

Übungen zu Grafikprogrammierung in C++

Abgabetermin: 03. Juli 2006

In dieser Aufgabe geht es darum, ein Aquarium zu implementieren. Dieses Aquarium kann ganz einfach durch einen blauen Hintergrund dargestellt werden. In dem Aquarium befinden sich Fische, die sich in einem Schwarm fortbewegen und dabei Luftblasen ausstoßen, die langsam nach oben steigen. Im Rahmen dieser Aufgabe werden Sie folgende Verfahren kennen lernen und umsetzen:

- Partikelsysteme

Aufgabe 22 (H) Partikelsysteme - Fische und Luftblasen

Mit Partikelsystemen kann man Bewegungen bzw. Krafteinwirkungen auf Objekte einer Szene modellieren. Beispielsweise könnte man Regen, Schneetreiben oder Bewegungen von Schilf im Wind visualisieren. In dieser Übung soll ein Fischschwarm mittels Partikelsystemen modelliert werden. Dabei soll der Schwarm im Aquarium einer fest vorgegebenen Wegstrecke folgen und die einzelnen Fische sich gleichzeitig relativ zueinander bewegen, ohne dass es den Schwarm auseinander treibt. Um die Aufgabe ein wenig zu erleichtern, wird ein Programmierrahmen zur Verfügung gestellt. Dieser beinhaltet bereits das Laden der Fischobjekte, so dass nur noch die Methode `draw()` aufgerufen werden muss, um einen Fisch um den Mittelpunkt des Koordinatensystems zu zeichnen. Zusätzlich werden auch schon die zu schwimmende Wegstrecke und die dazugehörige Interpolation vorgegeben. Der gezippte Code befindet sich auf der Webseite. Die Unterordner sind in `bin` (alle ausführbaren Dateien), `include` (alle header Dateien), `src` (alle `.cpp` Dateien) und `vs` (alle IDE Dateien) aufgeteilt. Alle neu zu erstellenden Klassen sollen in das bestehende Framework eingebunden werden. Entpacken Sie die Ordnerhierarchie in `TeamX/Blatt9/Augabe22`.

- Design:** Entwerfen Sie eine Klassenhierarchie mit Oberklasse `Particle` und zwei Unterklassen `FishParticle` und `BubbleParticle`. In `Particle` sollen alle Attribute und Methoden implementiert werden, die Partikel gemeinsam haben, z.B. Geschwindigkeit eines Partikels, Ausrichtung, Position, etc. Es soll zwei abstrakte Methoden geben, `render()` und `update()`, die in den Unterklassen implementiert werden sollen.
- Rendering:** Für die Zeichnung der Fische ist abgesehen von der Ausrichtung in der Szene nur noch die Methode `draw()` des Fischobjektes `m_obj_fish` in der Klasse `Aquarium` aufzurufen. Texture Mapping und Normalenbestimmung sind darin ebenfalls schon implementiert.

Die Darstellung der Bubbles (Luftblasen) kann durch die in Glut vorgefertigte 3D Struktur `gluSphere` in Verbindung mit Alpha Blending realisiert werden. Alpha Blending bedeutet,

dass dem Material der Bubbles zusätzlich zu den drei Farbkomponenten RGB noch ein vierter Wert, der Alpha Wert, zugeordnet wird. Ein Alpha Wert von 1 steht hierbei für ein komplett opakes Objekt, ein Alpha Wert von 0 für ein komplett transparentes Objekt. Mit anderen Worten, implementieren Sie die Bubbles so, dass sie teilweise transparent sind und eventuelle Objekte im Hintergrund durch die Bubbles hindurch zu sehen sind. Verwenden Sie dazu die OpenGL-Funktionen `glBlendFunc(GLenum sfactor, GLenum dfactor)` bzw. `glEnable(GL_BLEND)`.

Bitte beachten Sie, dass sowohl bei der Zeichnung der `FishParticle`, als auch bei der Zeichnung der `BubbleParticlen Display Lists` verwendet werden sollen, um eine gute Performance zu gewährleisten.

- c) **Kräfte:** Die Wegstrecke, die der Fischschwarm schwimmen soll (entspricht einer 8) ist bereits im Programmierrahmen festgelegt und in dem Vektor `m_path` in der Klasse `Aquarium` abgespeichert. Die Einträge in diesem Vektor entsprechen Winkeln und geben die Richtung an, in die ein Fisch schauen soll. Zwischen den Winkeln wird in Abhängigkeit der Zeit interpoliert, so dass es zu einer fließenden Drehbewegung des Fisches kommt. Die eigentliche Bewegung soll nun nach dem Prinzip "Ich schwimme dorthin, wohin ich schaue" realisiert werden. Um die aktuelle Richtung zu erhalten, muss in der Klasse `FishParticle` eine Instanz der Klasse `PathInterpolator` erzeugt werden, diesem der vorgegebene Pfad `m_path` übergeben werden und schließlich in der Methode `update()` regelmäßig die Methode `interpolate()` des `PathInterpolator` aufgerufen werden. Als Rückgabewert erhält man dann die aktuelle Blickrichtung des Fisches. Um die Geschwindigkeit anzupassen kann die Interpolationsgeschwindigkeit mit der Methode `setInterpolationSpeed(float)` verändert werden.

Als zusätzliche Kraft soll die Bewegung der einzelnen Fische relative zueinander implementiert werden. Dies kann beispielsweise über das Federgesetz realisiert werden. Ziel ist es, die Bewegungen so zu gestalten, dass der Schwarm niemals auseinander schwimmt, einzelne Fische jedoch auch niemals zusammenstoßen.

Die Geschwindigkeit, mit der die Luftblasen aufsteigen sollen, soll abhängig von ihrem Umfang sein. D.h. je größer die Luftblase, desto schneller steigt sie nach oben.

- d) Die Klasse `Aquarium` soll als Hauptklasse dienen und sowohl die `main()` Methode als auch die Instanzen der einzelnen Partikel beinhalten. Grundlegende Dinge wie die Zuordnung der `reshape()`, `update()` und `draw()` Funktionen sind bereits im Programmierrahmen realisiert. Ihre Aufgabe ist es nun diese Klasse zu vervollständigen und alle benötigten Komponenten für Beleuchtungsmodelle, Partikelsysteme, User-Interaktion, etc. hinzuzufügen.