High Quality Surface Splatting on Today's GPUs

Mario Botsch, Alexander Hornung,
Matthias Zwicker, Leif Kobbelt

Introduction

- Point-based representations suitable for massive data sets
- Rendering is important for interactive applications
  - High performance
  - High visual quality

Current PBR methods

- High quality
  - Projectively correct rasterization
  - Phong shading
  - EWA anti-aliasing
- ... or high performance

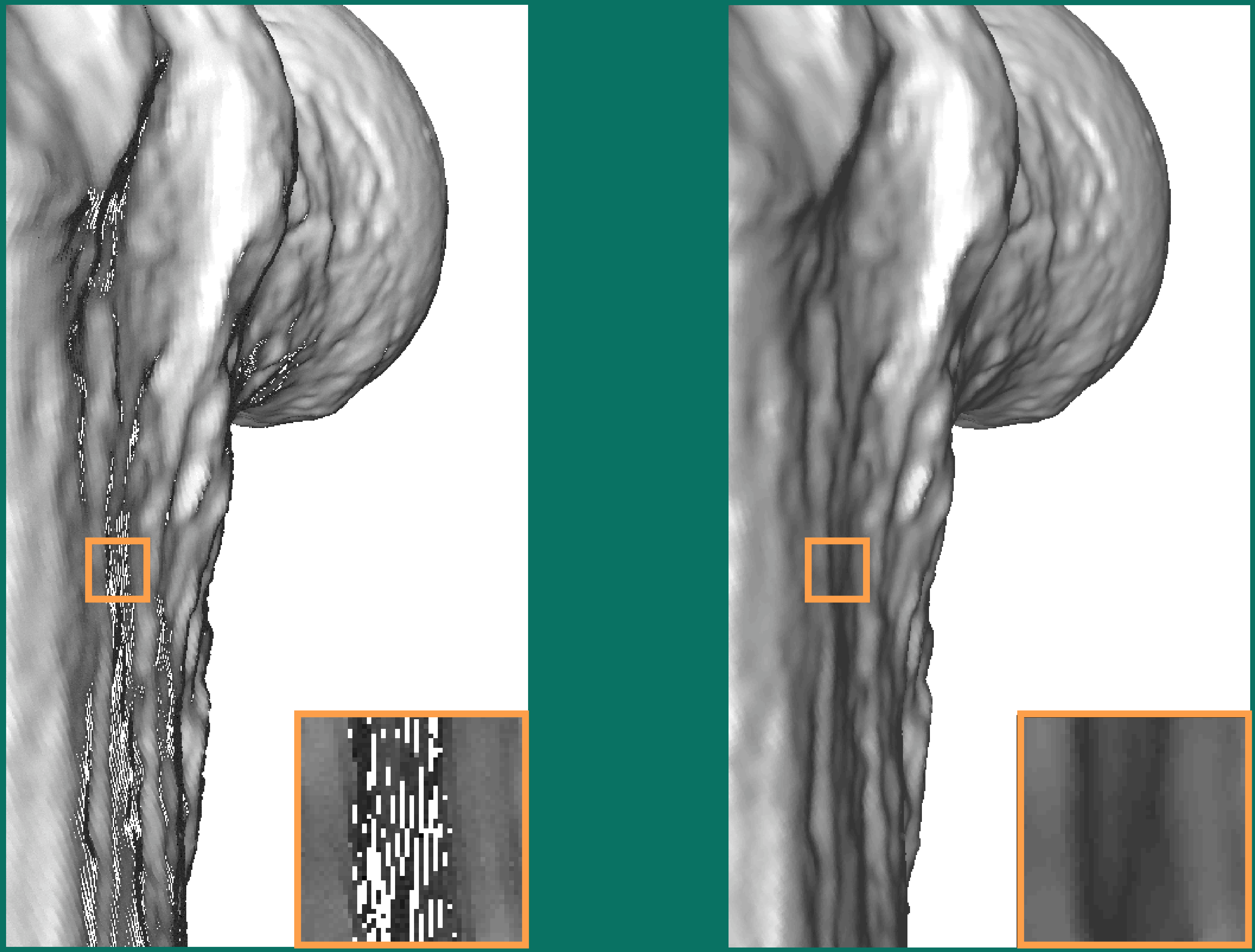
Current PBR methods

- Trade-off performance against quality
  - Limited by available GPU features
- Exploit latest GPUs' features to get both

Overview

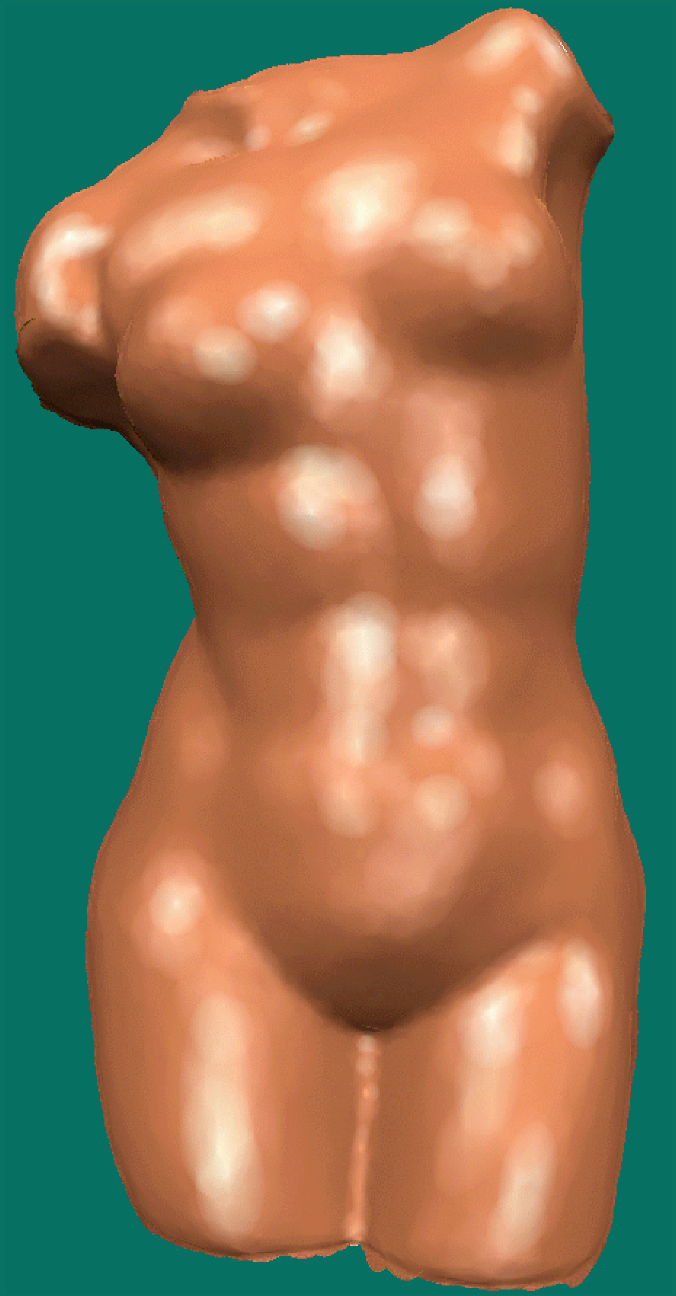
- Related Work
- Deferred Shading
- EWA Filter Approximation
- Results

