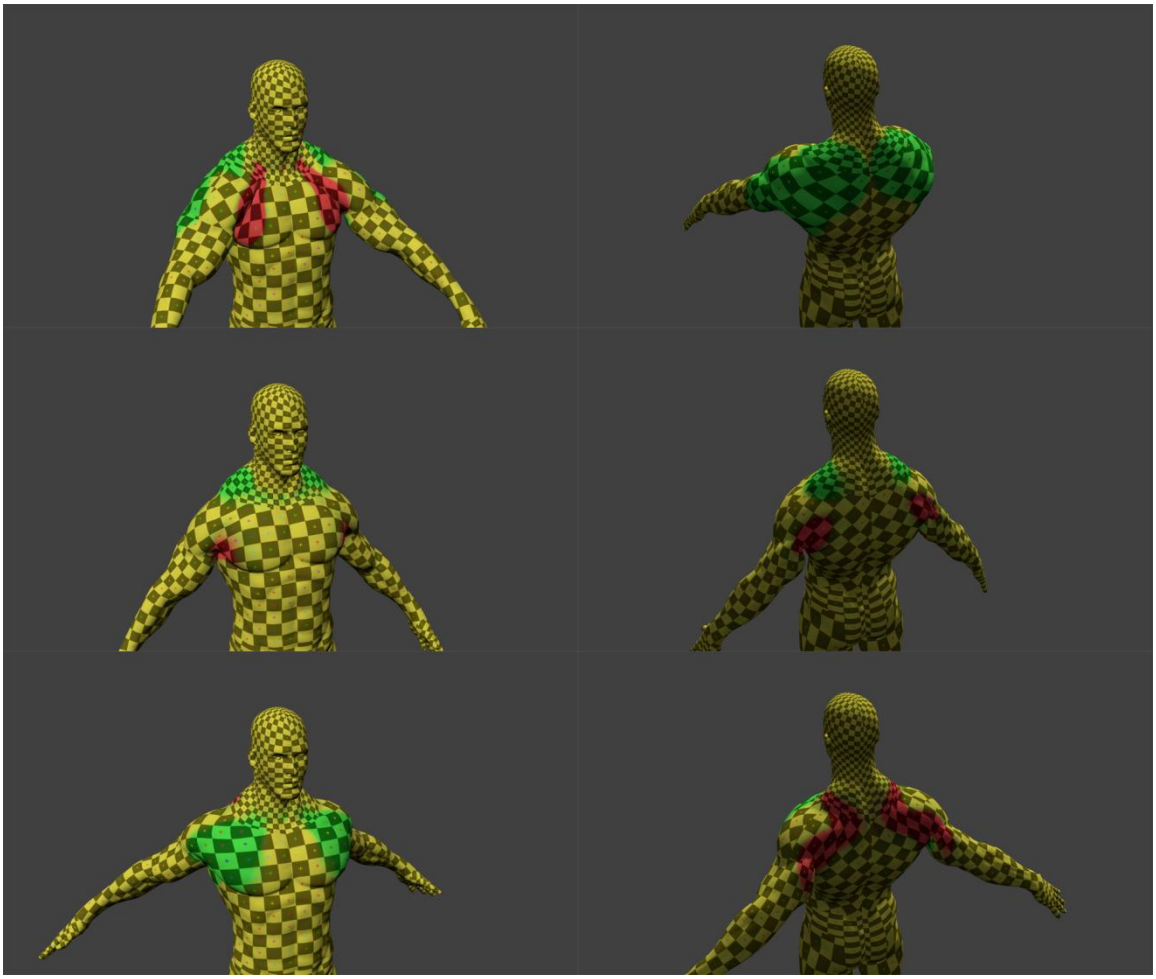

Figure 6. Previewing the texture blending on the rig without a muscle animation.

When the shoulders move forward the textures blend at the chest muscle area but not at the center. At the backside textures blend at right places but because scapula area stretches unnaturally texture blending would not probably work. When shoulders are moved backwards the textures blend at the chest muscle area but not at the center. At the back side textures blend in a strange pattern and at lower back textures do not blend at all. These dynamic masks would not be suitable for blending textures during animation. (Figure 6.)

