An Interactive method to control Computer Animation in an intuitive way.

Andrea Piscitello
University of Illinois at Chicago
1200 W Harrison St, Chicago, IL
apisci2@uic.edu

Ettore Trainiti
University of Illinois at Chicago
1200 W Harrison St, Chicago, IL
etrain2@uic.edu

ABSTRACT
Human Computer Interaction is a rising field of computer science and perfectly fits among Computer Graphics scopes. The video-game industry is focusing on this field in order to provide always more realistic and engaging game experiences. The bond between animation and user movement seems to be destinated to be closer in future.

Author Keywords
Keyframing, Rigging, Mophing, Warping, Blending.

INTRODUCTION
Computer Animation is the process of generating animated images in computer graphics scenes. It is one of the most ancient branches of Computer Graphics and its foundations can be traced in 60's, when at Bell Labs, scientist were studying the possibilities of computer computations. The first remarkable example of computer-generated animation, however, is the 1972 short film "A Computer Animated Hand", produced by Edwin Catmull and Fred Parke at the University of Utah [1]. This film is very important since it contains many techniques and approaches which are the ancestors of the ones used until today. Mostly because of the poor technologies and computational power available at that time, the majority of the works conducted to make the animation were done manually. In fact a real model of the hand was constructed and polygons were drawn in ink on it.
Since 70's the exponential growth of computational power opened the door to advanced tools that accomplishes heavy tasks, like animation in an automatic way.
Today's most used techniques are morphing and skeletal animation. The first one basically consists in recording the position of all the vertexes of the mesh in some keyframes. The positions of vertexes in the frames between keyframes are calculated using interpolation. Keyframing and interpolation, together with other main techniques are described in the Pixar paper "Principles of Traditional Animation Applied to 3D Computer Animation" [2]. Morphing, which requires the recording of big quantity of data and the modeling of each keyframe by defining the position of each vertex, may result in very tedious task to perform. For this reason, skeletal animation is often preferred. This method is composed of two phases: rigging and animation. Rigging consists in defining a skeleton made of bones. Parts of the mesh are associated to each of these bones in order to define which bone is responsible of deforming a part of the object. The animation is then designed by defining the position of each bone in each keyframe and interpolating the inbetweens. For big productions like movies and videogames, the animation design is executed by means of more powerful techniques like for instance motion capture. Motion Capture is a technique, which takes advantage of special suits that actors dress and that capture every movement they do. This process can be also executed for capturing facial expression by simply putting special indicators on actors' faces.
With the technique described until now, it is possible to define just an animation at a time. In order to mix together two or more animations a different approach is needed. In Motion Warping paper by Andrew Witkin and Zoran Popovic [3] a simple technique for editing captured or keyframed animations is introduced. The idea is the same of image morphing applied to 3D animated geometries. Once a set of keyframe-like constraints is defined, this technique derives a smooth deformation that preserves the structure of the original motion. The authors' goal is in fact to derive new motion curves for inbetweens based on the specified key frames. Each motion curve is treated independently from each other and therefore each joint can be considered separately. A timewarp function can also be set to better fit the animations needs. The authors in the paper demonstrate that a wide range of realistic motions can be created or derived by warping and joining keyframes or motion clips. A limitation of this approach is that motion warping is a purely geometric technique and is totally unaware of the motion structure. The authors work also shares the same problem that image morphing has: extreme warps are prone to look distorted and unnatural.

METHOD
The method proposed in this paper include some of the techniques previously described and in particular animation is accomplished using keyframing and the skeletal approach. Different animations are then blended together using the motion warping technique.
The aim of this paper is also to provide the final user with some controls to drive the animation and manage the blending factors of different animations. Some examples of interactive computer graphics applications are provided in "Mo-

tion Sketching for Control of Rigid-Body Simulations" [4] and "Motion Doodles: An Interface for Sketching Character Motion" [5]. The first one provides a method for defining a physics-based application exploiting mouse motion or hand gestures. The second proposes an approach based on some pre-recorded animation that are mapped to user inputs performed through pen gestures on a tablet. These two approaches though consist in two main steps: animation description and successive execution.
The method proposed in this paper, on the contrary, provide a real-time technique to command the animation. Using a device which is equipped with appropriate sensors and/or actuators is very useful to properly control the animation.
The user interaction is a key point in the proposed method since allows to exploit animations in a real-time way and opens the door to a set of applications in the entertainment or educational field. Moreover, the abundance of sensors in common mobile devices may result the added value in the view of a wide commercial distribution.

