中国游戏开发者大会

CHINA GAME DEVELOPERS CONFERENCE

《战争雷霆》中的海洋模拟和渲染

Tim Tcheblokov, NVIDIA

战争雷霆

地面战和空战模拟游戏

坦克模式

- 拥有射击游戏一样的细节
- 可分辨特性精细到2-3厘米

战争雷霆

地面战和空战模拟游戏

被坦克摧毁，
轰炸机的反击!

- 空战模式
  拥有飞行模拟器的细节
- 视野范围高达160 公里
- 可分辨特性高达数公里

战争雷霆

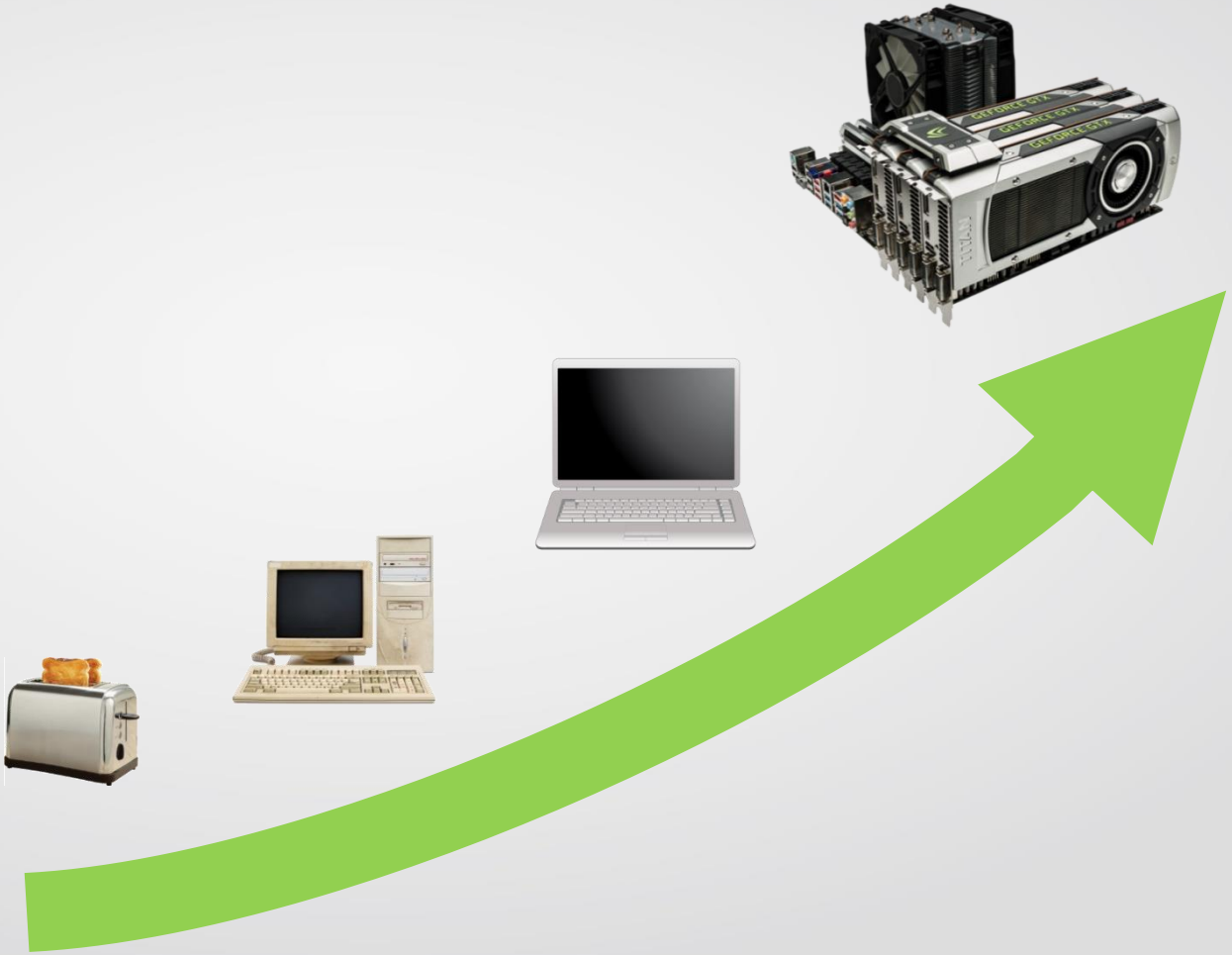
免费大型多人在线游戏

- 与付费游戏不同，玩家只有在喜欢这个游戏后才会消费
- 需要在任何客户端上流畅运行，并且效果出众

战争雷霆

可扩展性是关键

- DX11, DX9, GL, GLES, PS4
- Windows, Linux, Mobile, Mac及其它主机平台
- 从烤面包机到Titan显卡



战争雷霆

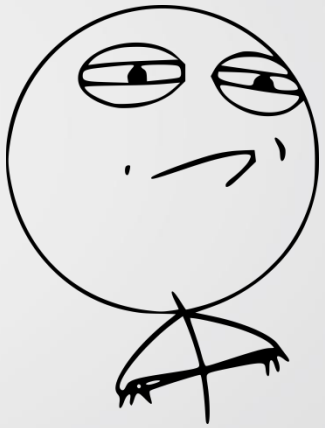
引擎永远都需要提升

- 新的效果
- 性能优化
- 水面技术进展缓慢
- 我们需要新的水面技术!

战争雷霆

新的水面技术所面临的挑战

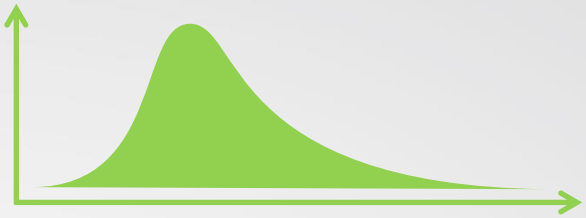
- 逼真的效果
- 爆炸，尾流，海岸，河流和湖泊
- 高性能的模拟和渲染
- 游戏内置全套物理模拟
- 可适用于每一位用户的硬件能力



接受挑战

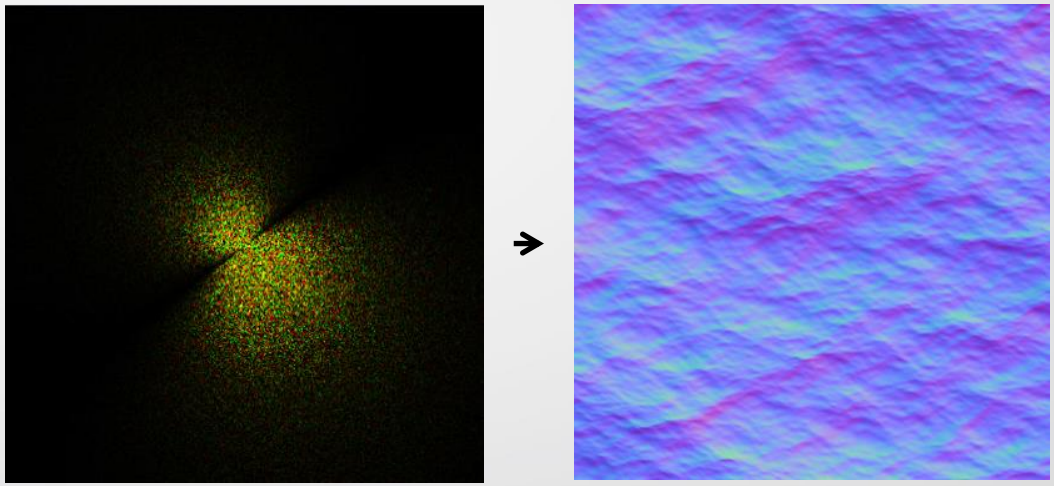
海洋模拟

经典方法



iFFT

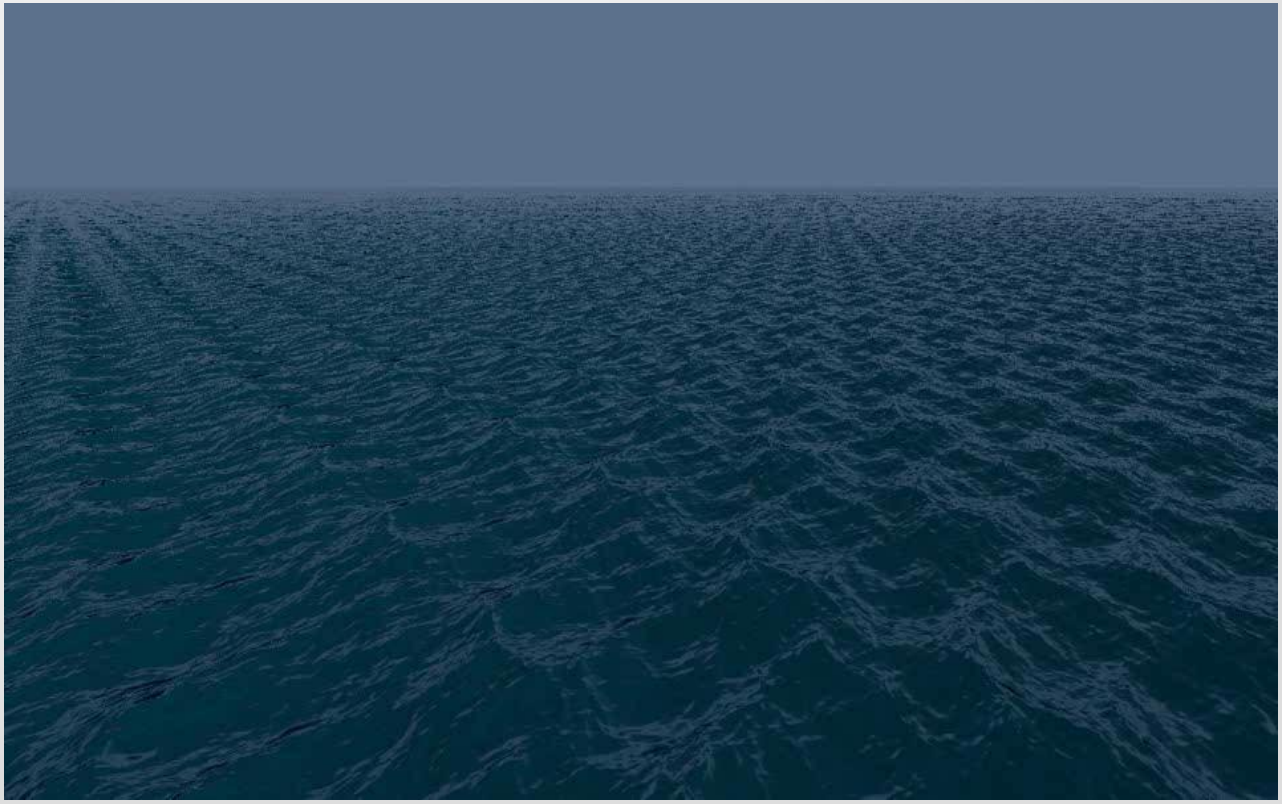
- Jerry Tessendorf的论文:
  Simulating Ocean Water
- 基于经验模型:
  Phillips 谱
- 在频域中进行模拟,
  通过FFT反向转换到空域
- 结果存储到纹理中
  R, G, B = Dx, Dy, Dz



