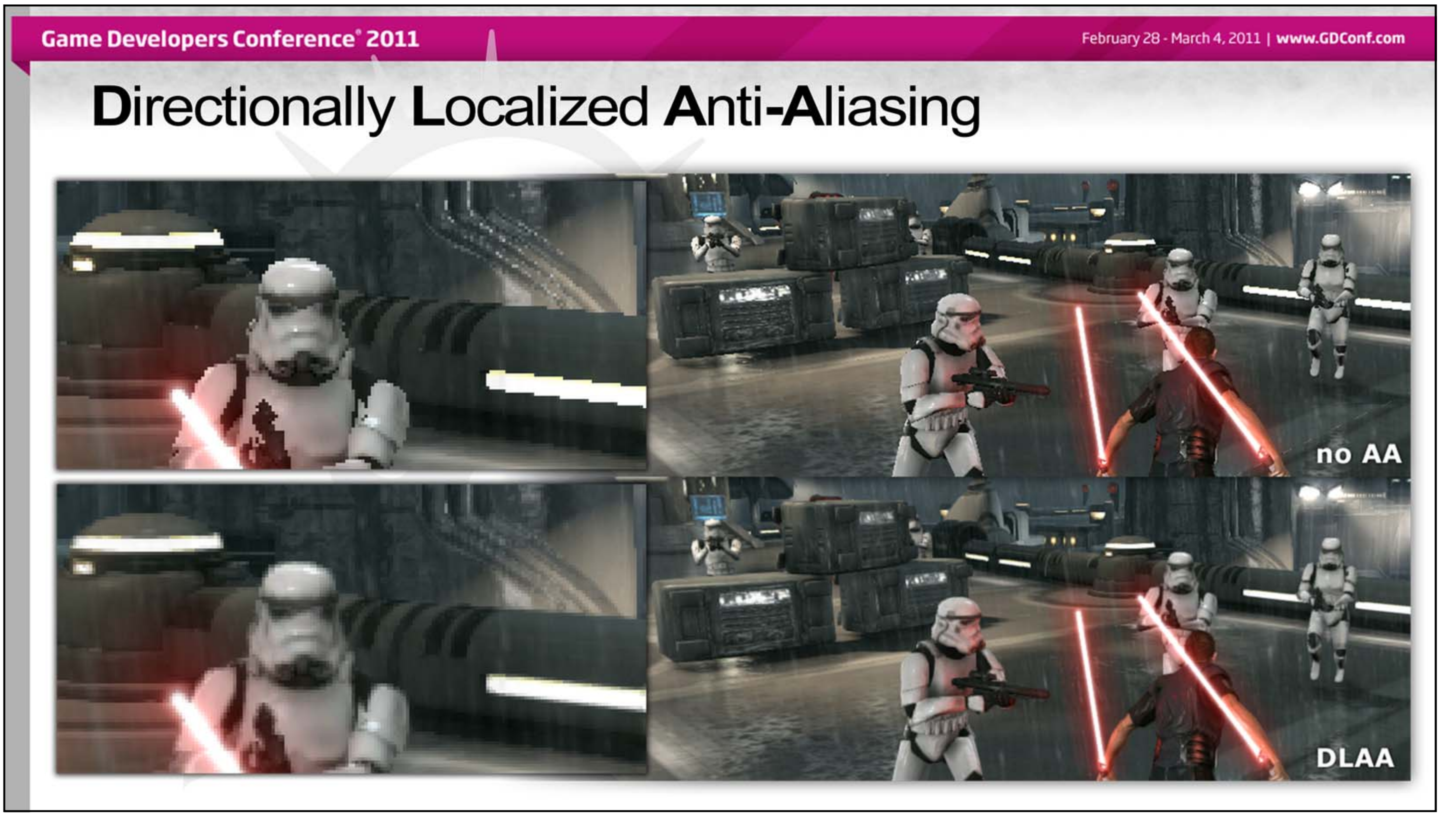
SIGGRAPH2011 VANCOUVER

Filtering Approaches for Real-Time Anti-Aliasing

http://www.iryoku.com/aacourse/

Final version with in-depth commentaries is available at
http://and.intercon.ru/


Game Developers Conference® 2011
February 28 - March 4, 2011 | www.GDConf.com
Directionally Localized Anti-Aliasing
no AA
DLAA

Agenda

- Aliasing & Anti-Aliasing
- Alternative Solutions
- Exploration
- DLAA
- PS3 & X360

The Idea Is ...

Blur Edges Along Their Directions

Blurred. Done !

Aliasing

- Signal Processing
  - Indistinguishable signals when sampled
  - Artifact of reconstruction
- Graphics
  - Pixel "noise"
  - Edge jaggies

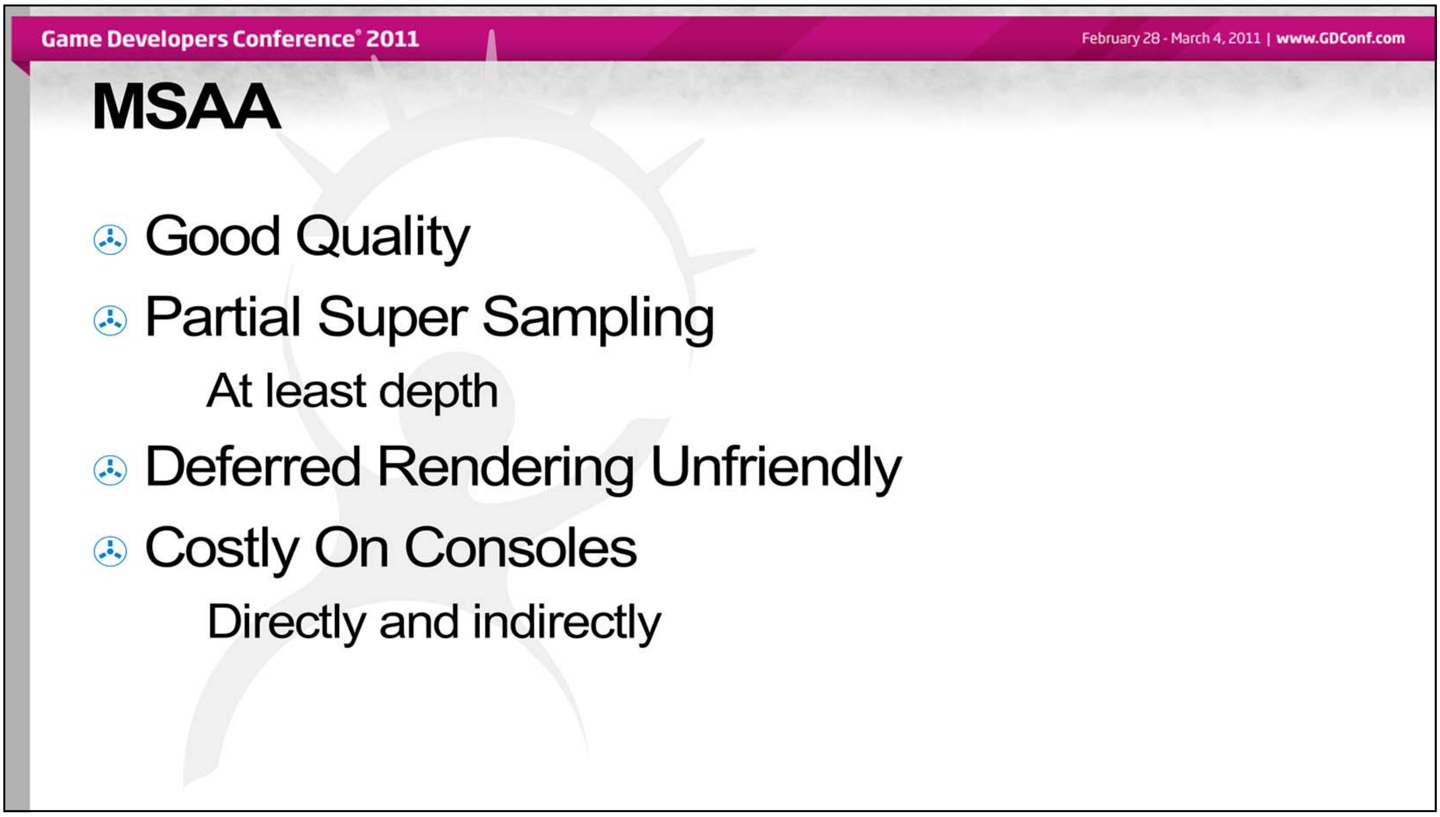
Anti-Aliasing I

- Reduce Higher-Frequencies
- Oversample And "Blur"
  - Temporal in audio
  - Spatial in optics
- No Perfect Filter Exists
  - Sampling theory
  - Sharp (aliased) vs soft (anti-aliased)

