

硕士研究生学位论文

|  |  |
| --- | --- |
| 题目： | **基于UE5的真实感水体** |
|  | **渲染技术研究与实现** |

|  |  |
| --- | --- |
| 姓 名： | 邱松涛 |
| 学 号： | 2001210398 |
| 院 系： | 软件与微电子学院 |
| 专 业： | 计算机技术 |
| 研究方向： | 虚拟与增强现实 |
| 导师姓名： | 王平 教授 |
|  | 许捷 讲师 |

二〇二三 年 六 月

版权声明

任何收存和保管本论文各种版本的单位和个人，未经本论文作者同意，不得将本论文转借他人，亦不得随意复制、抄录、拍照或以任何方式传播。否则，引起有碍作者著作权之问题，将可能承担法律责任。

# 摘 要

水体渲染不论在离线渲染还是实时渲染中都是十分重要的一个课题，一直是计算机图形学和游戏开发领域的核心难点之一。随着计算机技术及硬件的不断发展，水体渲染的技术在不断发展着。目前业界主流的水体渲染技术主要分成了几大类，比较常用的如基于波形叠加方法的正弦波和Gerstner波，基于统计模型的快速傅里叶变换方法，基于粒子物理的欧拉方法、拉格朗日方法等。其中基于波形的方法适用于大规模的水体，如海洋等的渲染，但是在互动和水流的物理逼真程度上不如基于粒子物理的方法。基于粒子物理的方法则是利用流体力学的原理，模拟粒子之间的黏性和碰撞，水互动方面也是基于物理实现，效果十分逼真。但缺点在于性能耗费较高。

目前计算机硬件的逐渐强大，同时Unreal Engine提供了强大粒子物理引擎Niagara，所以本文将利用Niagara粒子引擎强大的并行特性，在其上构建一套基于流体力学模拟的水体系统，其中包括水体粒子的粘度、压力、速度解算，水体材质渲染、场景交互实现等一系列工作。本文也将探索和解决基于流体模拟的真实感水体系统的性能优化方案，以便在实时渲染项目和游戏中推广普及该方法，使得虚拟世界变得更加真实。

本文将介绍项目背景和研究意义，并分析相关技术的发展现状以及市场案例。同时会给出水体模拟的理论依据，以及在UE的Niagara中的具体实现。同时还会对整个系统进行优化，提升性能和效果表现。

关键词：实时渲染，流体模拟，水体渲染

Research and Implementation of Realistic Water Simulation based on Unreal Engine 5

Qiu Songtao( Computer Technology )

Directed by Xu Jie

# **ABSTRACT**

KEY WORDS: , ,

# 目 录

[第一章 绪论 1](#_Toc126921126)

[1.1 研究背景与意义 1](#_Toc126921127)

[1.2 研究现状与文献综述 2](#_Toc126921128)

[1.2.1 研究现状 2](#_Toc126921129)

[1.2.2 水体渲染相关案例分析 3](#_Toc126921130)

[1.2.3 文献综述 5](#_Toc126921131)

[1.3 本文主要工作 6](#_Toc126921132)

[1.4 本章小结 6](#_Toc126921133)

[第二章 流体模拟的理论基础和技术分析 7](#_Toc126921134)

[2.1 流体模拟理论基础 7](#_Toc126921135)

[2.1.1 Navier-Stokes方程 7](#_Toc126921136)

[2.1.2 基于拉格朗日视角的SPH方法 7](#_Toc126921137)

[2.1.3 基于拉格朗日视角的PBF方法 7](#_Toc126921138)

[2.1.4 基于混合视角的APIC方法 7](#_Toc126921139)

[2.2 Unreal Engine 5的Niagara粒子系统 7](#_Toc126921140)

[2.2.1 Niagara的基础使用 7](#_Toc126921141)

[2.2.2 Niagara中的空间加速结构 7](#_Toc126921142)

[2.2.3 Niagara中的Simulation Stage 7](#_Toc126921143)

[第三章 水体渲染系统的架构和实现方案 8](#_Toc126921144)

[3.1 水体渲染系统需求分析 8](#_Toc126921145)

[3.1.1 水体波形 8](#_Toc126921146)

[3.1.2 水体材质渲染 8](#_Toc126921147)