海洋模拟

经典方法

- 近看较自然
- 远看会出现重复
- 可以增大FFT尺寸，但是
- 大尺寸FFT开销巨大

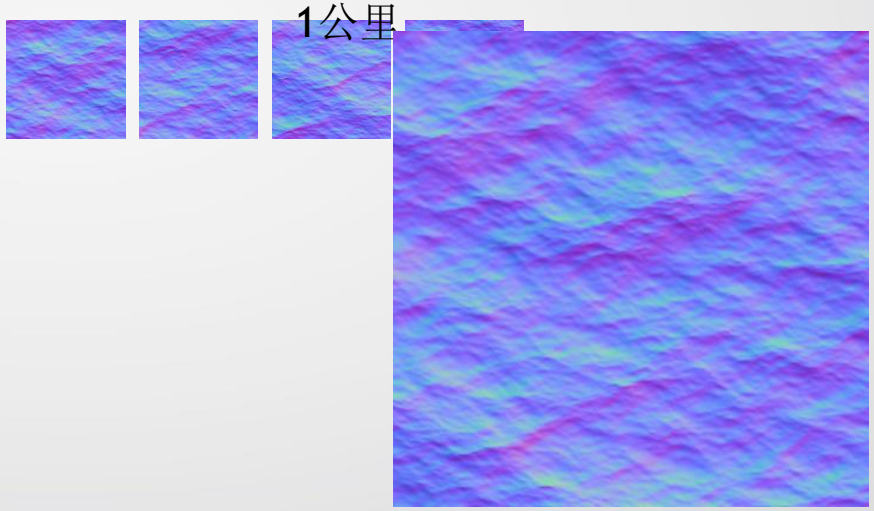


海洋模拟

基于多级叠加的方法

- 将谱分为4个层级
- 在频域中进行4次模拟，然后转换到空域
- 输出一组纹理
- 叠加各层级结果得到最终结果

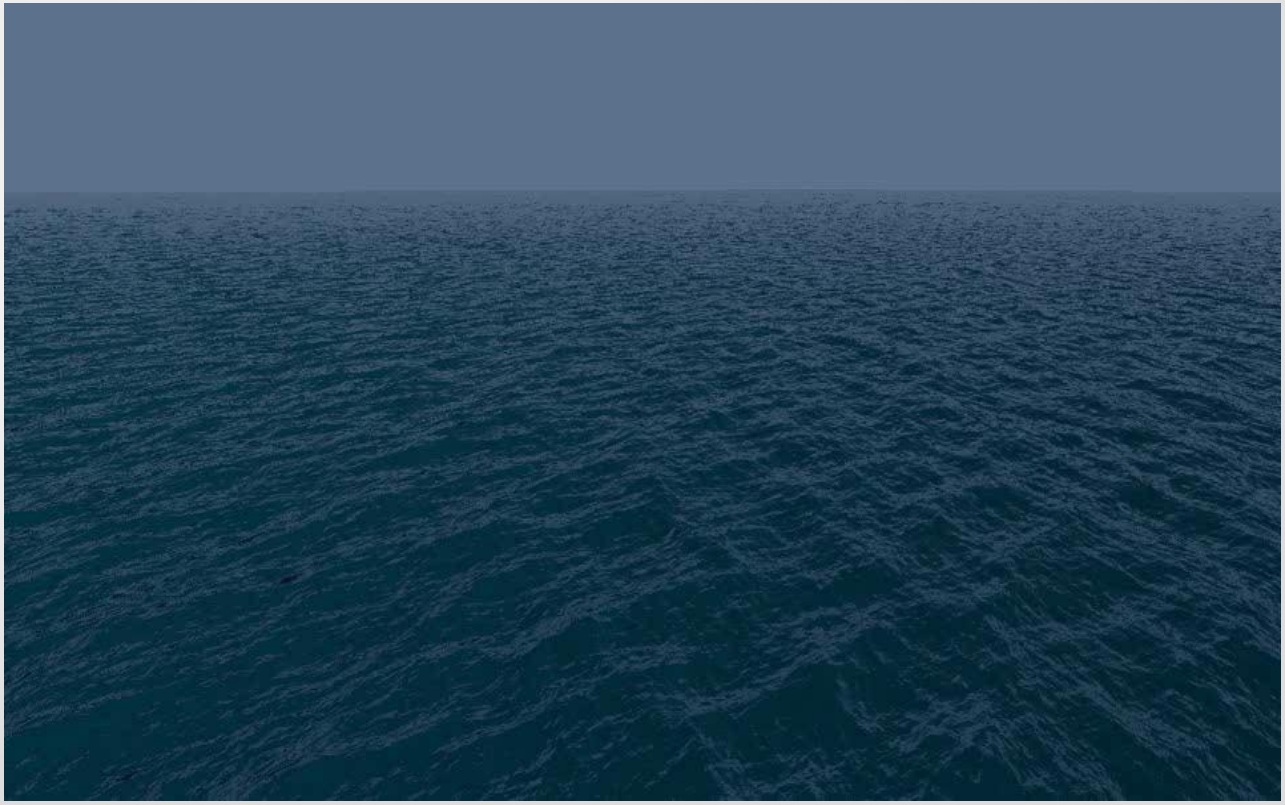
iFFT



海洋模拟

基于多级叠加的方法

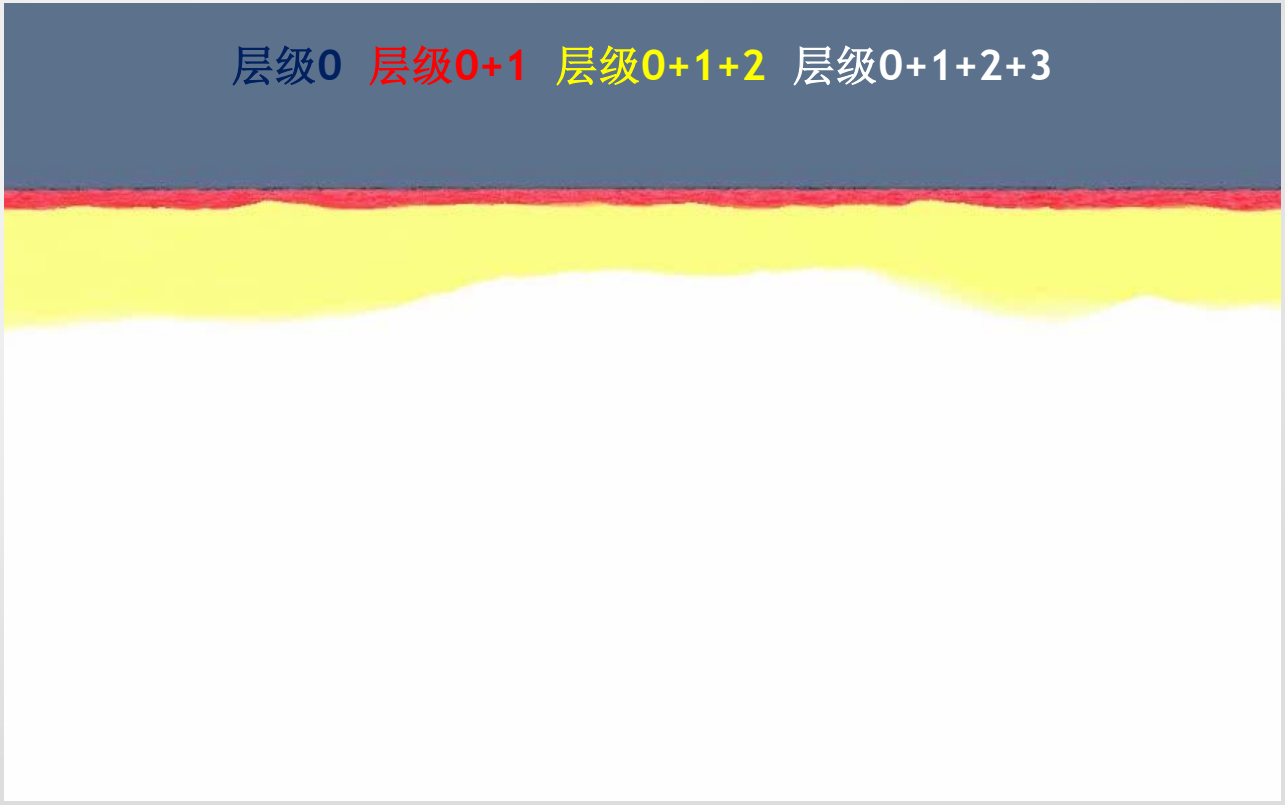
- 高动态范围的波长
- 近距离观察有更多细节
- 远距离观察时极少重复



海洋的模拟和渲染

基于多级叠加的方法

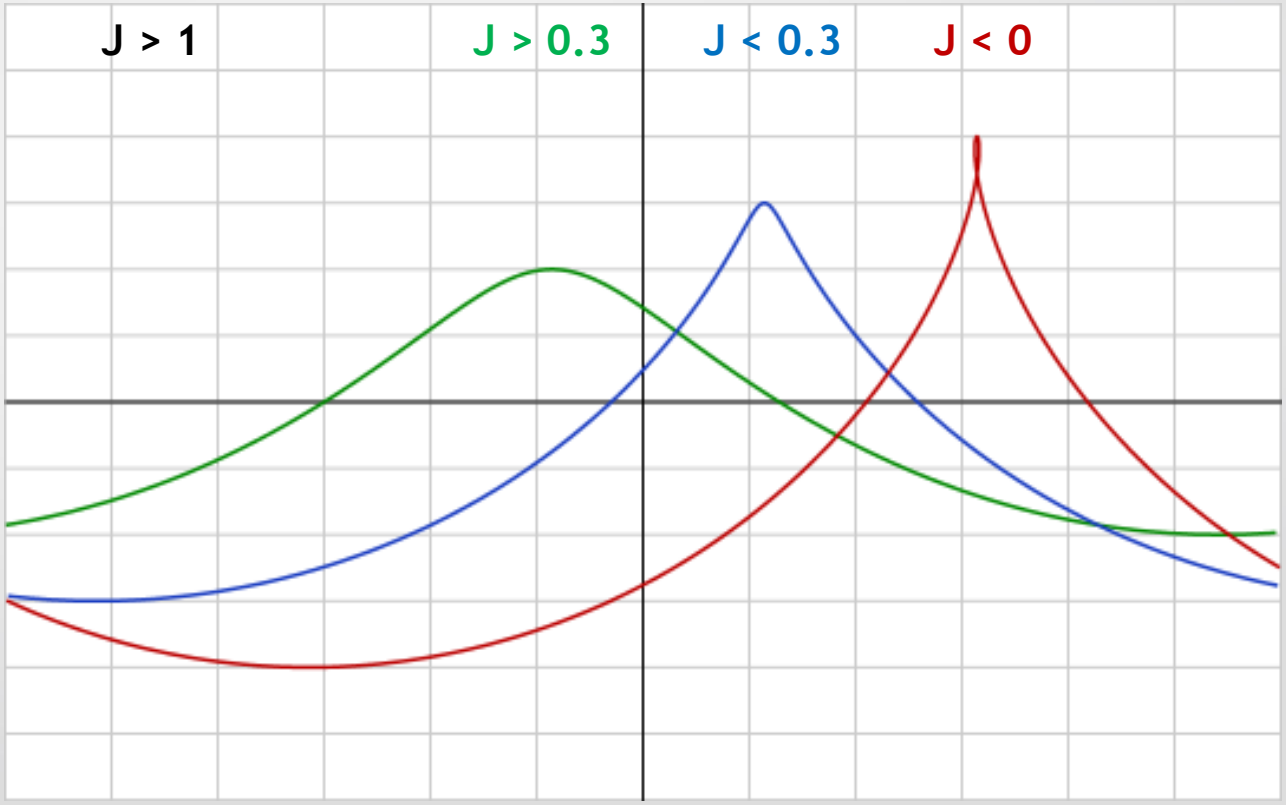
- 逐渐淡出。在远距离处逐渐排除层级叠加
- 噪点更少
- GPU的计算量更少



海洋的模拟和渲染

基本泡沫模拟

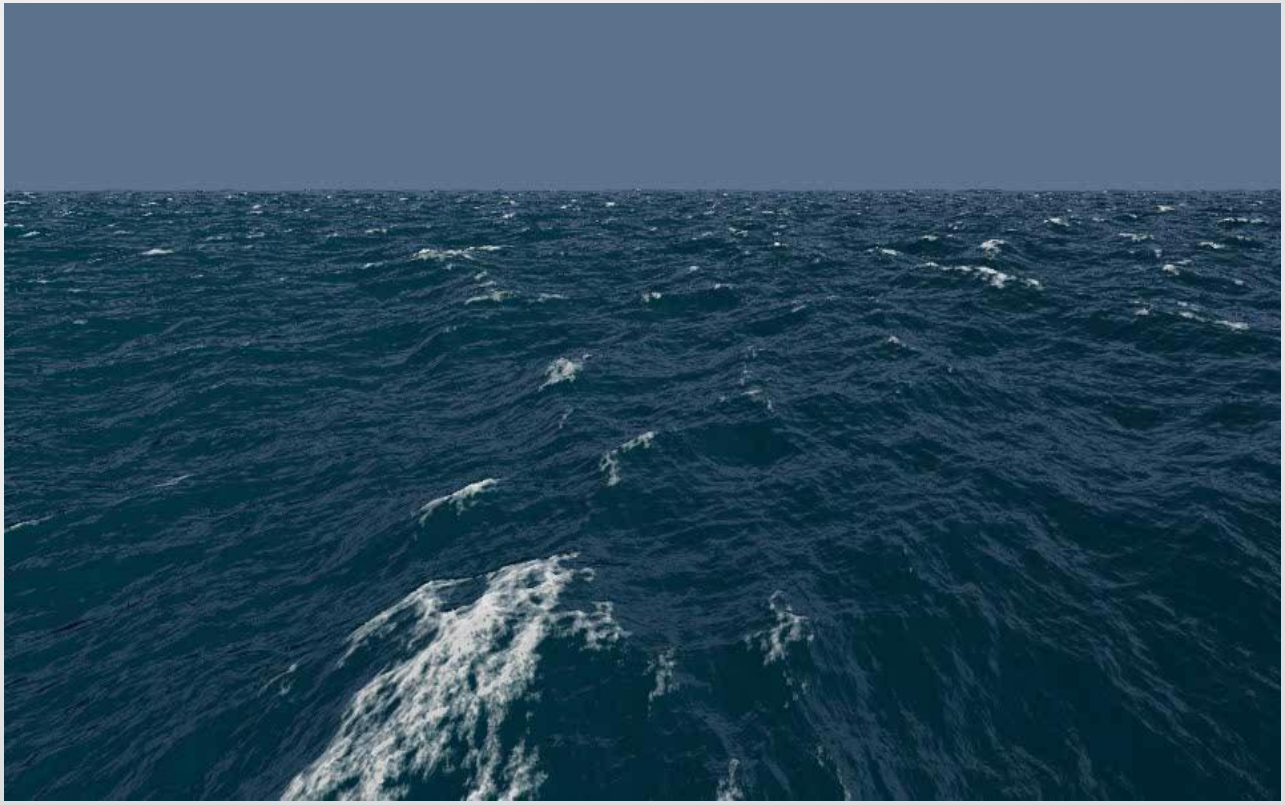
- Jerry Tessendorf 建议:
  使用雅各比变换来产生位移信息
- J > 1: 拉伸
  J < 1: 挤压
  J < 0: 重叠
  出现破浪!
- 我们使用 J < M
  M ~ 0.3 .. 0.5



海洋的模拟

基本泡沫模拟

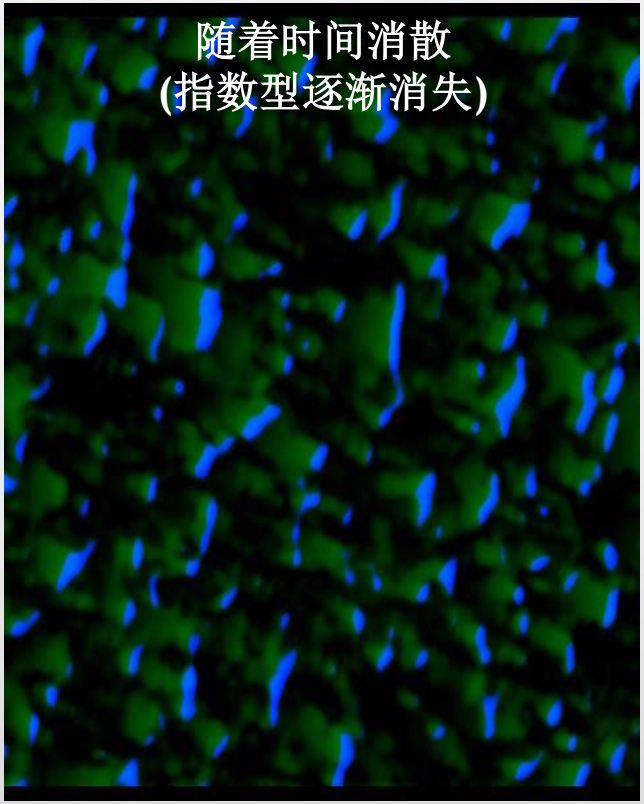
- 通过 saturate(k*(-J+M)) 来混合泡沫纹理
- 破浪区域看起来效果更好
- 泡沫却不见了!