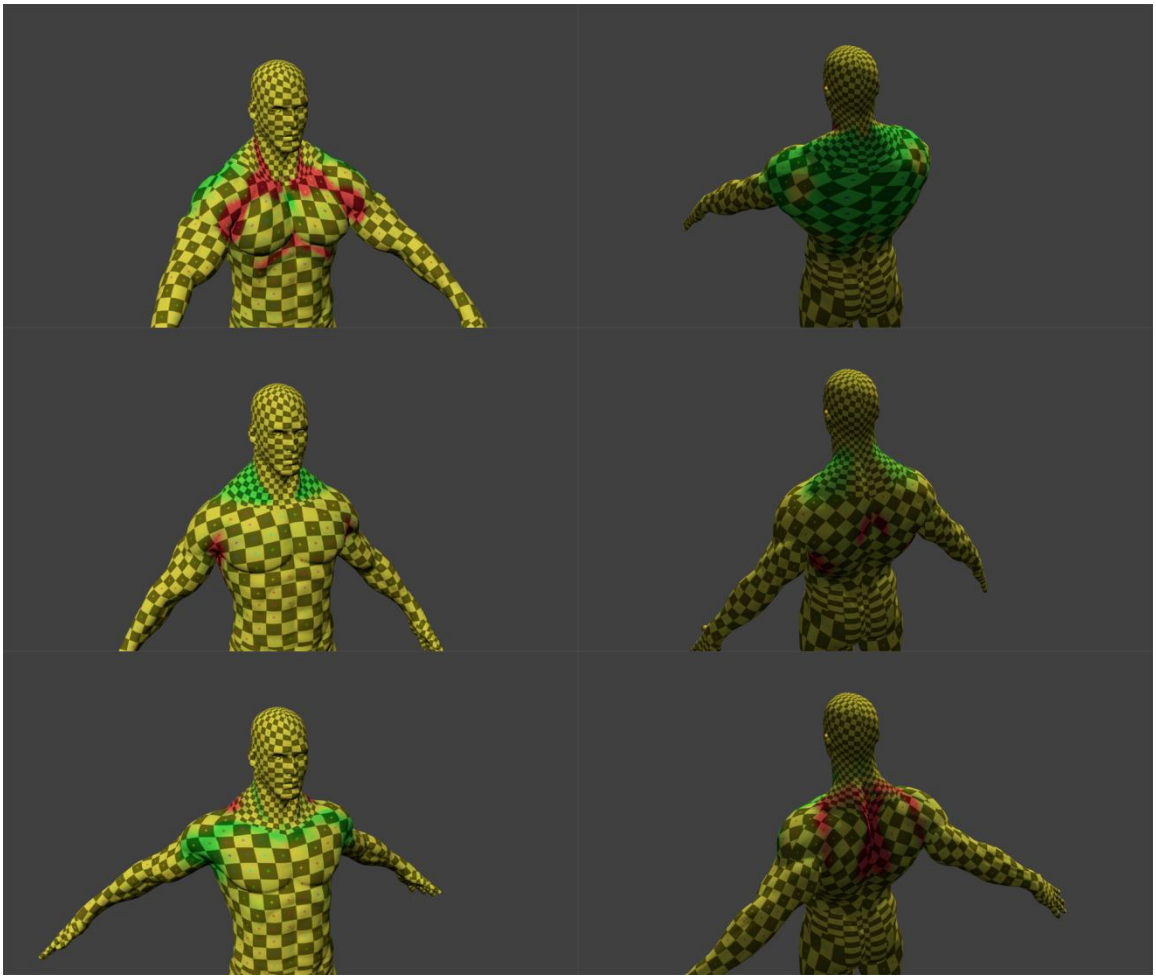

Figure 7. Previewing the texture blending on the rig with muscle animation.

When the shoulders are moved forward, the areas around the chest muscles squash accordingly and blend to the red texture. The chest muscles blend to the green texture as it should, but only a little and is barely noticeable in the Figure. Because the deformation of the mesh was correct, the texture blending threshold should be adjusted instead of revising the whole underlying bone structure. The back area stretches at the right places and the scapula area stays intact. When shoulders are moved backwards the chest area stretches very nicely. The whole chest area blends to the green texture but at the lower part it is barely noticeable. Reason for this is that the blending threshold is too big. This is the same problem as in the shoulders forward pose, adjusting the blending threshold is again needed. With little adjustments the chest area works as it should. The texture blending happens in the right places and could be used for texture blending. (Figure 7.)