[3.1.3 场景交互设计 8](#_Toc126921148)

[3.2 水体渲染系统设计 8](#_Toc126921149)

[3.2.1 静态水体 8](#_Toc126921150)

[3.2.2 河流 8](#_Toc126921151)

[3.3 水体渲染系统具体实现 8](#_Toc126921152)

[3.3.1 粒子模拟 8](#_Toc126921153)

[3.3.2 水体渲染 8](#_Toc126921154)

[3.3.3 场景交互 8](#_Toc126921155)

[第四章 水体渲染系统的测试和性能优化 9](#_Toc126921156)

[4.1 测试方案设计 9](#_Toc126921157)

[4.1.1 算法性能测试 9](#_Toc126921158)

[4.2 系统优化 9](#_Toc126921159)

[4.2.1 渲染效率优化 9](#_Toc126921160)

[4.2.2 性能优化 9](#_Toc126921161)

[第五章 总结与展望 10](#_Toc126921162)

[5.1 项目总结 10](#_Toc126921163)

[5.2 未来展望 10](#_Toc126921164)

[参考文献 11](#_Toc126921165)

[附录 A 12](#_Toc126921166)

[致 谢 13](#_Toc126921167)

[北京大学学位论文原创性声明和使用授权说明 14](#_Toc126921168)

# 主要符号对照表

# 绪论

## 研究背景与意义

电子游戏伴随着信息产业的发展而诞生，至今已有数十年的历史。其发展至今，不仅仅成为了人们生活娱乐的方式，本身对文化的承载也让它成为了国家民族文化输出的载体。在我国，2022年游戏用户已达到了6.66亿人的规模，仅上半年就创造了1477.89亿元的销售收入[1]。电子游戏产业的不断发展和用户需求的不断变化也要求游戏工业技术的不断迭代。图形硬件和相关的图形开发技术与游戏产业相辅相成，共同促进，不断带给玩家更为逼真和精致的游戏画面。

现在的真实感渲染技术力求做到照片般的画面，采用了PBR（基于物理的渲染 ）技术，包括基于物理光照系统、基于物理的材质、基于物理的相机。而PBR所能构建的只是一个静态的世界。真实的世界不仅仅包括各种光照和材质效果，同时也包括和世界中各种物体运动的交互，所以基于物理的模拟也是让虚拟世界真正活起来的关键所在。水是生命之源，在真实世界中无处不在，发挥着重要作用，点缀着绚丽多彩的世界。所以在虚拟世界中，对于水体的渲染也是我们在还原真实世界的工作中必不可少的一部分。

水体渲染技术伴随着电子游戏技术的发展一同进步，目前市面上大多数游戏的水体渲染还是通过波形叠加对网格体进行偏移实现水波，然后编写相应的着色器实现水体效果，例如经典的Gerstner Wave方法。这一方法简单高效，但是在水体交互以及水流模拟上由于只能采用贴图绘制的方法而显得不够真实。基于流体力学的流体模拟技术则提供了另外一种水体渲染的方法，它通过流体动力学中的Navier-Stokes方程[2]，根据流体粒子之间的粘度、压力等物理特性，计算出粒子的速度场从而实现流体的模拟。这种方法在物理上是准确的，同时对水体交互也有很好的表现效果，但是由于有大量的粒子需要参与解算，性能上耗费很高。

得益于当前GPU和游戏引擎技术的发展，GPU的并行性得到了充分的利用，最新的DirectX12图形API可以实现CPU和GPU之间完全的并行，极大解放了GPU的算力。同时Unreal Engine（以下简称UE）提供了Niagara粒子系统，可以充分利用GPU强大的并行计算能力，实现各种复杂的粒子效果。这些都为本文的项目提供了坚实的技术基础。

UE发展至今已经迭代到了5.1版本，在渲染方面，最新的版本提供了诸如Lumen、Nanite等强大的功能特性，使得虚拟世界更加真实。同时它也更新了在4.26版本中加入的Niagara粒子系统，提供了最新的空间网格加速结构Neighbor Grid，极大加速了对粒子进行物理模拟的效率。