IMPLEMENTATION

In order to provide a real feedback of all techniques introduced in this paper, a simple implementation has been produced. The demo that has been implemented, regards the modeling and the animation of a character walking and tilting arms, in such a way to recall a person moving on a narrow surface like a rope.

The workflow consisted in different steps:

- Object texturing
- Skeleton rigging
- Keyframing animation
- Scene programming
- Animations blending
- Mobile Controller Programming

The first three phases have been executed using Blender, a professional free and open-source 3D computer graphics software used for creating animated films, visual effects, interactive 3D applications and video games. For the following two steps *three.js*, a JavaScript library for WebGL, has been used. Three.js allows the user to easily produce graphics projects that are completely cross-platform exploiting the power of the browser programming.
Finally the mobile controller has been implemented as an application installed on an Android device.
The first step has been to find a model object to animate.

Object Texturing

After having selected the object model for the animation a simple color texture has been created and then applied to it. This is a very basic task to execute in Blender and it has taken just the time to learn the specific commands.

The object that has been used had been designed keeping the different parts of the body separated. In this way it has been very simple to apply the right texture color to the right part of the object. After the texturing process, the different components of the body have been merged together in order to be managed in an easier way during the following steps.

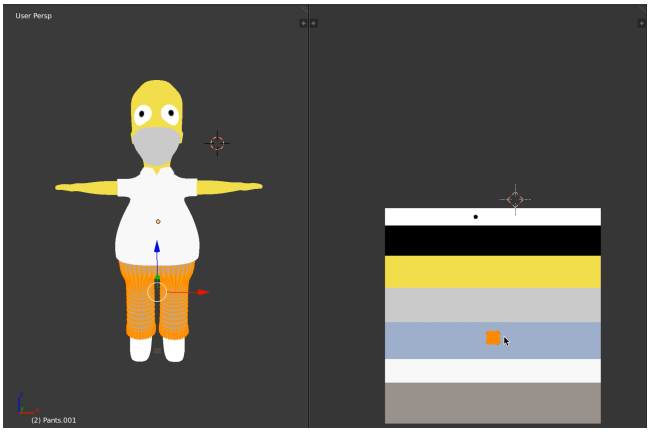

Figure 1. Texture image (right) and partial application of it (left).

Skeleton Rigging

This has probably been the most difficult and time consuming part. Very basic knowledge of human anatomy are sufficient to describe a working skeleton for a humanoid character. Although it can be accomplished in a relatively short time (once learned all the Blender commands and shortcuts), in order to obtain a realistic result, many corrections and optimizations have been necessary.
This process is basically made of two sub-steps. The first one consists in actually building the skeleton bone by bone. Each bone that has been created, correspond to the real human counterpart.

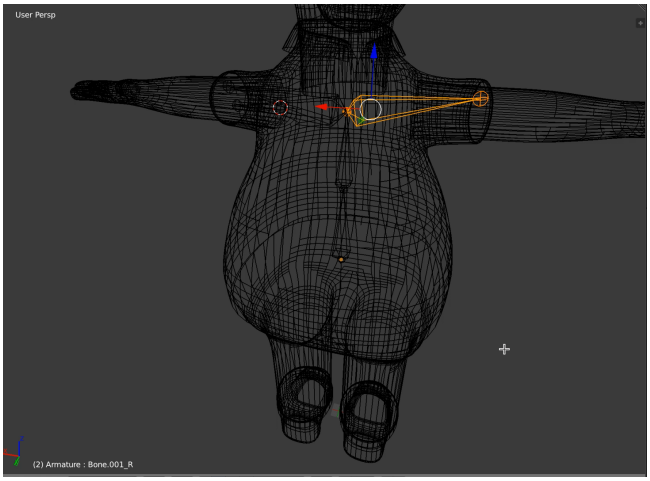

Figure 2. Arm bone positioning.

