Summed Area Tables using Graphics Hardware

Simon Green, NVIDIA

What are Summed Area Tables (SATs)?

- Pre-integrated texture representation
- Invented by Frank Crow, 1984
- Each texel is the sum of all texels below and to the left of it
- Allows rapid box filtering (average) over any rectangle for a fixed cost
- Based on the algebraic identity:

$$(x+a)(y+b) - (x+a)y - x(y+b) + xy = ab$$

Summed Area Table

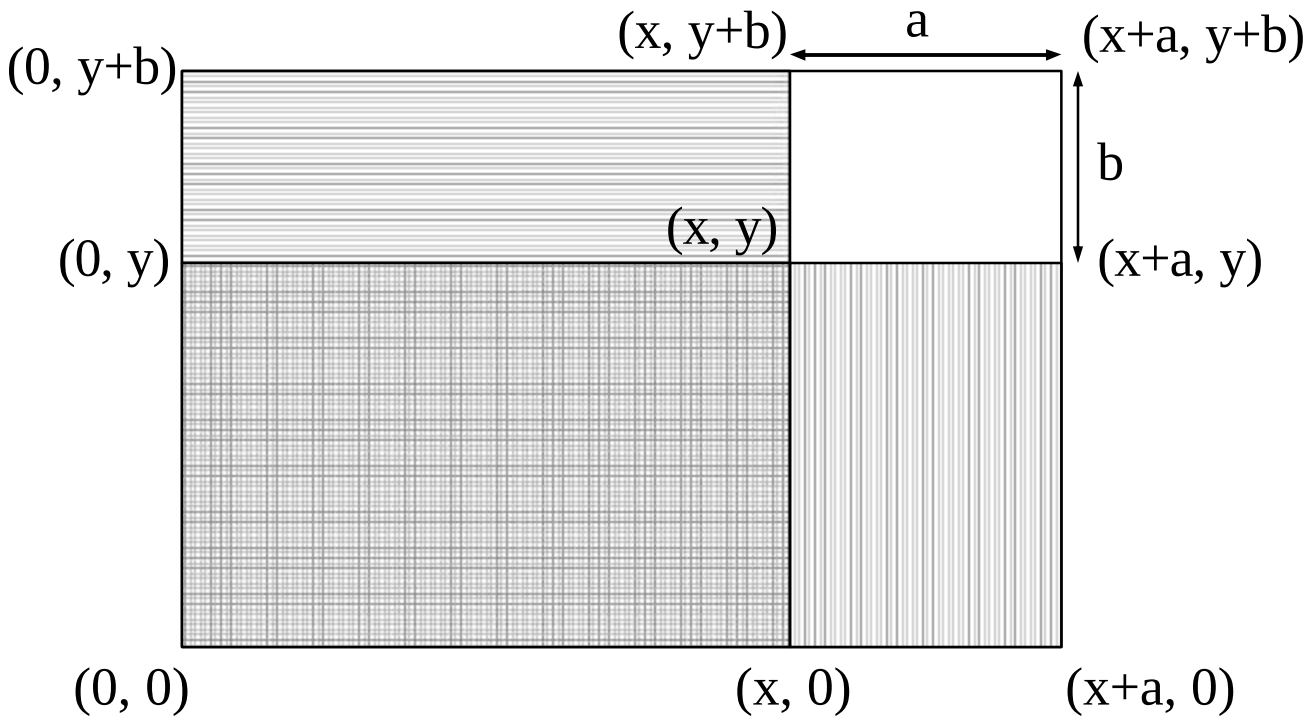
Regular image

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

Summed area table

4	8	12	16
3	6	9	12
2	4	6	8
1	2	3	4

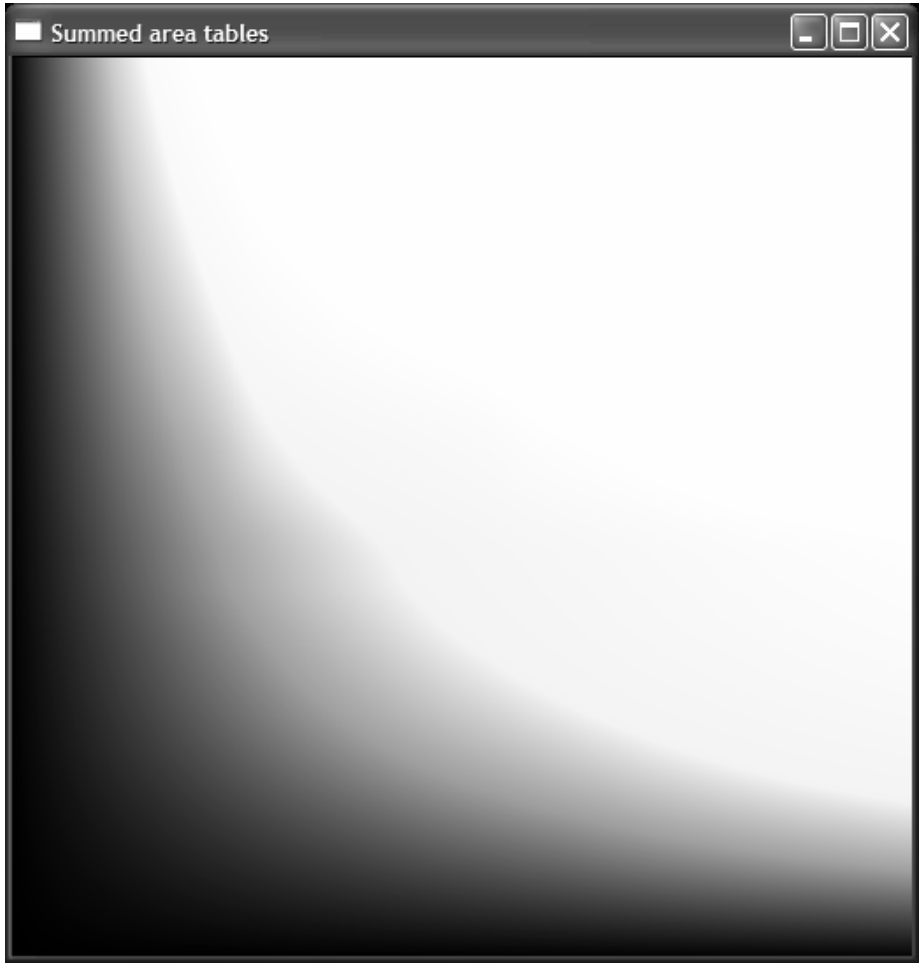
Summed Area Table



$$ab = S[x+a, y+b] - S[x+a, y] - S[x, y+b] + S[x, y]$$

Original Image

Summed Area Table (scaled)



SATs on Graphics Hardware

- Floating point textures have sufficient precision to represent SATs
- Can decode SAT using a fragment program