Game Developers Conference® 2011 February 28 - March 4, 2011 | www.GDConf.com

Anti-Aliasing II

- Texture
  - Mip-mapping
- Shading
  - Specular, rim lighting
  - Avoid manually
- Geometry Edges
  - Multi-sampling (MSAA)
  - Custom solutions

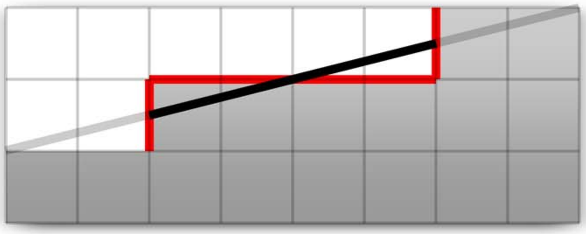

Game Developers Conference® 2011
February 28 - March 4, 2011 | www.GDConf.com
MSAA
Good Quality
Partial Super Sampling
At least depth
Deferred Rendering Unfriendly
Costly On Consoles
Directly and indirectly

Alternatives

- Screen-Space Filtering
  - Perception based
  - Hide jaggies
  - Morphological AA (MLAA)
- Temporal (Crysis 2, Halo)
- Edge-Based AA

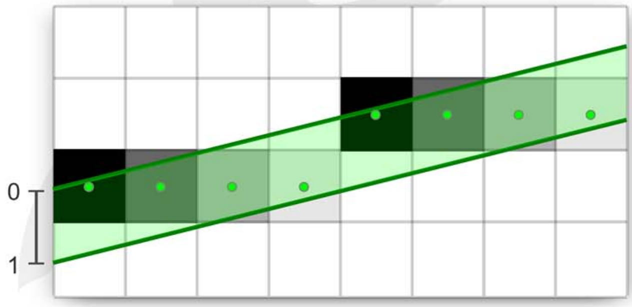
MLAA

- Morphological Anti-Aliasing (Intel)
  - Reconstruct original geometry
  - Re-blend neighbors
- CPU Friendly
  - The Saboteur
  - GoW3 (4ms / 5 SPUs)
- XBox360 GPU (> 3.7 ms)



Edge-Based

- XBox360 SDK Sample
  - Render one-pixel wide polygons
  - Texcoord as pixel coverage
  - Re-blend neighbors



Could Not Use

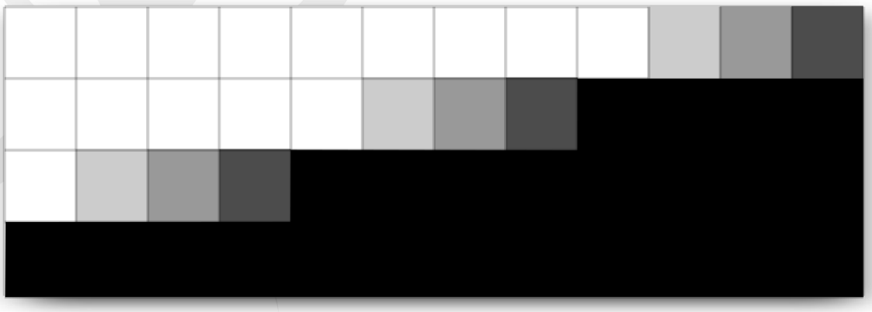
- MLAA
  - Unstable
  - Tough on X360
- Edge-Based
  - Extra GPU cost on PS3
- Temporal
  - Dynamic resolution adjustment in TFU2
  - Motion vs resolution

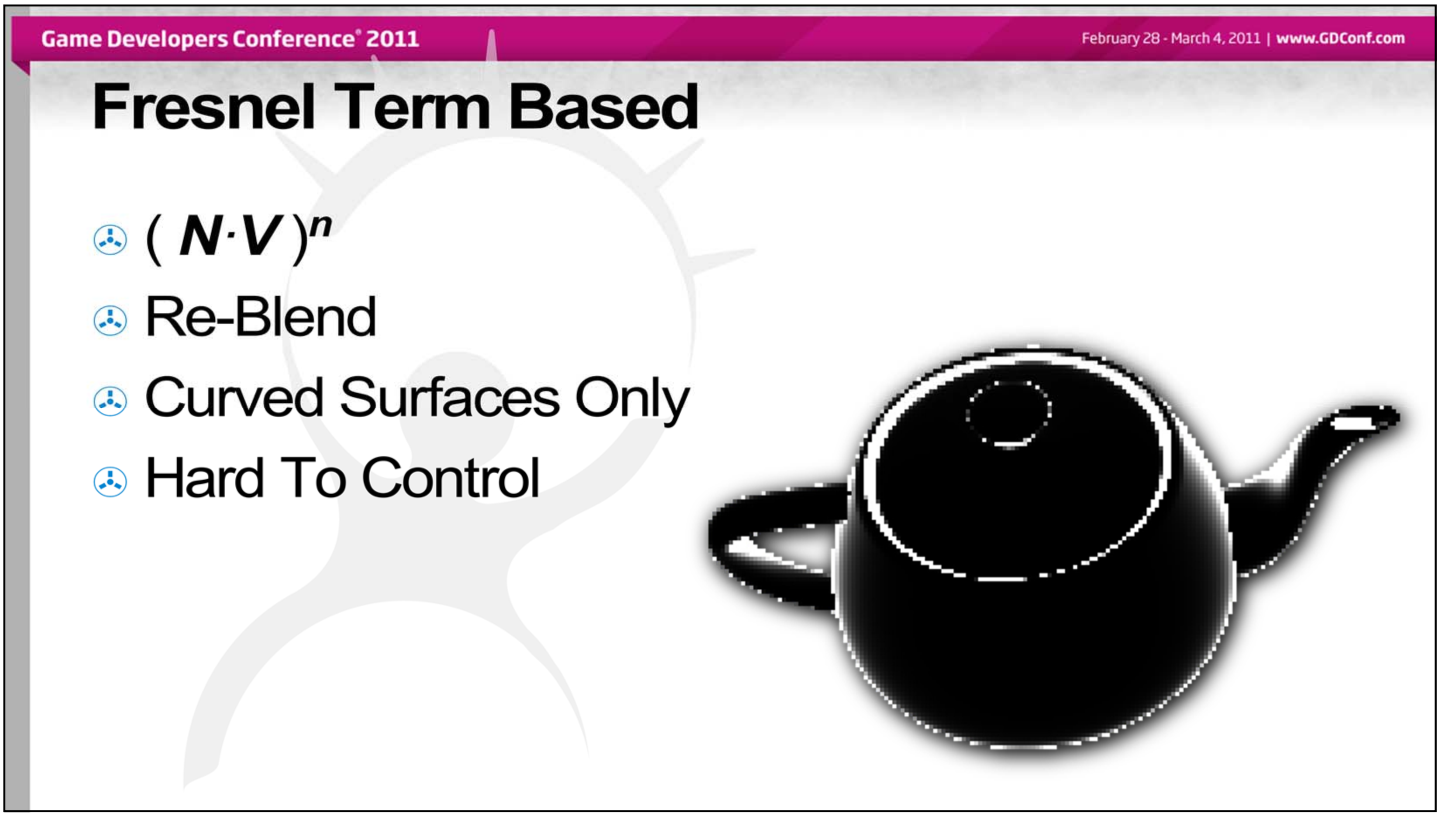
"Ideal" AA Filter

- Multi-Platform
  - GPU, SPU
  - Reliable in production
- Temporally Stable
- Perception Based
  - Hide jaggies
- Good Quality For Low Cost

