Yiwei Tu

CSSSKL 594

# METHODOLOGY/SYSTEM DESIGN

This section presents the requirements of the proposed watercolor painting application, identified from literature analysis, technologies selected to accomplish the application and the structure of the application.

1. Application Requirements

As stated in the Chapter 1 (introduction), the deliverables of this project is a functional real-time watercolor painting application. At the application layer, users should be able to choose different colors, the size of brush, and control the amount of pigments on the brush to draw paintings on a digital canvas. The digital canvas should be able to simulate watercolor pigment fluid in real time and produce the watercolor patterns. Therefore, the application should:

* Render watercolor patterns in real-time.
* Allow modification on painting settings.
* Implement user interface for interactively drawing activities.
* Display watercolor patterns generated by algorithms.

1. Implementation platform - Unity3D

To meet above application-level requirements, a few technical conditions should be considered while choosing an implementation platform. More specifically, the platform should:

* Consist of rendering components to visualize watercolor patterns
* Support UI components for user interactions.

The platform chosen to build the watercolor painting application is Unity3D. Unity3D is a real-time development platform, which helps create and grow real-time 3D games, apps, and experiences for entertainment, film, automotive, architecture, and more.

Reasons for choosing Unity:

* Support ready-to-use real-time rendering solution and UI components
* User-friendly coding interface, and straightforward architecture for real-time application development
* Used in the previous course works → familiarity
* Unity supports user-defined HLSL shaders and multi-pass rendering technique, which helps move the computation stress from the CPU to the GPU and improve the performance.

Alternatives:

* OpenGL: a fast, low-level, and cross-platform graphics library to render 3D graphics
  + Intensive foundation building and system configuration required in order to build a 3D rendering pipeline and UI
* GIMP: GNU Image Manipulation Program, a freely distributed program for such tasks as photo retouching, image composition and image authoring
  + But it is likely to be used like an image converter for watercolor stylization
* Processing: a free graphic library and integrated development environment built for the electronic arts, new media art, and visual design communities with the purpose of teaching non-programmers the fundamentals of computer programming in a visual context.
  + All computation work on the CPU side

1. Fluid Simulation model - Lattice Boltzmann equations (LBE) vs. traditional Navier-Stokes (N-S) equations

Numerical fluid simulations have been used by graphics experts in the last ten years to increase the fidelity of computer-generated animations. They often adopt the conventional methodology, beginning with the Navier-Stokes (N-S) equations, which provide a macroscopic description of the fluid. Given how challenging it is to solve the N-S equations, numerous approximate numerical approaches have been put forth. The lattice Boltzmann method, which has its origins in Ludwig Boltzmann's 1872 Boltzmann equation, is the technique utilized in this project to simulate fluid dynamics. The Boltzmann equation uses kinetic theory to explain the microscopic behavior of a gas. In a single-particle phase space, it provides the statistical distribution of the particles. The main idea of the LBE approach is to model the fluid dynamics using a simplified particle kinetic model.

Introduction on the LBE

* Introduction on lattice - diagram, symbol meaning e0 - e8, directions
* Particle distribution function within a lattice -> density and velocity equations based on the distribution functions
* Streaming step in the LBE - diagram and equations
* Collision step in the LBE - diagram and equations

Why LBE?

1. It does not involve Poisson equations
2. All operations are simple and local
3. It is easy to incorporate physics that is hard to describe macroscopically
4. Kubelka-Munk Reflectance Model

In order to display the color variance on the watercolor painting generated by the computer, I used the Kubelka-Munk Reflectance (KM) model. The KM model is a physically based model that simulates the scattering and absorption of light by materials. There is an alternative approach to rendering the resulting watercolor artwork, employing the CMY color system to calculate the resulting color. However, CMY color synthesis works best for purely transmitting materials, and pigmented surfaces have both transmitting and reflecting characteristics. Therefore, the KM model, which calculates the transmittance and reflectance of the colored layer, is proposed to calculate the illumination of colorful pigment.

Introduction on the KM model

* Define absorption coefficient K and scattering coefficient S using user-defined values Rw and Rb
* Demonstrate the calculation of Transmittance T and Reflectance R of one layer

1. Programming on GPU