Motion Blur - as a post processing effect

Presented by Richard Fredriksson and Fredrik Wendt
Original paper by Gilberto Rosado

What is Motion Blur?

- Artifact of the camera
- Shutter speed
- Speed of objects
- Camera tracking

Motion Blur Photo

Motion Blur - Why simulate an image artifact?

- Enhanced game immersion
- Illusion of speed
- Reduced stuttering, feels more natural
- Increasing frame rate does not eliminate the need for motion blur
- Dramatic effect

Motion Blur in a game

Killzone 2

Motion Blur

- Techniques mentioned in paper
  - Multiple render passes
    - Velocity buffer
      - Used for motion blur of a dynamic rigid bodies
  - Post processing effect
    - This paper

- Other techniques
  - Model and render the blur
    - Geometry shader
  - Accumulation buffer
    - Slow but perfectly correct solution
  - Motion blurring textures
    - Sets of preblurred images

Multiple Render Passes

Advantages

- Good image quality
- Only blurs selected objects
- Can handle dynamic rigid bodies

Disadvantages

- Slow, sometimes we cannot afford to go through the entire graphics pipeline more than once
- A lot of work to integrate into an engine
- Memory limited (without workarounds)

Motion Blur as a Post Processing Effect

Advantage

- Easily integrated into an existing engine
- Fast, offers better performance then multipass rendering

Disadvantage

- Only offers motion blur for camera movement
- Objets that should not be motion blurred must be masked

Details – Depth Buffer

- DX10
  - Direct sample from the depth buffer
- DX9
  - Write depth to texture
- Driver hack work around?

Details – World space positions

Z = Depth Texture sample

H = Viewport space position

H = (x * 2 – 1, (y-1) * 2 – 1, z, 1)

M = World-view-projection matrix

D = H * inverse(M)

WorldPosition = D / D.w

Details – Velocity of each pixel

CurrentPosition = H

OldM = previous view-port-projection Matrix

PrevPosition = WorldPosition * oldM

PrevPosition = PrevPosition / PrevPosition.w

Velocity = (CurrentPosition – PrevPosition) / 2

Details – Performing Motion Blur

Color = Sample the color buffer

TexCoord = TexCoord + velocity

Loop for NumberOfSamples

    CurrentColor = sample at TexCoord

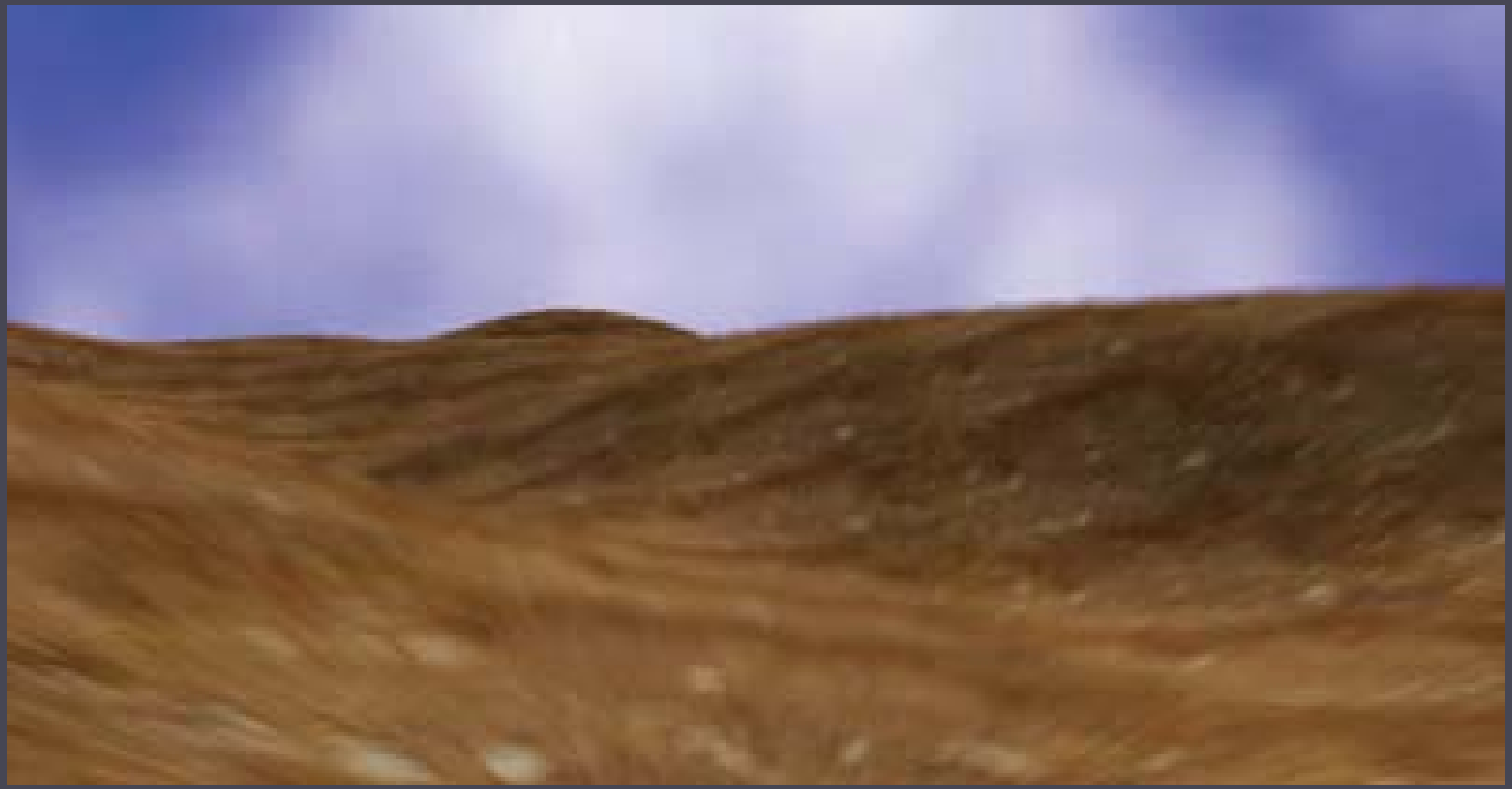
    Color = Color + CurrentColor

    TexCoord = TexCoord + velocity

FinalColor = Color / NumberOfSamples

Motion blur effect applied

GPU Gems



Other – Handling Dynamic Objects

- As mentioned this technique only takes camera movement into account

- Velocity buffer
  - Transform object by using both current and last frames view-projection matrix
  - Compute difference
  - Render color buffer
  - Use velocity buffer to blur at each pixel sampling from the color buffer render

Other – Masking Off Objects

- Often when using Motion Blur parts of the scene should not be blurred
- In order to achieve this a mask is used to determine what pixels should be blurred

Conclusion

- Easily integrated into an existing rendering engine
- Better performance than traditional multipass solutions

- No benchmark data
- No real comparison

Demo

Thanks for taking the time to listen