目前UE5.1版本提供了基于PIC/FLIP的流体模拟系统，但是由于FLIP算法性能耗费较高，导致无法在游戏项目中使用。同时，市面上的游戏水体渲染项目，基本以波形叠加和基于统计学模型的方法实现，在对水体流动的效果表现上不如流体模拟的方法。本项目的主要意义在于探索基于流体动力学的水体模拟和渲染技术在游戏项目中的运用，同时探索一个高性能的解决方案，让其可以较大规模地进行实装。同时本项目还会验证多种流体模拟算法在Niagara粒子系统中的性能表现，为以后的研发提供参考。

## 研究现状与文献综述

### 研究现状

水体渲染技术是计算机图形学中的一大分支，同时也是计算机图形学发展至今都在不断探索的研究方向。水体渲染技术主要涉及两个大方向，一是水体状态的模拟，例如波形的形成，水流的实现；另外就是水体材质的渲染，包括水体本身材质和水下后处理材质。近年来的很多研究主要集中在水体状态模拟以及如何加速渲染效率上。

Gerstner Wave理论最早于1804年被Gerstner F.J.提出，提供以摆线模拟周期性水波的方法[3]。 伴随着可编程渲染管线的发展与成熟，Laeuchli于2002年实现了根据Gerstner Wave计算水面波形的着色器。Gerstner Wave方法采用多个不同振幅、周期和频率的正弦波叠加的方式产生高度场。然后根据高度场计算出水面法线朝向，这样就可以完成水面的渲染。这种方法十分简单快速，适用于湖泊和简单的海面渲染，直到今天依然被大规模应用到各类游戏项目中。缺点在于它并非物理正确，对于河流等有具体流向的水体，无法表现流体的细节。目前UE中自带的水体系统使用的就是Gerstner Wave方法。

Fast Fourier Transform Ocean方法在SIGGRAPH 2001上由Jerry Tessendorf提出[4]。这种方法基于统计学模型，将三角函数所描述的海浪波形，包括振幅（波高）、频率（海洋运动速度）、相位（海洋的初相）由实数域通过欧拉公式转换到了复数域，从而可以用离散的复数域的点表示海洋波形，然后使用海洋波普和高斯噪声作为初始化的振幅。最后利用逆快速傅里叶变换就可以将波形从频域空间转换到时域空间，从而得到了随时间变化的海洋波形。这种方法因为需要实时进行计算，消耗较高，较早应用在离线渲染中。近年来由于实时渲染技术，特别是Compute Shader技术的发展，使得其得以在实时渲染项目中得到使用。它可以生成极为逼真的海浪波形，并具有十分优秀的动态效果。

基于流体动力学的不可压缩流体模拟通常由著名的Navier-Stokes方程来描述。求解该方程主要有两种方法，一种是基于网格的欧拉视角，另一种则是基于粒子的拉格朗日视角。对于水体模拟来说，一般使用基于粒子的拉格朗日视角来进行解算，例如光滑粒子动力学（Smoothed Particle Hydrodynamics，SPH）方法。它最早由Lucy、Gingold等人提出并应用于天体物理学领域[5]，后来被拓展到流体力学领域解决流体模拟问题[6,7]。它具有实现简单直观，材料边界划分准确，适用性广等优点。但是其效率不够高，同时存在数值计算不稳定的问题。针对它的缺点和不足，后续又有多种解决方案提出，其中在实时渲染中应用较多的则是基于位置的流体模拟方法（Postion Based Fluid，PBF）[8]方法，它来源于基于位置动力学方法（Position Based Dynamics，PBD）[9]。它通过约束实现粒子物理量的更新，可以有效解决模拟过程中的不稳定问题，同时允许较大的时间步长，进而可以提高实时模拟的效率。

目前游戏项目中使用最为广泛的还是Gerstner Wave方法，在一些大规模的水模拟中则会用到FFT方法。此外还有一些诸如波动粒子方法等[11]。而基于流体力学的水体模拟技术还没有在游戏中广泛使用。

### 水体渲染相关案例分析

对于工业化的游戏项目来说，优秀高效的实时水体方案是重中之重，通常他们会根据项目需求自行研发水体方案。但对于中小团队和个人开发者而言，使用市场已公开的解决方案并进行个性化定制更符合需求。