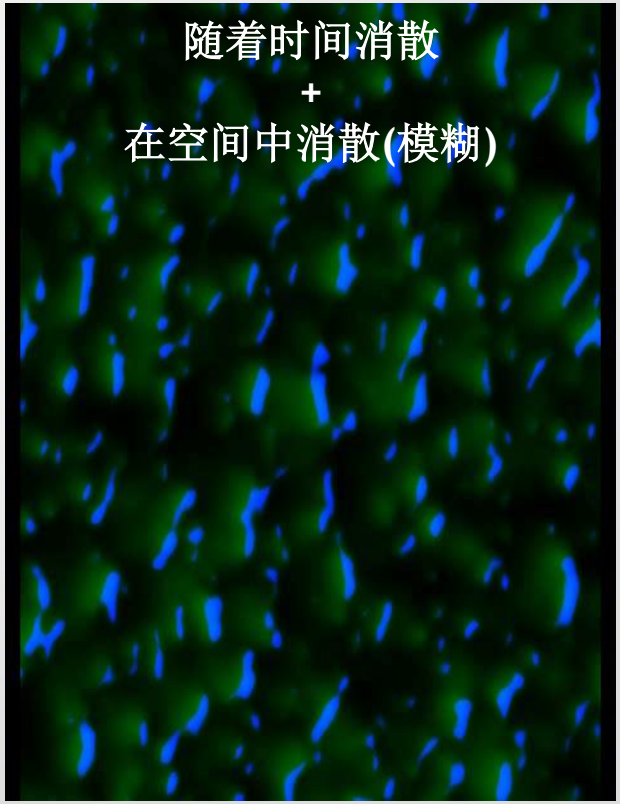


海洋的模拟和渲染

高级泡沫模拟

- 破浪将湍流能量注入到层级中
- 能量在空间中随着时间逐渐消散
- 使用PS为每一个层级进行模拟
- 将结果结合在一起





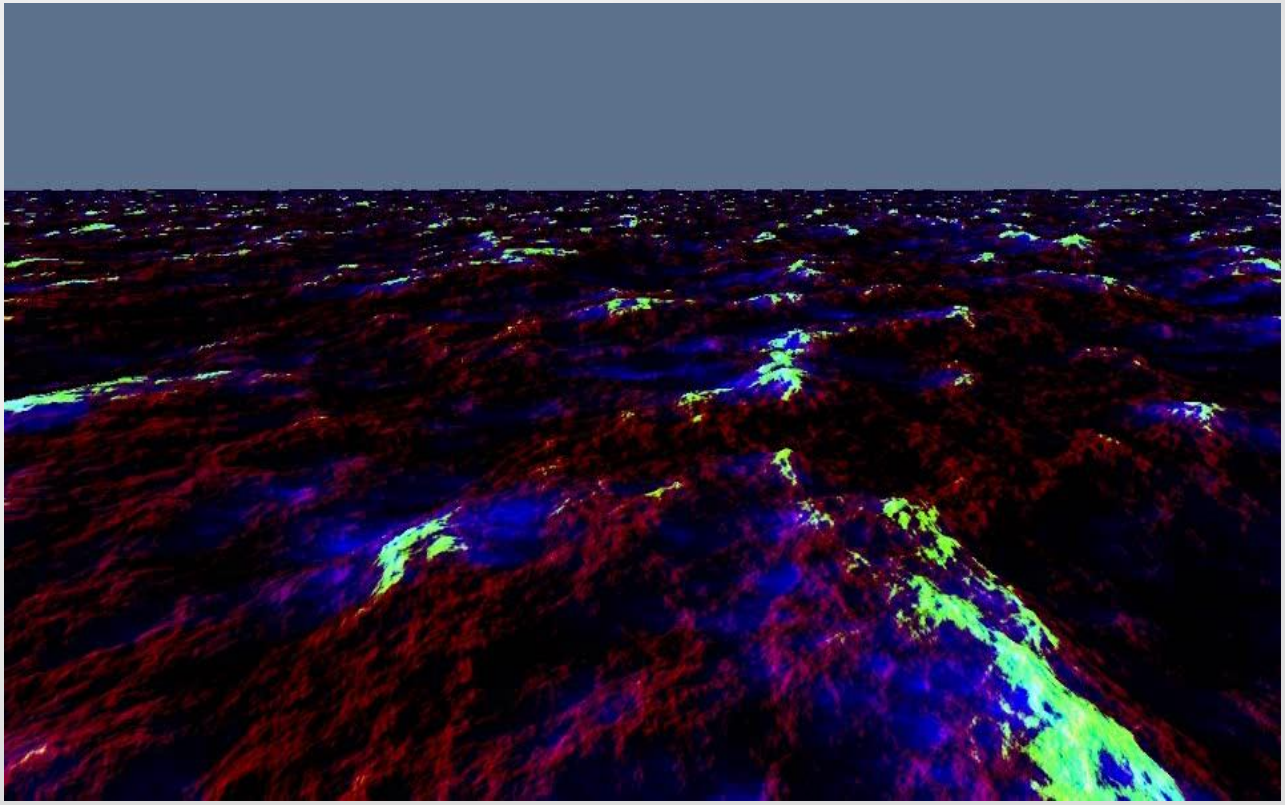
海洋的模拟和渲染

高级泡沫模拟

- 泡沫的模拟结果，使用颜色表示：

  破浪区域
  湍流能量
  表面拉伸

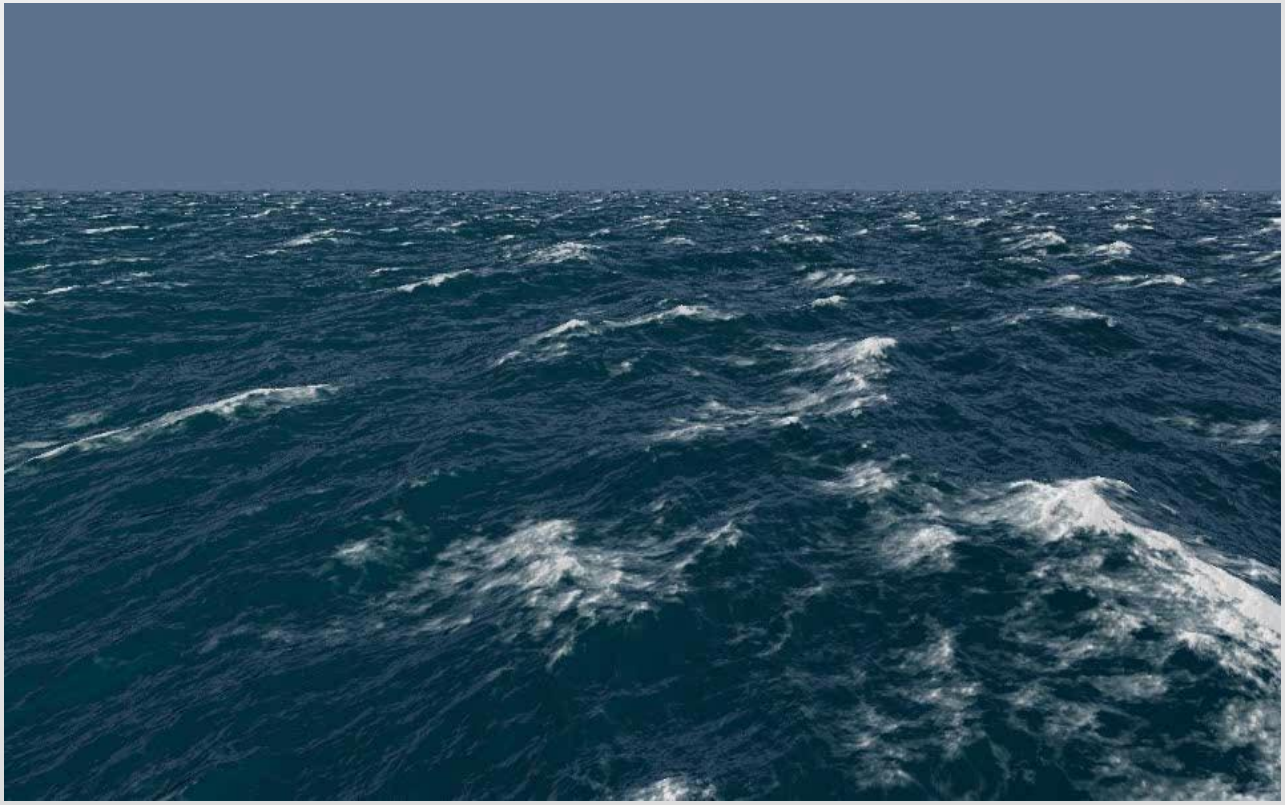
- 拉伸对泡沫层非常重要：
  拉伸 - 变薄
  挤压 - 变厚



海洋的模拟和渲染

高级泡沫模拟及渲染

- 湍流能量
  = 泡沫强度
- 通过能量和拉伸来混合泡沫纹理
- 在破浪区域添加密集的泡沫效果



海洋的模拟和渲染

高级泡沫模拟及渲染

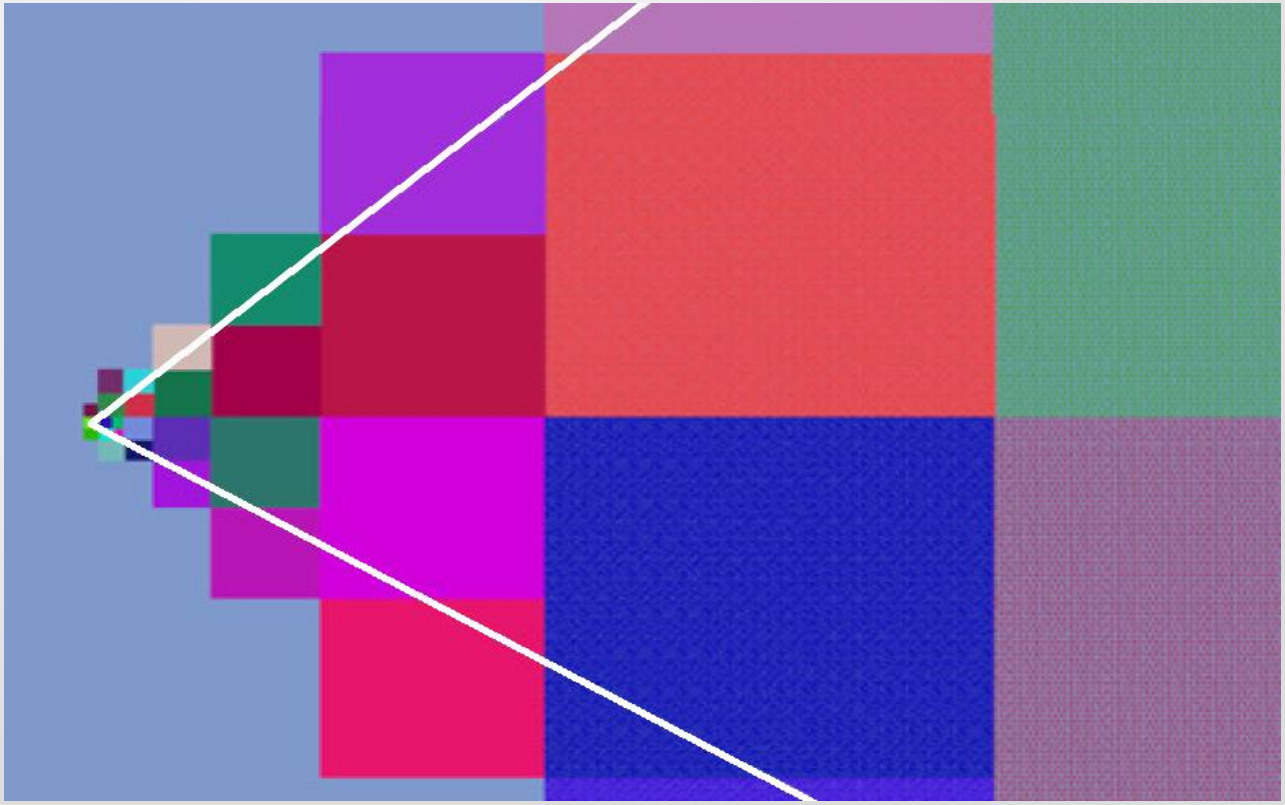
- 湍流能量
  = 泡沫强度
- ..别忘了把气泡混合进去!
- 水下泡沫的颜色体现在折射颜色中，按能量来混合“乳白色”

海洋的模拟和渲染

海洋表面网格

使用视锥体定义的四叉树

- 运行时快速创建
- 每个结点是规则的顶点网格 + 边界处用条带无缝连接
- 预先计算的 VB/IB, indexed triangle lists



海洋的模拟和渲染

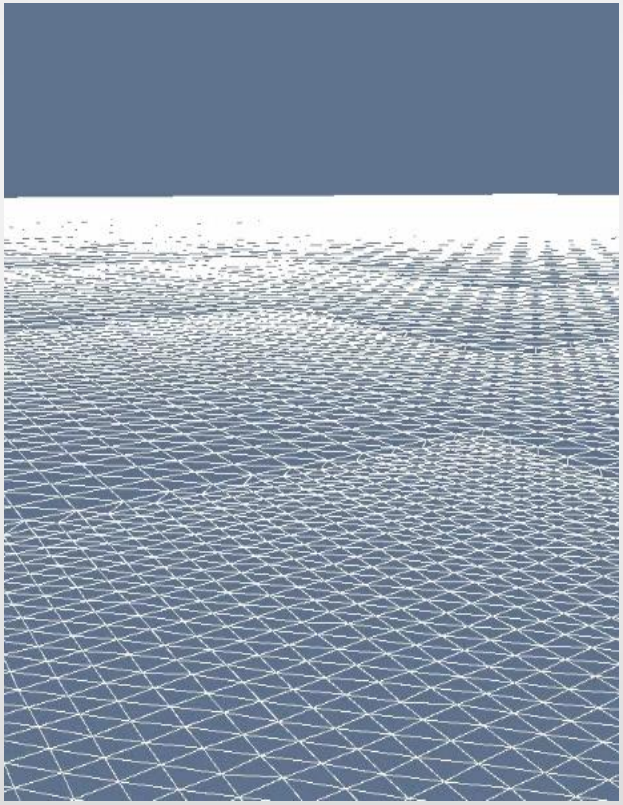
海洋表面网格

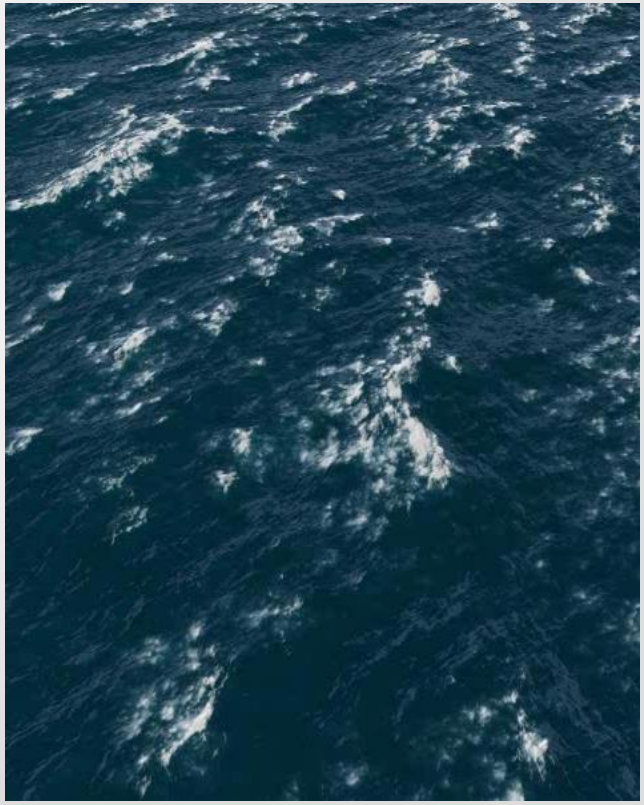
使用视锥体定义的四叉树

- 好的LOD平衡:

  近距离时较密集
  远距离时较稀疏

- LOD会产生突变!