Perspectively Correct Rendering



Per-Pixel Phong Shading

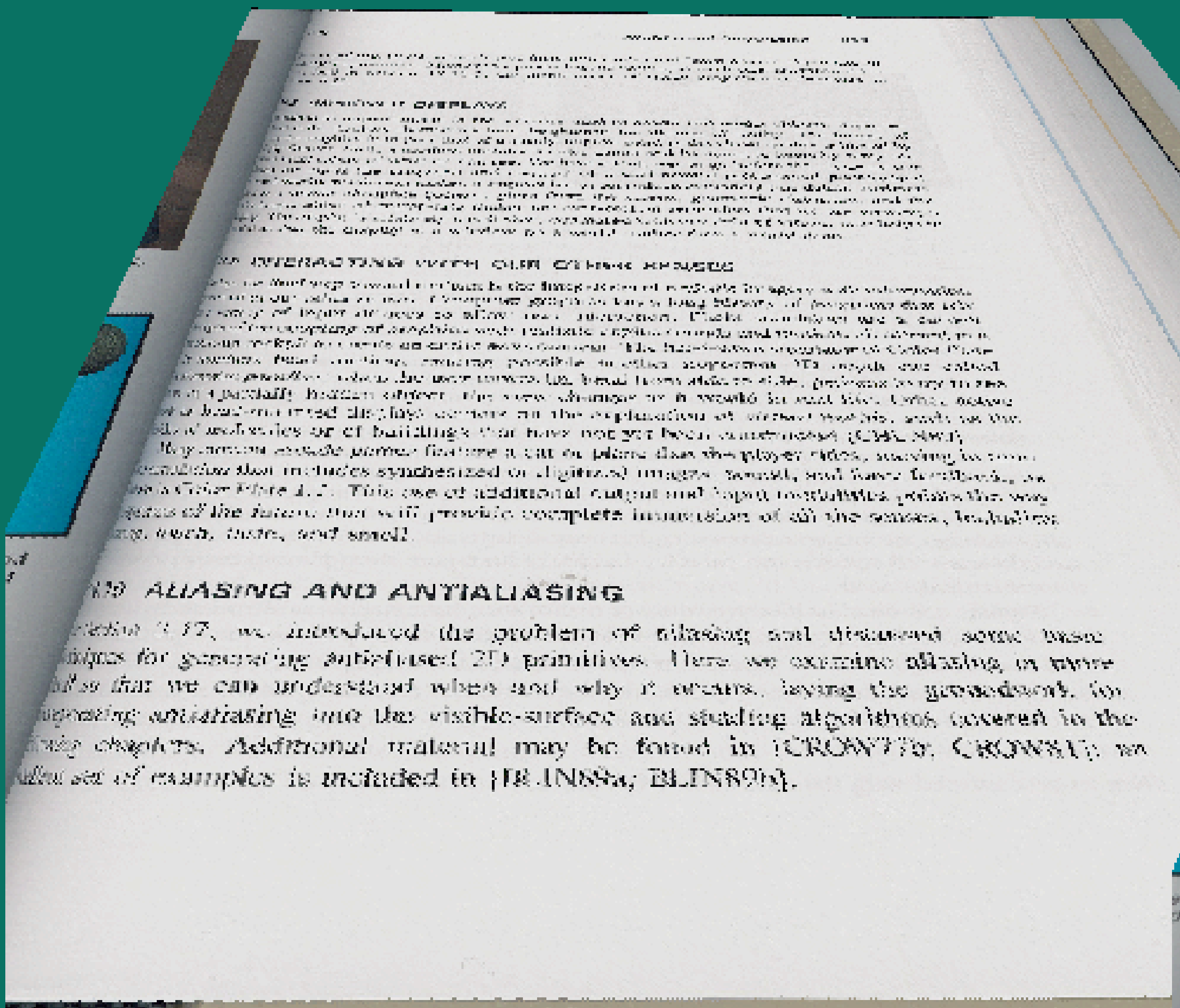
Flat Shading



Gouraud Shading

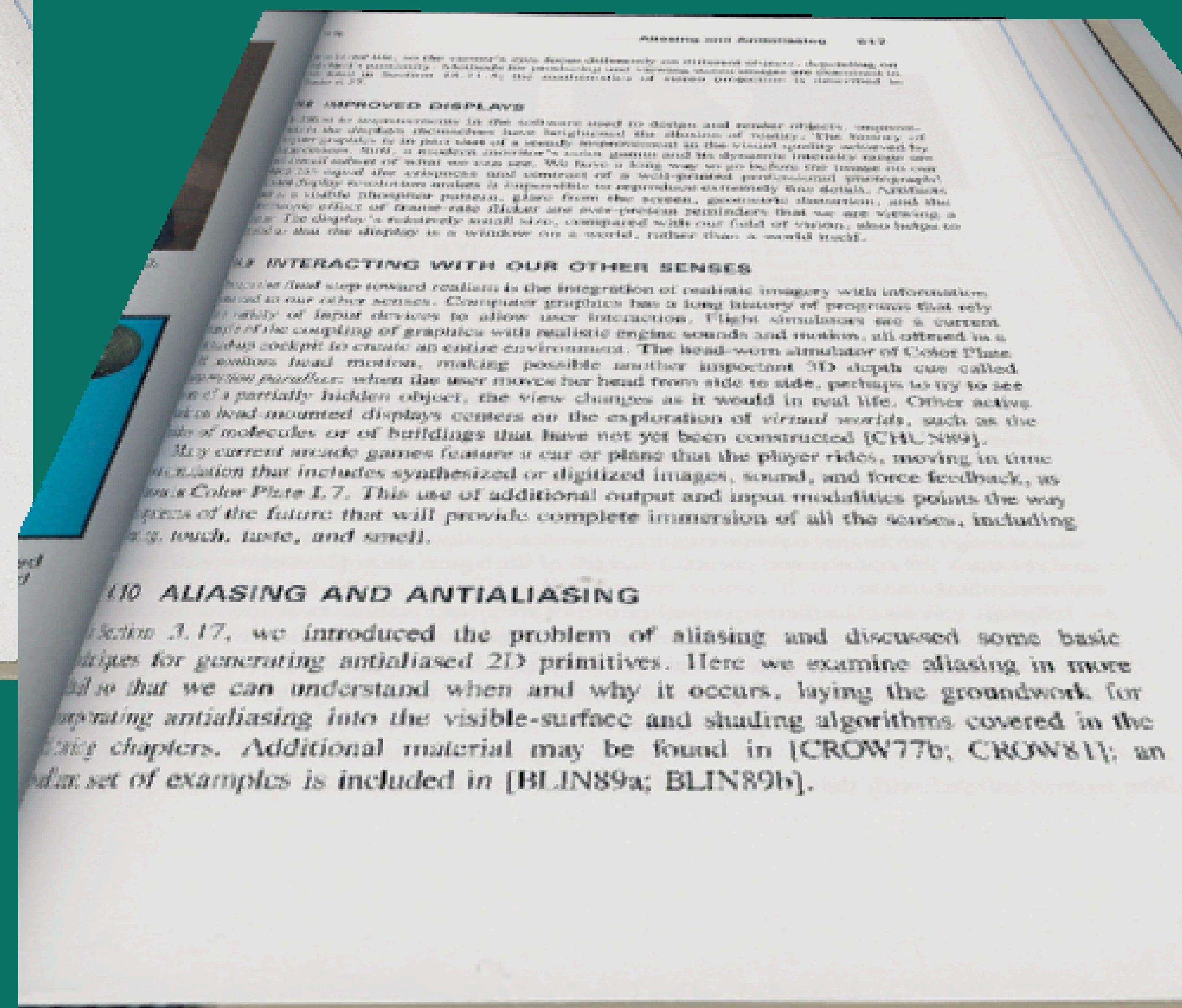
Phong Shading

Anti-Aliasing



No filtering

EWA filtering



Comparison

	Persp. Correct	Phong Shading	Anti-Aliasing	Splats/ sec
EWA Splatting	✗	✓	✓	1M
NV30 Splatting PG '03	✗	✗	✗	27M
Persp. Accurate GI '04	(✓)	✗	✓	5M
Phong Splatting PBG '04	✓	✓	✗	6M
