Simulation in Computer Graphics TamiFlu

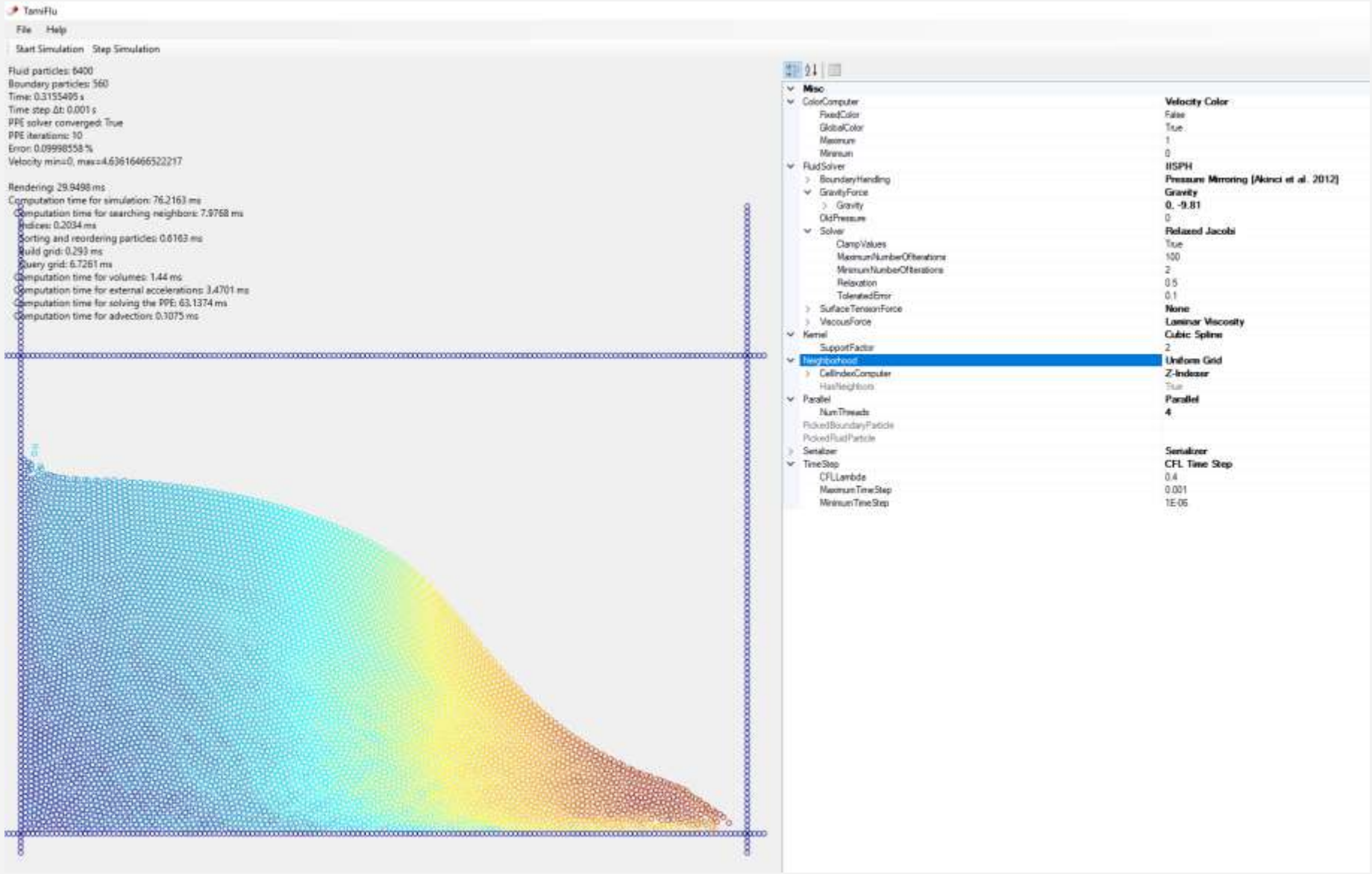
Matthias Teschner

UNI FREIBURG

Overview

- 2D fluid simulation framework
  - Written in C# / .NET 4.7.1
- Prerequisites
  - C# compiler, e.g. Microsoft Visual Studio 2017
- Author
  - Stefan Band