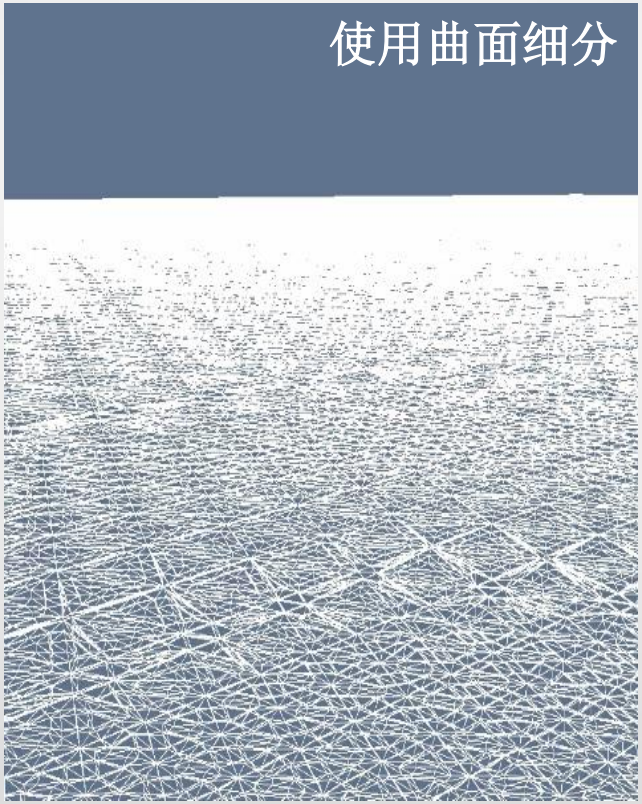


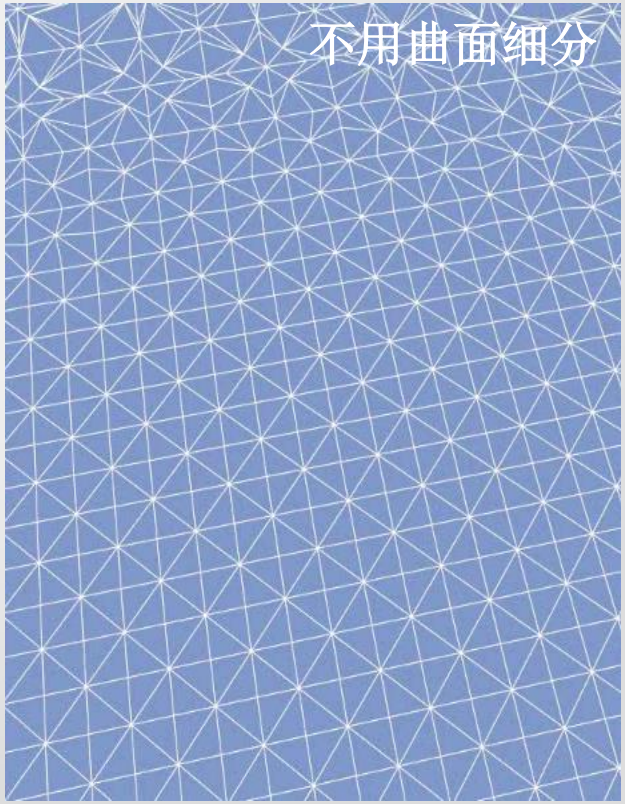
海洋的模拟和渲染

海洋表面网格

网格过渡

- 在不同网格LOD之间平滑过渡
- 在VS中完成
- 使用曲面细分技术更方便!





海洋的模拟和渲染

海洋表面网格

网格过渡处理后的效果：

- 平滑过渡自动调整的LOD
- 可对每个客户端调整网格密度

海洋的模拟和渲染

海岸交互

自然现象:

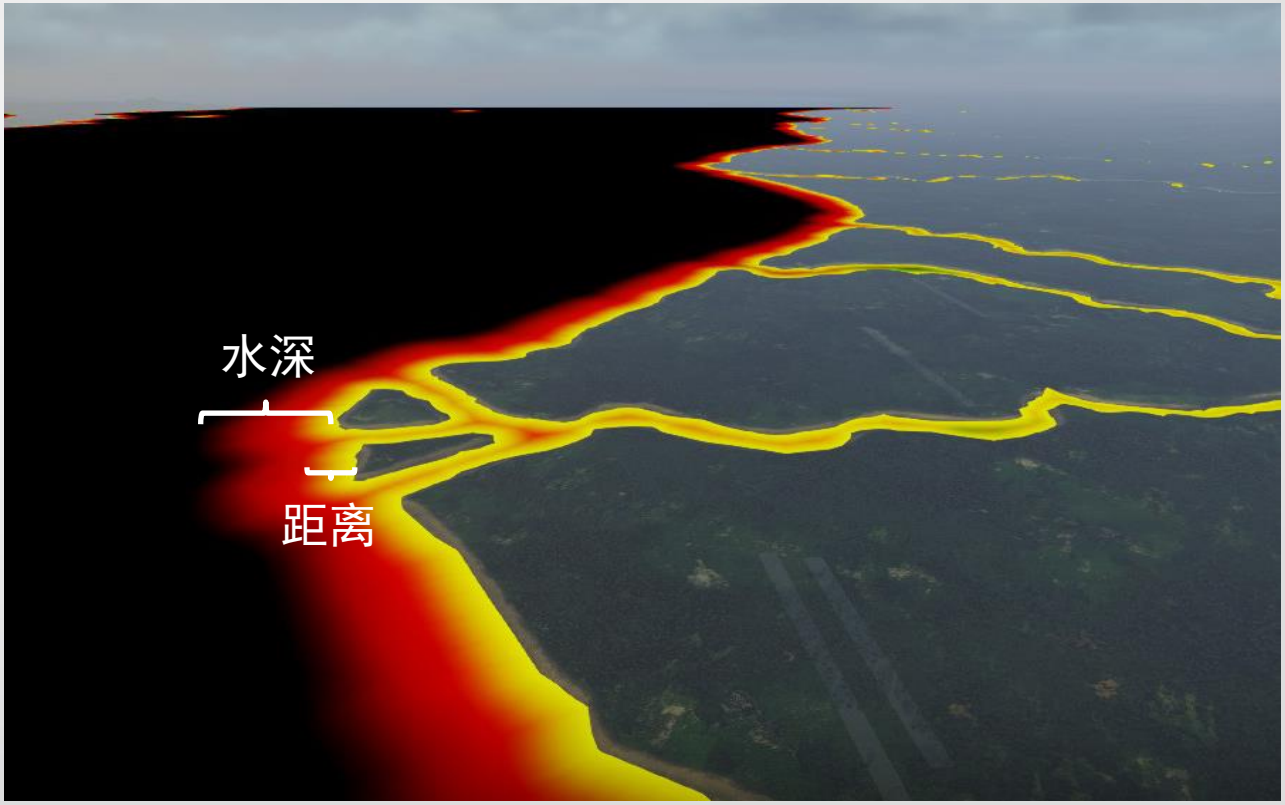
- 近岸水波出现在浅水区
- 基本与海岸线相平行
- 可用距离场进行描述!

海洋的模拟和渲染

海岸交互

距离场

- 将海岸距离作为水波的相位值
- 在GPU上，距离场通过加载4k*4k的纹理来描述65km*65km大小的世界
- RGBA8 纹理:
  R: 深度
  G: 海岸距离
  B,A: 距离场的梯度



海洋的模拟和渲染

海岸交互

河流和湖泊:

- 地表对风和海浪形成障碍
- 没有近岸水波
- 没有海浪

海洋的模拟和渲染

海岸交互

岛屿:

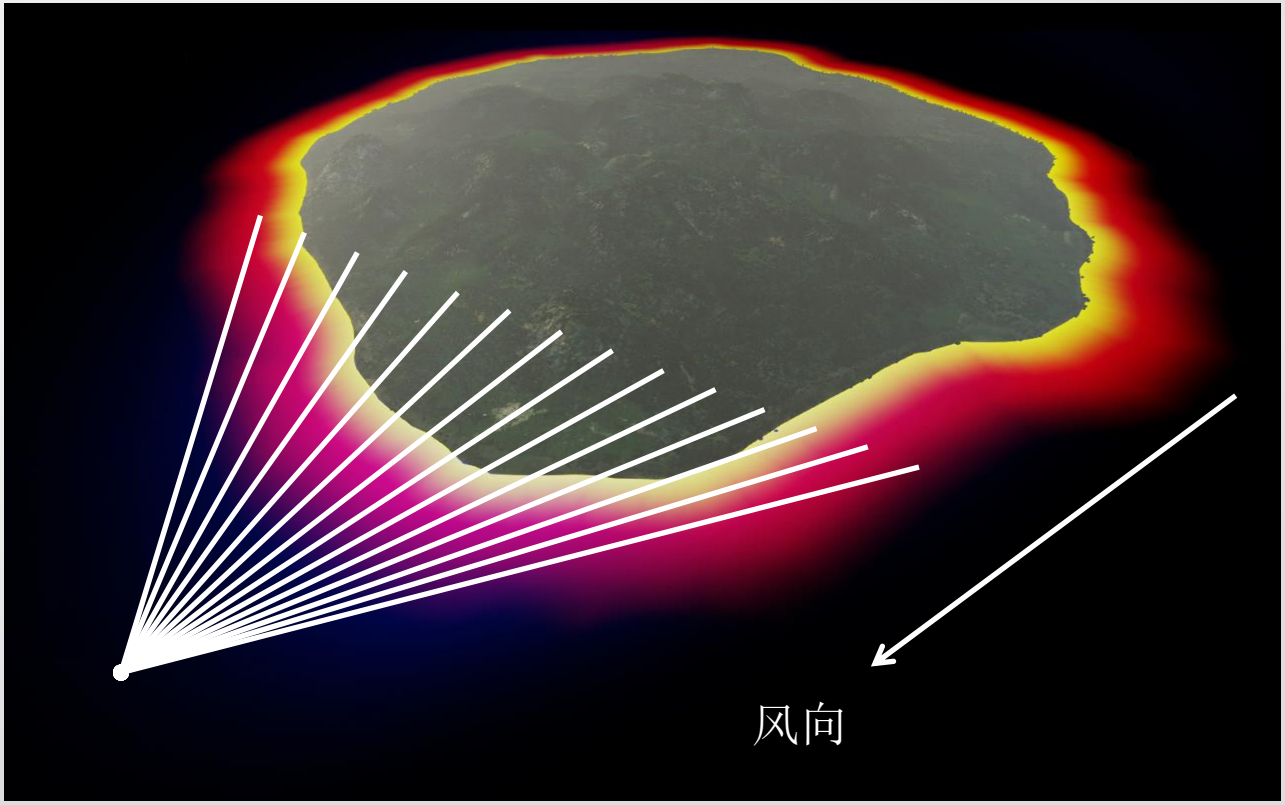
- 地表对风和海浪形成障碍物
- 较小的近岸水波
- 较小的海浪

海洋的模拟和渲染

海岸交互

地表对水波的阻碍作用有多大?