What If …

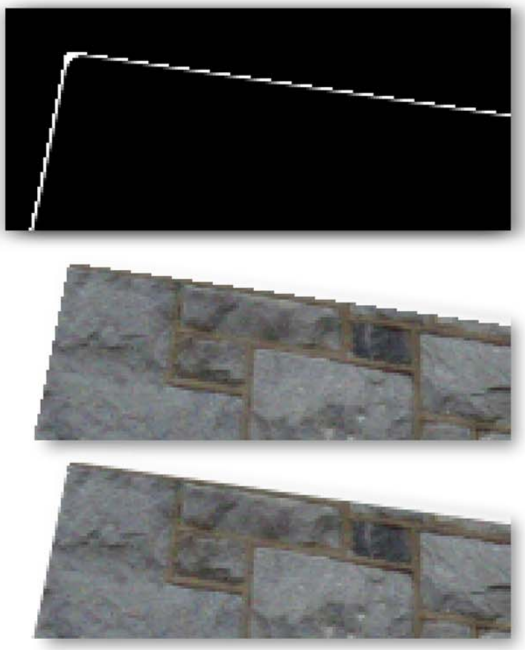
Create Pixel Coverage-Like Look




Game Developers Conference® 2011
February 28 - March 4, 2011 | www.GDConf.com
Fresnel Term Based
(N·V)n
Re-Blend
Curved Surfaces Only
Hard To Control

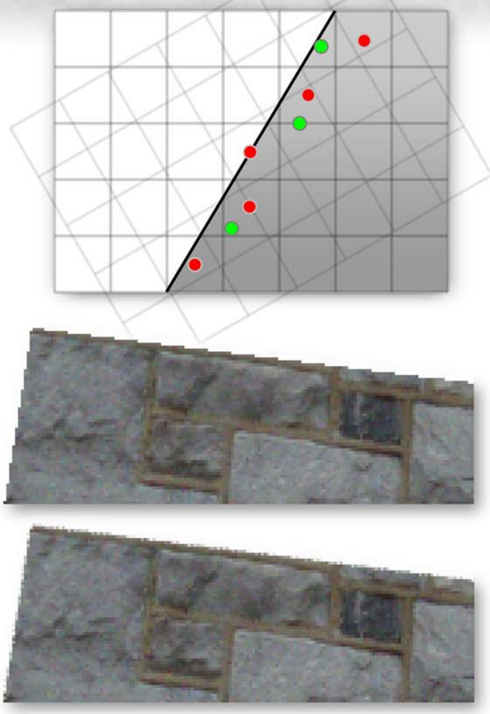
Depth Based Gradients

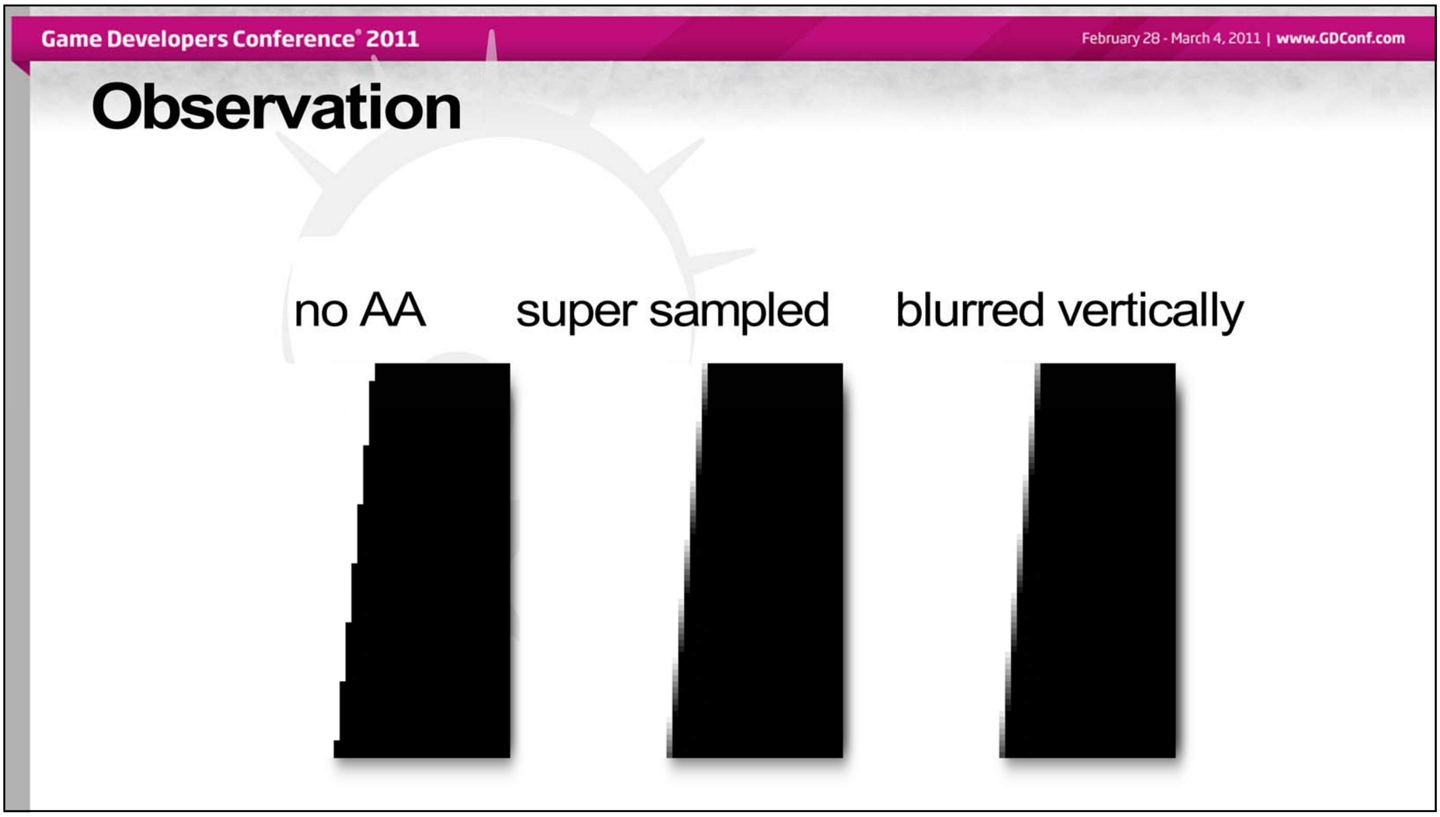
- Find Edge Gradients
  - Depth box-blur
  - Adjust levels locally
- Re-Blend
- Flat Surfaces



Depth Re-Sampling

- Render Alternative Depth
  - Rotated 2nd z-pre pass
  - Or 4x MSAA for depth
- Compute Pixel Coverage
  - Remap depth value
- Re-Blend




Game Developers Conference® 2011
February 28 - March 4, 2011 | www.GDConf.com
Observation
no AA
super sampled
blurred vertically

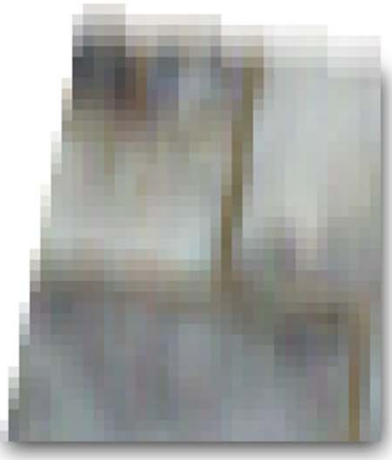
DLAA Prototyping I

- Photoshop
  - Layers vs Pixels
  - Hard to do complex things
  - Easy to implement IF works :)
- Filter / Other / Custom
  - Basic 5x5 convolution
  - Blurs, Edges, etc...