Screenshot


TamFlu
File Help
Start Simulation Step Simulation
Fluid particles: 6400
Boundary particles: 560
Time: 0.3155495 s
Time step Δt: 0.001 s
PPE solver converged: True
PPE iterations: 10
Error: 0.09998558 %
Velocity min=0, max=4.63616466522217
Rendering: 29.9498 ms
Computation time for simulation: 76.2163 ms
Computation time for searching neighbors: 7.9768 ms
Indices: 0.2034 ms
Sorting and reordering particles: 0.6163 ms
Build grid: 0.293 ms
Query grid: 6.7261 ms
Computation time for volumes: 1.44 ms
Computation time for external accelerations: 3.4701 ms
Computation time for solving the PPE: 63.1374 ms
Computation time for advection: 0.1075 ms
Misc
ColorComputer
Velocity Color
FixedColor
False
GlobalColor
True
Maximum
1
Minimum
0
FluidSolver
IISPH
BoundaryHandling
Pressure Mirroring [Akinci et al. 2012]
GravityForce
Gravity
Gravity
0, -9.81
OldPressure
0
Solver
Relaxed Jacobi
ClampValues
True
MaximumNumberOfIterations
100
MinimumNumberOfIterations
2
Relaxation
0.5
ToleratedError
0.1
SurfaceTensionForce
None
ViscousForce
Laminar Viscosity
Kernel
Cubic Spline
SupportFactor
2
Neighborhood
Uniform Grid
CellIndexComputer
Z-Indexer
HasNeighbors
True
Parallel
Parallel
NumThreads
4
PickedBoundaryParticle
PickedFluidParticle
Serializer
Serializer
TimeStep
CFL Time Step
CFLLambda
0.4
MaximumTimeStep
0.001
MinimumTimeStep
1E-05

SPH Fluid Solver

IFluidSolver

void Simulate(IParticleContext)

for all *particle* i **do**
find neighbors j

for all *particle* i **do**

$$\rho_i = \sum_j m_j W_{ij}$$
$$p_i = k(\frac{\rho_i}{\rho_0} - 1)$$

for all *particle* i **do**

$$\mathbf{a}_i^{\text{nonp}} = \nu \nabla^2 \mathbf{v}_i + \mathbf{g}$$
$$\mathbf{a}_i^{\text{p}} = -\frac{1}{\rho_i} \nabla p_i$$
$$\mathbf{a}_i(t) = \mathbf{a}_i^{\text{nonp}} + \mathbf{a}_i^{\text{p}}$$

for all *particle* i **do**

$$\mathbf{v}_i(t + \Delta t) = \mathbf{v}_i(t) + \Delta t \mathbf{a}_i(t)$$
$$\mathbf{x}_i(t + \Delta t) = \mathbf{x}_i(t) + \Delta t \mathbf{v}_i(t + \Delta t)$$

Neighbor Search

IParticleNeighborhood

void SearchNeighbors()

for all *particle* i **do**
find neighbors j

for all *particle* i **do**

$$\rho_i = \sum_j m_j W_{ij}$$
$$p_i = k(\frac{\rho_i}{\rho_0} - 1)$$

for all *particle* i **do**

$$\mathbf{a}_i^{\text{nonp}} = \nu \nabla^2 \mathbf{v}_i + \mathbf{g}$$
$$\mathbf{a}_i^{\text{p}} = -\frac{1}{\rho_i} \nabla p_i$$
$$\mathbf{a}_i(t) = \mathbf{a}_i^{\text{nonp}} + \mathbf{a}_i^{\text{p}}$$

for all *particle* i **do**

$$\mathbf{v}_i(t + \Delta t) = \mathbf{v}_i(t) + \Delta t \mathbf{a}_i(t)$$
$$\mathbf{x}_i(t + \Delta t) = \mathbf{x}_i(t) + \Delta t \mathbf{v}_i(t + \Delta t)$$