- 在一定方向范围内采样
- 1 (表示地表)
  0 (表示水域)
- 总和 -> [0..1]
- 值大的纹素 -> 模糊



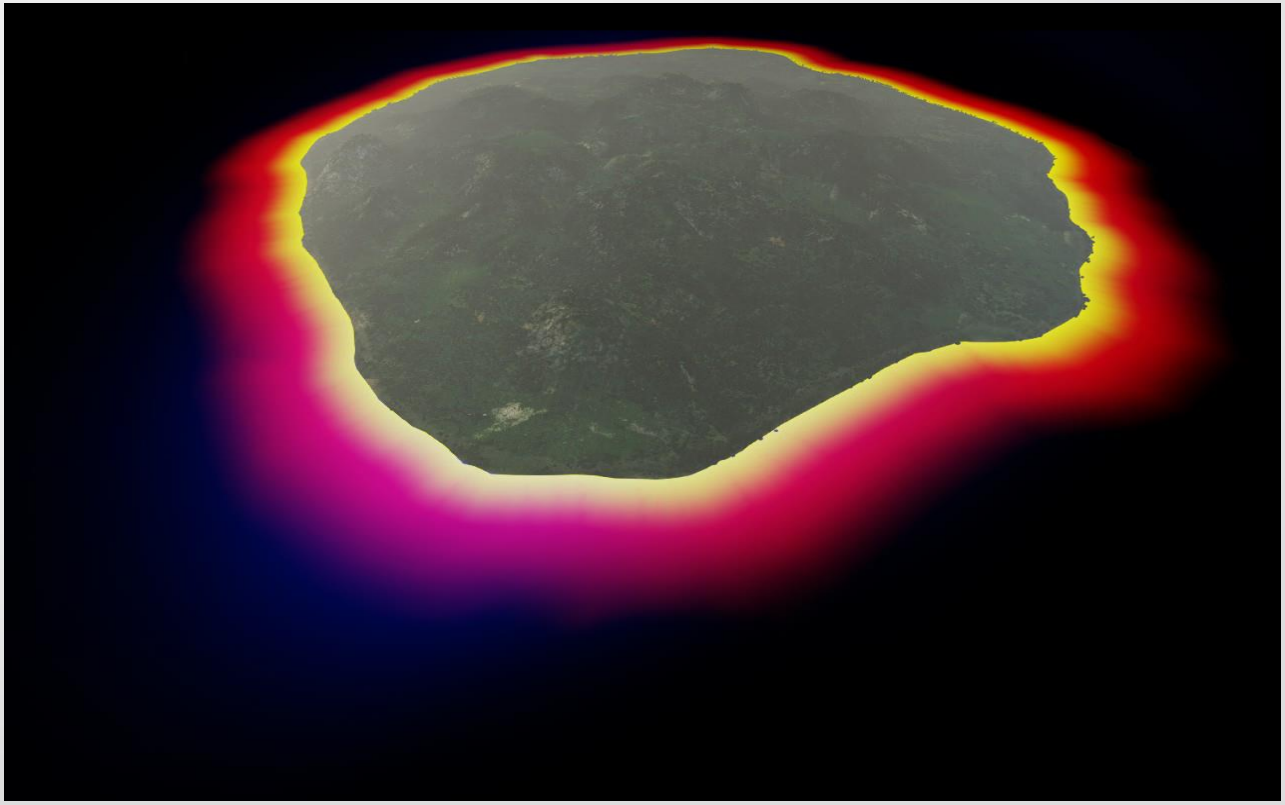
海洋的模拟和渲染

水面和海岸的互相交互

下风向纹理

一种测量“开放度”的方法

- 更小的海浪/近岸水波
- 在河流和湖泊中没有海浪/近岸水波!
- 河流和湖泊自动获取适当的“开放度” 数值

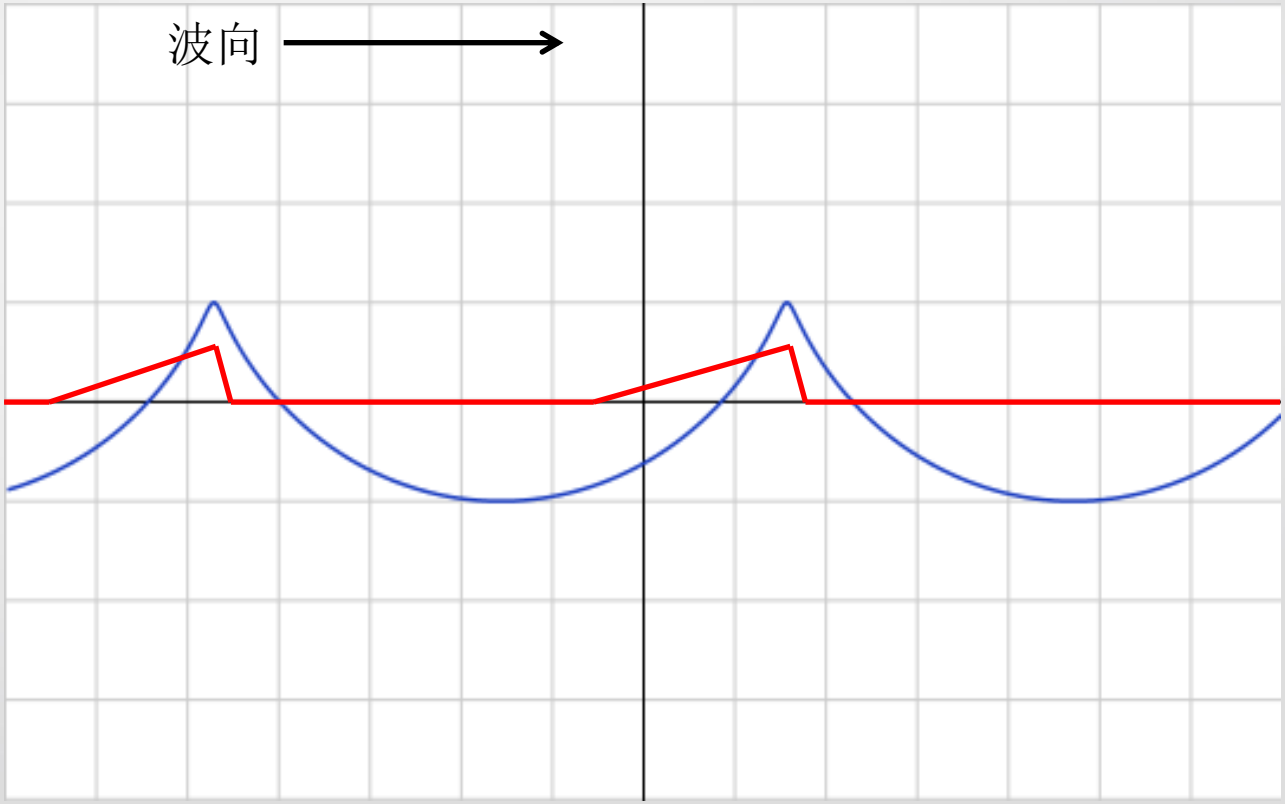


海洋的模拟和渲染

海岸交互

Gerstner 波

- A = 特征波幅
- ω, 从A导出的角速度
- 法线由解析法算出
- 泡沫使用锯齿波

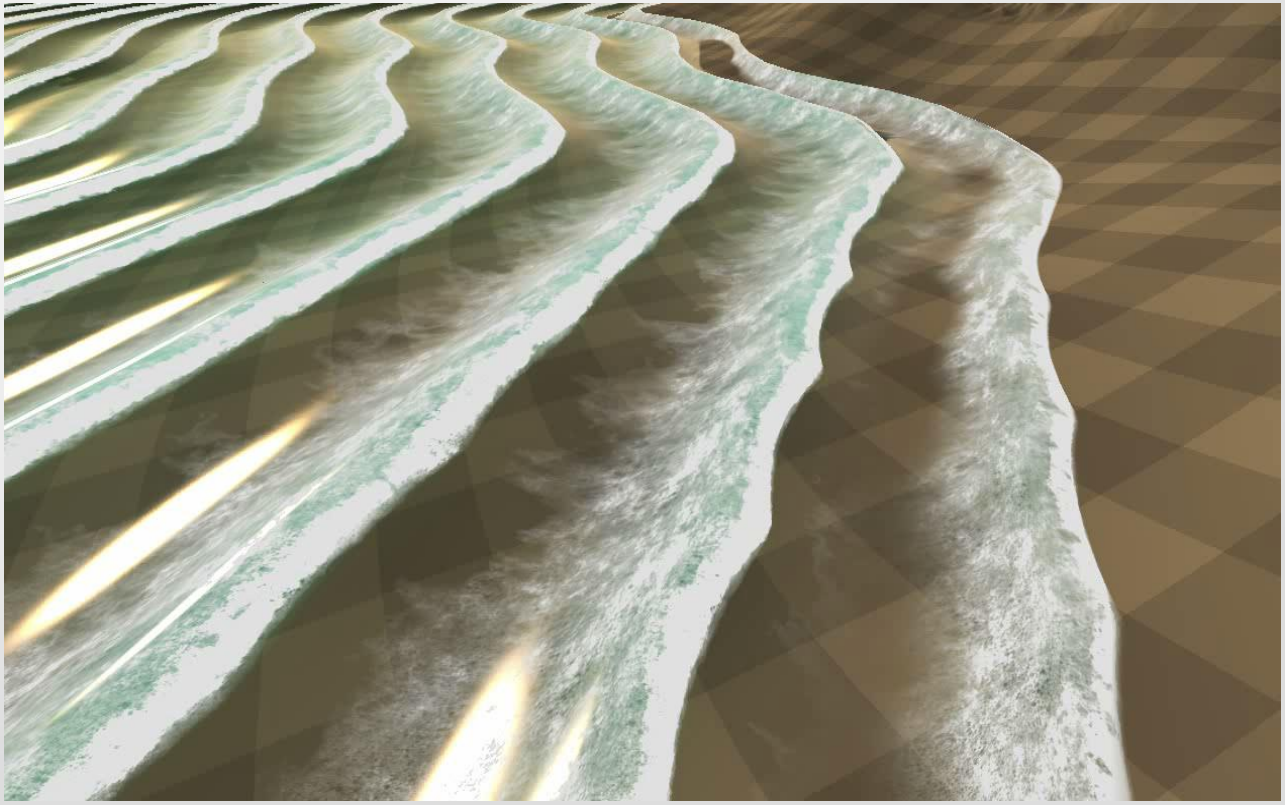


海洋的模拟和渲染

海岸交互

Gerstner 波

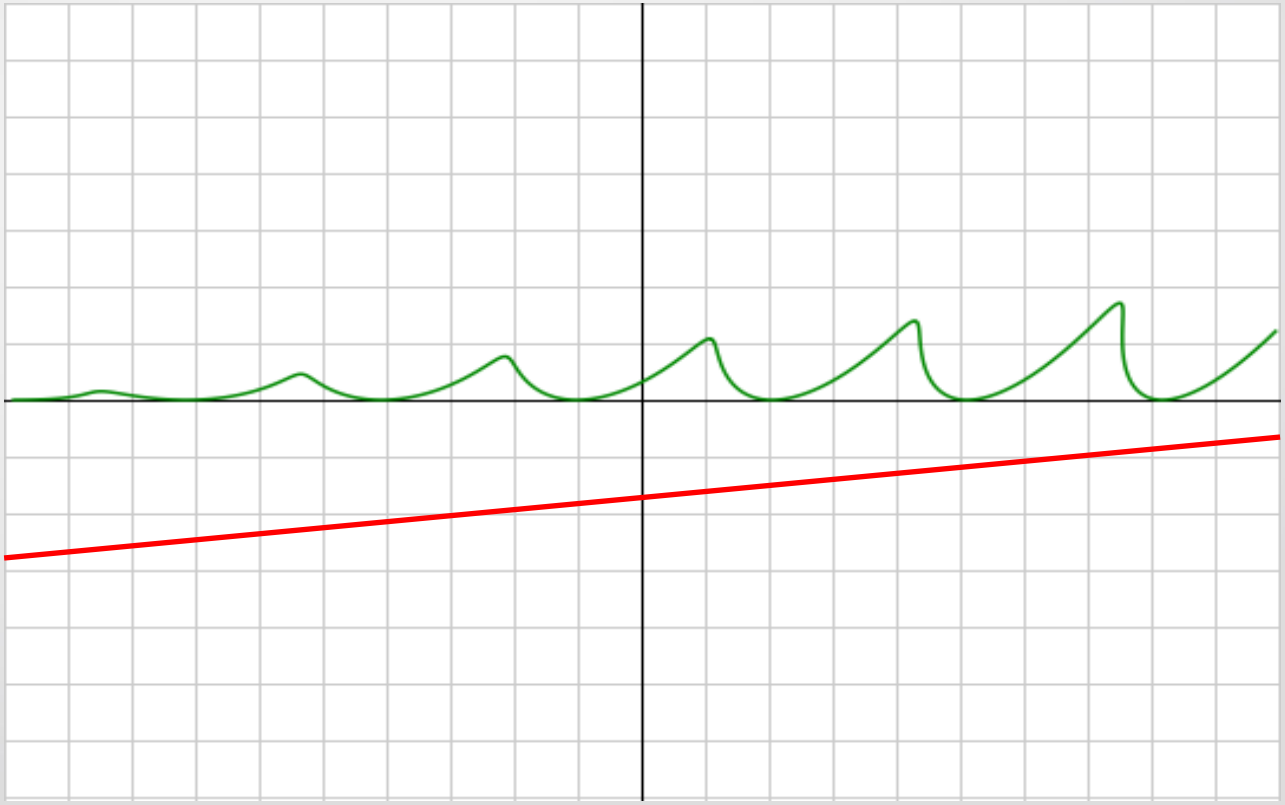
- A = 特征波幅
- ω, 从A导出的角速度
- 法线由解析法算出
- 泡沫使用锯齿波



