Realistic Water Simulation

Team Wavy

School of Data and Computer Science(SYSU)

July 3, 2018

Content

HOW to simulate?

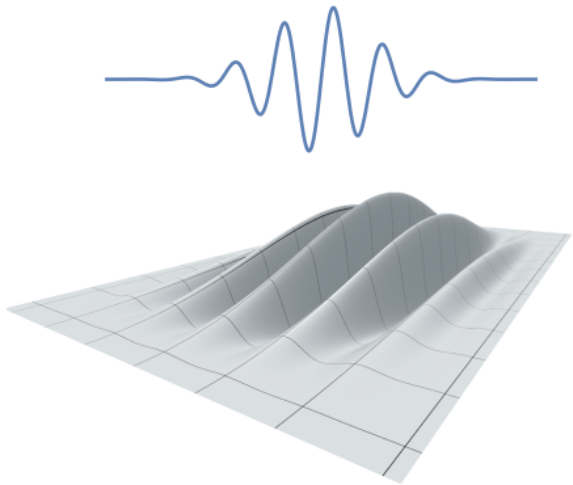
Quite difficult to deal with the **volatile** material...

- Texture-based method : minimize the calculation, wildly used in real time rendering
  - Blinn, 1978, Bump Mapping
- Construction-based method : more mathematicall
  - Cosine function superposition algorithm
  - Gerstner Wave
  - B-spline

HOW to simulate?

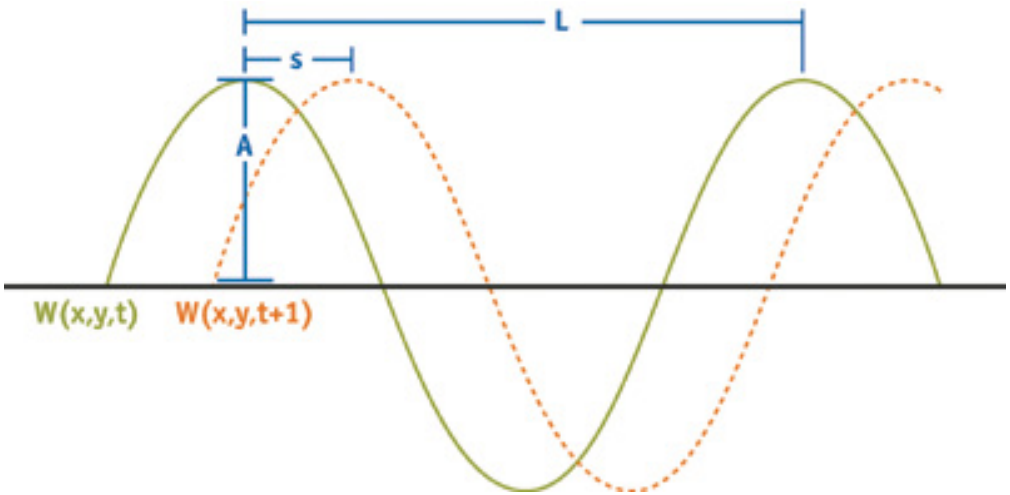
Quite difficult to deal with the **volatile** material...

- Based on physics models :
  Realistic and Lifelike(pick it!✓)



Introduction to Gerstner Wave

- Wavelength (L)
- Amplitude (A)
- Speed (S)
- Direction (D)



Introduction to Gerstner Wave

对于单个波而言，其函数表达式如下所示。

- A 振幅
- D 二维方向向量
- w 角速度
- φ 其值为 $S \times \frac{2}{L}$

$$W_i(x, y, t) = A_i \times \sin(D_i \cdot (x, y) \times w_i + t \times \varphi_i)$$

Introduction to Gerstner Wave

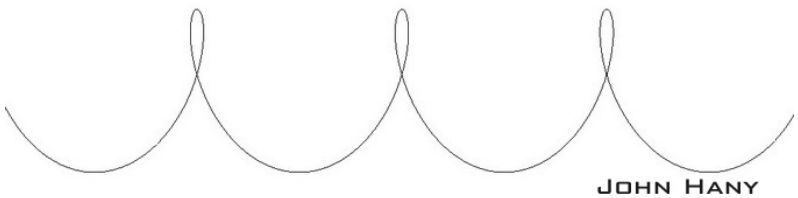
完整的 Gerstner Wave 函数如下所示。其为向量值函数。

- 输入
  网格化的坐标值
- 输入
  三维坐标系中的坐标 (x, y, z)