6.2 The hands

Figure 8. The mesh deformation comparison.

In the upper row the rig without muscle animation is used. Biceps and triceps are not moving which gives a toy like appearance. This is common situation in games released recently. In the lower row a rig with the muscle animation is used. The biceps and triceps muscles contract and expand depending on arm movement. The shoulder area and the chest muscles move also a little which gives an organic feel to the character. Deformation at the elbow area looks also more natural. The biceps movement is the most noticeable improvement when using rig with muscle animation. The bicep was the easiest muscle to animate. Lack of bicep movement can be very visible. Based on the results, game character biceps should be always animated if they are not covered by clothing. (Figure 8.)

Figure 9. The texture blending comparison.

In the upper row the rig without the muscle animation is used. This is a quite complex pose and blending textures using mask created from polygon area just does not work anymore when using standard rig. Without the muscle animation the mesh around the chest, back, triceps and biceps area do not stretch accordingly and texture blending does not happen at the right places. In the lower row the rig with muscle animation is used. The muscle rig stretches the mesh in the same way as the skin stretches in real life. The mesh around chest, back, triceps and biceps area stretch accordingly and these areas blend to green texture. Again the chest muscles blend to green texture, but too little. Thresholds need adjusting to get more texture blending at chest muscle area. (Figure 9.)

6.3 The legs

Figure 10. Previewing the mesh deformation on a rig without muscle animation.

At the legs, deformation looks good even without muscle animation. Despite of large muscles, lack of muscle movement in the legs are not that noticeable. Even though muscles stay stationary player would not probably notice it. (Figure 10.)

Figure 11. Previewing the texture blending on a rig without muscle animation.

Again blending textures using mask created from polygon area just does not work with standard rig. Polygons area changes only around joints and at best these masks could be used to create wrinkles around joints. Blending textures on top of muscles is not possible since at those parts mesh is stationary. (Figure 11.)