海洋的模拟和渲染

海岸交互

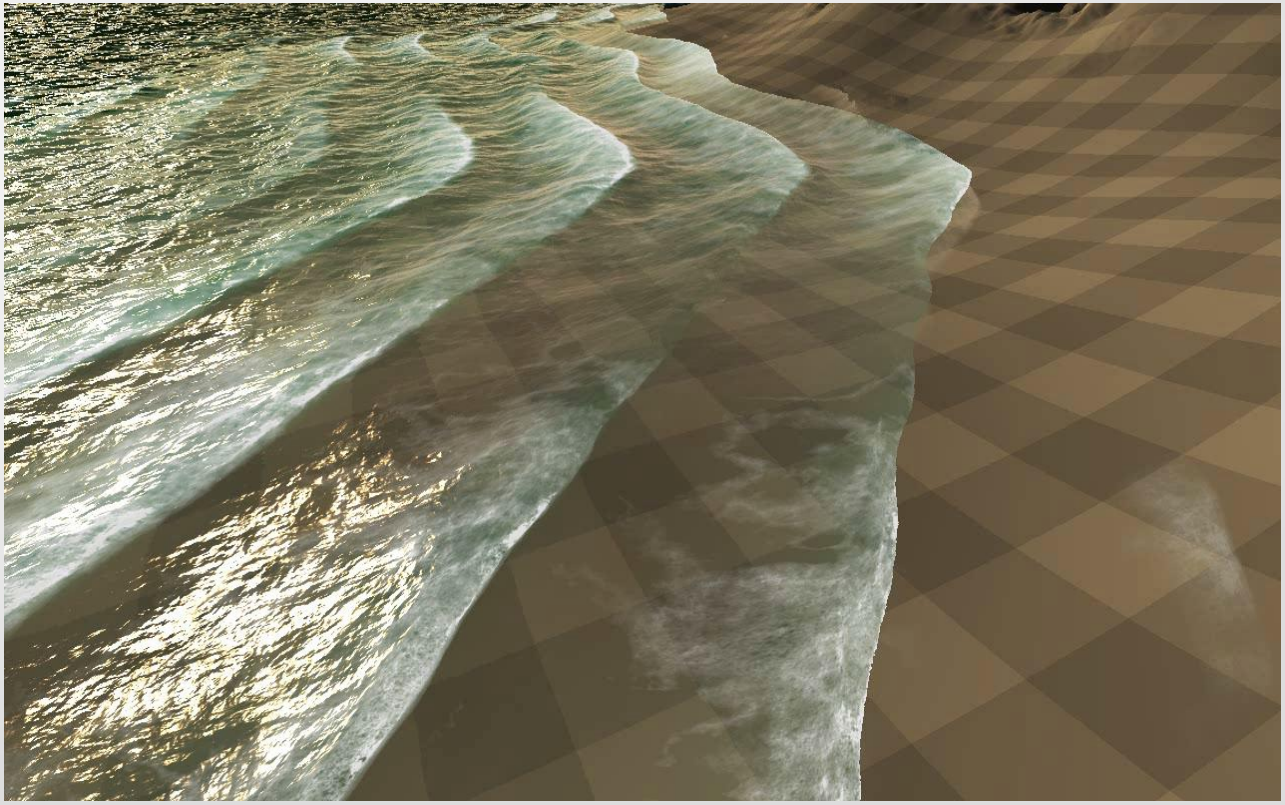
- 近岸水波按水深缩放
- 波峰向前倾斜
- 来自海床的牵引力



海洋的模拟和渲染

海岸交互

- 近岸水波按水深缩放
- 波峰向前倾斜
- 来自海床的牵引力
- 沿海洋->海岸方向对位移和法线进行插值

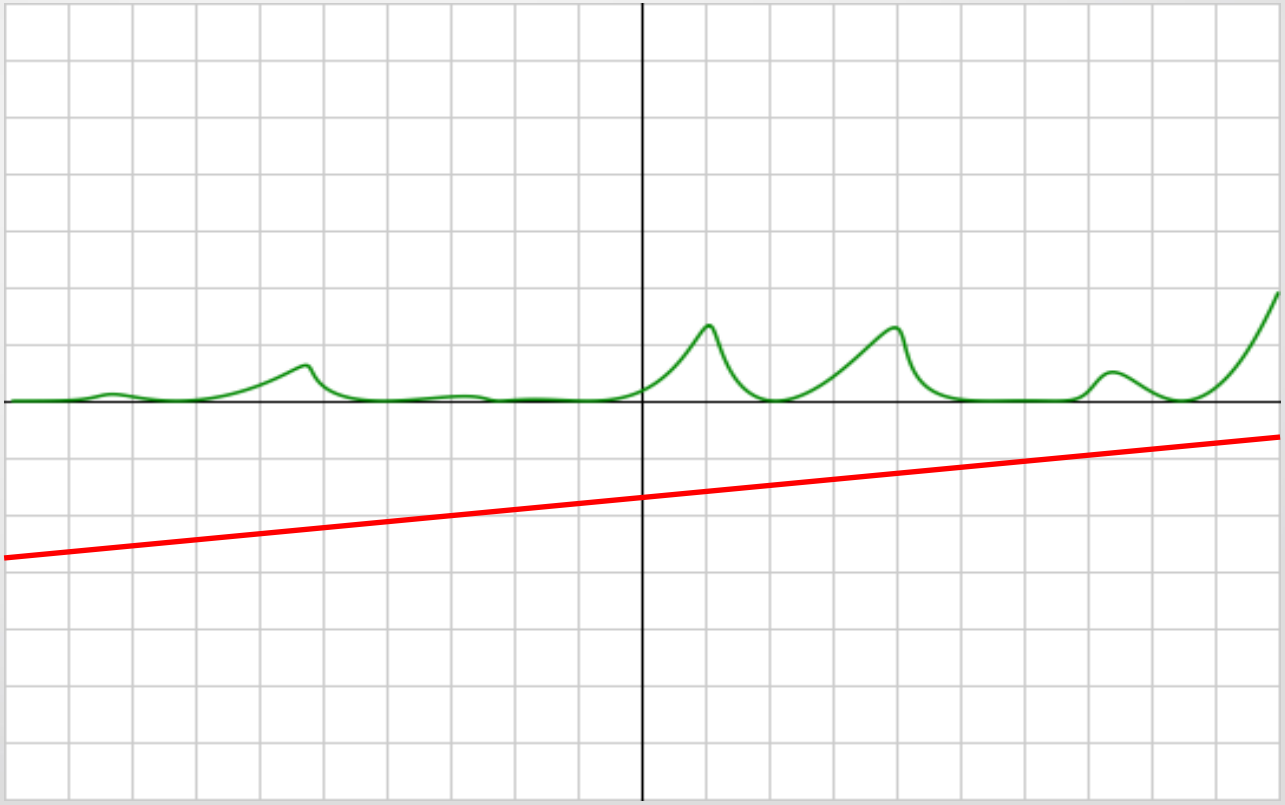


海洋的模拟和渲染

海岸交互

去除水波的规整性

- 添加噪音
- 应用群速度
  (相速度 / 2)

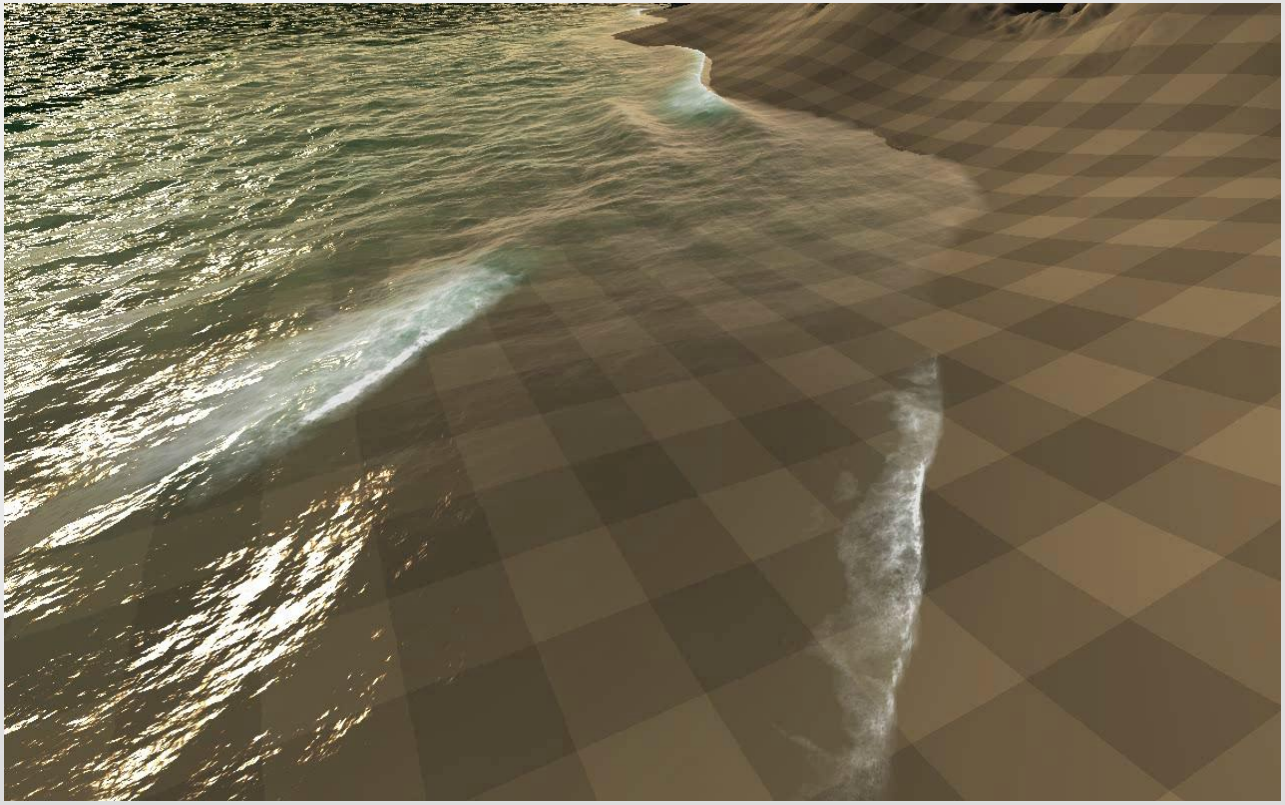


海洋的模拟和渲染

海岸交互

去除水波的规整性

- 添加噪音
- 应用群速度
  (相速度 / 2)



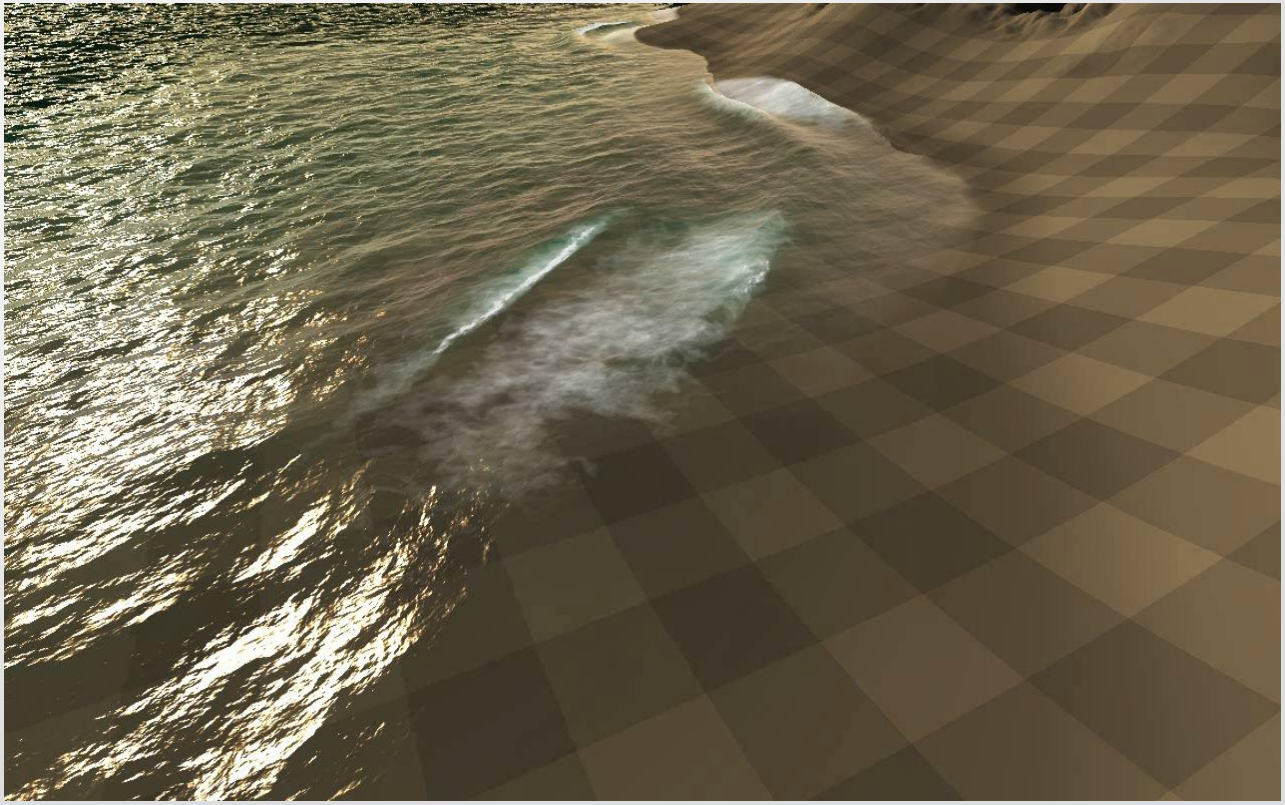
海洋的模拟和渲染

海岸交互

地表着色

沙滩变得潮湿且反光

- 水和泡沫被击退



海洋的模拟和渲染

海岸交互

远视角观察

海洋的模拟和渲染

渲染性能

性能 <->图形质量的调整

	近景	中景	远景
位移及法线	所有层级	部分层级	最大层级
水面网格	高密度网格	低密度网格	平面四边形
折射	扭曲效果	简单效果	无
反射	扭曲效果	模糊	模糊
海岸交互	全开	全开	法线 + 泡沫
尾浪及水花	全开	只计算法线	只计算法线
散射光	有	有	无

海面模拟与游戏物理的结合

物理模拟面临的挑战

- 从高端的PC到烤面包机，尽可能快:
  - WaveWorks: CUDA, DC 和 CPU 模拟
  - Gaijin: 对一些其它平台的GPU也提供了支持
- 每一个玩家都需要相同的物理模拟结果
  - 物理水面位移 vs 图形水面位移
- 与游戏世界交互
  - 将水面位移值反馈给船只和水上飞机