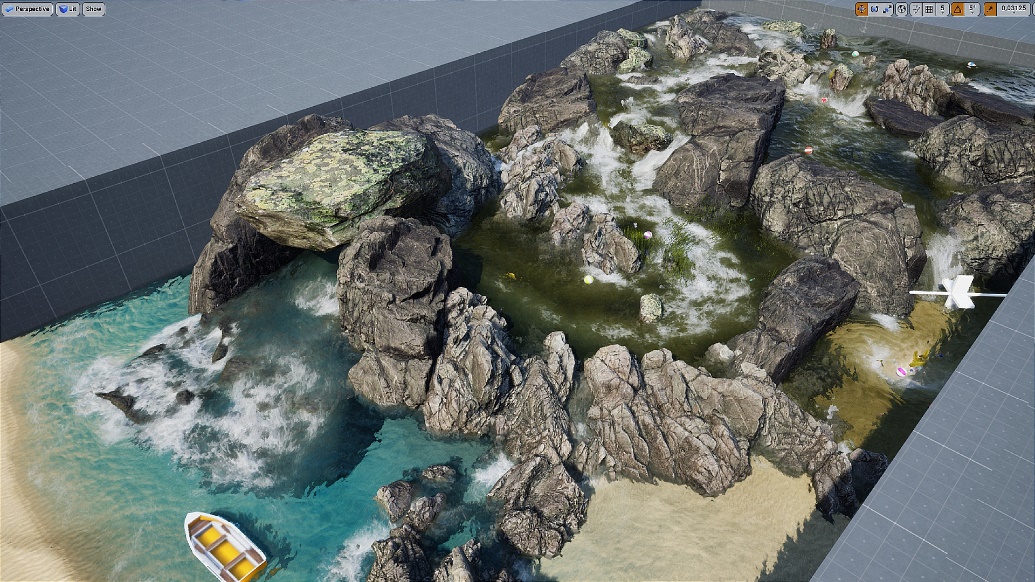
1. Fluid Flux[12]

图1.1 Fluid Flux示例关卡

该插件总体采用的是波形叠加的技术形成波形，实现重点在于大面积、实时的浅水区渲染，实时的水体交互，可烘焙成静态模型水面的工具等。它是目前市场上在水体渲染方面的标杆产品，集合了众多工具，方便开发者在任何地方都可以找到应用场景。

1. Waterline Pro[13]

图1.2 Waterline Pro

该插件是基于快速傅里叶变换实现的海洋模拟。主要功能包括：基于快速傅里叶变换的海洋波形实现；浅水模拟；水下和吃水线实现；浮力模拟；基于物理的水体材质等。基于以上功能，Waterline Pro提供了一套完整的实时海洋模拟方案，同时可以完美结合UE中的大气光照等功能，实现十分逼真的水体渲染效果。

1. UIWS(Unified Interactive Water System)[14]

图1.3 Unified Interactive Water System

不同于上面两款水体模拟项目，UIWS在波形生成方面比较简单一些，它着重于水体交互方面，提供了和场景、人物等碰撞交互功能。同时也提供了基于样条的瀑布和河流的实现。

以上三款都是在基于UE的水体渲染方向鼎鼎大名的水体模拟项目，它们均采取了比较成熟的基于高度图的波形生成方法。而基于流体力学模拟的水体渲染在UE中还并未开始流行，但是随着计算机硬件的发展，为该方法提供了基础支撑，同时它相比其他方法在物理上的准确性更高，可以得到精美的水体渲染效果。

### 文献综述

1. 《Fluid Simulation for Computer Graphics 2rd》[2]

本书是流体模拟的入门书籍。总共分为三个部分。第一部分介绍了流体模拟领域的基本方程，也就是Navier-Stokes方程的推导过程，以及欧拉视角和拉格朗日视角的求解方法。同时也介绍了数值模拟方法的原理和实现。第二部分则介绍了具体流体类型，如火、水等的模拟方法。第三部分介绍了涡方法、流体和固体的交互等算法。

本书深入浅出，由表及里地讲述了流体模拟的基础框架结构，十分适合入门流体模拟领域使用。同时其中也加入了很多优化方法和解决方案，例如水体表面张力的实现、海洋模型等。为读者研究流体模拟和理解相关概念提供了十分有利的参考。