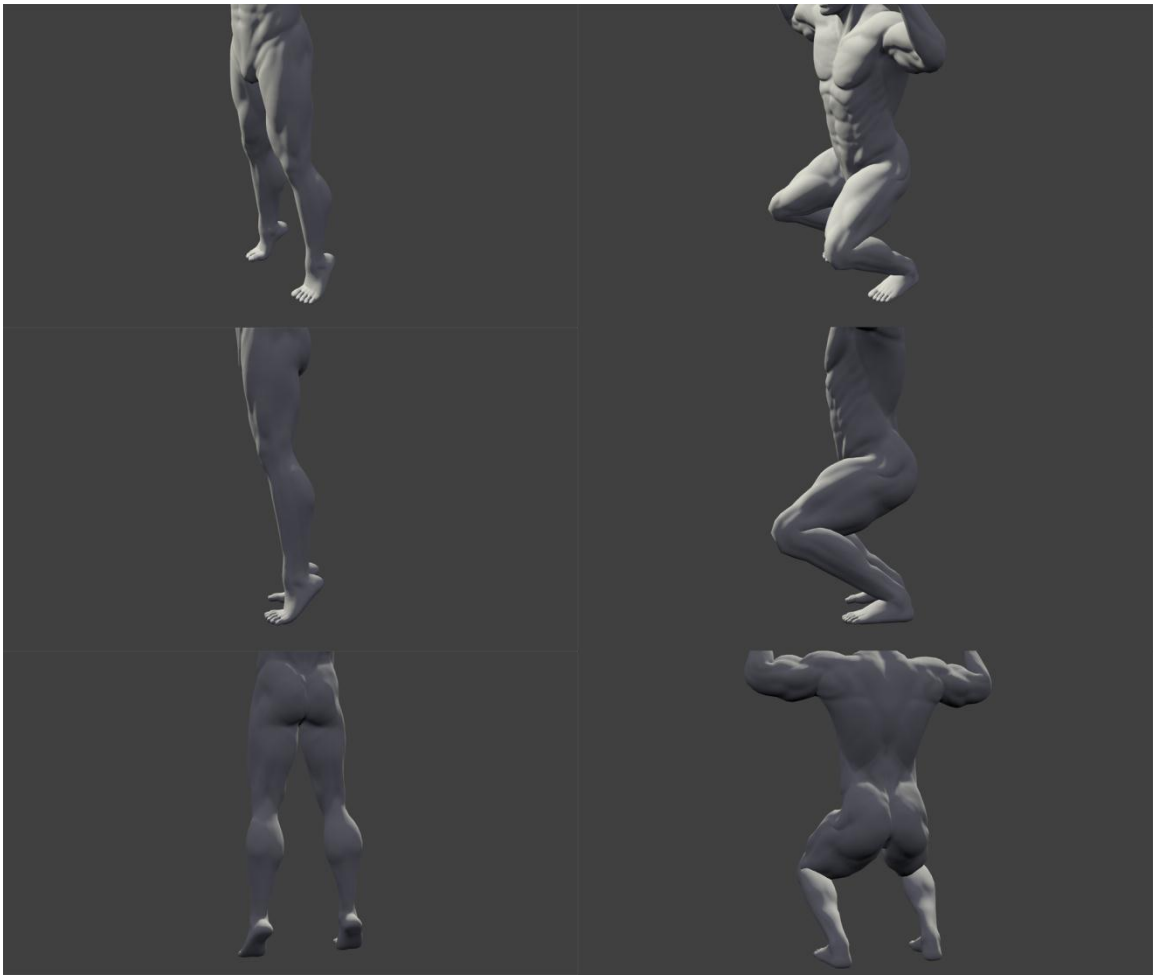

Figure 12. Previewing the mesh deformation on a rig with muscle animation.

The deformations look slightly better in the legs with muscle animation. The differences between muscle animation and no muscle animation are not that obvious in still images. Muscle flex can be seen when comparing the figure 10 and figure 12, but perhaps to only to trained eye or for those who are specifically looking for it. However, the muscle animation is really subtle effect when applied to legs. In animated tests something unexpected happens. The muscle flex itself is not very visible but great feeling of weight or mass is transmitted from the animation when muscle animation rig is used. I hope I had more time to investigate this phenomenon but I suspect that even small flexing in leg muscles is somehow picked up by our eyes. This is great news since feeling of weight can be hard to portray in animation but when using muscle rig this seems to happen automatically. (Figure 12.)

Figure 13. Previewing the texture blending on a rig with muscle animation.

The muscle rig stretches the mesh in the same way as the skin stretches in real life. The mesh around legs stretches accordingly and blends the textures nicely. Again there are some minor flaws, this time around the thighs. Area around the thighs blends to green, but in some parts too little. Once again threshold needs to be adjusted. Calf muscles are other problem area; partly because the mesh stretches in one way and contracts in the other, resulting in too small change in the polygon area to texture blending to start take effect. This is a quite rare situation but one of the drawbacks of this method and should be taken into consideration when building these kinds of rigs. In this case the problem is fixed by tweaking the muscle bone structure. The texture blending and deformations around ankles look a bit unnatural and need more work. (Figure 13.)

7 CONCLUSIONS

The muscle animation was easy and fast to implement and for a more experienced rigger it would be even more so. So far the muscle rig has worked as I anticipated and in some cases even better. Using muscle rig in game development takes more time and effort in some stages but in turn saves them in others. More research would be needed to see if texture blending looks good with texture sets with realistic muscle forms and wrinkles. The time it takes to create such texture sets is also an interesting question and should be investigated. For certain game types building muscle rigs would make lot of sense and I believe they will come more common.

Muscle animation can give the game characters more realism and believability. It is the small features like the muscle animation that make them appear more lifelike. Making game character feel lifelike is important when building immersion. The magic in games happens when player forgets for a moment that his looking at digital character.

8 FUTURE DEVELOPMENT

8.1 Applying motion capture data to the character rig

Now that the rig works as expected, next step would be to attach motion capture animation. Blender's motion capture tools are rather limited but extensive enough for game character animation. Motion builder can be used to along blender to edit motion capture files, but since bone constraints are not interchangeable, the animations must be finalized in Blender.

8.2 Additional controls for the muscles

Muscle contraction or expansion happens when moving or rotating a limb, but muscles can be flexed also when limb stays stationary. Muscle movement also depends on situation, for example when lifting a light object muscles stay relaxed and the muscle movement is not that obvious. For these situations custom animation controls should be created.