海面模拟与游戏物理的结合

游戏内物理的模拟

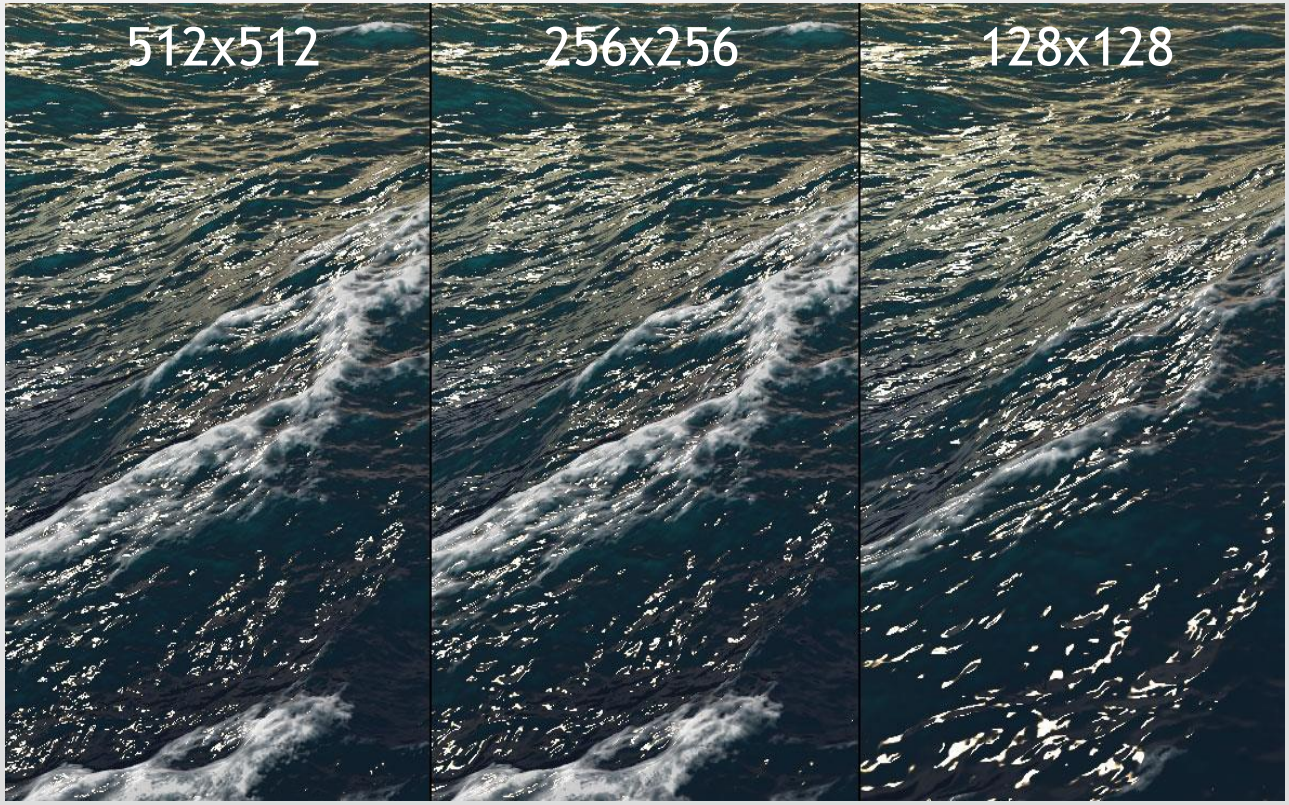
- 物理模拟运行于CPU上
  - 保证每个玩家有相同的物理模拟
  - 服务器 + 每一个客户端
  - 固定时长, 48次/秒
- 为图形而进行的模拟运行于CPU 或 GPU上
  - 尽可能使用GPU
  - 仅在客户端上模拟
  - 每帧时长可变

海洋模拟

游戏内物理的模拟

大尺寸 FFT 对CPU而言极其费时!

- 降低频谱尺寸
- 物理: 128x128
- 图形: 128x128 .. 512*512
- 非常接近! 3m波幅下差异< 5 cm



海面模拟与游戏物理的结合

游戏内物理的模拟

- 《战争雷霆》是一个在线游戏
- 客户端->服务器 延时达到 500 毫秒
  - 在服务器端，最差情况下需用高达24步来重新模拟
  - 正常的情况下需要0到2步来重新模拟 - 导致最高2倍的计算量
- 服务器->客户端 延时达到 1500 毫秒
  - 在客户端，最差情况下需用高达72步来进行重新模拟
  - 正常的情况下需要0到5步来重新模拟 - 导致最高5倍的工作量
- 重新模拟海洋表面非常的昂贵!

海面模拟与游戏物理的结合

游戏内物理的模拟

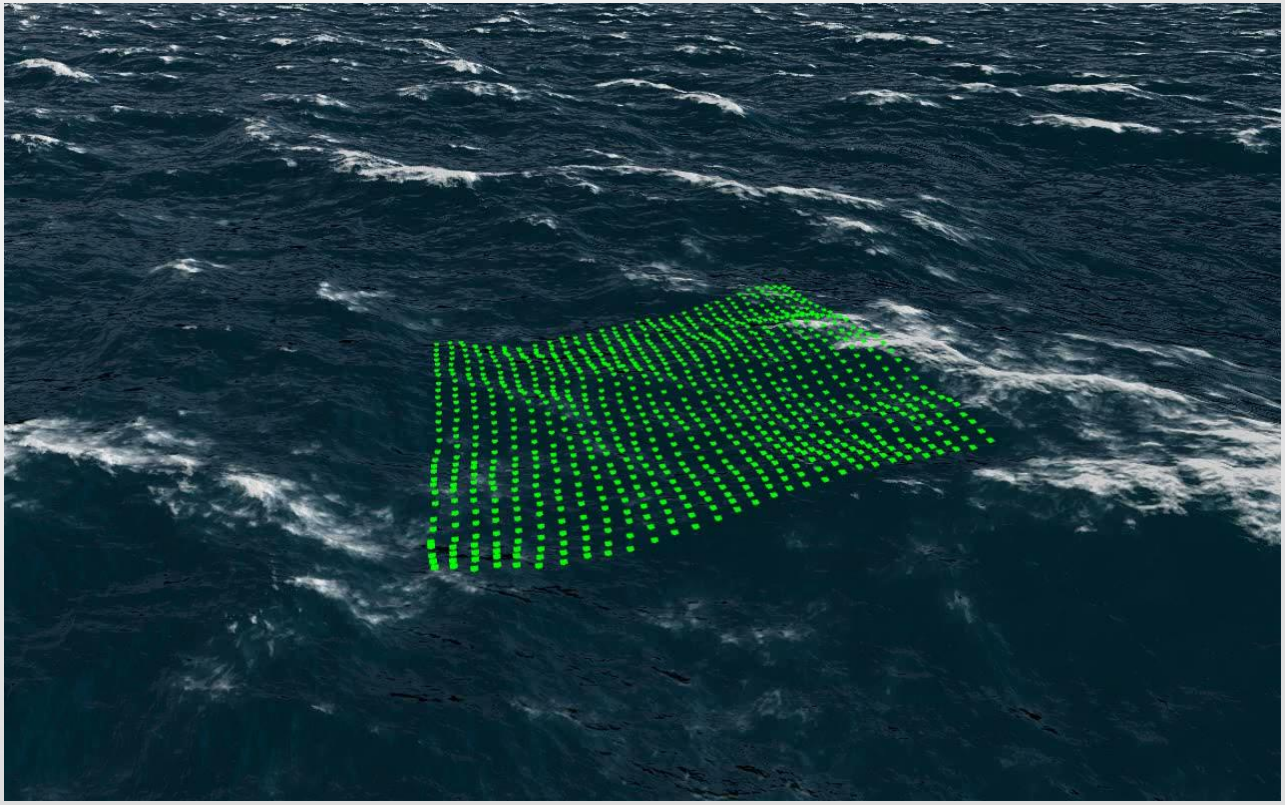
- 对模拟结果进行缓存
  - 模拟一次，保存结果。当需要“先前”的结果时，可重新读回
- 将模拟结果存放到数组中 (2D水面位移数组)
  - N(片) * 4(层级)*128x128(FFT尺寸)*3(dx,dy,dz)
  - 每一片需要~800 kb
- 最大延时 = 1.5秒, 48次/秒 -> 72切片 -> ~57MB
  - 可以去除小波浪的层级
  - 可以半速进行物理模拟

海面模拟与游戏物理的结合

为游戏内置物理提供数值反馈

为船只和漂浮物提供水面高度值反馈

- 从所有的层级得到位移，求和后返回给应用程序
- 在CPU端完成
- 少量的回读 + 一点数学运算 = 真实可信的船只运动
- 不适用于向水中投射物体!



海面模拟与游戏物理的结合

射线求交

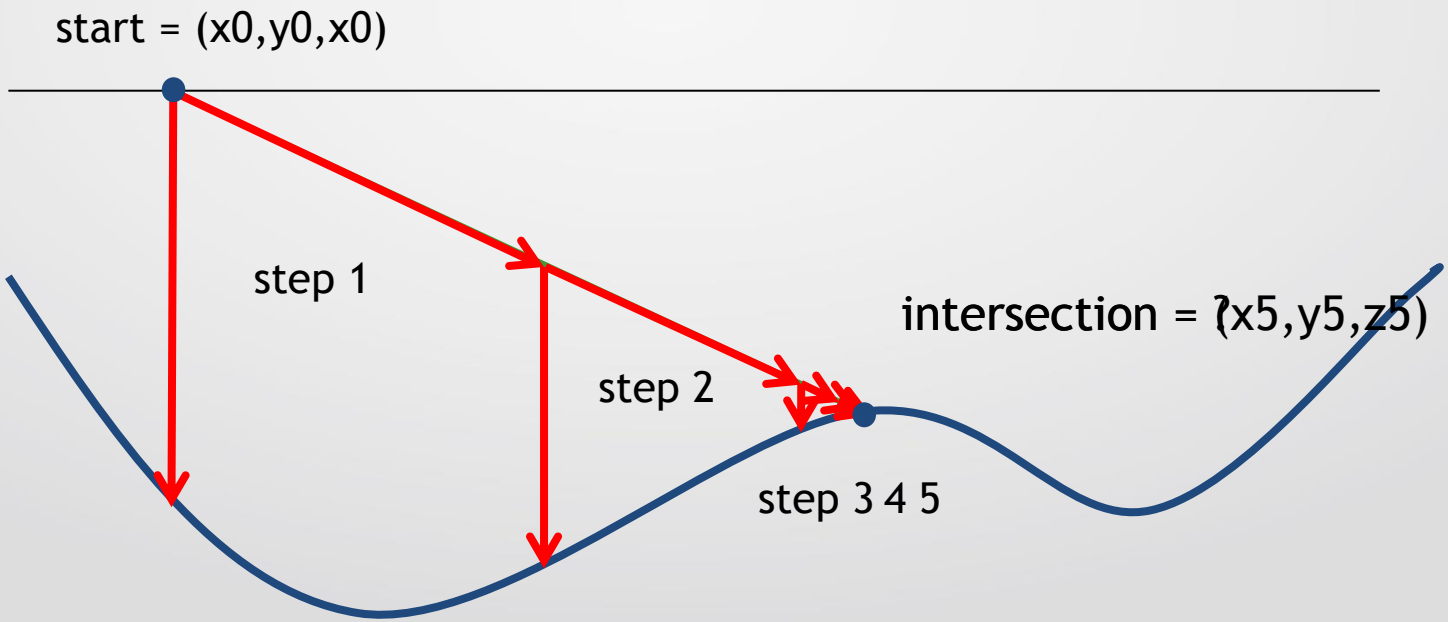
- 投射物需要做 射线/表面 的相交测试
- 投射物与水的碰撞通过射线求交来完成
  - 需要沿射线对水体进行追踪
- 这比数值反馈更费时
  - 可能出现无穷次迭代
    - 当前误差 < 阀值 (比如5 cm) 时，退出迭代
    - 或步数 > 预设值N时，退出迭代
- 每一步追踪也不是一个简单的反馈过程!

海面模拟与游戏物理的结合

射线求交

追踪水体

Position += direction * (current_Y - water_Y)

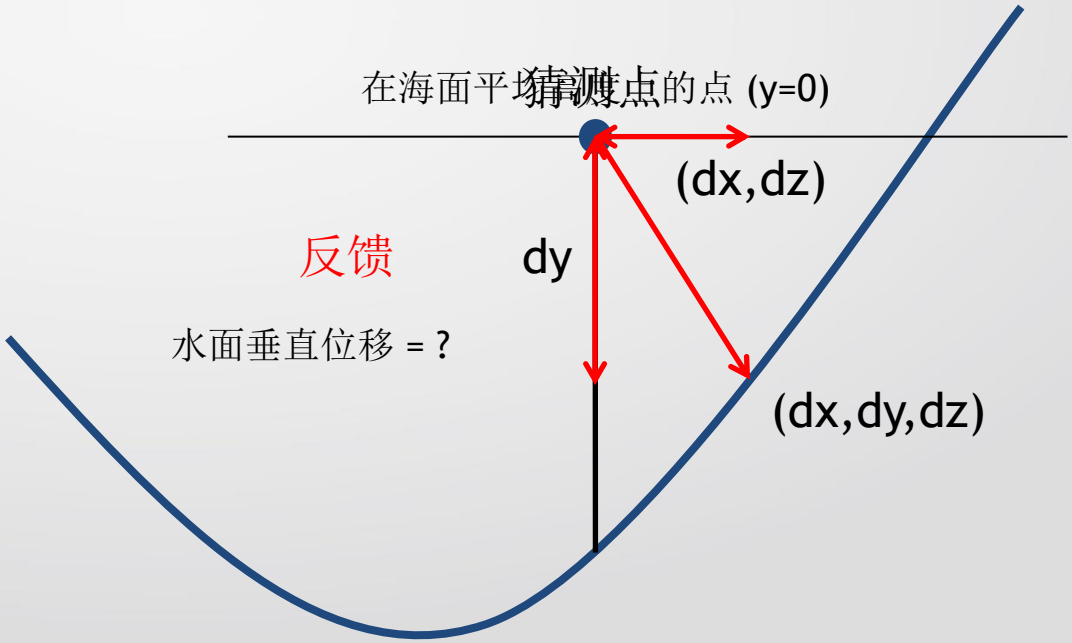


海面模拟与游戏物理的结合

射线求交

假设位移是局部线性的

- 在一个“猜测点” x,z处采样
- 读取“猜测点”的位移值，然后根据x,z的偏移量将“猜测点”往回移动
- 重复N次. 4 步足够了!