1. Smoothed Particle Hydrodynamics Techniques for the Physics Based Simulation of Fluids and Solids[7]

该论文是SPH的入门文章，详细讲述了SPH方法的前世今生。前两章主要介绍了SPH的原理以及离散化形式的求解方法。第三章介绍了近邻搜索等空间加速结构。第四章主要介绍压力求解，以及各种不同的求解方法，例如预测矫正的SPH，Divergence-Free SPH等。第五、六、七三章则分别介绍了边界条件、粘度求解、表面张力求解的方法。最后则是对于一些特殊流体和弹性物体的模拟。

该论文介绍的基础SPH方法原理简单，方便在UE中使用Niagara进行部署，搭建了流体模拟从理论到实践的桥梁。

1. Position Based Fluids[8]

该文章详细讲述了PBF中对不可压缩项、表面张力和粘度项的解决方法。同时给出了对比实验结果。PBF方法是基于PBD的流体模拟方法，它和SPH方法不同之处在于使用约束来实现对压力、粘度、密度等物理量的更新。弥补了SPH方法在交互反馈上的弱势。这也是本项目着重实验和使用的方法之一。

1. 《虚拟现实引擎特效制作》[10]

游戏特效中对于粒子效果的模拟是不可或缺的一部分，流体的模拟也可以用粒子实现。所以对于UE中粒子系统的学习和研究也是十分有必要的一环。

本书主要介绍了UE中Cascade和Niagara两种特效系统的基础使用方法。提供了包括力场特效、云层、角色粒子等特效案例，方便读者深入研究和理解Niagara粒子系统。

## 本文主要工作

本文的主要工作有：

1. 对实现基于流体模拟的水体渲染所需要的相关技术进行研究学习并分析游戏项目中的需求。主要包括利用UE进行游戏开发以及Niagara粒子系统相关技术。
2. 基于Niagara实现各种流体模拟算法，进行比较和测试，找到了适合实时交互的流体模拟、材质渲染、场景交互的方法。通过对各种算法的测试和改造，提升渲染和模拟的效率，使得可以在当前硬件水平流畅运行。
3. 利用UE的插件机制，部署水体渲染插件，并根据用户需求提供相关可调节参数。同时搭建示例关卡，展示水体模拟的不同使用场景。
4. 在项目中暴露的性能问题和渲染问题进行修复和调优。完成UE蓝图和特效系统的协作。

## 本章小结

本章主要讨论了水体渲染技术的背景，以及在游戏中的应用现状。不论是写实画风或者风格化的游戏项目，抑或其他实时渲染项目，水体渲染都是不可或缺的一环。基于流体力学的模拟方法在离线渲染中因其优异的视觉效果从而得到了广泛的应用，但在实时渲染中由于性能原因并未普及。但随着计算机软硬件技术的发展，在游戏中部署基于流体力学的水体渲染的时机越来越成熟，所以目前对于此方面的探索和开发也是十分有必要的。随后本章对现有的水体渲染技术方法，以及目前较为流行的水体渲染方案进行了探讨和论述，最后对主要参考文献进行了介绍。下一章将具体介绍流体模拟的理论基础和实现的技术基础。

# 流体模拟的理论基础和技术分析

## 流体模拟理论基础

### Navier-Stokes方程

### 基于拉格朗日视角的SPH方法

### 基于拉格朗日视角的PBF方法

### 基于混合视角的APIC方法

## 基础技术简介

### Unreal Engine

### HLSL着色器语言

### UAV

## Unreal Engine 5的Niagara粒子系统

### Niagara概述

Niagara粒子系统是UE最新最强大的特效系统，它提供了一套完整灵活的接口，封装了GPU并行编程的底层逻辑和数据结构，方便美术人员在不知道底层逻辑的情况下可以编写各种精美的视觉特效。同时得益于GPU的强大并行性，它还可以模拟海量的粒子运动，对其进行物理解算，可以模拟诸如布料、群集运动、流体、弹性体等真实世界中的各类物体。