NV40 Splatting PBG '05	✓			

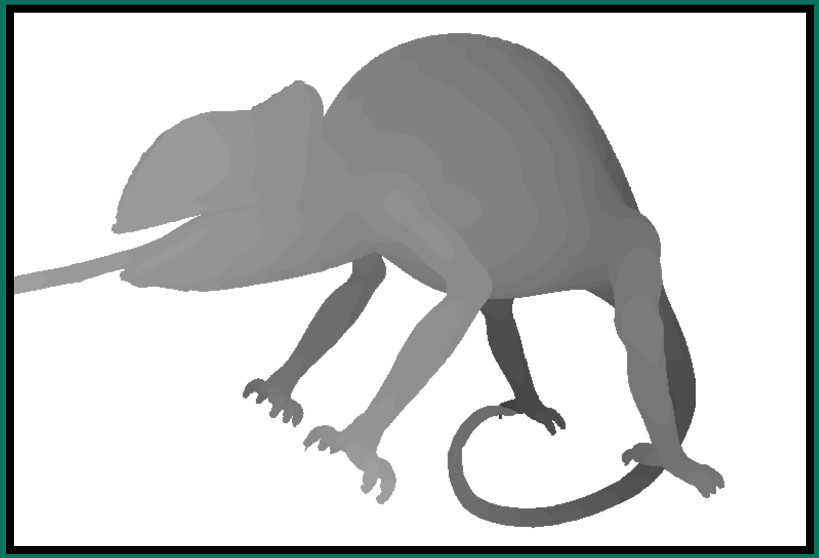
Overview

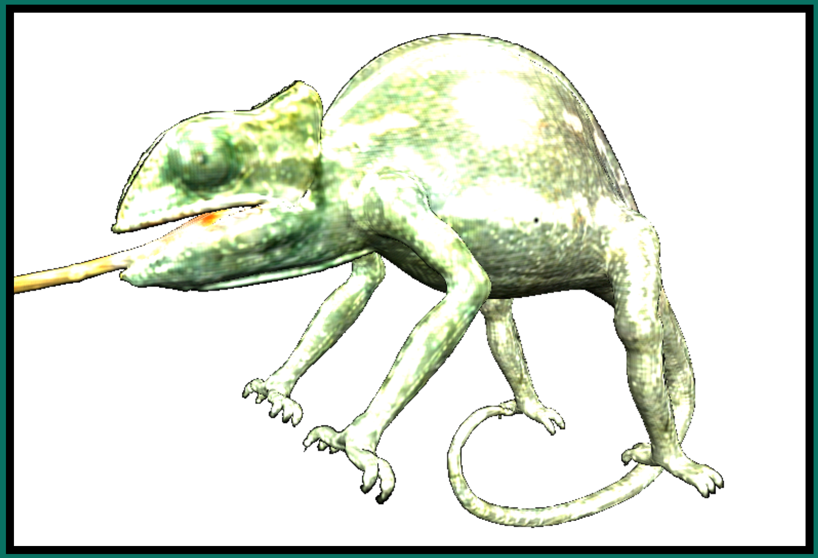
- Related Work
- Deferred Shading
- EWA Filter Approximation
- Results

Phong Shading for PBR

- Interpolate normal vectors
  - No connectivity like for meshes
- Assign linear normal field
  - Limited to static geometries
- Splat normals into framebuffer
  - Deferred shading