- Enables texture lookup with any width box filter for constant cost (4 texture lookups + math)
- Amount of blur can vary from pixel to pixel (spatially variant)

SAT Texture Decode

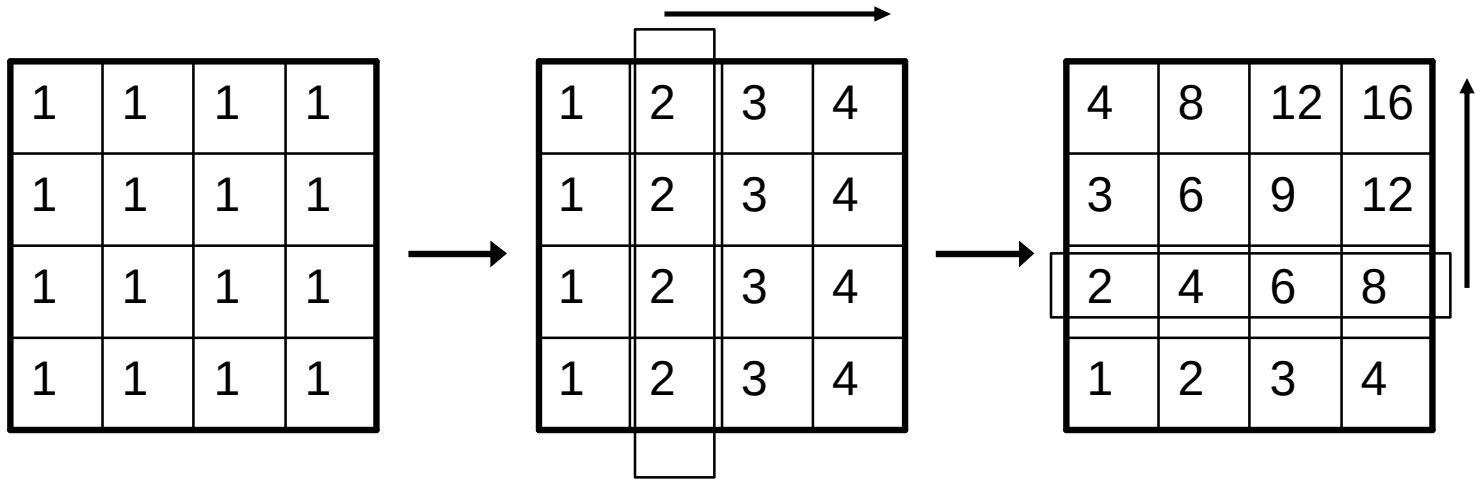
```
// look up into summed area table texture
// returns average of pixels in rectangle with corners (x,y) - (x+a,y+b)
float4 texRECT_summedArea(uniform texobjRECT tex, float2 xy, float2 ab)
{
   float4 r;
   r = f4texRECT(tex, xy + ab);                    // S[x+a, y+b]
   r = r - f4texRECT(tex, xy + vec2(ab[0], 0) ); // S[x+a, y]
   r = r - f4texRECT(tex, xy + vec2(0, ab[1]) ); // S[x, y+b]
   r = r + f4texRECT(tex, xy);                     // S[x, y]
   return r / (ab[0] * ab[1]);
}
```

- **Bilinear lookup requires 4 bilinear lookups into float texture, followed by SAT decode**
- **May also require clamping texture coordinates to avoid problems at edges**