Niagara的核心组件包括：系统、发射器、模块、参数[15]。

1. 系统

一个Niagara系统是一个容器，里面可以放入各种要添加到容器中的效果，也就是发射器。同时可以在系统层级对系统内所有的发射器进行一种全局的控制，主要包括参数传递和生命周期管理等。

图2. Niagara系统

1. 发射器

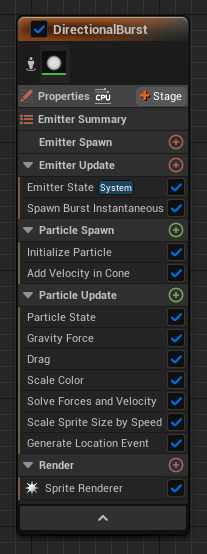
发射器是Niagara的主体单元，用来在Niagara中生成粒子，并控制粒子的生成、粒子在生命周期中的外观及行为。它采用类似瀑布流的方式组成堆栈，每个堆栈中可能有几个组，组中可以放入各个任务的实现模块。

图2. Unreal引擎官方Directional Burst发射器示例

其中比较重要的有一下几个堆栈：

1. 发射器生成（Emitter Spawn）

这个单元定义了在CPU上首次创建发射器运行的功能。一般用于一些值的初始化。

1. 发射器更新（Emitter Update）

这个单元定义了CPU上每一帧发射器模块的行为。通常用来定义粒子的产生方式。

1. 粒子生成（Particle Spawn）

当每个粒子生成时会调用这个单元对自身的属性进行初始化。例如粒子的颜色，生成位置，大小，以及其他特征。

1. 粒子更新（Particle Update）

每一帧上的每个粒子都会调用粒子更新模块。可以在此处定义粒子生命周期内的会逐渐变化的所有行为。例如粒子的颜色随时间推移的变化。或者粒子受到各种力的影响，例如重力、引力等。

1. 事件处理器（Event Handler）

在事件处理器中，可以定义一个监听状态，然后在其他的发射器中触发状态。常用的比如Generate Location Event会记录当前粒子的位置状态，其他发射器可以读取该发射器的粒子位置用于一些诸如跟随、索引等操作。

1. 渲染（Render）

渲染模块定义了粒子的显示方式。可以在这里为粒子设置一个或多个渲染器。最新的UE5.1中渲染模块包括Sprite Renderer、Ribbon Renderer、Component Renderer、Mesh Renderer、Light Renderer、Geometry Cache Renderer。它们可以渲染出各种需要的粒子显示内容。

1. 模块

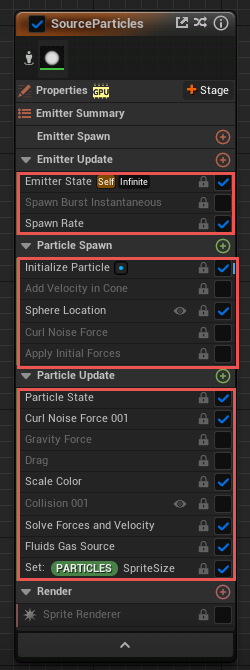
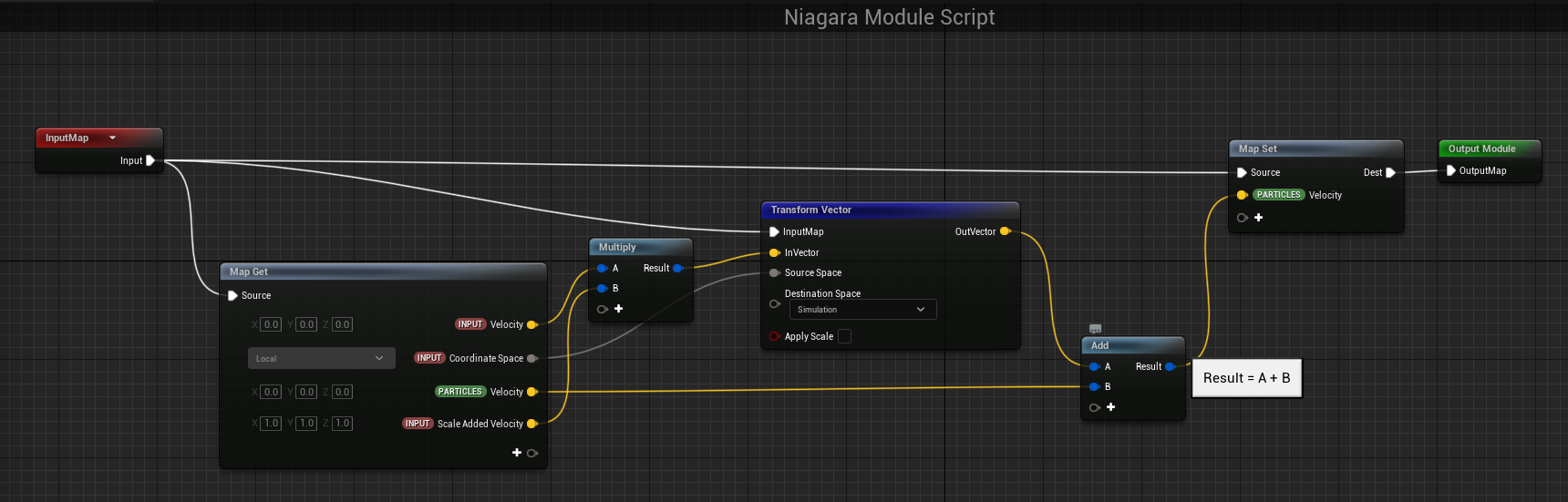
模块是Niagara效果实现的基础模块。可以将各类模块添加到组中构成堆栈。类似堆栈的结构，模块也是自上而下形成瀑布流的方式来运行。