3-Pass Shading



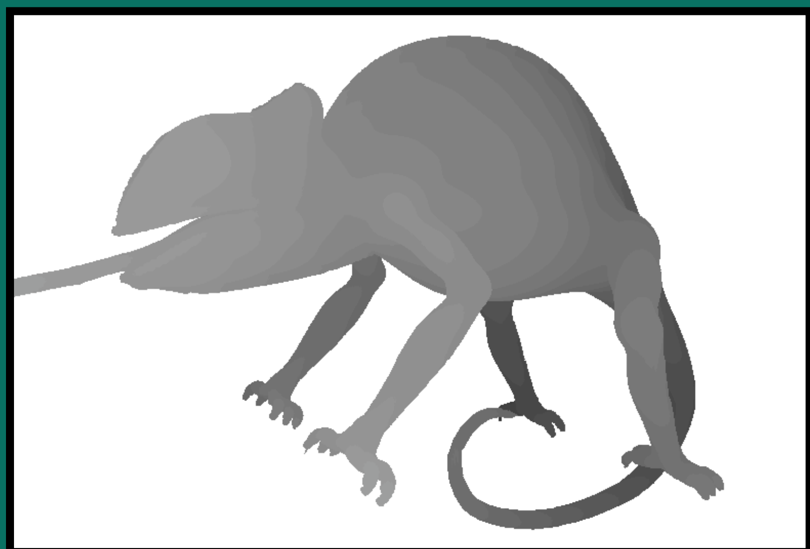


Visibility
Splatting

Shading
Blending

Normali-
zation

3-Pass Deferred Shading



Visibility Splatting

Attribute Splatting

Normalization Shading

Deferred Shading

- Compute lighting for each *image pixel*, not for each *generated fragment*

- Splats mutually overlap
  - #fragments ≈ 7 · #pixel

- Rasterization shader is bottleneck
  - Keep it small by deferring shading

➡ Performance does hardly depend on surface shader complexity

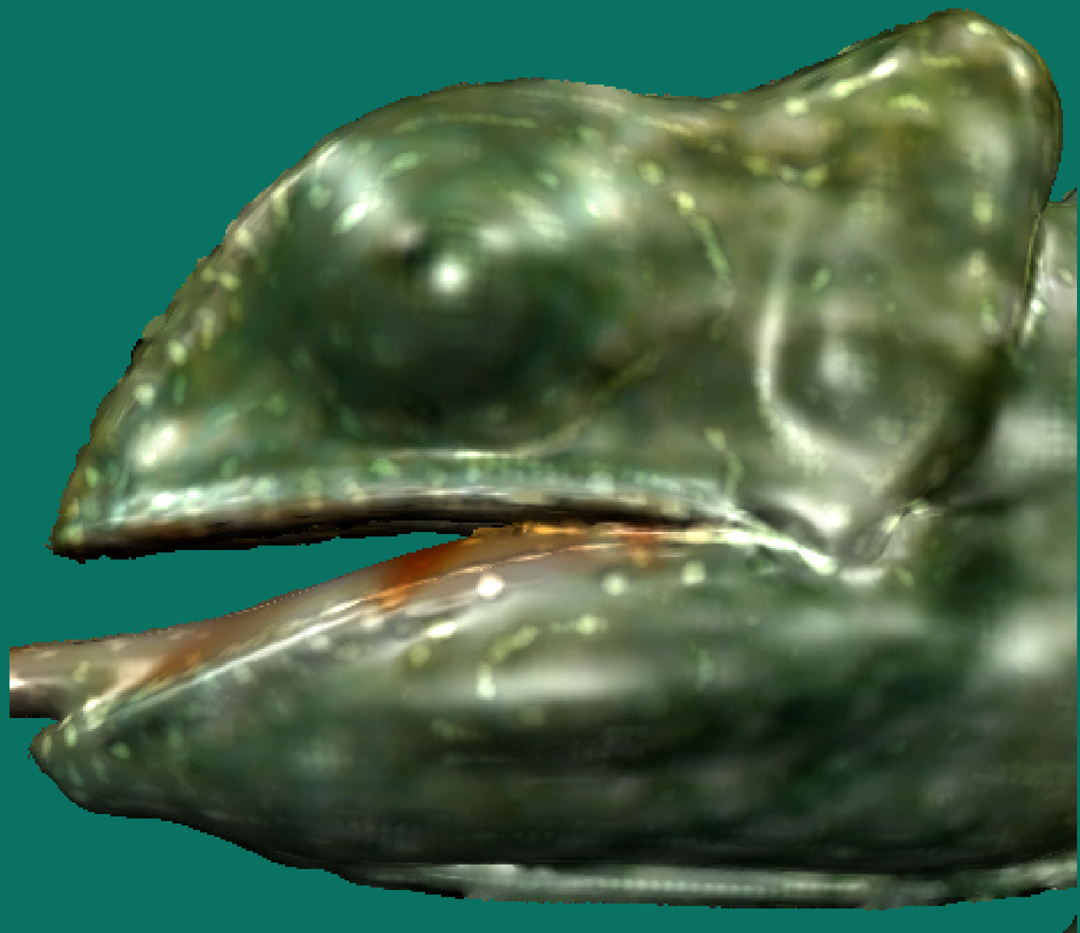
Deferred Shading

- Clear separation between rasterization and surface lighting / shading
- Simplifies shader development
  - Same shader as for meshes

Required GPU Features

- Render attributes to several buffers
  - Multiple render targets (MRT)
- Accumulate at high precision
  - Floating point arithmetic
  - Floating point buffers
  - Floating point textures
  - Floating point blending

Floating Point Precision



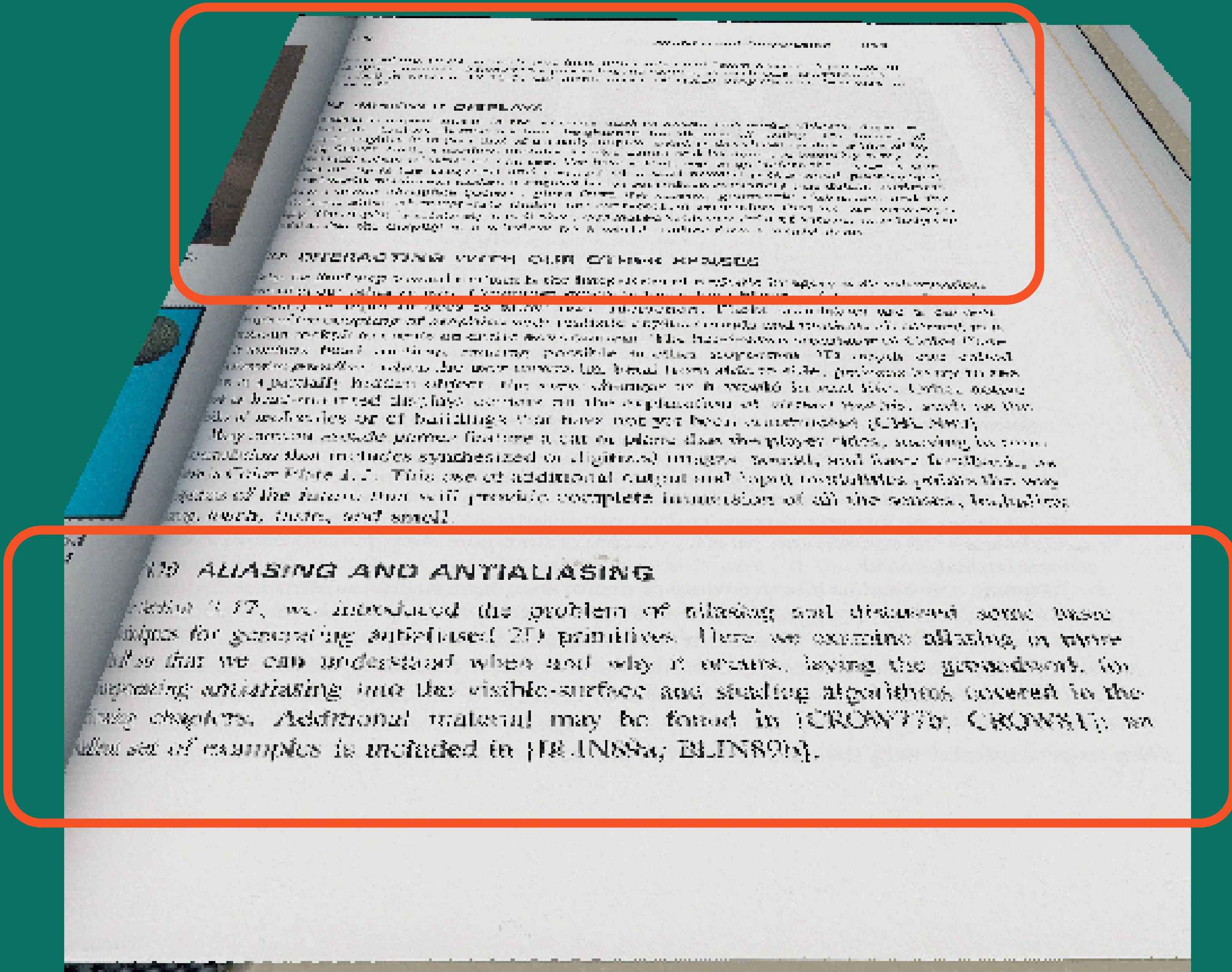
8 bit ubyte
clamped to [0,1]

