11.4.2011

TEAM 1 INTERIM REPORT – MORPH MADNESS

Game Programming Lab 2011
Stefan Wenger, Matthias Hofstetter, Lars Schnyder, David Meier

Index

Game Instructions

The core gameplay is implemented. Up to 4 players can accelerate/decelerate their sphere with the Buttons A/B. The steering can either be done with the left thumbstick or the digital pad (both X-axis). The player is also able to grow and shrink his vehicle with the right and left trigger buttons. The jumping is set to the button X.

Changing of the material and shooting is not yet implemented.

Design revisions

Sound

We decided to move the implementation of basic sounds to the desirable target, because it doesn't make much sense to distinguish between basic and advanced sound, as we had planned previously.

Challenges

Camera

For a good game play it seems to be essential that the movements of the camera are handled in a proper and stable way. To do this we defined a local coordinate system in our sphere, where the first vector represents the velocity. This is the direction in which the sphere moves. The second vector is given by the normal of the spheres collision with the floor. This coordinate system allows defining the desired position of the camera with respect to the spheres movement.

This works very well for flat scenes. But whenever there is a bumpy floor the velocity changes very fast in y direction. To overcome this problem it seems to be essential that there is some kind of intelligent path planning. That is we have to collect the calculated desired positions for our camera and then try to find and estimate the proper position of the camera without any fast changes of the positions.

Actually we investigate some methods include regression and motion planning to calculate these positions. But as it seems to be a rather complex task there is actually no usable solution.

Physically-based environment

Due to the fact that all the objects in the world have to be synchronized with the models in the physics engine a lot of work went into fusing the graphical and physical appearance. To get the physics engine running on the Xbox we have to represent all objects in the physics system as simple geometric primitives like cylinders, spheres or boxes. It would be much easier to model everything as triangle meshes but collisions with this kind of objects cause the physics system to crash if the game is running on the Xbox.

Up to now we didn't have to change/hack the physics system to enable important features of our game. Since it is our intention to use a consistent world we think this is good and we want to keep it that way.

In addition we didn't encounter any computation problems with the system but are planning to multithread physics in case there are some issues.

Lava

We have planned to make physically-based animated lava. But we don't know yet how to accomplish this in a physically correct way. Just adding more friction or stiffness to a water simulation doesn't give nice results. So probably we stick to an animation without correct physics for this particular case.

Balancing

We did not have much time to set up a good balancing yet. Before we focus on this task we have planned to implement all game elements which have impact on the balancing (size, materials…). But now after that (in few days) we will put strong efforts on in a good balancing because it it's the most important part of our gameplay.

Modeling/Texturing

For the modeled spheres (3D Studio Max) we were able to use an additional content pipeline which includes texturing and normal mapping. This is currently the only part in the game which has been created with modeling software.

Next to the sky-box which is implemented straight-forward, the world is created with height map which is textured with a multi-texture terrain shader as proposed in the game class. For this part we're currently also implementing normal mapping.

Project Status

Functional Minimum

Functional Minimum	Status	Comments
Player can move sphere (all directions and jumping)	100%	
Player can grow and shrink sphere	100%	
Sphere can catch and move flag	100%	
Player can bump into opponent spheres to destroy them	100%	
Basic map (Non textured Platforms with simple connections)	100%	
Respawn of destroyed sphere	100%	

Low Target

Low Target	Status	Comments
Player with flag is slower than others	0%	Has to be done in the balancing step
Big sphere is slower than a smaller one	80%	Some balancing has to be done
Physical behavior with material properties (Wood, Metal…)	60%	
Basic sounds	0%	Rescheduled to desirable target
Basic HUD (Points, Time, …)	100%	
Basic shading and lighting	100%	
Random placement of the flag	100%	
Enhanced arena (low physical effects)	100%	