图2. Niagara中的模块

模块可以看作一种运算器，接收一些变量，然后计算并输出，同时数据也可以在模块间传递。模块本身时使用高级着色器语言HLSL进行构建的，UE内部提供了一套图形化编程的接口来方便技术美术人员或者其他不了解编程的人员使用。在模块中除了UE提供的原生节点，还可以在CustomHLSL节点中内嵌HLSL以实现更高的自由度。

图2. Add Velocity模块

在运行时，模块脚本首先检索输入的速度，然后变换到正确的坐标空间最后更新到粒子的速度上。所有的模块都按照这种方式进行构建，只是复杂度不同。

1. 参数

Niagara中的参数是参与运算的单元，是数据的抽象化表现。参数主要分为四种类型：图元、枚举、结构体、数据接口。其中最常用的是图元类型，它表示具有各种精度和宽度的数值数据。数据接口则比较特殊，通常表示UE外部传入的数据源，比如Render Target等。

### Niagara的Simulation Stage

Simulation Stage是Unreal Engine 4中推出的试验性功能，它极大加强了粒子系统的灵活性，是整个粒子系统执行顺序和逻辑的核心。

在上一小节中我们提到，粒子核心逻辑在Particle Update中，在这个堆栈里，粒子每帧按照从上到下的顺序执行Update中的模块，更新自身的速度、大小、颜色等属性。这个过程是分批并行的，每一批粒子中的所有粒子同时从第一个模块开始向下执行。由于系统内部调度和算法的复杂度不同，我们无法得知哪个粒子先完成了当前模块的计算进入下一个模块。这也意味着，并行运算的粒子之间在同一帧是无法交叉影响的，每个粒子只能读取Emitter、System或者自己的属性，并对其进行修改，而无法得知其他粒子的属性。虽然在Niagara中可以通过Attribute Reader获取其他粒子的属性，并根据得到的数据进行运算，但由于Particle Update是每帧执行的，我们无法控制每个粒子的先后执行顺序，所以通过Attribute Reader获取到的是粒子在当前帧的初始值，在同一帧内如果粒子A要读取粒子B经过运算后的属性是做不到的。这也就是Particle Update的局限性所在了，由此UE引入了Simulation Stage。

根据上面的论述，我们需要在同一帧下，所有的粒子执行完模块A的计算后更新自身属性，然后我们利用更新的属性来计算模块B，当所有的模块执行完成后，我们才更新当前帧的所有粒子属性。这就是Simulation Stage的作用。

我们可以向一个Emitter中添加多个Simulation Stage，每个Simulation Stage类似一个同步节点，所有粒子执行完一个Simulation Stage前必须等待上一个Simulation Stage结束。所以事实上Simulation Stage会存在一些性能上的损失，所以只有需要运行的逻辑对当前帧其他粒子的属性有依赖时才会使用，例如水体模拟中对压力的解算就需要其他粒子在当前帧的状态，所以需要用Simulation Stage。