16 bit float
un-clamped

Overview

- Related Work
- Deferred Shading
- EWA Filter Approximation
- Results

EWA Anti-Aliasing



Screen-space low-pass filter

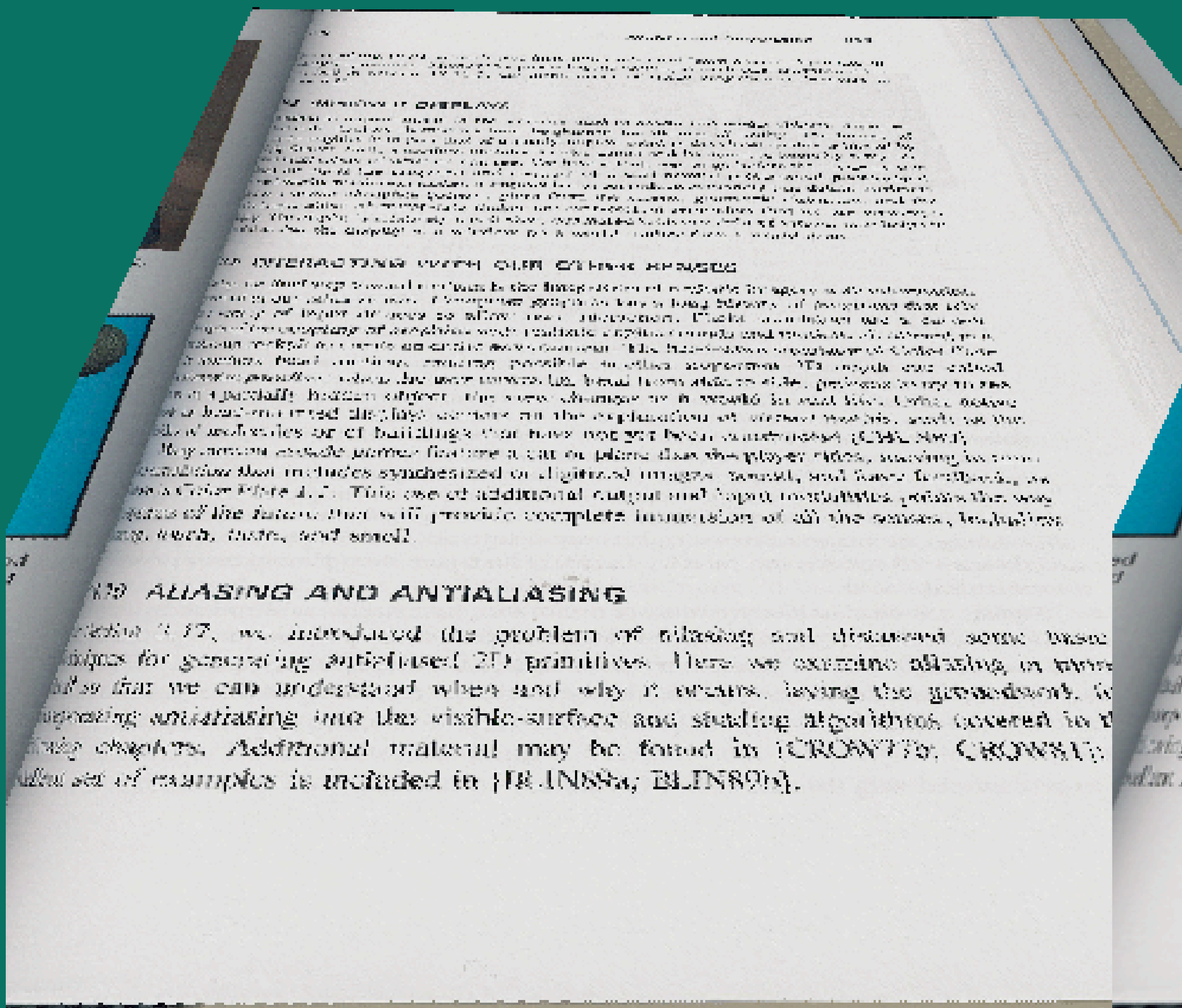
Object-space reconstruction filter

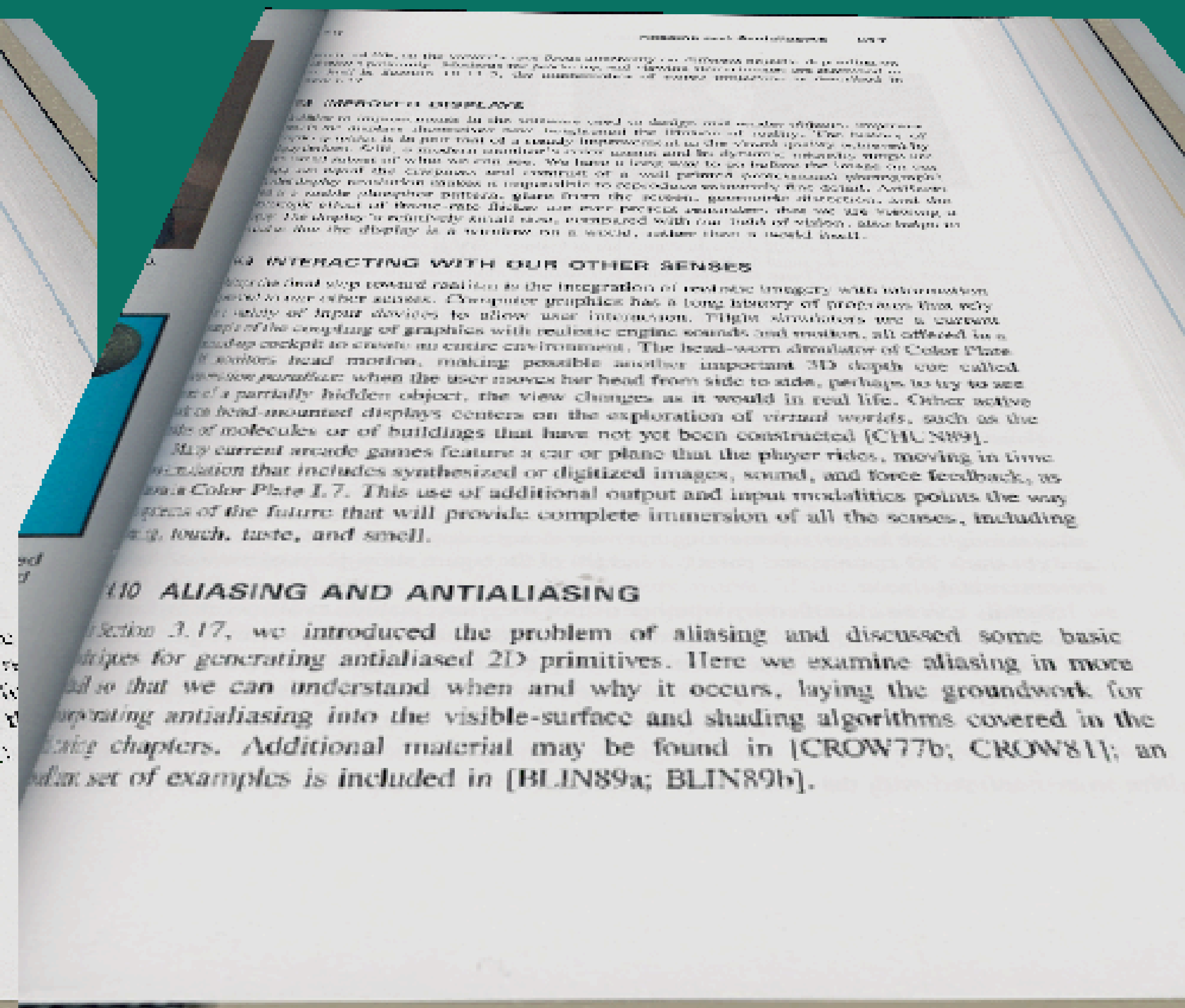
EWA Filtering

- Combine both filters into one
  - Choose both as Gaussians
  - Affine approximation to projection
- GPU implementation in GI ‘04
  - Complex computations
  - About 5M splats/sec