Building Summed Area Tables using Hardware

- For dynamic applications (e.g. depth of field), we need to be able to create SAT textures on-the-fly
- Possible using render-to-texture
  - Sum columns first, and then rows
  - Each row or column is rendered as a line primitive
  - Fragment program adds value of current texel with texel to the left or below
  - Can be done using render to active texture, but *warning - results undefined!*

Building Summed Area Table

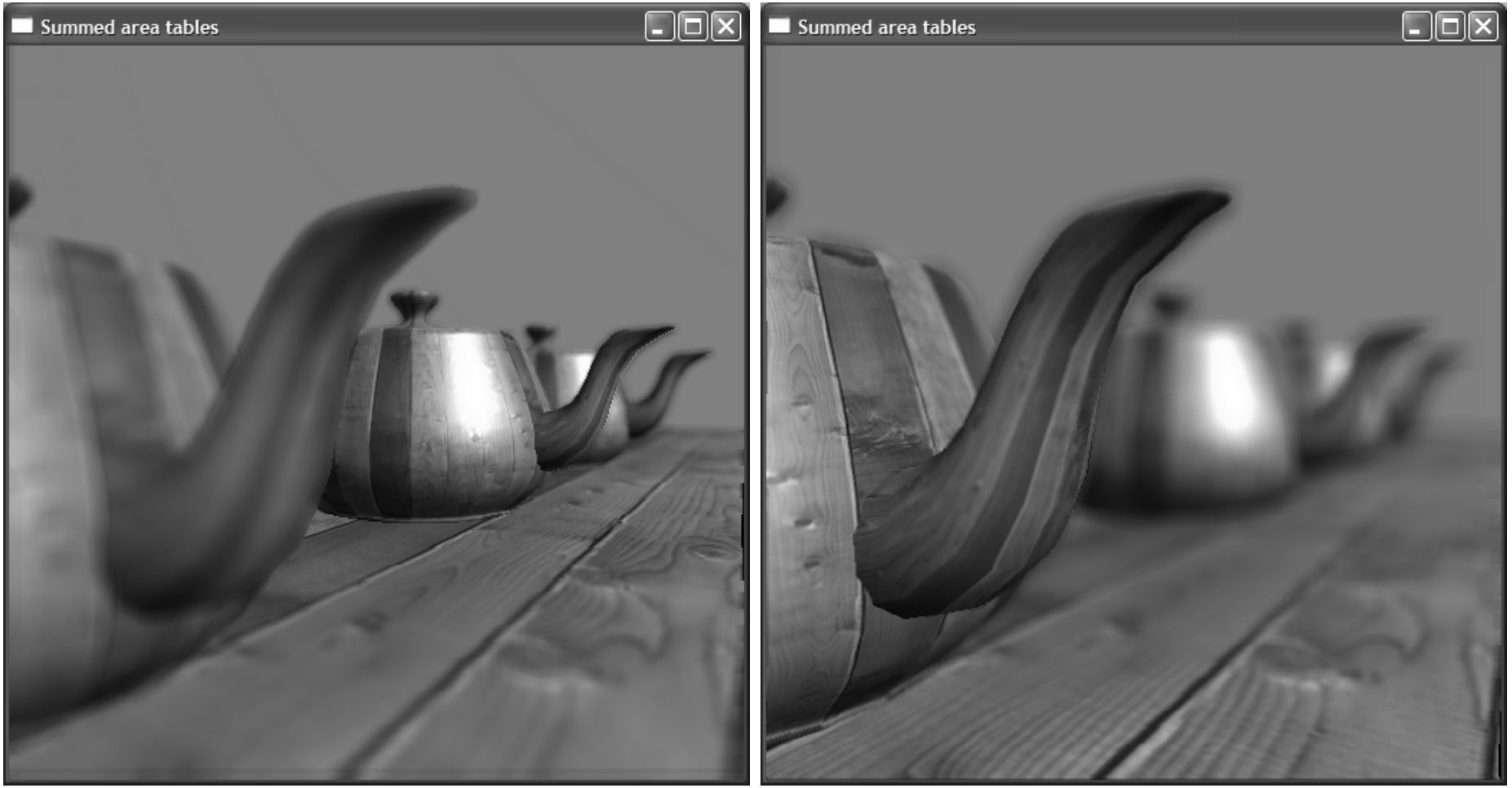


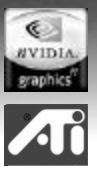
- **For n x m image, requires rendering 2 x n x m pixels, each of which performs two texture lookups**
- **Other algorithms may be possible**

Applications of SATs

- **Anisotropic texture filtering**
  - **Can be better than mip-mapping**
  - **Rectangle may not match projection of pixel exactly**
- **Depth of field**
  - **Blur image of scene based on distance**
  - **Higher quality blur than using mipmaps**
- **Soft shadows**
  - **Blur image of shadow based on distance from occluder**

Depth of Field Demo





References

- **Crow, Frank. *Summed-Area Tables for Texture Mapping*. Proceedings of SIGGRAPH `84. In Computer Graphics 18, 3 (July 1984), pp. 207-212.**