之前说到Simulation Stage可以读取当前帧的粒子状态，这个状态的存储或者缓存是通过Niagara中加入的Grid结构实现的，这个结构可以在任意位置读写，而且可以在Simulation Stage内进行迭代更新。

### Niagara的缓存和空间加速结构

# 水体渲染系统的架构和实现方案

## 水体渲染系统需求分析

### 水体波形

### 水体材质渲染

### 场景交互设计

## 水体渲染系统设计

### 静态水体

### 河流

## 水体渲染系统具体实现

### 粒子模拟

### 水体渲染

### 场景交互

# 水体渲染系统的测试和性能优化

## 测试方案设计

### 算法性能测试

## 系统优化

### 渲染效率优化

### 性能优化

# 总结与展望

## 项目总结

## 未来展望

# 参考文献

1. 2022年1-6月中国游戏产业报告,2022
2. Bridson, Robert. Fluid simulation for computer graphics. CRC Press, 2015
3. [Gerstner, F.J.](https://en.wikipedia.org/wiki/Franti%C5%A1ek_Josef_Gerstner" \o "František Josef Gerstner) "Theorie der Wellen", Abhandlunger der Königlichen Böhmischen Geselschaft der Wissenschaften, Prague. Reprinted in: Annalen der Physik 32(8), pp. 412–445, 1809
4. Jerry Tessendorf. Simulation Ocean Water, SIGGRAPH 2001
5. Lucy L B. A numerical approach to the testing of the fission hypothesis. The Astrophysical Journal. 1977, 8(12): 1013-1024.
6. Matthias Müller, David Charypar and Markus Gross. Particle-Based Fluid Simulation for Interactive Applications. Eurographics/SIGGRAPH Symposium on Computer Animation (2003)
7. Koschier D, Bender J, Solenthaler B, and Teschner M. Smoothed Particle Hydrodynamics Techniques for the Physics Based Simulation of Fluids and Solids. In: Eurographics Proceedings. Tutorials. 2019
8. Miles Macklin, Matthias Muller. Position Based Fluids. ACM Transcations on Graphics(TOG) 32.4 (2013):104
9. Matthias Müller Bruno Heidelberger Marcus Hennix John Ratcliff. Position Based Dynamics. 3rd Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2006)
10. 刘跃军, 龙姝羽. 虚拟现实引擎特效制作.
11. 浅墨. 真实感水体渲染技术总结. [EB/OL]. [https://zhuanlan.zhihu.com/p/95917609. 2021](https://zhuanlan.zhihu.com/p/95917609.%202021)
12. Epic Game Marketplace. Fluid Flux. [EB/OL]. <https://www.unrealengine.com/marketplace/zh-CN/product/fluid-flux>
13. Epic Game Marketplace. Waterline Pro. [EB/OL]. <https://www.unrealengine.com/marketplace/zh-CN/product/waterline>
14. Epic Game Marketplace. UIWS-Unified Interactive Water System. [EB/OL]. <https://www.unrealengine.com/marketplace/zh-CN/product/uiws-unified-interactive-water-system?sessionInvalidated=true>
15. Epic Document. Niagara概览. [EB/OL]. https://docs.unrealengine.com/4.27/zh-CN/RenderingAndGraphics/Niagara/Overview/



# 致 谢

# 北京大学学位论文原创性声明和使用授权说明

**原创性声明**

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品或成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名： 日期： 年 月 日

**学位论文使用授权说明**

本人完全了解北京大学关于收集、保存、使用学位论文的规定，即：

* 按照学校要求提交学位论文的印刷本和电子版本；
* 学校有权保存学位论文的印刷本和电子版，并提供目录检索与阅览服务，在校园网上提供服务；
* 学校可以采用影印、缩印、数字化或其它复制手段保存论文；
* 因某种特殊原因需要延迟发布学位论文电子版，授权学校□一年/□两年/□三年以后，在校园网上全文发布。

（保密论文在解密后遵守此规定）

论文作者签名： 导师签名：

日期： 年 月 日