DLAA Prototyping II

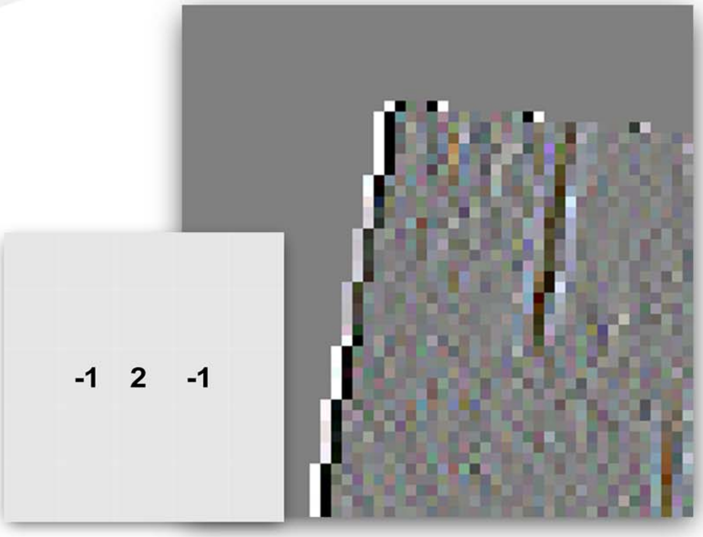
- Blur Vertically

1
1
1
1
1



DLAA Prototyping III

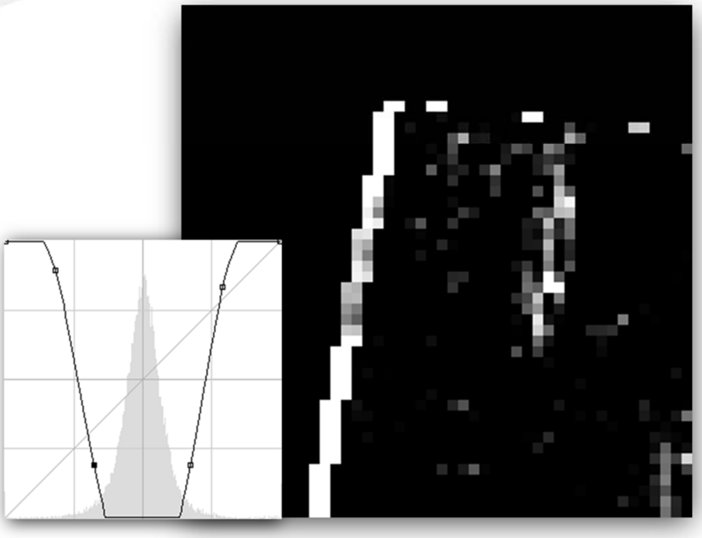
- Blur Vertically
- Find Vertical Edges



DLAA Prototyping IV

- Blur Vertically
- Find Vertical Edges
- Build Edge Mask

  saturate(abs(x) · a – b)

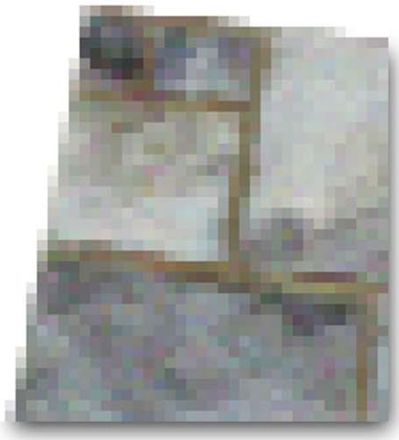


DLAA Prototyping V

- Blur Vertically
- Find Vertical Edges
- Build Edge Mask

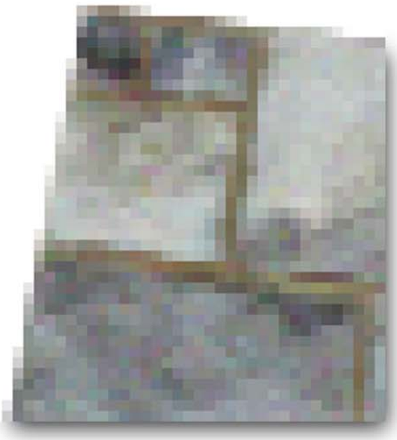
  saturate(abs(x) · a – b)
- Blend With Original Layer
- Same Horizontally