8.3 Properties of the custom shader

Research should be made on regarding the properties of the custom shader needed to blend the texture maps. Things that should be researched include: What requirements of the shader from the hardware and software, how big are the costs in computational resources and how complicated is it to write the shader.

REFERENCES

3d kingdoms 2006 skeletal animation #1. Available: http://www.3dkingdoms.com/weekly/weekly.php?a=4. (Read 25.4.2012).

Autodesk n/d. Maya Muscle Advanced Techniques. Available: http://images.autodesk.com/adsk/files/mayamuscleadvancedtechniques.pdf. (Read 15.3.2012).

Barreby, M. 2009. When is it Necessary to Use Muscle Systems to enhance 3D Animation? Available: http://hig.diva-portal.org/smash/get/diva2:222923/FULLTEXT01. (Read 24.3.2012).

Chadwick, J., Haumann, D. & Parent, R. 1989. Layered Construction for Deformable Animate characters. Available: http://chrisevans3d.com/files/reference/layered89.pdf. (Read 20.5.2012).

Harkins, J. 2003. Rigging and Anatomy: Making your characters move convincingly. Available: http://www.animationartist.com/2003/08_aug/tutorials/rigging_anatomy.htm. (Read: 5.3.2012).

Hess, R. 2009. Animating with Blender. USA: Elsevier Inc.

InnerBody n/d Human Joints and Mechanical Equivalents. Available: http://www.innerbody.com/image/skel07.html. (Read 25.4.2012).

Lee, G. & Hanner, F. n/d. Practical Experiences with Pose Space Deformation. Available: http://www.disneyanimation.com/library/poseSpaceDef.pdf. (Read 19.5.2012).

Murdock, Kelly. 2008. 3ds Max 2009 Bible. USA: Wikley Publishing Inc.

Nieminen, Mikael. 2009 Lowpoly-hahmon riggaus 3Ds Max-ohjelmassa. Available: https://publications.theseus.fi/handle/10024/3371. (Read 25.4.2012).

Oat, C. 2007 Real-Time Wrinkles. Available: http://developer.amd.com/media/gpu_assets/Oat-Wrinkles(Siggraph07).pdf. (Read 13.11.2011).

Physical & Sports Therapy n/d. Anatomy. Available: http://www.physicalandsportstherapy.co.uk/SkeletalMuscle.html. (Read 10.2.2012).

Radoff, J. 2008 anatomy of an MMORPG. Available: http://radoff.com/blog/2008/08/22/anatomy-of-an-mmorpg/. (Read 25.4.2012).

Ritchie, K., Callery, J. & Biri, K. 2007. The art of rigging: volume 1. Available: http://www.cgtoolkit.com/book1.htm. (Read 18.5.2012).

Ross, B. 2011. Fight Night Champion Art Blog. Available: http://www.ea.com/fight-night/blog/fight-night-champion-art-blog. (Read 10.12.2011).

Sanders, A-L. n/d. Animating for Video Games vs. Animating for Movies. Available: http://animation.about.com/od/videogameanimation/a/gamesvsmovies.htm. (Read 11.1.2012).

Sifakis, E., Neverov, I. & Fedkiw, R. 2005. Automatic determination of facial muscle activations from sparse motion capture marker data. Available: http://physbam.stanford.edu/~fedkiw/papers/stanford2005-04.pdf. (Read 20.5.2012).

Stinson, W. & Thuriot, P. 2005. Bulging Muscles and Sliding Skin: Deformation systems for Hellboy. Available: http://www.chrisevans3d.com/files/reference/065-stinson.pdf. (Read 19.5.2012).

Sony 2010. The people vs. red dead redemption. Available: http://uk.playstation.com/ps3/news/articles/detail/item281934/The-People-vs-Red-Dead-Redemption/. (Read 10.12.2011).

Williamson, J. 2011 Blender: Introduction to Character Rigging. Available: http://cgcookie.com/blender/2011/12/12/blender-introduction-to-character-rigging/. (Read 2.3.2012).

Zhou, X. & Lu, J. 2005. NURBS-based galerkin method and application to skeletal muscle modelling. Available: http://www.cin.ufpe.br/~sbm/CGartigos/p71-zhou.pdf. (Read 20.5.2012).

Zuccarello, N. 2012. Human Male Athletic Final Model. Available: http://nickzucc.blogspot.com/search/label/3D. (Read 5.11.2011).