➡ Screen-space filter often omitted

Object-Space Filter Only

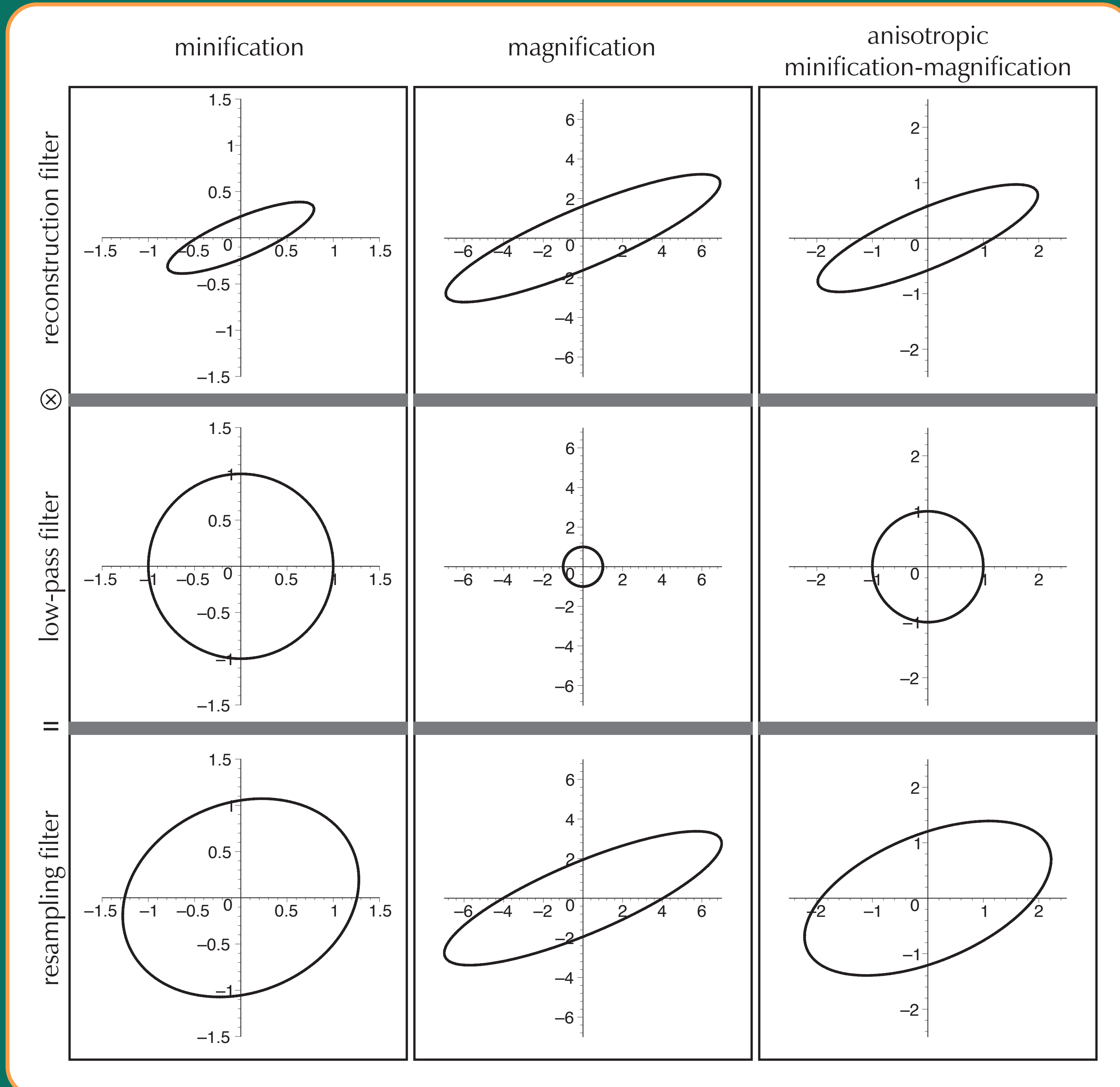




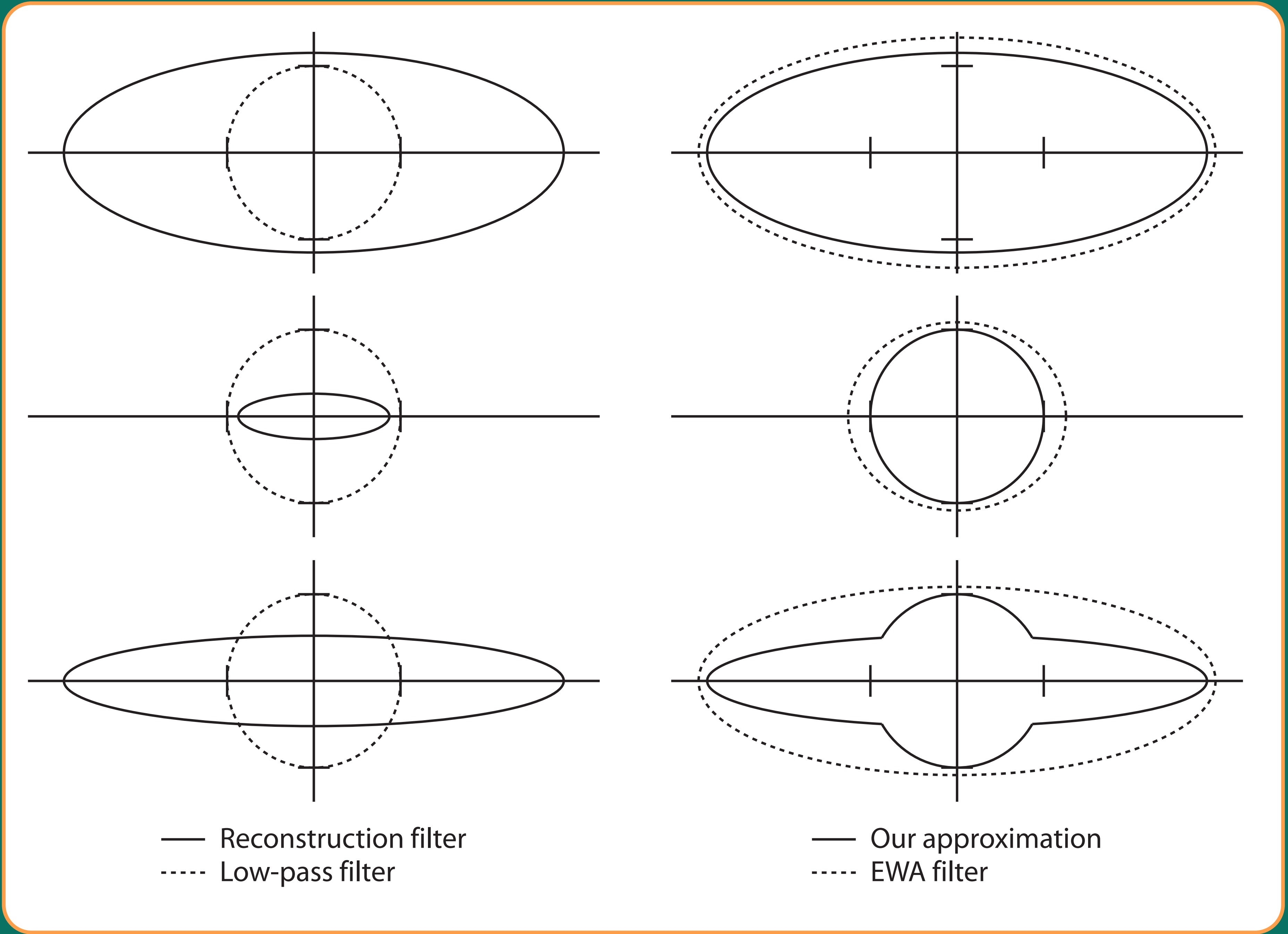
No filtering

Object-Space

EWA Filtering



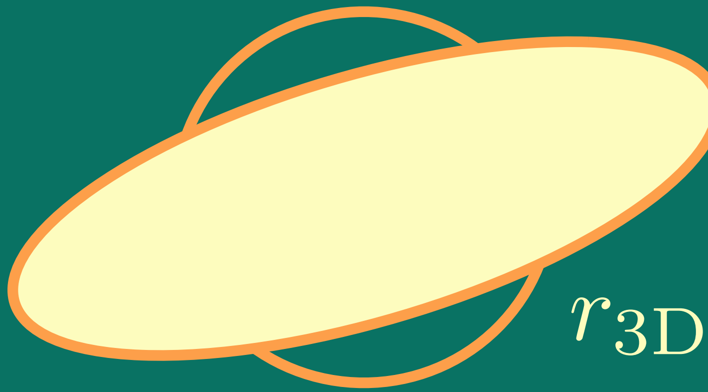
Approximate EWA Filtering



Approximate EWA Filtering

- Reconstruction filter radius

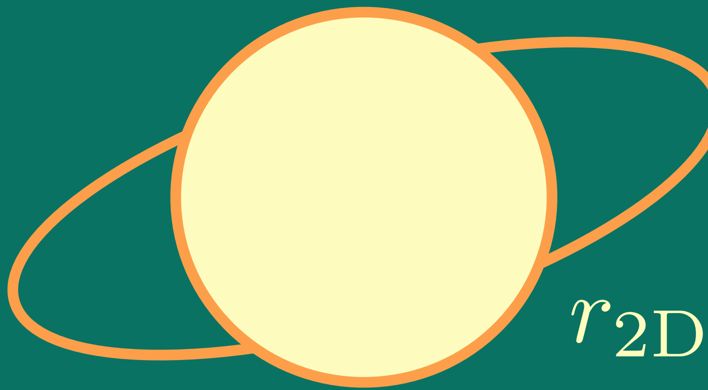
$$r_{3D} := \sqrt{u^2 + v^2}$$



$r_{3D} \leq 1$

- Screen-space filter radius