Short Edges Only

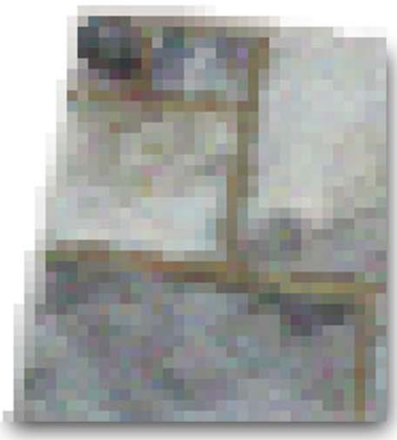


Two Cases

5-Pixel Wide

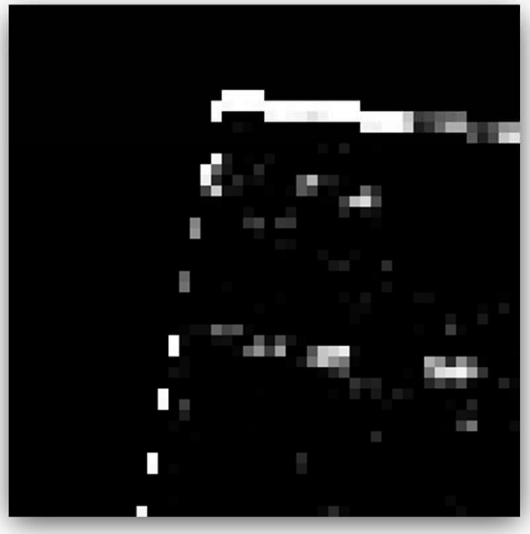


16-Pixel Wide



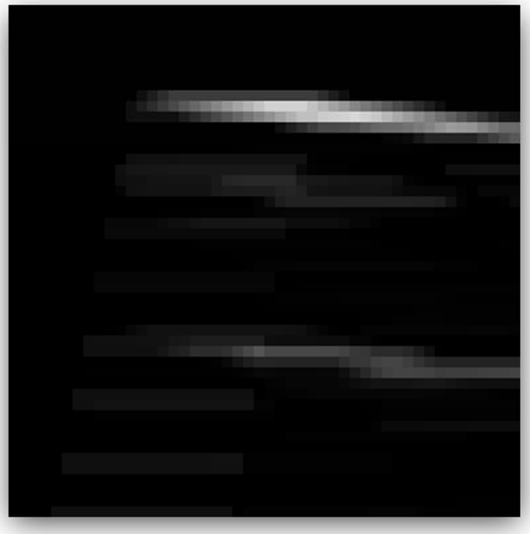
Long Edge Detection I

- Take High-Pass Mask



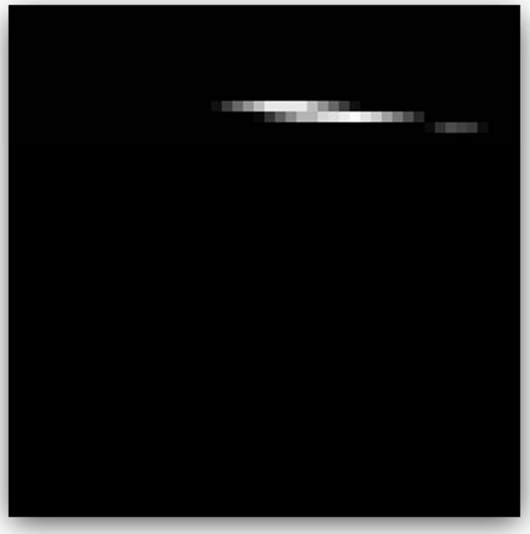
Long Edge Detection II

- Take High-Pass Mask
- Blur



Long Edge Detection III

- Take High-Pass Mask
- Blur
- Adjust Contrast



Long Edge Detection IV

- Take High-Pass Mask
- Blur
- Adjust Contrast
- Apply Long-Edge Filter
  - Where it's needed

Game Developers Conference® 2011

February 28 - March 4, 2011 | www.GDConf.com

Long Edge Filtering I

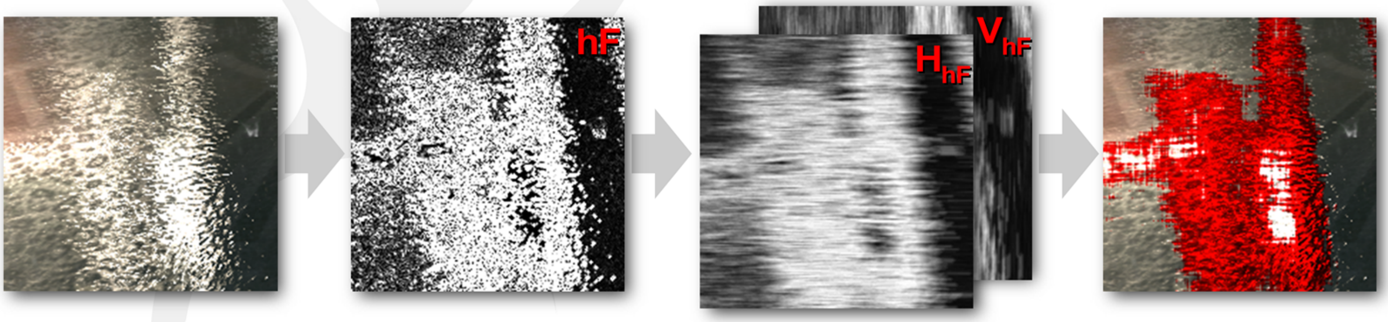
- Color Bleeding

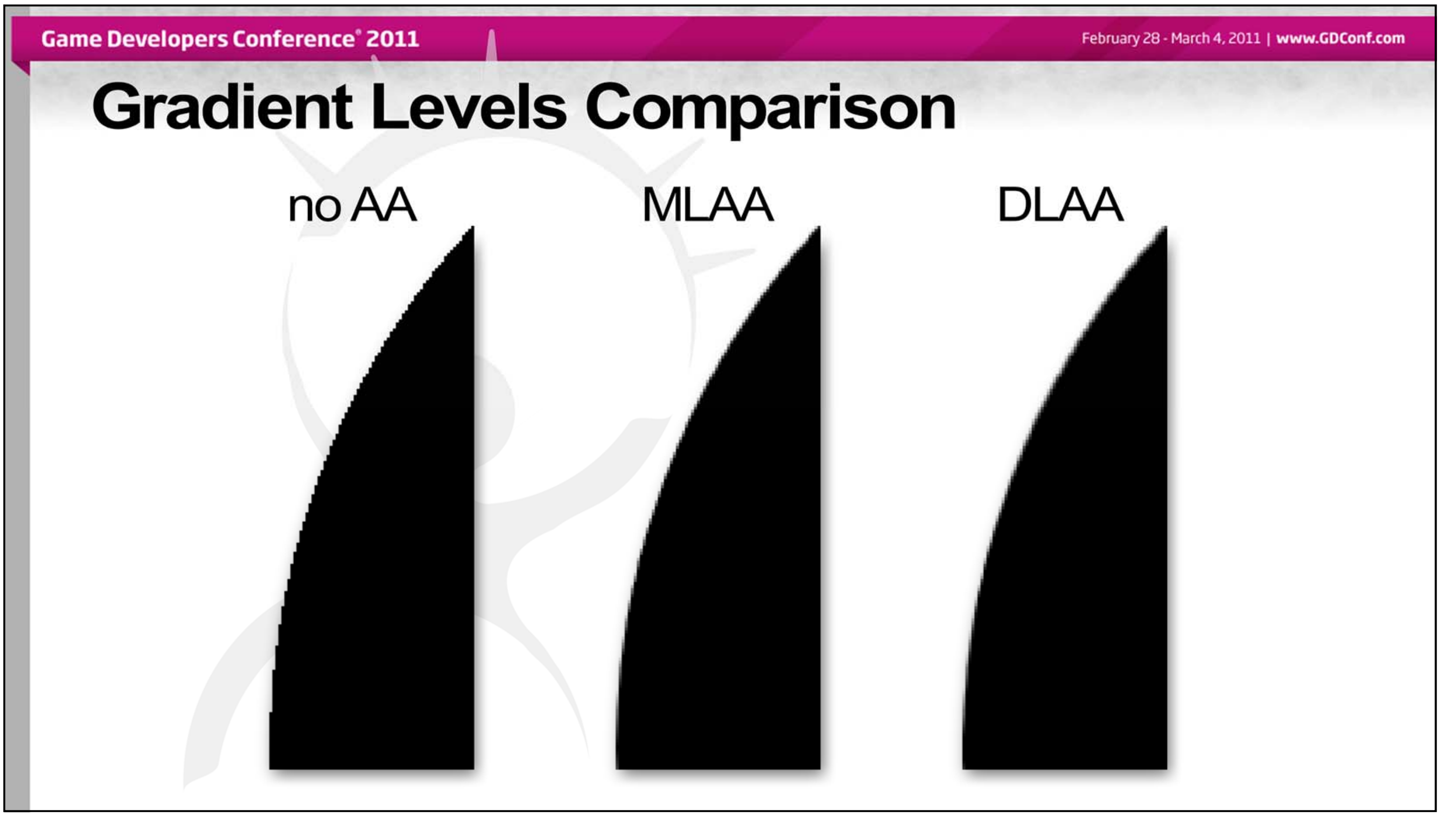
Long Edge Filtering II

- Color Bleeding
- Luminosity Blending Mode
- Blurred luminance As Target
  - Find local pixel that matches it

Noise Level Estimation

- Exclude Noisy Regions
  - Have long vertical and horizontal edges
  - $||H_{hF} - V_{hF}|| > \lambda$




Game Developers Conference® 2011
February 28 - March 4, 2011 | www.GDConf.com
Gradient Levels Comparison
no AA
MLAA
DLAA

