PhysBAM: Physically Based Simulation
Michael Lentine, Craig Schroeder
Outline

• The PhysBAM Release
  • What is PhysBAM
  • How to get and use PhysBAM

• Algorithms
  • Levelset
  • Water
Outline

- PhysBAM Overview
- PhysBAM_Tools
- PhysBAM_Geometry
- Running PhysBAM
- Viewing the results
- Make your own simulation
What is PhysBAM?

- Code package used for physical simulation
- Developed at Stanford
  - Industry Collaboration
- Handles many types of simulations
  - Solids
  - Fluids
  - Coupling
What is PhysBAM?

- Used in a wide range of academic papers
  - [Lentine et al. 2011; Schroeder et al. 2011; Lentine et al. 2011; Kwatra et al. 2010; Lentine et al 2010; etc...]
  - 18 SIGGRAPH Papers, 14 SCA Papers, etc...
- Used in a number of corporations
  - Industrial Light & Magic, Pixar Animation Studios, Walt Disney Animation, Intel Corp.
PhysBAM Release

• PhysBAM Tools was released in the winter
  • Smoke
  • OpenGL
  • Ray Tracing

• PhysBAM Geometry will be released during SIGGRAPH 2011
  • Levelsets
  • Surfaces
PhysBAM Structure

- PhysBAM
  - Projects
    - Advection
    - Projection
    - Smoke
    - Water
  - Public_Library
  - Scripts
    - PhysBAM_Tools
    - PhysBAM_Geometry
PhysBAM Tools

- Core set of tools for PhysBAM
- Data Structures
  - Arrays, Hashtables
- Math primitives
  - Matrices, Vectors, Quaternions
- Math Tools
  - Linear solvers, Optimization algorithms
PhysBAM Tools

- Core set of tools for PhysBAM
- Simulation objects
  - Grids, Particles
- Simulation algorithms
  - Advection, Interpolation
- Convenience tools
  - Parsing, I/O, Debugging
PhysBAM Geometry

- Builds on the tools library
- Simple primitives
  - Spheres, Boxes, Cylinders
- Meshes
  - Triangulated surfaces, Tetrahedralized volumes
- Acceleration Structures
  - Hierarchies, Partitions
PhysBAM Geometry

- Builds on the tools library
- Extended simulation objects
  - Rigid Bodies, Deformable Objects, Levelsets
- Extended simulation algorithms
  - Rasterization, Intersection, Collision Detection
Downloading PhysBAM

- Downloading the source
  - http://physbam.stanford.edu/
- Required prerequisites
  - GCC 4.5.2
  - Scons
- Optional prerequisites
  - libpng, libjpeg
  - libgl, libglu, libglut
  - libz, libpthread, libmpi
Downloading PhysBAM

- Create directory $PHYSBAM
- Download the library into this directory
- Extract
  - Public_Library
  - Projects
  - Scripts
Compiling PhysBAM

- Use Scons
  - python Scripts/scons/setup_scons.py
  - $PHYSBAM/Sconstruct
  - $PHYSBAM/Sconstruct.options
- Each project has a Scons script
- `scons -Q -u -j #procs CXX="/usr/bin/g++"`
Compiling PhysBAM

- Environment vars
  - PLATFORM=nocona (64bit)
  - PLATFORM=pentium4 (32bit)

- Build options
  - shared
  - shared_objects
Compiling PhysBAM

• Compile Flags
  • Parallelism
    – USE_MPI, USE_LAM, USE_PTHREADS
  • Formats
    – USE_LIBPNG, USE_LIBJPEG, USE_FFMPEG
  • Others
    – USE_FFTW, USE_LIBZ
An Example

• Download the Smoke project
  • http://physbam.stanford.edu/~mlentine/smoke.tar.gz
  • Extract inside $PHYSBAM/Projects
  • Comes with INSTALL file
• cd $PHYSBAM/Projects/Smoke
• scons -Q -u -j 2 CXX="/usr/bin/g++"
• Creates smoke_{nocona,pentium4}
• Build files are in $PHYSBAM/build
Running PhysBAM

• Usage instructions
  • Projects come with README files
  • -h will display help
  • -scale <int> sets the resolution
  • -2d, -3d sets the dimensions
  • -e <int> sets the last frame
  • -restart <int> sets a frame to start from
  • -substeps <int> outputs data every step
An Example

- ./smoke_pentium4
- Default parameters
  - Resolution is 50x100
    - 3d is 50x100x50
  - Starts from frame 1
  - Ends at frame 100
An Example

- Writes to directory \$PHYSBAM/Smoke/output
  - A directory for each frame with frame specific data
    - \texttt{mac\_velocity}, density
  - A common directory for all frames
    - grid
Projects

- Tools projects
  - Smoke
  - Projection
  - Advection
- Geometry Projects
  - Water
  - Levelset
Viewing the Results

- View the results from the output directory
- OpenGL viewers
  - opengl_1d
  - opengl_2d
  - opengl_3d
- Ray tracer
  - Only works in 3d
OpenGL

- Three independent viewers
2D Viewer
3D Viewer
3D Viewer
Commands

- **Playing**
  - “p” play
  - “s” step
  - “ctrl+s” step back
  - “r” restart
  - “z” end
  - “g” goto
  - “ctrl+d” capture*
Commands