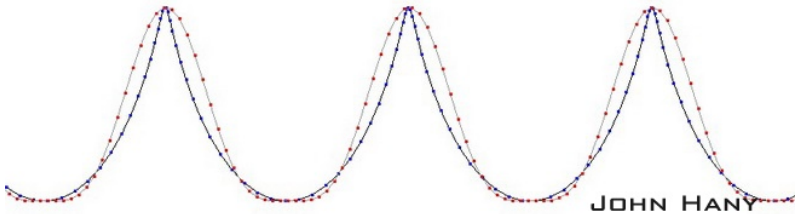
$$P(x,y,t) = \begin{pmatrix} x + \sum(Q_i A_i \times D_i[x] \times \cos(w_i D_i \cdot (x,y) + \varphi_i t)) \\ y + \sum(Q_i A_i \times D_i[y] \times \cos(w_i D_i \cdot (x,y) + \varphi_i t)) \\ \sum(A_i \sin(w_i D_i \cdot (x,y) + \varphi_i t)) \end{pmatrix}$$

参数特点

- Q 较大
  波峰尖锐，类似于真实水面
- Q 过大
  产生环，破坏波的形状
- Q 为 0
  退化为简单三角函数的叠加



The first film



The second film

Advantages

- 波峰尖锐，波谷平缓
- 允许多波产生叠加效果, 交互感真实
- 参数众多，可调节性强
- 性能较高，开销较小

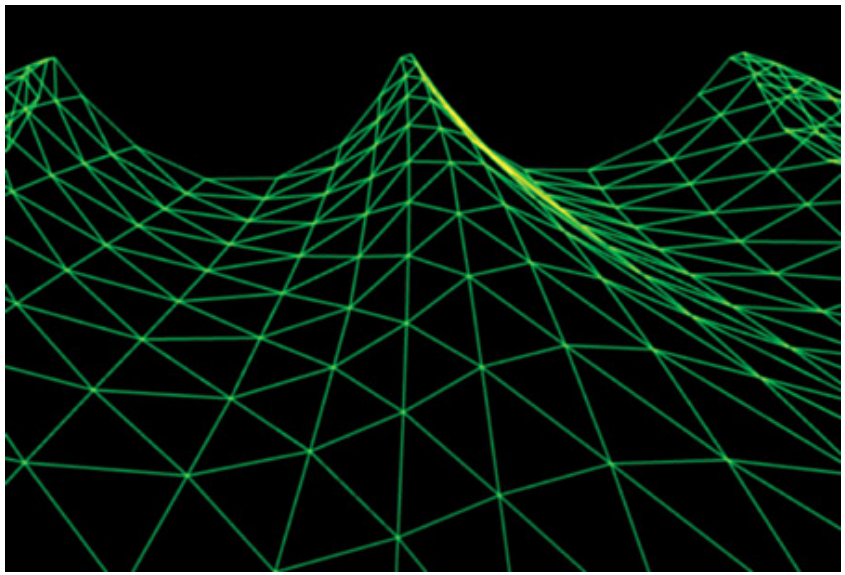


Figure: Gerstner Wave

Disadvantaged

- 水面永动，水面远处永远“自发地”泛起波纹
- 波峰低的波会产生大面积规律褶皱
- 无法仿真不同向的波间的碰撞和能量损耗。当波间相差的角度较大时，水面效果较差
- 整体上看，规律性过强