Visual Results

Game Developers Conference® 2011 February 28 - March 4, 2011 | www.GDConf.com

Reflections Anti-Aliasing

Execution Results @ 720p

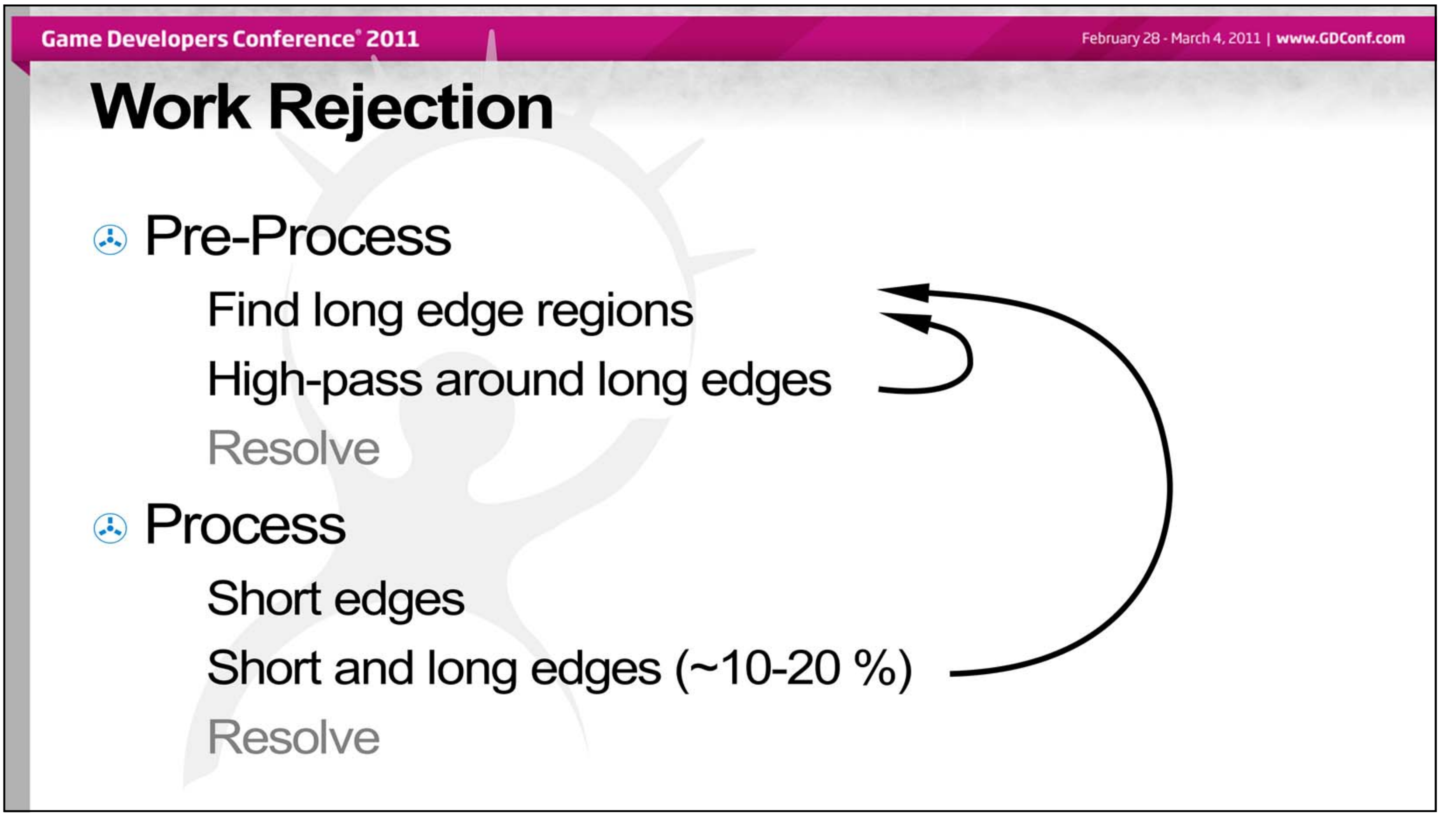
- XBox360 **2.2 ± 0.2 ms**
- PlayStation3 **1.6 ± 0.3 ms** (5 SPUs)

- Project Time

Research	8 weeks (part time)
X360	2 weeks
PS3 (SPU)	> 3 weeks

Implementation Strategies

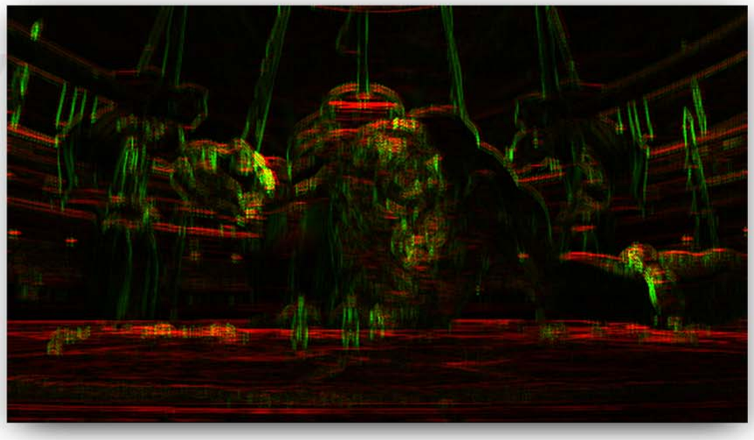
- Execution Time
  - Reuse samples
  - Reject as much work as possible
  - Balance pipelines
- Memory Usage
  - Reuse textures and buffers
  - Pack data by usage
- Global Pipeline Optimizations


Game Developers Conference® 2011
February 28 - March 4, 2011 | www.GDConf.com
Work Rejection
Pre-Process
Find long edge regions
High-pass around long edges
Resolve
Process
Short edges
Short and long edges (~10-20 %)
Resolve

Long Edge Estimation I

- Find Long Axial Edges Directly
  - At lower resolution (e.g. from HDR reduction)

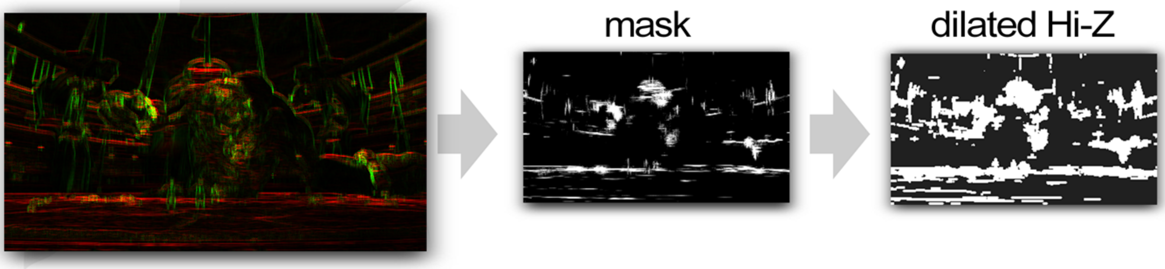
1	1	1	1	1	1
-1	-1	-1	-1	-1	-1



Game Developers Conference® 2011 February 28 - March 4, 2011 | www.GDConf.com

Long Edge Estimation II

- Transfer Into Hi-Z (4x4 pixel blocks)
  - 4x MSAA trick
- Flip Hi-Z Test With Depth Trick
  - Using D3DHIZFUNC



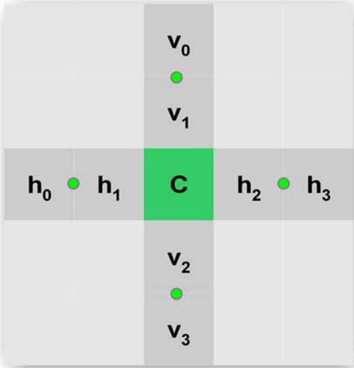
High-Pass Filter

- 5 Bi-Linear Samples
- Around Long Edges Only
- Store In Alpha

1	2	1
2	-12	2
1	2	1

Short Edges

- Low And High-Pass Filters
  - Reuse vertical and horizontal samples
- Normalized Blending Coefficients
  - $t_h = (\lambda \cdot L(edge_h) - \varepsilon) / L(blur_h)$
  - $L(x)$ - intensity function
- Re-Blend
  - $c = lerp(c, blur_h, saturate(t_h))$



