Vulkan and Animation

Natasha A. Anisimova

(Particle systems in Vulkan)
Intel Game Dev


The Loop

Vulkan Cookbook
OPENGL VS VULKAN

OpenGL
- Hand holding
- Safety net

Vulkan
- Efficient/Effective
- Offloads the CPU/Uses the GPU
- Steep Learning Curve
Compute Shaders

What is it?

is used entirely for computing arbitrary information. While it can do rendering, it is generally used for tasks not directly related to drawing triangles and pixels.

-Definition pulled from OpenGL wiki

How they help?

Animation

- Compute the positions
- Write updated positions for the next frame
Particle simulation

Computer shaders handle the physics and update positions

Particles are rendered as points
Vulkan Renderer Setup (for CPU sim.)

5 components

- Create Instance Object
- Enable Validation Layers
  - Vulkan does not check the validity of the application code
- Create the Virtual Device
  - Surface Object Instance
  - Device Object
- Scene Setup
- Scene Element
- Graphics Memory Allocation & Performance
Multithreaded Approach to Computing Particle System

1. Parallelize the workload across the array of particles, calculating the interactors’ contribution and Euler step as a single work unit.

   2a. Parallelize the workload across interactors, calculating the effect of each interactor at every particle as a single work unit.

   2b. Further parallelize Option 1, above, by splitting interactor contributions into separate work units.

3. Parallelize the sorting algorithm.

4. Parallelize the generators’ contribution.
Recommended Approach

1. Parallelize the workload across the array of particles, calculating the interactors’ contribution and Euler step as a single work unit.
Vulkan Compute in Modeling Particle System

The simulation will be moved onto the GPU

The sorting and generation of particles will be on the CPU

1 Use GPU memory buffer for computation and visualization

2 Moving computation from the CPU to compute shader on the GPU

3 Structure of the compute shader-oriented simulation

*** Note: Intel mentioned that RenderDoc is extremely helpful for debugging Vulkan ***
1 Use GPU memory buffer for computation and visualization

Repurpose the buffer structure that holds particle data

- SSBO (Shader Storage Buffer Object)

Buffer Class needs to be constructed

Vulkan virtual device must be created (this has to happen first)

AppInstance is the very first thing
2 Moving computation from the CPU to compute shader on the GPU

Interactor logic is in the compute shader

- They need to be split by their type
- Specific data is needed for all of them
- Arrays are created for each type of interactor

Vulkan API: Compute stage and graphics stage cannot be bound into a single pipeline so two pipelines would have to be created

Compute pipeline is less complicated

Once pipeline is created, data has to be ready for execution
Two Concepts

- Push Constants
  - Data delivery can be scheduled right within the command buffer without the need for memory mapping
  - Simpler process than descriptor sets
- Division of Workload
  - Work groups are scheduled to execute on the compute device
  - The size is specified within the computer shader
  - API is given the number of work groups scheduled for execution of the compute job
  - Important to remember that these numbers can be limited by your hardware
3 Structure of the compute shader-oriented simulation

Interactors take the form of two components

- Interactor Parameters (delivered through the uniform buffer)
- Function set wherein each function maps to a specific interactor and calculates the acceleration vector for the processed particle

Computing step is done

Renderer is invoked

- Uses the same buffer but accesses memory through vertex buffer interface to render the scene (Not the SSBO)
What Intel is looking to improve on

Completely remove dependencies on using the CPU

- Move all work onto the GPU and use compute shaders for the rest of the computations

Parallelize the Performance Path

- Double Buffering
A piece of advice….

“Understand the hardware you are working with. Not all the techniques will work equally well on every type of GPU. Although all Vulkan devices expose the same API, their properties and limitations might be fundamentally different from each other.”

-Tomasz Chadzynski

Integrated Computer Solutions Inc.
Now what would you use this for?

- Water
- Fire
- Black hole
- Explosions
- Galaxies
- Snowflakes
Questions?