After having defined the whole skeleton, for each bone an area of interest is defined. This task can be accomplished quite easily by using some brush tools that allow to increase or decrease the influence of each bone on a specific area of the mesh. It is a very intuitive task, since the level of influence

of each bone is represented using a range of colors that goes from blue (0) to red (MAX).

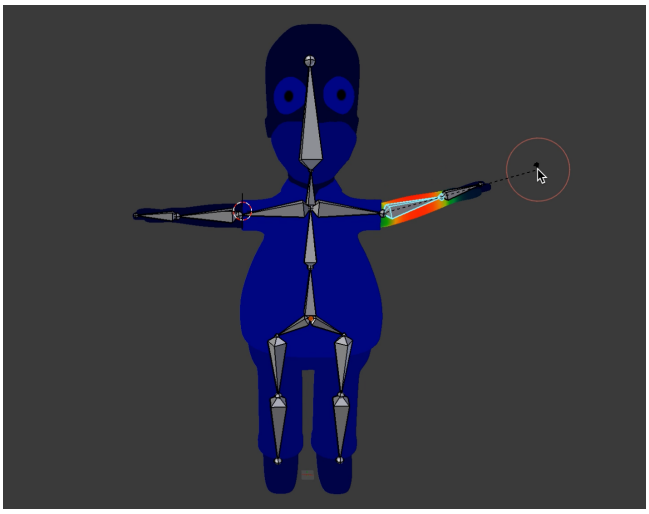

Figure 3. Area of influence of the left forearm.

The process have been reiterated different times because of some problems mostly due to the three.js exporter plugin for Blender. The right combination of some exporting parameters is needed to make animation work into three.js.

Keyframing animation

The animations have been designed with the keyframing technique. Few crucial keyframe have been defined leaving the interpolation task to the software responsible of the animation.

Five animations have been modeled:

- Idle position
- Left tilt
- Right tilt
- Forward Walk
- Backward Walk

As can be easily noticed, some of them are simply symmetric to each other.

Walk Animation Design

The walk animation design deserve a separate paragraph. Even if it seems to be an easy animation, its modeling is not properly straightforward. First of all it has been necessary to clearly figure out which bones are involved in walking and how they actually move. After a brief empiric study a first semi-realistic walk has been designed.

Taking advantage of some of the concepts described in the paper "Goal-Directed, Dynamic Animation of Human Walking" by Armin Bruderlin [6], a more realistic walk has been produced.

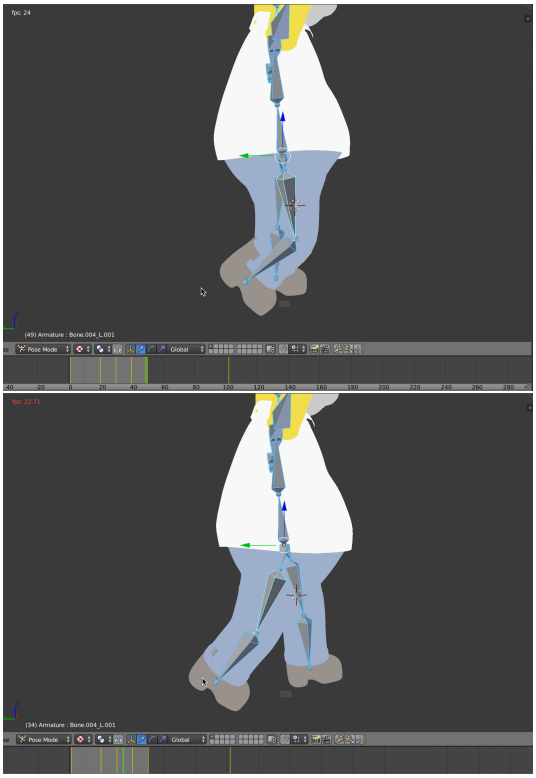

Figure 4. Some frames from the walk animation modeling.