Long Edges I

- Sparse Sampling On GPU
  - Reuse short samples
  - Extra 4 bi-linear samples

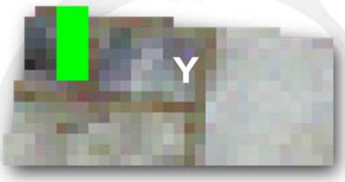
- Discard If Horizontal And Vertical
  - [branch] based on blurred high-pass

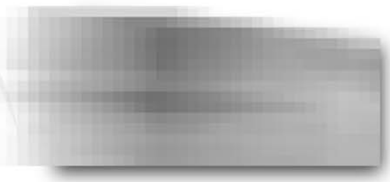
Long Edges II

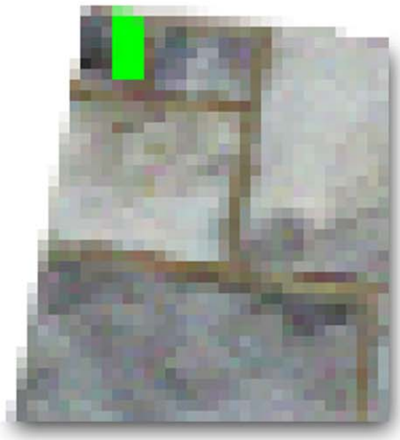
- Find Local Pixel That Matches Blurred Intensity

$$\text{blurred}_{lum} = \text{lerp}(\, \mathbf{X}_{lum}, \mathbf{Y}_{lum}, \mathbf{t} \,)$$

$$\text{color} = \text{lerp}(\, \mathbf{X}, \mathbf{Y}, \mathbf{t} \,)$$







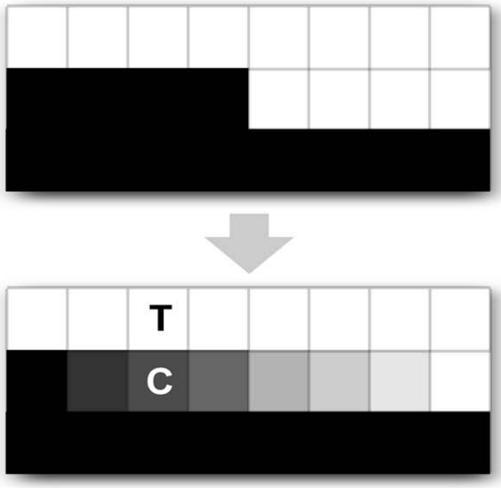
Long Edges III

- Find Local Pixel That Matches Blurred Intensity

  $\text{blurred}_{lum} = \text{lerp}(\, \mathbf{X}_{lum}, \mathbf{Y}_{lum}, \mathbf{t} \,)$

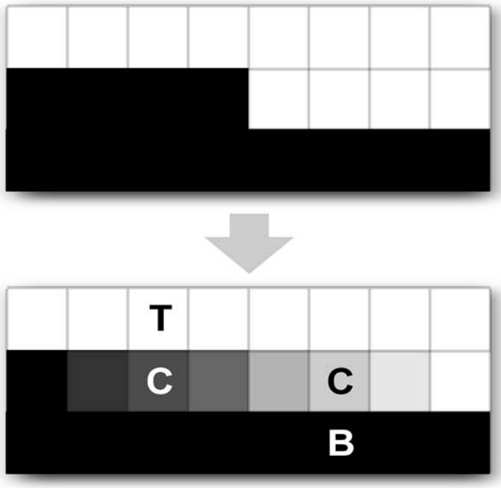
  $\text{color} = \text{lerp}(\, \mathbf{X}, \mathbf{Y}, \mathbf{t} \,)$

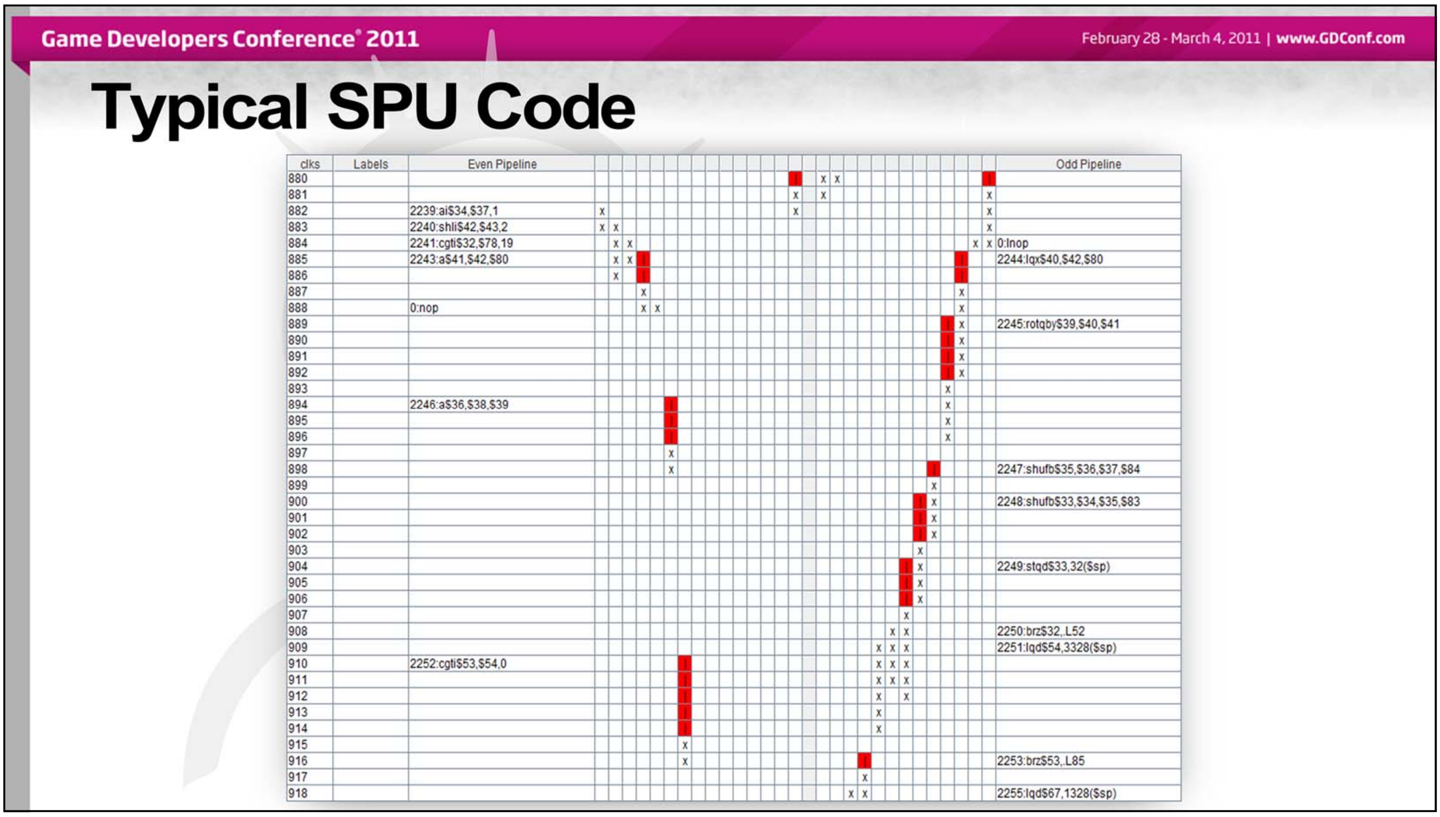
- Two Search Cases



Long Edges IV

- Find Local Pixel That Matches Blurred Intensity
  - $\text{blurred}_{\text{lum}} = \text{lerp}(\ \mathbf{X}_{\text{lum}},\ \mathbf{Y}_{\text{lum}},\ \mathbf{t}\ )$
  - $\text{color} = \text{lerp}(\ \mathbf{X},\ \mathbf{Y},\ \mathbf{t}\ )$
- Two Search Cases
  - Top and bottom neighbors
- Re-Blend
  - Based on longEdgeMask