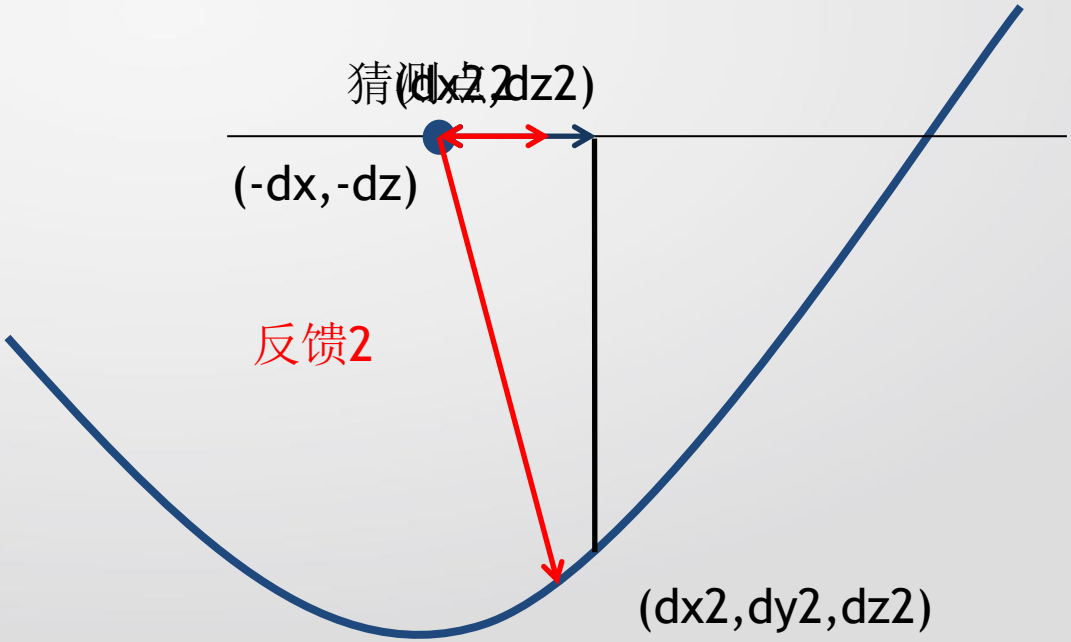


海面模拟与游戏物理的结合

射线求交

假设位移是局部线性的

- 在一个“猜测点” x,z处采样
- 读取“猜测点”的位移值，然后根据x,z的偏移量将“猜测点”往回移动
- 重复N次. 4 步足够了!

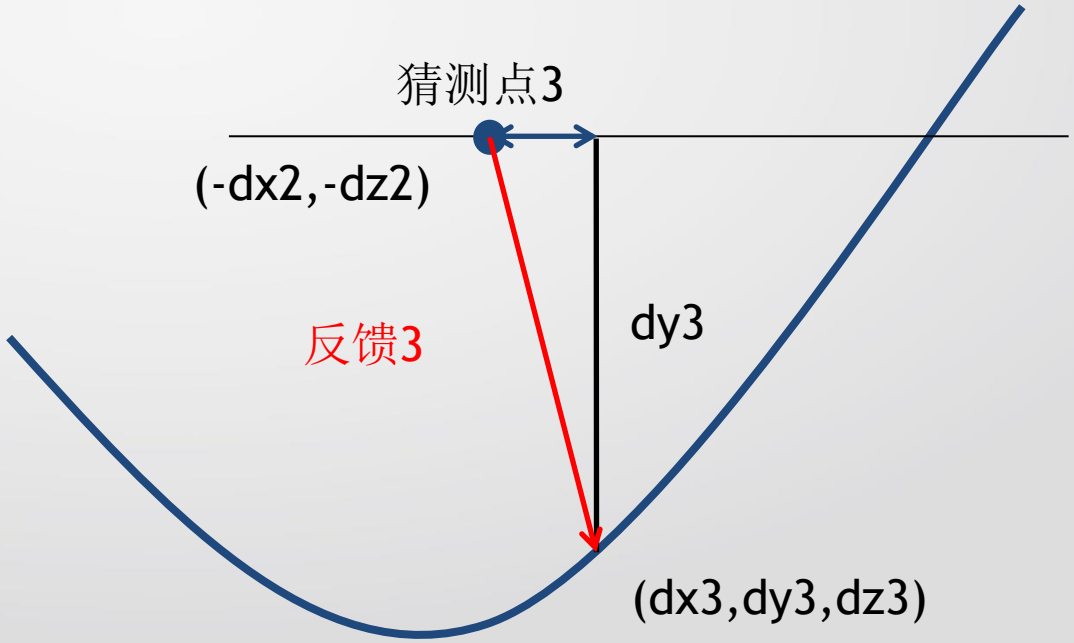


海面模拟与游戏物理的结合

射线求交

假设位移是局部线性的

- 在一个“猜测点” x,z处采样
- 读取“猜测点”的位移值，然后根据x,z的偏移量将“猜测点”往回移动
- 重复N次. 4 步足够了!

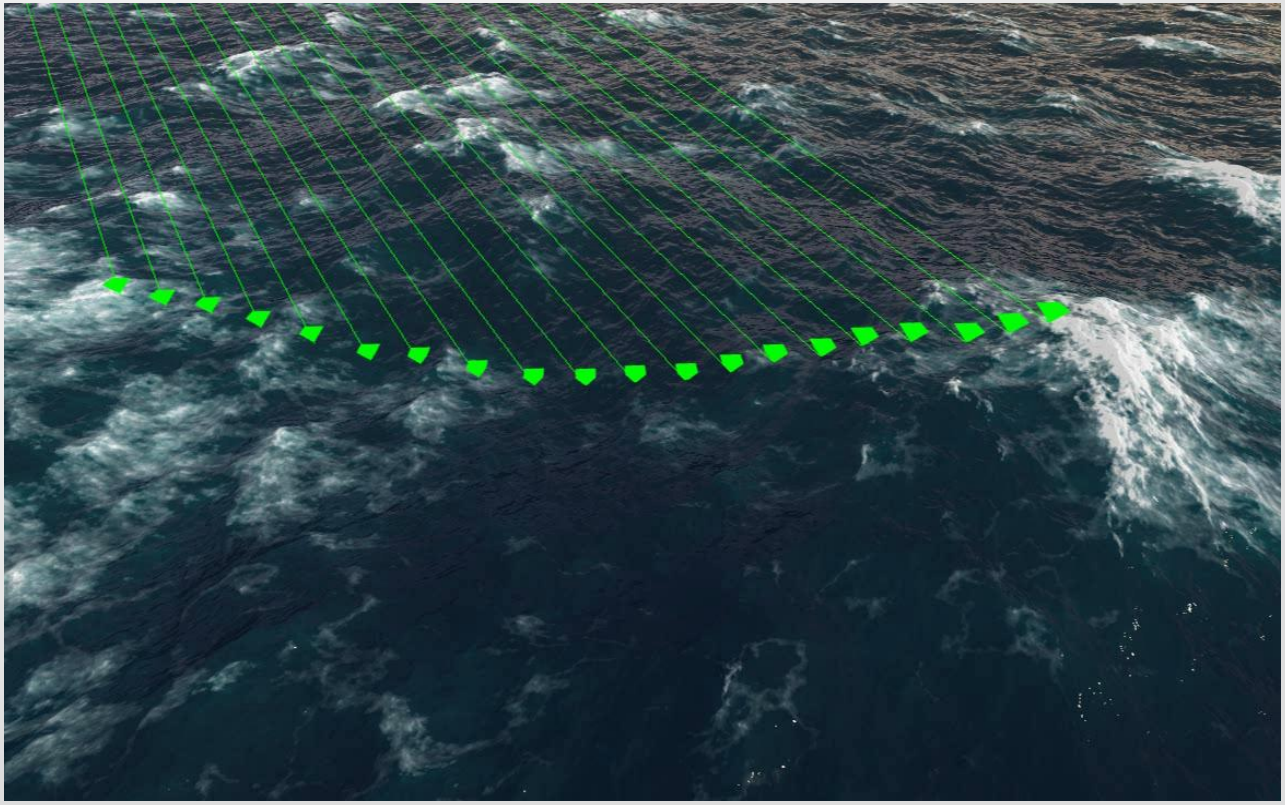


海面模拟与游戏物理的结合

射线求交

结果

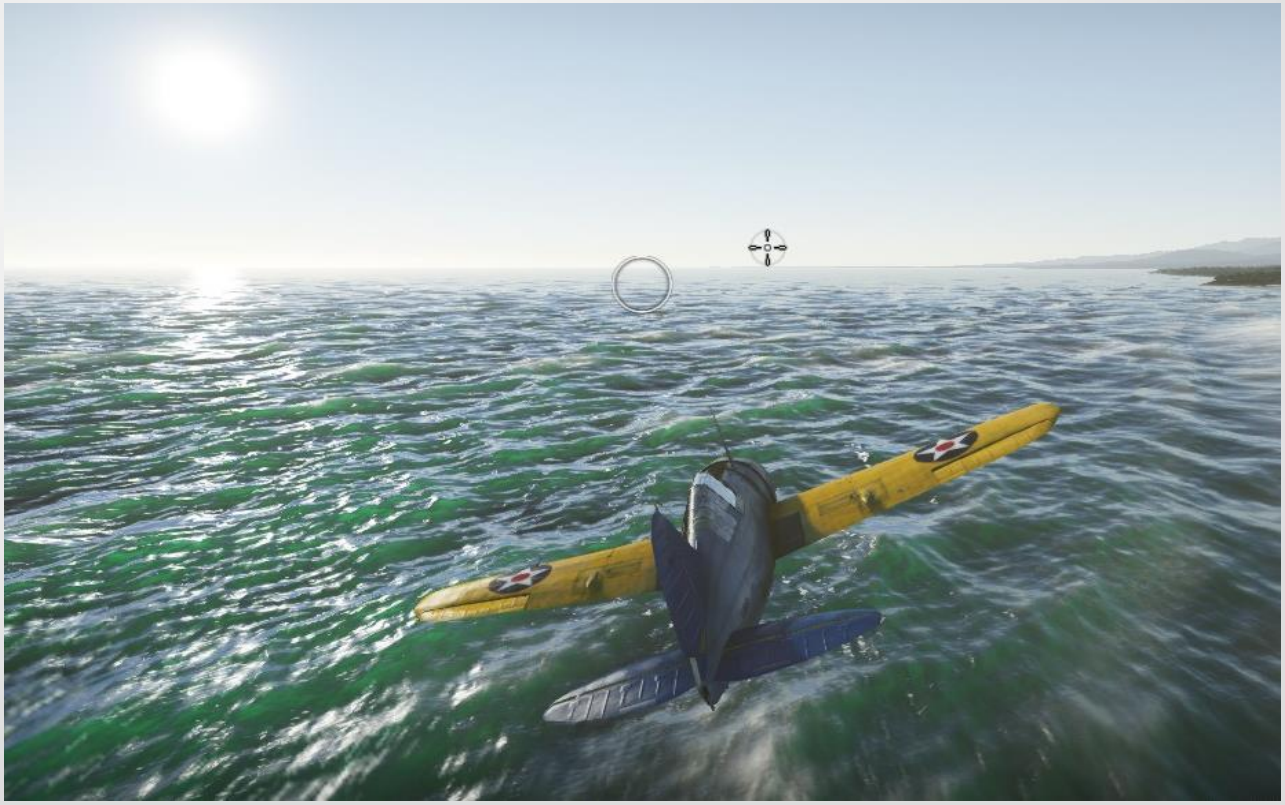
- 计算量不小，
  但是只有少数飞行中的投射物需要射线求交
- 在每秒进行48次模拟的频率下，最多支持~2000个飞行中投射物



已集成至《战争雷霆》中

尚未公开

WaveWorks发行版中
包含演示及示例程序

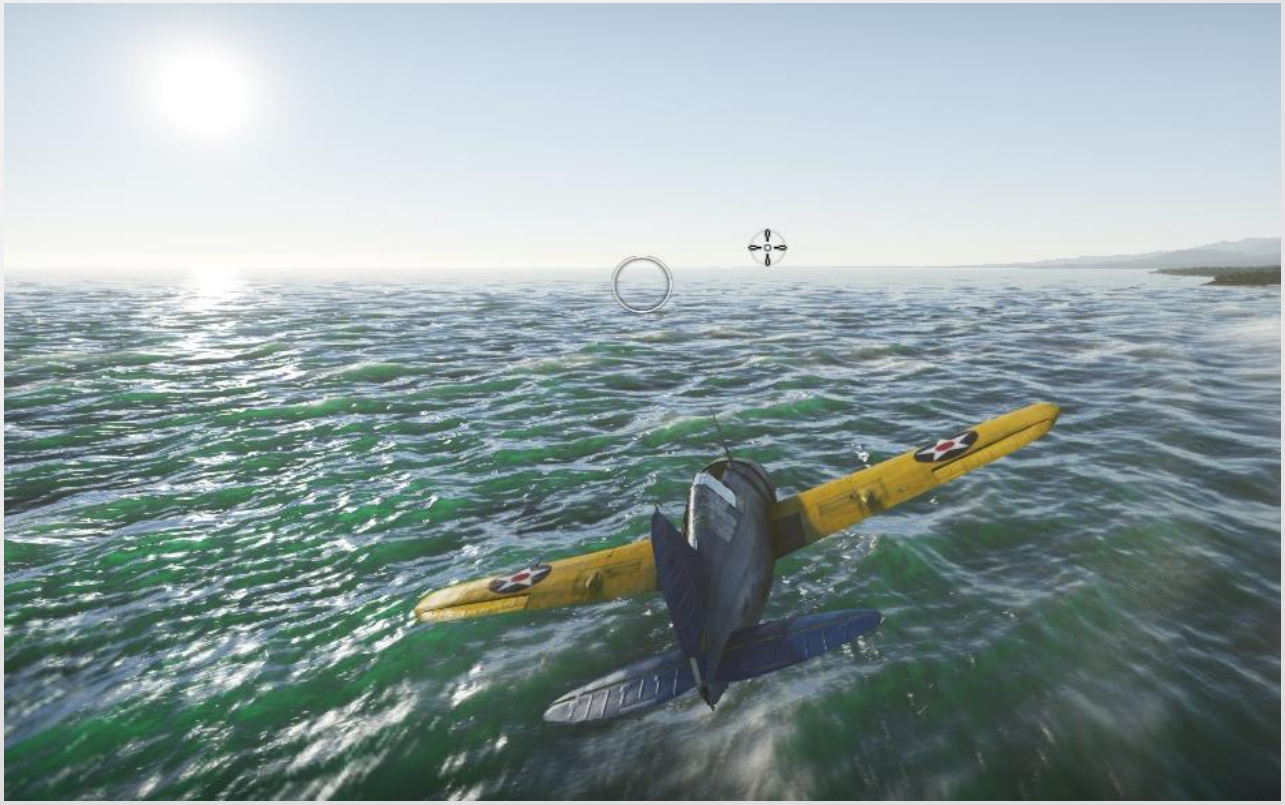


已集成至《战争雷霆》中

尚未公开

WaveWorks发行版中
包含演示及示例程序:

- 海洋模拟
- 渲染
- 几何过渡
- 海岸交互
- 射线求交
- 数值反馈



演示

D3D11 WaveWorks 示例程序
Android 测试程序

谢谢!

问答时间

Tim Tcheblokov, NVIDIA


CGDC 2015 中国游戏开发者大会
CHINA GAME DEVELOPERS CONFERENCE


分享智慧
LET US SHARE