Scene programming

This is the first step that have been actually executed on the three.js framework.
A simple scene is created with few commands. The ambient light is defined and two directional lights are added in different positions, in order to have a good overall illumination. Finally the mesh is loaded from the JSON file that has been previously exported from Blender.

Listing 1. Basic scene programming

```
scene = new THREE.Scene();

scene.add(new THREE.AmbientLight(GRAY));

var light = new THREE.DirectionalLight(WHITE,
    1.5);
light.position.set(0, 0, -1000);
scene.add(light);

var light2 = new THREE.DirectionalLight(WHITE,
    1.5);
light.position.set(0, 0, 1000);
scene.add(light2);

blendMesh = new THREE.BlendCharacter();
blendMesh.load("model/homer.json",start);
```

A simple menu with basic controls to command the animation has been implemented. It allows to define the weights of all the animation that have to be blended in the final one.

Animations blending

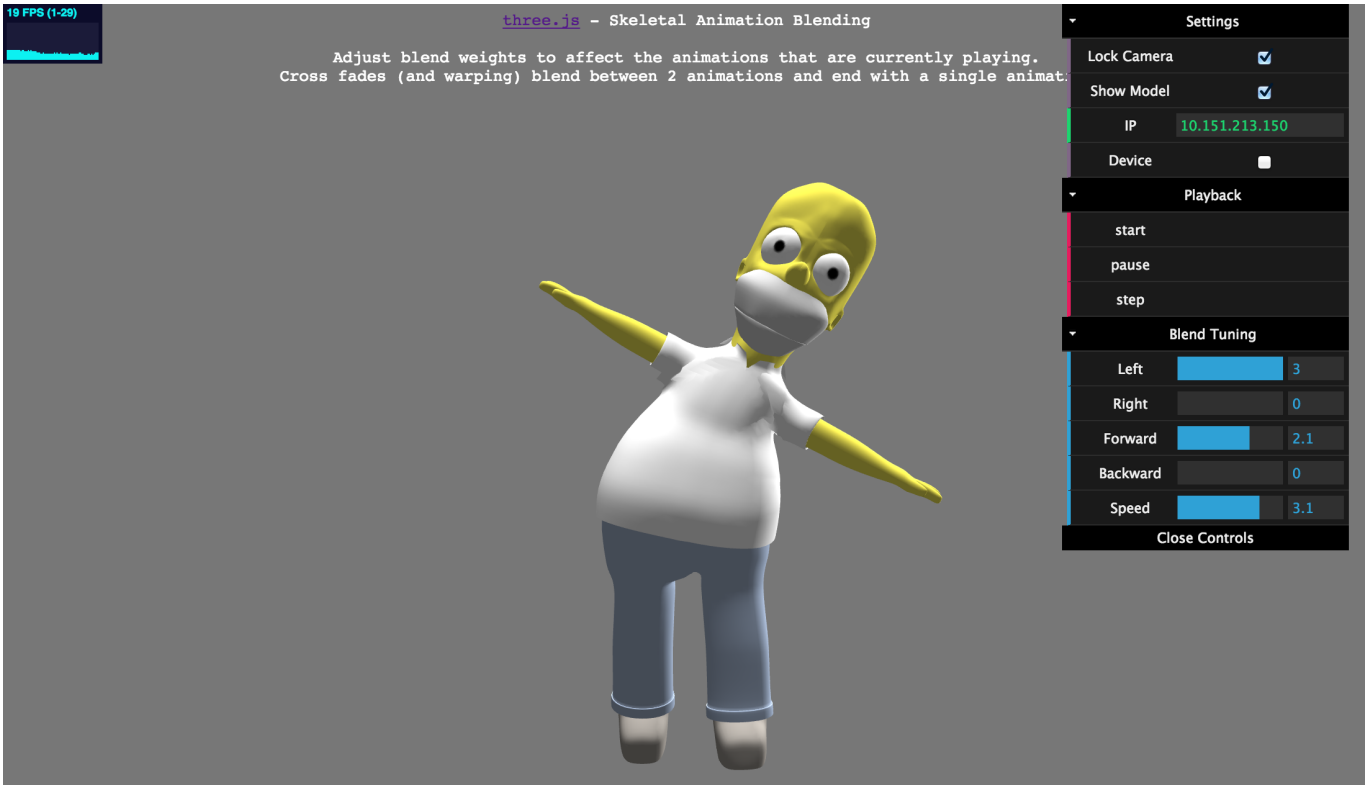

Figure 5. Final result

The animations are merged together accordingly to the weigths previously assigned.
The following code represent the process of starting an animation setting weights to each component.

Listing 2. Animations blending

```
function onStartAnimation( event ) {

    var data = event.detail;
    blendMesh.stopAll();

   for(var i = 0; i < data.anims.length; ++i) {
      blendMesh.play(data.anims[i],
          data.weights[i]);
   }

   isFrameStepping = false;
}
```

Mobile Controller programming

In order to provide the user with some basic controls to command the animation evolution, a simple Android application has been developed. This application basically consists in a HTTP Server that has the role of respond to HTTP REST calls invoked from a dedicated module inside the three.js program. The application sends accelerations values of the three axis of the device accelerometer every 10ms. These values are filtered by the responsible module of the program. Then they are mapped in ranges which are compatible with the weights of the animations and finally are communicated to the main program which updates the animation.

All this process results in a very simple but intuitive behavior: the user is able to command the animation by simply moving his Android device. In fact by tilting the device forward or backward he can control the forward/backward walk and by tilting it right or left he can control the side movements of the character.

