GPUVision

Image Processing on the GPU

Michael Lehr *michael.lehr@gmail.com*
Ikkjin Ahn *ikkjin@gmail.com*
Paul Turner *turnerpd@seas.upenn.edu*
May 5, 2005
CIS 700 – GPU Programming and Architecture
University of Pennsylvania

Overview

- Purpose/Description
- Related Work
  - Basis
- Design
- Supporting Filters – Results – Performance
  - Convolutions
  - Edge/Feature Point Detection
  - Matrix/Vector Dense/Sparse mult/sum/max
    - Solver – Conjugate Gradient
  - Image Segmentation
  - Disparity Map
- Problems
- Future

Most Tests on 2 GHz Athlon, NVidia Quadro 3400 PCIe

Purpose/Description

"To create a windows based GPU Accelerated Image Processing Framework"

- Users
  - Filter users
    - no CG knowledge
    - Template based graphics knowledge
  - Filter creator
    - No RenderTexture knowledge
    - Template based Filter Creation
      - Can create new filter structure (not CG) in under 5 min
        - » Focus on CG code

Related Work

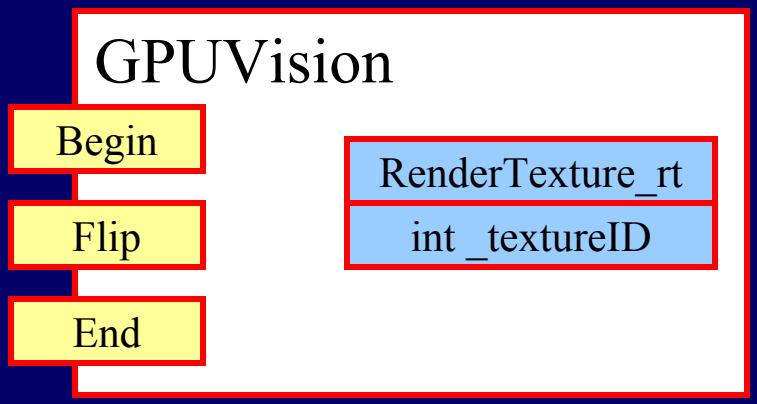
- OpenVidia
  - Linux based
  - Video processing computer vision
  - Some of the Algs are actually not quite right
  - Some CG code is portable
- Mac OSX ‘Tiger’
  - Core Image
    - Found after we came up with structure and Concept of Operations
    - Only on Mac
    - Very very similar
      - I guess we have a good structure =)
      - ‘Image Units’ versus ‘Filters’
- PyFx : Python IP
  - Very well abstracted

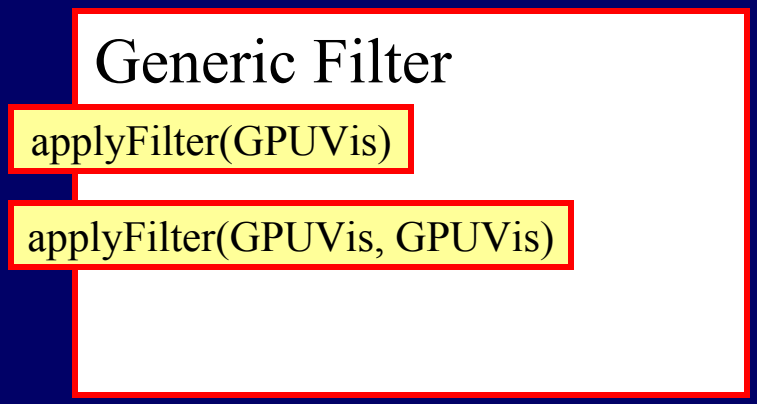
Basis

- Image Segmentation
  - "Isoperimetric Graph Partition for Data Clustering and Image Segmentation" Leo Grady and Eric Schwartz, 2003
- Harris Corners
  - "A Combined Corner and Edge Detector" Chris Harris & Mike Stephens, 1988
- Color spaces
  - http://www.couleur.org/index.php?page=transformations
- Various other sources from computer vision (non GPU)