Refine

- 波随机过滤
  对小波采取随机化地去除。
- 动态修改参数
  动态地略微地修改各个波的方向、振幅等参数。

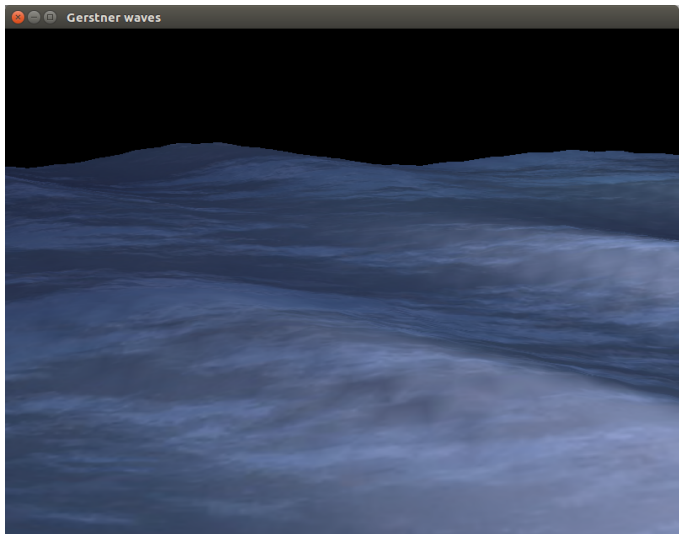


Figure: 程序运行图片

Three ways to implementation

water wave packet 论文中提到了三种方法用来实现水面

- 无限的波列 (infinitely long wavetrains)
- 单波峰 (single wave crests)
- 波包 (packets of waves)

我们自己根据 Gerstner 波的相关知识，使用无限的波列渲染了简单的水面。
后来，论文的思想，进一步启发了我们使用单波峰完善了水面的渲染。
最后尝试了波包在水面渲染上的应用。