POSSIBLE FUTURE WORKS

This implementation is only intended to provide a basic feedback for the proposed method. Many works can be further conducted on this demo in order to improve the accuracy or to show different capabilities of it.
In order to augment the accuracy of the mapping between the controller device and the character movements, some further filtering can be implemented. A mobile mean on the last values acquired from the accelerometer is straightforward to be added and can provide a really cleaner effect.
In order to show the real applications of this method, a little video-game with some basic physics can be implemented. An example could be a mini-game in which the player's goal is to make the character cross a cliff by walking on a rope. Some animations like the fall of the character can be easily added.

CONCLUSIONS

The proposed method aims to be inserted in the Human Computer Interaction field by providing Real-Time rendered 3D scenes which the user can interact with. This approach may result to be very useful in many various contexts like home entertainment, education or exhibitions. Thanks to its versatility it can be installed on almost every browser-enabled

platform and, by exploiting the power of common devices like smartphones, it can be used by everyone who can install a simple application. For specific purposes, ad-hoc hardware controllers can be built quite easily.
Many other different scenarios may be simulations of particular situations by means of actors that interact in real-time with the 3D scene or also the development of innovative Human Input Devices.

REFERENCES

1. Catmull E., *A System for Computer Generated Movies*. University of Utah, 1972.
2. Lasseter J., *Principles of Traditional Animation Applied to 3D Computer Animation*. Pixar, San Rafael, California, 1987.
3. Witkin A. and Popovic Z., *Motion Warping*. Carnegie Mellon University, Pittsburgh, 1995.
4. Popovic J., Seitz S. M., Erdmann M., *Motion Sketching for Control of Rigid-Body Simulations*. Massachusetts Institute of Technology, 2003.
5. Thorne M., Burke D., van de Panne M., *Motion Doodles: An Interface for Sketching Character Motion*. University of British Columbia, 2004.
6. Bruderlin A. and Calvert T. W., *Goal-Directed, Dynamic Animation of Human Walking*. Simon Fraser University, Burnaby, British Columbia, Canada, 1989.