Prof. G. Zachmann
C. Schröder (schroeder.c@cs.uni-bremen.de)

University of Bremen
School of Computer Science
CGVR Group
January 20, 2016

Winter Semester 2015/16

Assignment on Virtual Reality and Physically-Based Simulation - Sheet 6

Due Date 3. 2. 2016

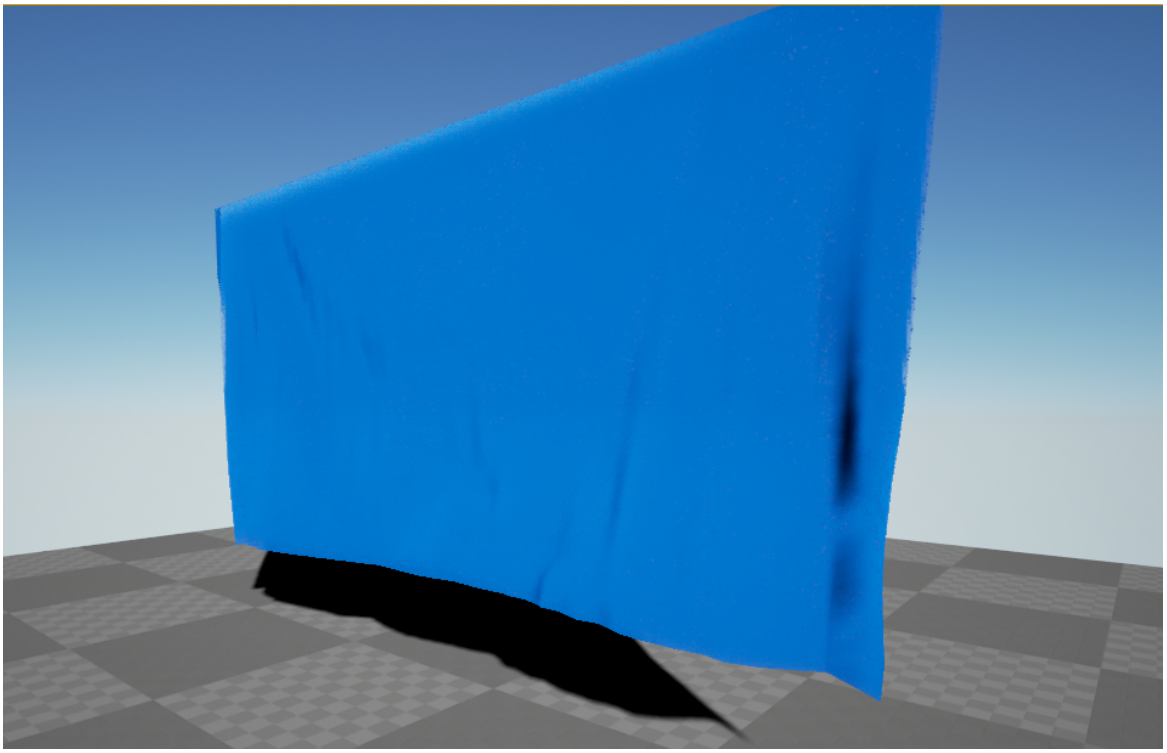

Figure 1: Final mass-spring system.

Exercise 1 (Mass-Spring-Systems, *8 Credits*)

The goal of this exercise is to implement a mass-spring system. On our website you can find an Unreal project with most parts already implemented.[1] When you recall the definition of a mass-spring system from the lecture slides, you will recognize the two main components in the Spring.{h,cpp} and MassPoint.{h,cpp} files. The SpringMassActor.cpp glues the system together. It initializes the mass points and springs, calls the update methods, and further creates a mesh to visualize our system (`ASpringMassActor::initSpringSystem`).

[1] `http://cgvr.cs.uni-bremen.de/teaching/vr_1516/uebungen/vr06_spring_mass.zip`

In the provided level, you can press `F` to apply a force to the center of the mesh. The logic behind it is implemented in the `ASpringMassActor::Touch` function. By pressing the keys `0` and `9`, you can switch the rendering between wire-frame and lit mode.

To generate the Visual Studio project files, do a right-click on the spring_mass.uproject file and select *Generate visual studio project files*. Afterwards open the spring_mass.sln file and start the unreal editor from Visual Studio. When you changed code, save the file and click on the compile button in the unreal editor. A notification in the bottom right corner informs you about whether the code was reloaded successfully. This way you do not have to restart the editor for each change an you can use the debugger from within Visual Studio. I am confident that the process on a MAC is similar. If there are problems, please feel free to contact me for further assistance.

a) Implement the force calculation for each spring and add it to the mass points (`Spring::Tick`). You can access the members of the connected mass points (`m_m1`, `m_m2`) directly, as `class Spring` is a friend of `class MassPoint`.

b) Add a gravitational force to each mass point in `MassPoint::updateGravity`.

c) Implement a perturbed gravitation vector[2] instead of a constant one to each mass point.

d) Change the integration method in `MassPoint::updateCurPos` to use the approximate midpoint method from the particle system slides (p 14).

Exercise 2 (Bonus: System stability, *2 Credits*)

Explain why the system explodes for large damping factors.

[2] You can add a small, random vector offset to the gravitation vector.