CSM Scrolling

An acceleration technique for the rendering of cascaded shadow maps
CSM Scrolling by:
  Mike Day  mday@insomniacgames.com

CSM Caching by:
  Al Hastings  afh@insomniacgames.com

Who am I?
  Mike Acton  macton@insomniacgames.com
Quick Background
From light POV, imagine whole world as single mega shadow texture
On any particular frame, a shadow map represents a 2D rectangular slice of that volume.
Cascade refers to multiple resolutions of that slice
Assumptions
Most of the time, the camera does not make radical changes across frames.
Most geometry is relatively static across frames.
Geometry which has changed from the previous frame can be identified
The light direction and shape is relatively stable across frames
Results of spatial queries can be used in the same frame as shadow rendering.
Geometry is divided into small* instances
Concept
Store “static” geometry from previous frame in cached map.
Scroll cached map to account for change in camera view
Render additional “static” geometry into edges exposed by scrolling
Render newly “static” geometry in cached area
Copy map to use as final shadow map for current frame
Render non-static geometry into final shadow map for frame
CSM Caching
Assumption: Camera is not moving (much)
Store “static” geometry from previous frame in cached map
Store “static” geometry from previous frame in cached map

“static” = not moved for t time. (e.g. 5 seconds)
Each frame, render non-static geometry on to cached copy.
Cache of previous frame shadow map
Cache of previous frame shadow map

Invalid if...
Cache of previous frame
shadow map

Invalid if...
• Camera moves
• Camera FOV changes
Cache of previous frame shadow map

Invalid if...
- Camera moves
- Camera FOV changes
- “Static” geometry moves
Render newly “static” geometry in cached area
Query for state of “static” geometry
Diff current “static” versus previous “static” query results
Dynamic occlusion system used
Create copy new map cache to use this frame
“Dynamic” geometry rendered to temporary shadow map
CSM Scrolling
Assumption: Camera moves a lot (but slowly*)
Insert in to CSM Caching:
1. Scroll map
2. Render into exposed edges
Scroll cached map to account for change in camera view
Sample shadow texels from previous frame
Scrolled area is clamp-to-border (color=1.0)
Observe:
Camera motion is 3D
Observe:
Camera motion is 3D
  • Lateral scrolling
  • Depth scrolling
Lateral scrolling

Translation perpendicular to light rays

```cpp
float ScrolledDepth_LateralOnly( input )
{
    float2 uv = input.xy;
    return SampleShadowMap(uv);
}
```

UV translated by delta camera in light frame
Lateral scrolling

*Translation perpendicular to light rays*

```cpp
float ScrolledDepth_LateralOnly( input )
{
    float2 uv = input.xy;
    return SampleShadowMap(uv);
}
```

Simple texture lookup
(Point sampling)
Depth scrolling

*Translation parallel to light rays*

```cpp
float ScrolledDepth( input )
{
    float2 uv = input.xy;
    float depth_offset = input.z;
    float old_depth = SampleShadowMap(uv);
    return old_depth + depth_offset;
}
```

Additional handling needed for depth scroll
Depth scrolling

Translation parallel to light rays

```cpp
float ScrolledDepth( input )
{
    float2 uv = input.xy;
    float depth_offset = input.z;
    float old_depth = SampleShadowMap(uv);
    return old_depth + depth_offset;
}
```

Delta camera depth in light frame
Depth scrolling

*Translation parallel to light rays*

```cpp
float ScrolledDepth( input )
{
    float2 uv = input.xy;
    float depth_offset = input.z;
    float old_depth = SampleShadowMap(uv);
    return old_depth + depth_offset;
}
```

Offset all previous depths (scroll depth)
Depth scrolling

*Translation parallel to light rays*

**Gotchas:**

- Near plane
- Far plane
Depth scrolling

*Translation parallel to light rays*

**Gotchas:**

- Near plane
- Far plane
  
  Clamp to 0.0
Depth scrolling

*Translation parallel to light rays*

Gotchas:

- Near plane
- Far plane

Problem 1.0 = buffer clear
Depth scrolling

*Translation parallel to light rays*

Gotchas:
- Near plane
- Far plane

Problem 1.0 = buffer clear

```c
float ScrolledDepth( input )
{
    float2 uv = input.xy;
    float depth_offset = input.z;
    float old_depth = SampleShadowMap(uv);
    float new_depth = old_depth + depth_offset;
    return (old_depth < 1.0) ? new_depth : 1.0;
}
```
Render additional “static” geometry into edges exposed by scrolling.
Scrollled in area divided into slabs (thin OBBs)
‘Static’ geom with overlapping bounding volume rendered
Observe: Coarseness of geometry relative to view
Very few overlapping volumes
Jagged pattern not relevant: Using square geom tiles
Render newly “static” geometry in cached area.
Copy map to use as final shadow map for current frame.
persistent

temporary

scroll

render ΔS

render A

CSM Scrolling
Each map (512x512) PS3/360
Another view…
Wrap up
Straightforward addition to CSM Caching
Key:
Like 2D bitmap scrolling
Do not render ~70% of ‘static’ geometry in to CSM
Detailed paper:
bit.ly/QIoBr9