$$r_{2D} := d\left(x, y\right) / \sigma$$

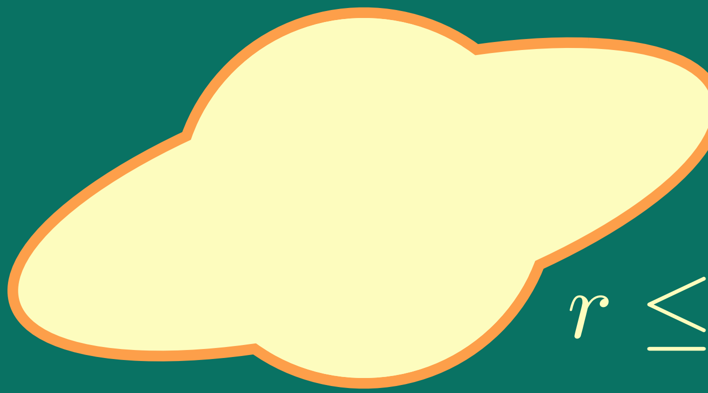


$r_{2D} \leq 1$

- Combined filter

$$r := \min\left\{r_{3D},\ r_{2D}\right\}$$

$$w := \text{Gauss}\left(r\right)$$



$r \leq 1$

Approximate EWA Filtering

- Restrict minimum projected splat size to $2\sigma \times 2\sigma$ pixels
  - Ensure enough fragments for AA
  - Done in vertex shader
- Combine minimum of radii
  - Done in fragment shader
  - 3 additional instructions only