实现效果

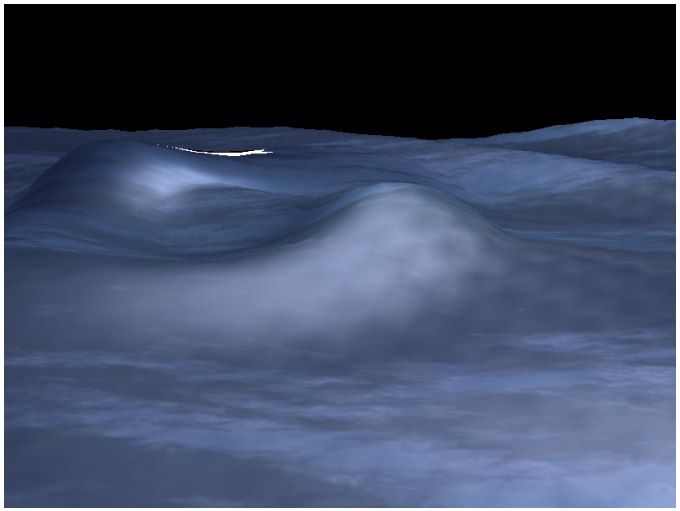


Figure: 单个波浪

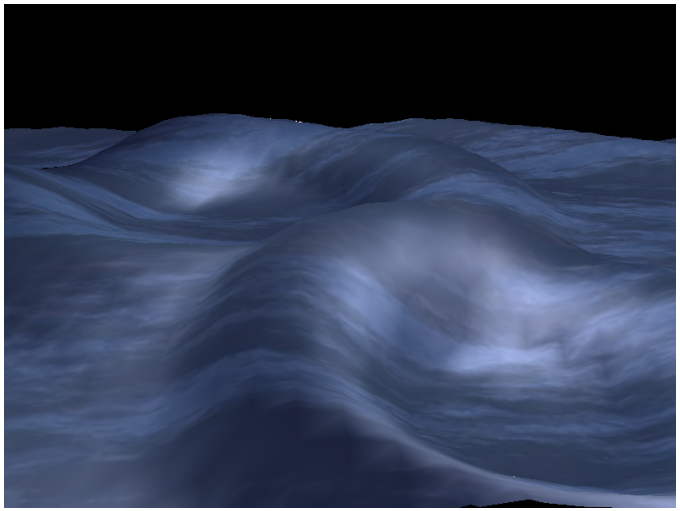


Figure: 多个波浪

我们使用单波峰实现了对水面扰动的仿真，具体实现特性有以下几点：

- 按一次键盘-> 扰动-> 波浪
- 扰动强弱的控制
- 水面扰动能量的衰减
- 多个扰动间的叠加

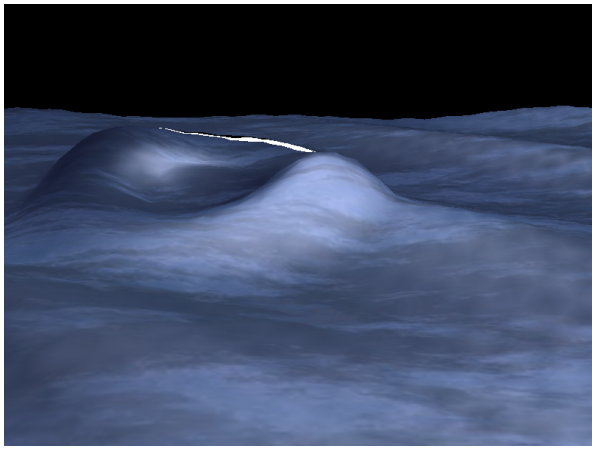


Figure: 单个扰动刚出现

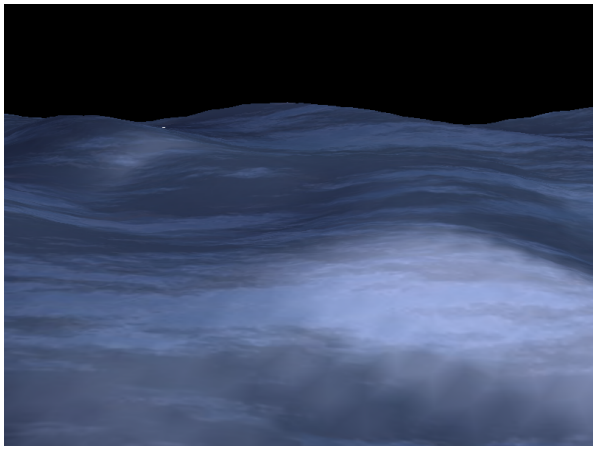


Figure: 扰动接近消失

大致实现思路

将波独立成波峰或者波包来实现

- 对波的控制更为细致
- 易于实现能量衰减
- 不同波之间的相互影响有着成熟的理论体系

使用波包来对水面进行模拟，能够基于物理知识，渲染出更加真实且效果更佳丰富的水面。
在进一步的实现中，我们仅使用了单波峰来完善我们的水面的渲染。

具体实现思路

- 实现了两个类
  - Packet 类
  - PacketManager 类

Packet 类

- 存有该波峰的能量，波长，振幅，起始坐标等
- 存放于 PacketManager 类中的 vector 容器中
- 用户不直接使用 Packet

PacketManager 类

- 管理该水面具有的 Packet
- 增加新的 Packet
- 遍历 vector 容器，更新当前水面信息
- 检测到波峰能量为 0 时，将该波峰从容器中删除

波高信息的更新

在每一帧，程序的主渲染循环都会调用 PacketManager 的 `update_data` 接口。
正是这关键的操作，实现了波峰的移动及消失

```
delay = glfwGetTime() - start_time;      You, a few seconds ago • Uncommitted
last_time = glfwGetTime();
double max_delay = 5;
double cur_r = 0.3 - 0.3 * decrease_function(delay,max_delay);
double e_fraction = decrease_function(delay, max_delay);
cur_energy = this→energy * e_fraction;
// 根据波的能量大小还有半径，计算当前波的形状
for (int i = 0; i < STRIP_COUNT; i++){
    for (int j = 0; j < STRIP_LENGTH; j++){
        float one_range = DIS_TO_COORDINATE * (cur_r);
        float packet_range = 2 * one_range;
        float d = sqrt(pow(i-x, 2) + pow(j-y, 2));
        point_height[i][j] = cur_energy * wave_packet_function(one_range, 3 *
        one_range, d, packet_range);
    }
}
```

Figure: 一处关键代码

不足与优点

在我们的实现中，我们认为有如下优点。

- 不使用波序列，而使用单个的波峰进行模拟
- 能量衰减实现效果自然

不过，仍有以下不足。

- 物理背景缺乏，较难实现与刚体碰撞后波的进一步反弹与传播
- 单波峰扰动效果一般，波峰过于圆润，不够真实

Demo

Now is the demo time!

Group member

李新锐 15323032　孙一言 16337216
王锦鹏 16337232　王永锋 16337237
颜彬 16337269　韦博耀 16337242

Thank you

Thank you for listening!