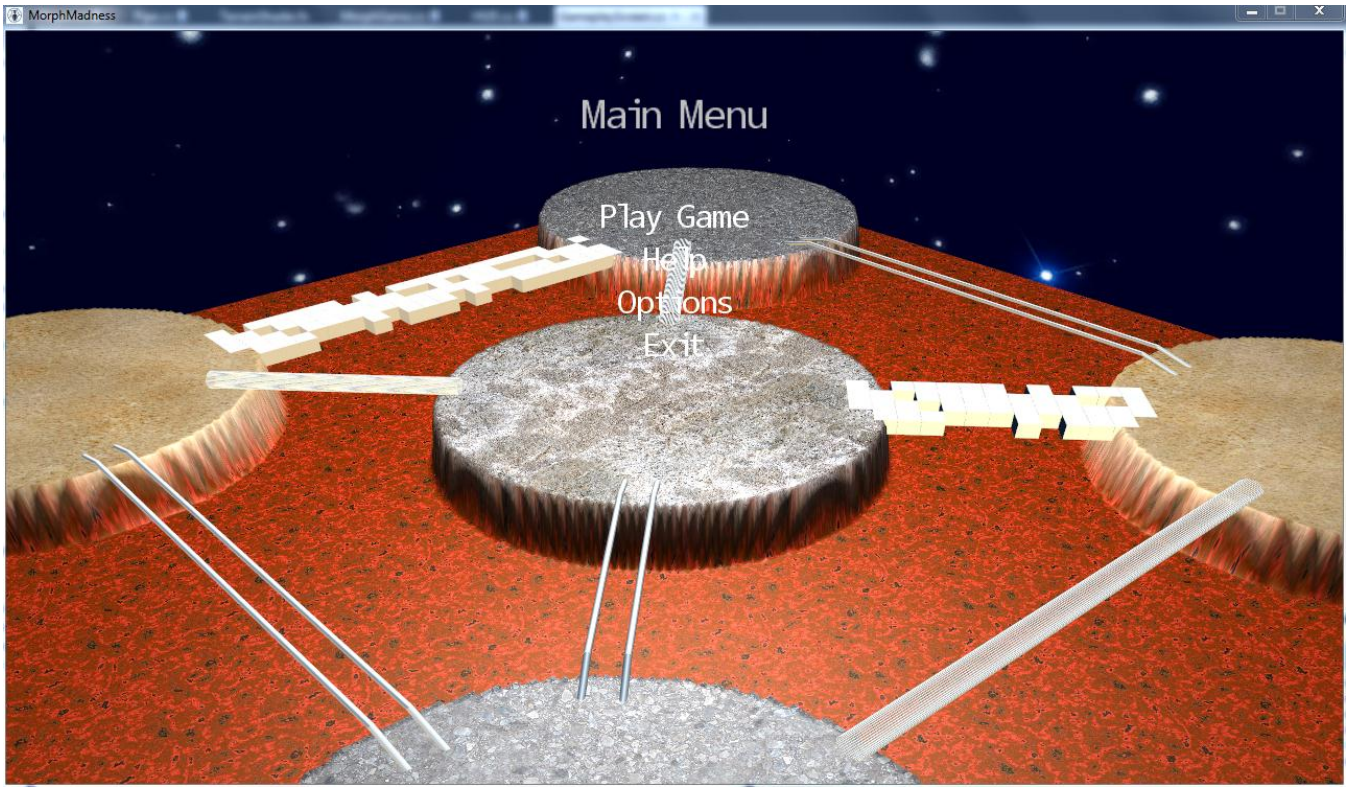
Desirable Target

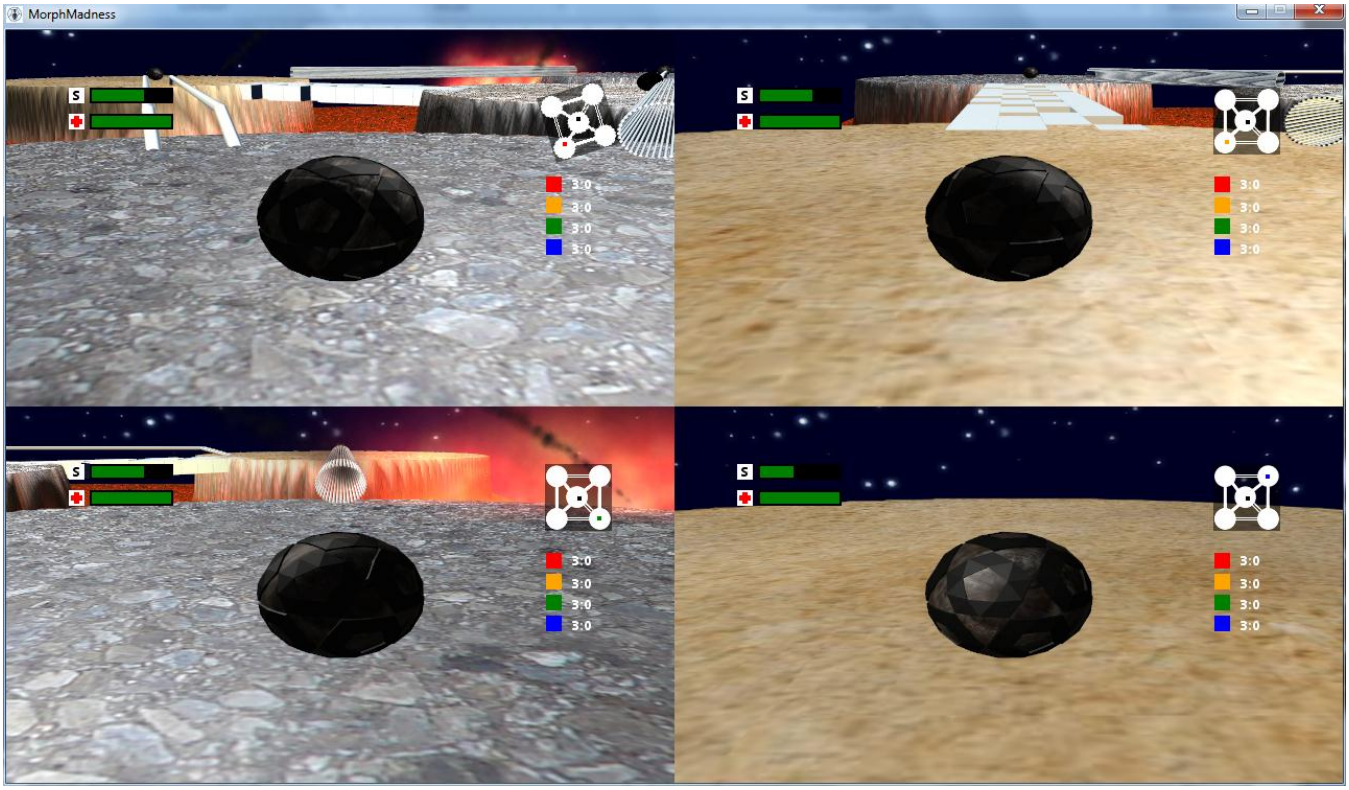
Desirable Target	Status	Comments
Final HUD	70%	
Game menu, GUI	80%	
Texturing of Spheres and Scene	70%	
Advanced Shading and Lighting	70%	
Advanced sounds	0%	
Enhanced map with physically based water, lava…	0%	
Damaged spheres are moving and transforming slower (different meshes for different degree of destruction)	0%	

High Target

High Target	Status	Comments
Marketplace for upgrades and updates	0%	
Achievement system (Money collection)	0%	
Weapons and effects	0%	
Real-time deformed meshes for damages spheres (mesh reconstruction)	0%	

First Impressions




MorphMadness
3:0
3:0
3:0
3:0