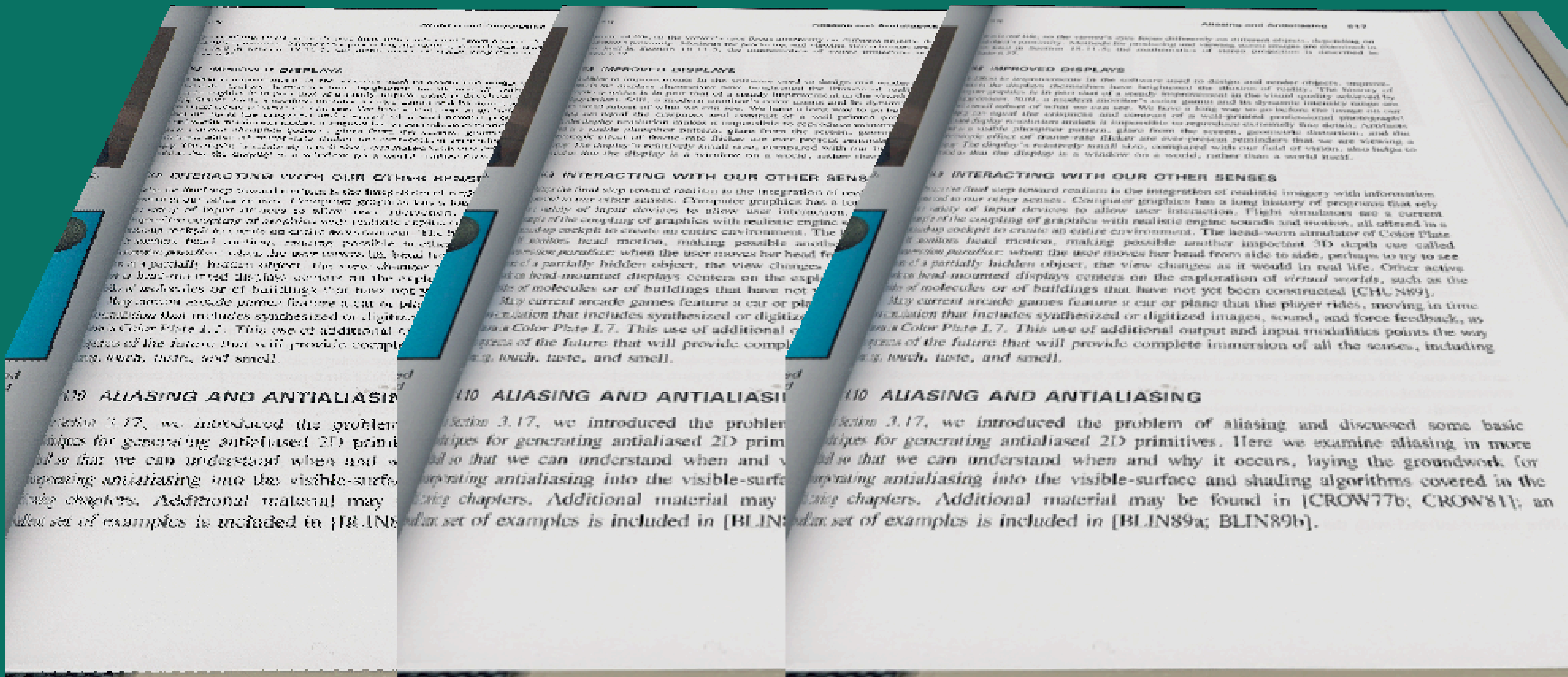
Approximate EWA Filtering

- Simple approximation to exact EWA
  - Efficient implementation
  - Removes (most) aliasing
- Generates much more fragments
  - $\#fragments \approx 30 \cdot \#pixel$
  - Deferred shading!

Overview

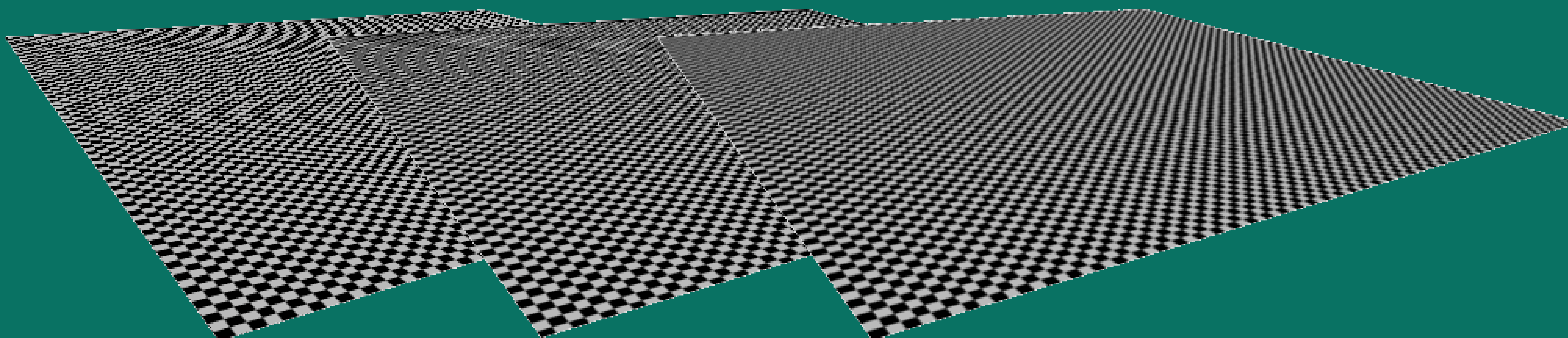
- Related Work
- Deferred Shading
- EWA Filter Approximation
- Results

Approximate EWA Filtering



No filtering | Object-Space | Object-Space Screen-Space

Approximate EWA Filtering



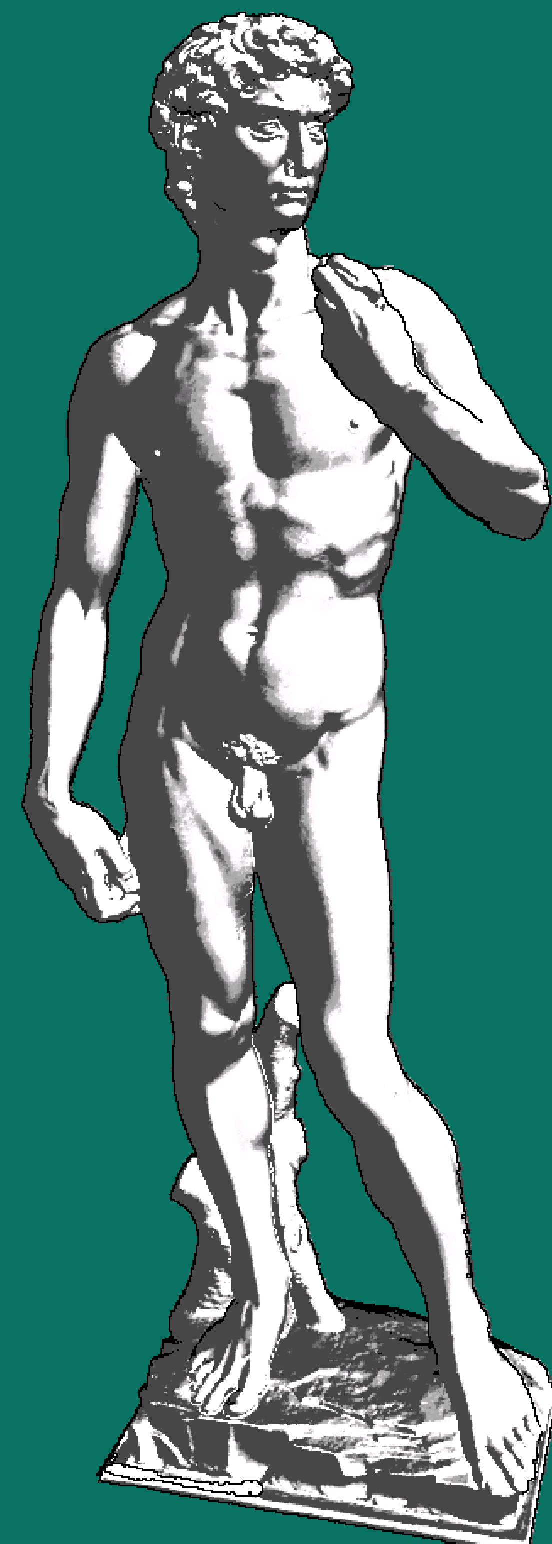
Object-Space

Object-Space
FSAA

Object-Space
Screen-Space

Phong Shading

NPR Shading



Timings & Comparison

- Models from 100k to 14M splats
- Different surface shaders (GeForce 6800 Ultra)

	#instructions	Msplats/sec
Phong	8	22.8
NPR	48	21.3
Phong+Shadow	23	15.7
NPR+Shadow	61	15.0
PG ‘03		25.0
PBG ‘04		5.2

Comparison

	Persp. Correct	Phong Shading	Anti-Aliasing	Splats/ sec
EWA Splatting	✗	✓	✓	1M
NV30 Splatting PG '03	✗	✗	✗	27M
Persp. Accurate GI '04	(✓)	✗	✓	5M
Phong Splatting PBG '04	✓	✓	✗	6M
NV40 Splatting PBG '05	✓	✓	(✓)	23M

Conclusion

- Latest GPUs offer important new features
  - Multiple render targets
  - Floating point pipeline
- Allows for fast & high quality splatting
  - Deferred shading
  - EWA approximation