- **Visualization**
  - “V” show velocity
  - “d” show smoke
  - “6” show grid
  - “-” decrease velocity size
  - “=” increase velocity size
  - “ctrl+h” slice mode
  - “[“ and “]” increment and decrement slices
  - “\” change slice axis
An Example

- ./opengl_1d_pentium4
- ./opengl_2d_pentium4
- ./opengl_3d_pentium4

Takes in directory as argument
  - Use directory $PHYSBAM/Projects/Smoke/output
Ray Tracing

- OpenGL is good for testing/debugging
- Ray tracing produces better results
  - Higher quality images
  - Slower
- Uses a scene file to draw objects
Ray Tracing
Ray Tracing
Scene File

- Set of general options
  - Sky, rays per pixel
- Camera
- Lights
  - Point lights, area lights, spotlights
- Materials
- Shaders
Camera

```
Camera{
    Location=[x y z]
    Look_At=[x y z]
    Pseudo_Up=[x y z]

    Field_Of_View=d
    Focal_Distance=d
    Aspect_Ratio=d

    Width=i
    Height=i
    Output_Filename="str"
}
```
Point Light

Light{
    Name="str"
    Type="Point"
    Color=[r g b]
    Position=[x y z]
    Power=i
    Casts_Shadows=b
}
Light{
    Name="str"
    Type="Spotlight"
    Color=[r g b]
    Position=[x y z]
    Power=i
    Casts_Shadows=b
    Direction=[x y z]
    Cone_Angle=a
    Penumbra_Angle=a
}
Scene File

• Simulation objects
  • Deformable bodies
  • Rigid bodies
  • Levelsets
  • Voxel data
Object{
    Name="str"
    Type="Voxel_Data"
    Grid_Filename="output/common/grid"
    Density_Filename="output/%d/density"
    Volume_Shader="SmokeShader"
    Volume_Step=d
}

Voxel Data
Deformable Object

List_Object{
  Name= "str"
  Type= "Deformable_Object"
  Prefix= "simulation/path"
  Shader= "Shadername"
  Smooth_Normals= b
  Preserve_Creases= b
  Range= 1-5
}

Rigid Bodies

List_Object{
  Name="str"
  Type="Rigid_Body_List"
  Prefix="simulation/path"
  Shader="ShaderName"
  Smooth_Normals=b
  Subdivide_Geometry=b
  Preserve_Creases=b
  Range=1,3-10
}

Scene File

- Other objects
  - Meshes
  - Cylinders
  - Spheres
- Unsimulated objects
Sphere

Object{
    Name="str"
    Type="Sphere"
    Position=[x y z]
    Radius=d
    Shader="ShaderName"
}

Triangulated Surface

Object{
    Name="str"
    Type="Triangulated_Surface"
    Smooth_Normals=b
    Filename="surface.tri.gz"
    Shader="ShaderName"
}

Scene File

• Shaders
  • Colors
  • Textures
  • Transparency
  • Reflection
  • Blending
Color Shader

Material{
    Name="str"
    Type="Color"
    Color= [r g b]
}

Lambertian Shader

Material{
   Name="str"
   Type="Lambertian"
   Shader="ShaderName"
   Ambient=[r g b]
   Reflectivity=d
}

Phong Shader

Material{
    Name="str"
    Type="Phong"
    Shader="ShaderName"
    Diffuse=[r g b]
    Specular=[r g b]
    Specular_Exponent=d
}

Voxel Shader

Volume_Material{
    Name="SmokeShader"
    Type="Voxel_Shader"
    Absorption=i
    Scattering=i
    Inscattering_Amplification=i
}

Transparency and Reflection

Material{
   Name="str"
   Type="Transparent"
   Reflectivity=d
}

Blend Shader

Material{
    Name="str"
    Type="Blend"
    Shader1="ShaderName1"
    Shader2="ShaderName2"
    Blend_Fraction=d
}

An Example

- `./ray_tracing_pentium4 <scene> <frame>`
- Use the scene file for smoke
  - Voxel data points to the output directory
  - Output path is relative
- Generates `frame.<frame>.png`
Make Your Own

• Can now download, run, view data
• Want to change simulation
• PhysBAM structure
  • main.cpp
  • TYPE_EXAMPLE.{h,cpp}
  • TYPE_DRIVER.{h,cpp}
main.cpp

- Parse input
- Creates Example
  - Sets parameters
- Creates Driver
  - Runs simulation
TYPE_EXAMPLE

- Stores Variables
  - Smoke density, velocity
- Adds Callbacks
  - Get_Boundary_Conditions
TYPE_DRIVER

- Executes simulation
  - Advection
  - Forces
  - Projection
How to Add a Sphere

- Files to modify
  - SMOKE_EXAMPLE{.h,.cpp}
- Use a SPHERE<TV> object
- Works with other geometry
How to Simulate Particles

- Files to modify
  - SMOKE_EXAMPLE{.h,.cpp}
  - SMOKE_DRIVER.cpp
- Stored as an array of positions
How to Modify the viewer

- How to add visualization
- Files to modify
  - main.cpp

Can use OPENGL_COMPONENTS
How to Add Particles
How to Add Particles
PhysBAM

- PhysBAM is a code package for simulation
  - http://physbam.stanford.edu/
- Tools and Geometry are released
  - Smoke
  - Water
  - Opengl viewing
  - Ray Tracing
Whats Next?

- Portability
  - Tablets, Phones
- Functionality
  - Meshing
  - Rigids
  - Deformables
Thank You!