Test task for recruitment of a developer

This task should give you a small insight into the kind of work we do. Furthermore we want to see how you solve the given programming task. For assessment of your solution we focus not only on the correctness/robustness of your program(40%) but also on aspects that are important for development of successful software such as: coding style and maintainability(20%), used algorithms and performance(20%), appropriate use of language features and standard library(10%) and documentation(10%).

Task

A 3d Euclidian point cloud aligned in main axis directions (x, y, z) and with a constant distance grid (Figure 1) starting at a given reference point (the point with indices 0, 0, 0 is located at reference point) is intersected by a move of sphere (Figure 2), where the path of the sphere center is defined by a user given formula $\vec{x} = f(t)$ where t is in the interval between t_0 and t_1. The function $f(t)$ can be handled as a discrete function with a user given Δt. Points that intersect with the sphere move are considered as deleted (Figure 3 middle).

- Only the first layer of points (which remains *visible/undeleted*) from top view must be written to a file as ASCII data (the skin of the point cloud from top view, see Figure 3 right). The file format is defined as follows:
  - Each line contains a single point.
  - The point definition contains x, y and z coordinates delimited by space characters.
  - Each line ends with a new line character.
- Create a small documentation(1 page with 2 pictures) to present the mathematical approach of the sphere move point intersection. It should clearly communicate the mathematical approach and how the *mathematical* code is generated from that.
- Please discuss briefly in 4-5 sentences what problems can arise by using a discrete stepping Δt.

Given

- Point class for the definition of a point in 3d and (some) methods for vector algebra
- Visual Studio 2008 and 2013 solutions, Makefile and qmake project file
- High level test function *CreateSkin(. . .)*(in CreateSkin.cpp) to test the resulting component. This defines also the *interface* of the component to be written.
- Program named *PointVisualizer* which allows you to view the results you obtain in 3d.

Hints

- Use of vector algebra (dot product, cross product etc.) is highly recommended. The use of sin and cos functions is not desired.
- Calculation speed and memory footprint are important but secondary in comparison to clearly readable code. E.g. a sphere, linear move of a sphere, point writer etc. might be modeled as classes. Prefer compact classes and functions over large classes and functions.
- Comments in the source code are welcome. Please use *good* names for classes and functions.
- Move of a sphere between $f(t)$ and $f(t + \Delta t)$ can be assumed as a linear move of the center of the sphere.

Illustrations

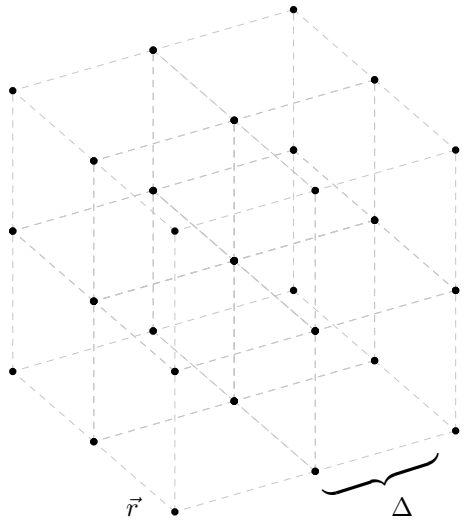

Figure 1: 3d point cloud defined by reference point $\vec{r}$ and point grid distance Δ.

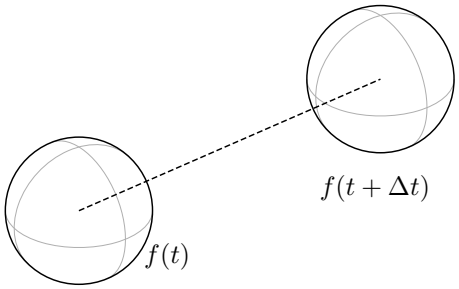

Figure 2: Linear move of a sphere with start point $f(t)$ and end point $f(t + \Delta t)$.

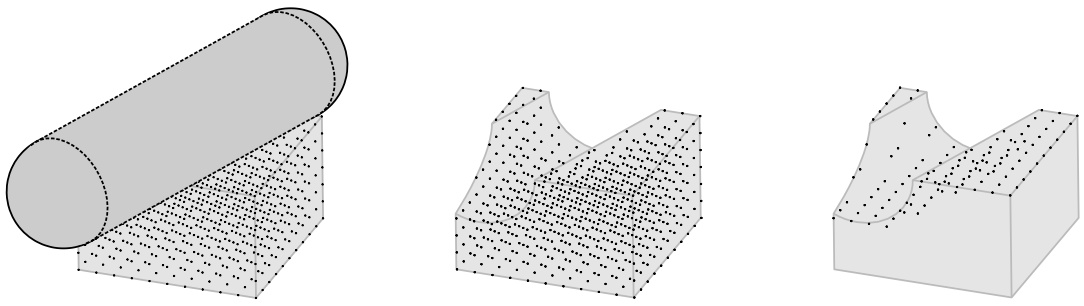

Figure 3: Left to right: complete point cloud, without deleted points, top skin of the point cloud.