GPUVision Design -Framework

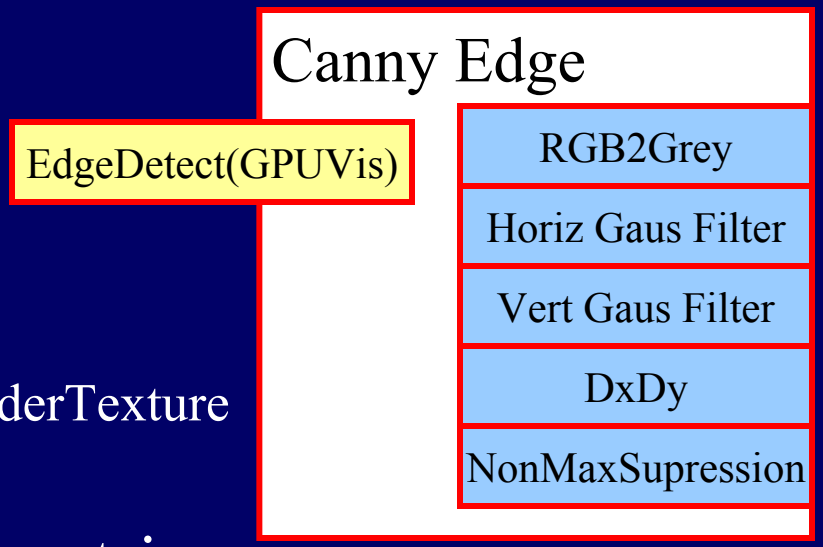
- GPUVision class encapsulates
  - Up/down textures to GPU
  - Ping/Pong
  - Drawing to screen
- Generic Filter is basis for all filters
  - Filters hold CG code
  - Generally can be applied to one or two GPUVis
    - Different for each filter
    - Would like to make this more user-friendly
  - A GPUVision is applied to a Filter
  - GPUVision RenderTexture has results of filter
  - Filters can be chained