Pressure Force

IPressureForce

void ApplyToFluidParticles()

for all *particle* i **do**
find neighbors j
for all *particle* i **do**
$\rho_i = \sum_j m_j W_{ij}$
$p_i = k(\frac{\rho_i}{\rho_0} - 1)$
for all *particle* i **do**
$\mathbf{a}_i^{\text{nonp}} = \nu \nabla^2 \mathbf{v}_i + \mathbf{g}$
$\mathbf{a}_i^{\text{p}} = -\frac{1}{\rho_i} \nabla p_i$
$\mathbf{a}_i(t) = \mathbf{a}_i^{\text{nonp}} + \mathbf{a}_i^{\text{p}}$
for all *particle* i **do**
$\mathbf{v}_i(t + \Delta t) = \mathbf{v}_i(t) + \Delta t \mathbf{a}_i(t)$
$\mathbf{x}_i(t + \Delta t) = \mathbf{x}_i(t) + \Delta t \mathbf{v}_i(t + \Delta t)$

External Forces

IExternalForce

void ApplyToFluidParticles()

for all *particle* i **do**
find neighbors j
for all *particle* i **do**
$\rho_i = \sum_j m_j W_{ij}$
$p_i = k(\frac{\rho_i}{\rho_0} - 1)$
for all *particle* i **do**
$\mathbf{a}_i^{\text{nonp}} = \nu \nabla^2 \mathbf{v}_i + \mathbf{g}$
$\mathbf{a}_i^{\text{p}} = -\frac{1}{\rho_i} \nabla p_i$
$\mathbf{a}_i(t) = \mathbf{a}_i^{\text{nonp}} + \mathbf{a}_i^{\text{p}}$
for all *particle* i **do**
$\mathbf{v}_i(t + \Delta t) = \mathbf{v}_i(t) + \Delta t \mathbf{a}_i(t)$
$\mathbf{x}_i(t + \Delta t) = \mathbf{x}_i(t) + \Delta t \mathbf{v}_i(t + \Delta t)$

TamiFlu – Fluid Solver Step

```
particleContext.Neighborhood.SearchNeighbors(particleContext);

particleContext.ForEachFluidParticleInParallel((ref FluidParticle f) => f.Pressure
= stateEquation.ComputePressure(f.Density,f.Properties.RestDensity));

particleContext.ForEachFluidParticleInParallel((ref FluidParticle f) =>
f.Acceleration = Vector.Zero);
GravityForce.ApplyToFluidParticles(particleContext);
ViscousForce.ApplyToFluidParticles(particleContext);
SurfaceTensionForce.ApplyToFluidParticles(particleContext);
PressureForce.ApplyToFluidParticles(particleContext);

particleContext.ForEachFluidParticleInParallel((ref FluidParticle f) =>
{ f.Velocity += timeStepValue * f.Acceleration;
  f.Position += timeStepValue * f.Velocity;});
```

TamiFlu – Boundary Handling

```
particleContext.ForEachFluidParticleInParallel((ref FluidParticle f) => {

   var numberDensity = selfKernelValue;

   foreach (var ffIndex in f.FluidNeighbors) {
      ref readonly var ff = ref fluidParticles[ffIndex];
      numberDensity += kernel.ComputeValue(f.Position, ff.Position); }

   foreach (var fbIndex in f.BoundaryNeighbors) {
      ref readonly var fb = ref boundaryParticles[fbIndex];
      numberDensity+=(fb.Volume/restVolume)*kernel.ComputeValue(f.Position,fb.Position); }

   f.Volume = 1f / numberDensity; });
```

SPlisHSPlasH

- SPH Framework
- Author: Jan Bender, RWTH Aachen
- https://github.com/InteractiveComputerGraphics/SPlisHSPlasH