Game Developers Conference 2011
February 28 - March 4, 2011 | www.GDConf.com
Typical SPU Code
clks
Labels
Even Pipeline
Odd Pipeline
880
881
882 2239:ai$34,$37,1
883 2240:shli$42,$43,2
884 2241:cgti$32,$78,19 0:lnop
885 2243:a$41,$42,$80 2244:lqx$40,$42,$80
886
887
888 0:nop
889 2245:rotqby$39,$40,$41
890
891
892
893
894 2246:a$36,$38,$39
895
896
897
898 2247:shufb$35,$36,$37,$84
899
900 2248:shufb$33,$34,$35,$83
901
902
903
904 2249:stqd$33,32($sp)
905
906
907
908 2250:brz$32,.L52
909 2251:lqd$54,3328($sp)
910 2252:cgti$53,$54,0
911
912
913
914
915
916 2253:brz$53,.L85
917
918 2255:lqd$67,1328($sp)

SPU Post Processing

- Software Pipelining
  - Hide latency
- Balance Even And Odd Instructions
- Stream Processing
- Tiled RSX Surfaces
  - 0.3 ms to copy from VRAM
  - Partial untiling with DMA

DLAA On SPUs I

- No Need to Handle Overlaps
- Short Edges
  - Byte operations → 4 RGBA pixels / clk
  - (1 2 1) = **AVGB**(**AVGB**(l, c), **AVGB**(c, r))
  - $\|x - y\|$ = **ABSDB**(x, y)
- Long Edges
  - blur(x) = $\sum f(x + dx)$
  - blur($x + 1$) = blur(x) $\boldsymbol{- f(\, x - r\,) + f(\, x + 1 + r\,)}$

DLAA On SPUs II

- Quick Luminance

  SUMB (G, R, G, B) → 0.25 R + 0.5 G + 0.25 B

- Quick Saturate

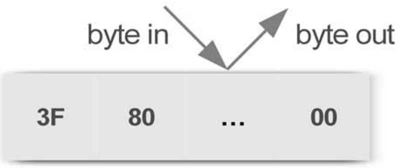
  CFLTU x, x, 32; **CUFLT** x, x, 32

- Quick Interpolation

  r = lerp(x, y, t)

  FS r, Y, X **SHUFB** X, x, _, _

  FMA r, t, r, X **SHUFB** Y, y, _, _



Typical SPU Code

clks	Labels	Even Pipeline	Odd Pipeline
880			
881			
882		2239:ai$34,$37,1	
883		2240:shli$42,$43,2	
884		2241:cgti$32,$78,19	0:lnop
885		2243:a$41,$42,$80	2244:lqx$40,$42,$80
886			
887			
888		0:nop	
889			2245:rotqby$39,$40,$41
890			
891			
892			
893			
894		2246:a$36,$38,$39	
895			
896			
897			
898			2247:shufb$35,$36,$37,$84
899			
900			2248:shufb$33,$34,$35,$83
901			
902			
903			
904			2249:stqd$33,32($sp)
905			
906			
907			
908			2250:brz$32,.L52
909			2251:lqd$54,3328($sp)
910		2252:cgti$53,$54,0	
911			
912			
913			
914			
915			
916			2253:brz$53,.L85
917			
918			2255:lqd$67,1328($sp)

Efficient SPU Code

clks	Labels	Even Pipeline	Odd Pipeline
690		1973:absdb$87,$96,$37	1974:shufb$57,$24,$24,$41
691		1975:sumb$68,$90,$91	1976:shufb$58,$23,$23,$41
692		1977:avgb$73,$83,$84	1978:shufb$40,$36,$36,$41
693		1979:avgb$71,$80,$81	1980:shufb$76,$89,$89,$41
694		1981:absdb$74,$88,$122	1982:shufb$75,$87,$87,$41
695		1983:absdb$70,$77,$127	1984:shufb$39,$37,$37,$41
696		1985:clgthi$69,$68,60	1986:lqd$21,0($46)
697		1987:absdb$64,$71,$105	1988:shufb$38,$122,$122,$41
698		1989:sumb$65,$75,$76	1990:shufb$63,$74,$74,$41
699		1991:absdb$67,$73,$101	1992:shufb$62,$70,$70,$41
700		1993:selb$35,$68,$50,$69	1994:shufb$15,$127,$127,$41
701		1995:sumb$33,$57,$58	1996:shufb$55,$64,$64,$41
702		1997:clgthi$66,$65,60	1998:shlqbii$34,$35,2
703		1999:sumb$27,$62,$63	2000:shufb$56,$67,$67,$41
704		2001:selb$5,$65,$50,$66	2002:lqd$16,0($49)
705		2003:rotmahi$18,$33,-2	2004:lqd$121,16($46)
706		2005:or$3,$34,$35	2006:shlqbii$26,$5,2
707		2007:sumb$7,$55,$56	2008:lqd$4,0($48)
708		2009:clgthi$32,$27,60	2010:shufb$115,$3,$3,$42
709		2011:sumb$14,$39,$40	2012:shufb$124,$18,$18,$42
710		2013:selb$6,$27,$50,$32	2014:shufb$10,$101,$101,$41
711		2015:or$9,$26,$5	2016:shufb$19,$105,$105,$41
712		2017:clgthi$25,$7,60	2018:shlqbii$8,$6,2
713		2019:rotmahi$11,$14,-2	2020:shufb$98,$9,$9,$42
714		2021:selb$119,$7,$50,$25	2022:hbrr.L98,.L62
715		2023:rothmi$12,$21,-8	2024:lqd$113,16($49)
716		2025:sumb$13,$15,$38	2026:shufb$17,$24,$18,$51
717		2027:rothi$2,$115,8	2028:shufb$22,$23,$18,$52
718		2029:andhi$20,$21,255	2030:lqd$103,32($46)
719		2031:or$79,$8,$6	2032:lqd$28,16($48)
720		2033:sfh$125,$20,$16	2034:shlqbii$118,$119,2
721		2035:rotmahi$123,$13,-2	2036:shufb$106,$11,$11,$42
722		2037:rothmi$109,$121,-8	2038:shufb$80,$79,$79,$42
723		2039:sfh$117,$12,$4	2040:lqd$84,48($46)
724		2041:sumb$110,$19,$10	2042:lqd$96,32($49)
725		2043:or$126,$2,$124	2044:lqd$76,48($49)
726		2045:rothi$29,$98,8	2046:lqd$91,32($48)
727		2047:ah$120,$124,$125	2048:lqd$70,48($48)
728		2049:andhi$114,$121,255	2050:stqd$22,0($47)

Conclusion

- DLAA

XBox360	**2.2 ± 0.2 ms**
PlayStation3	**1.6 ± 0.3 ms** (5 SPUs)

- End Of Console Life Cycle
  - Every millisecond counts
  - Tricks are inevitable
  - Different solutions & different thinking

Acknowledgments

Szymon Swistun

Ruslan Abdikeev

Axel Wefers

Jerome Scholler

Tom Madams

Anti-Aliasing Community

Thank You

Thank You

Questions ?

dandreev@LucasArts.com