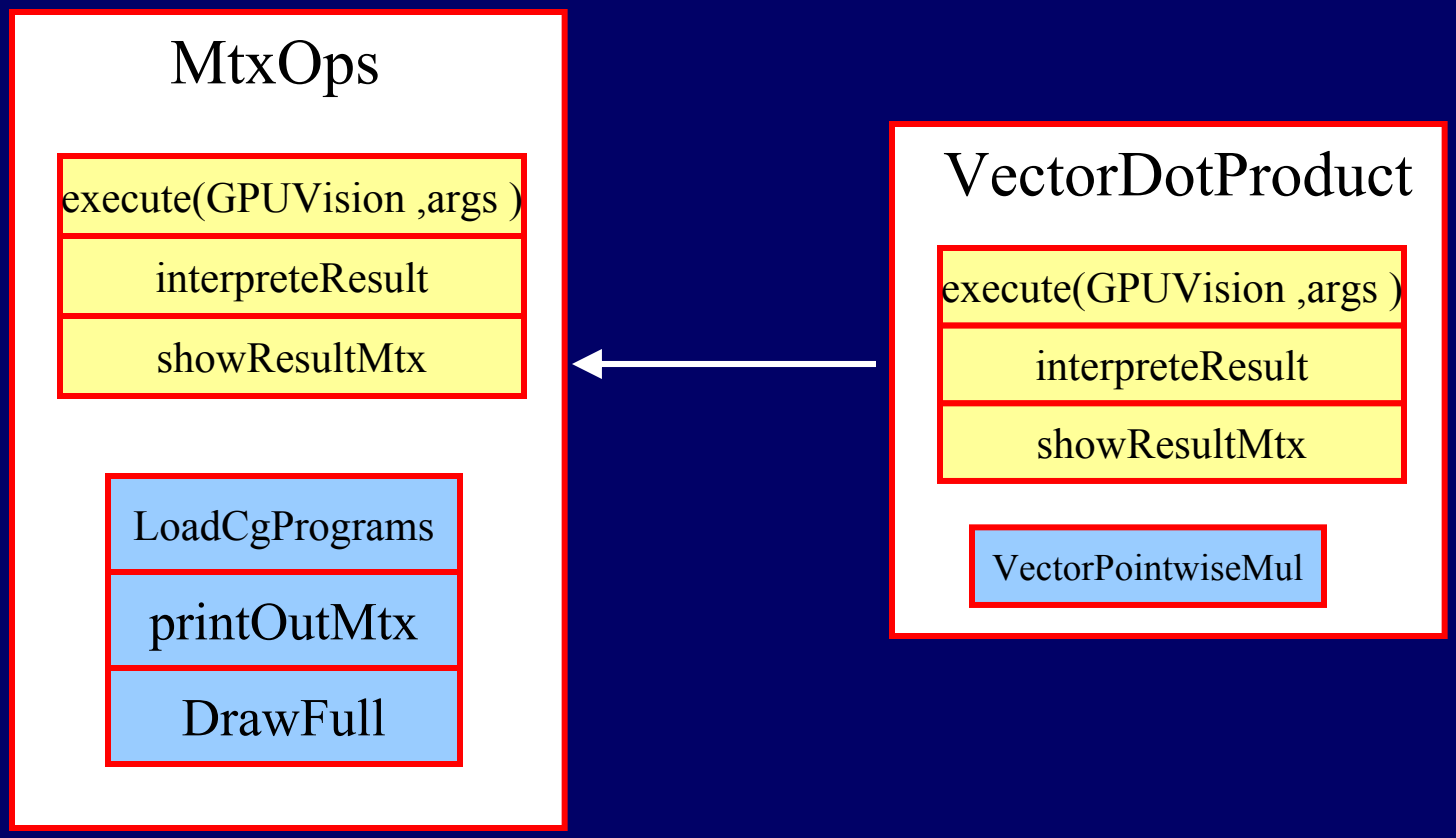


GPUVision Design - Filters

- Can make more complex ‘Filters’ which are pre-defined sequence of filters
- Canny Edge uses 5 filters
  - Only one context switch
  - 5 ‘Flips’
    - Flip resets Read/Write and RenderTexture Target Buffer
- Can create any single filter or string filters together to make complex composite filters



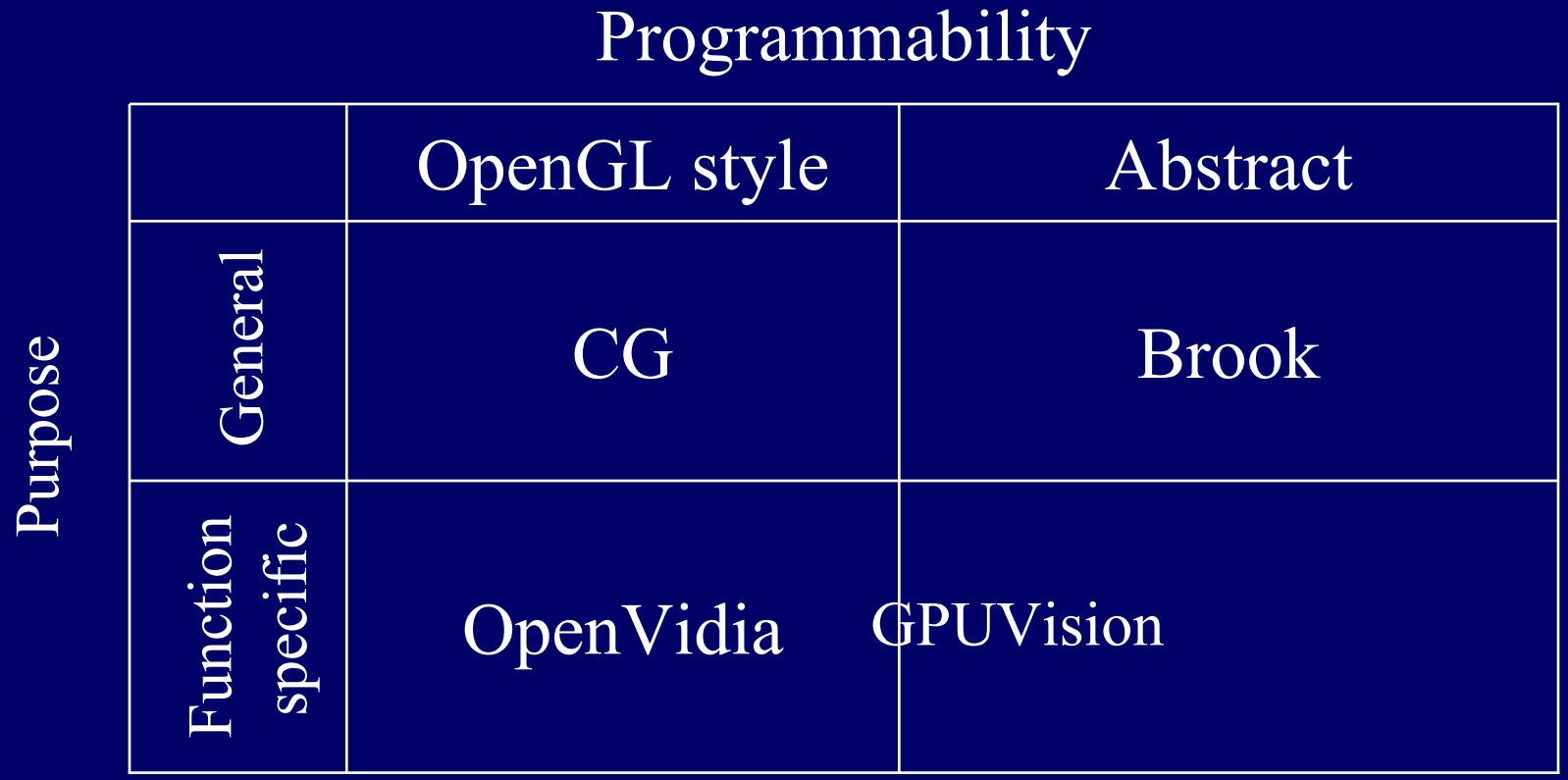
GPUVision Design – Matrix Ops



GPUVision: Amt of Code

- Total
  - 150 files
  - 27 filters
  - 20 Matrix functions
  - 22 test class

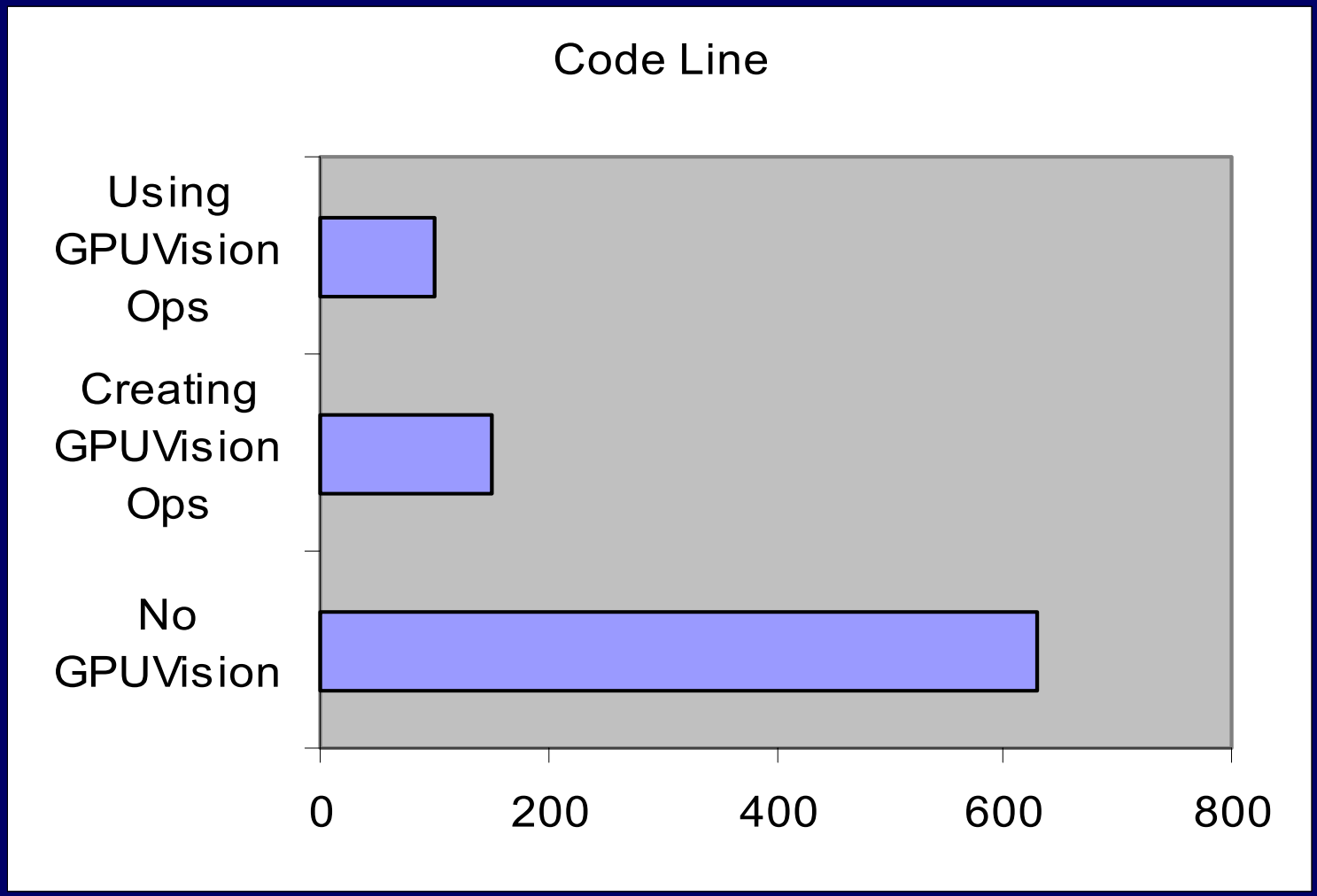
GPUVision vs OpenNvidia



Real Code Example

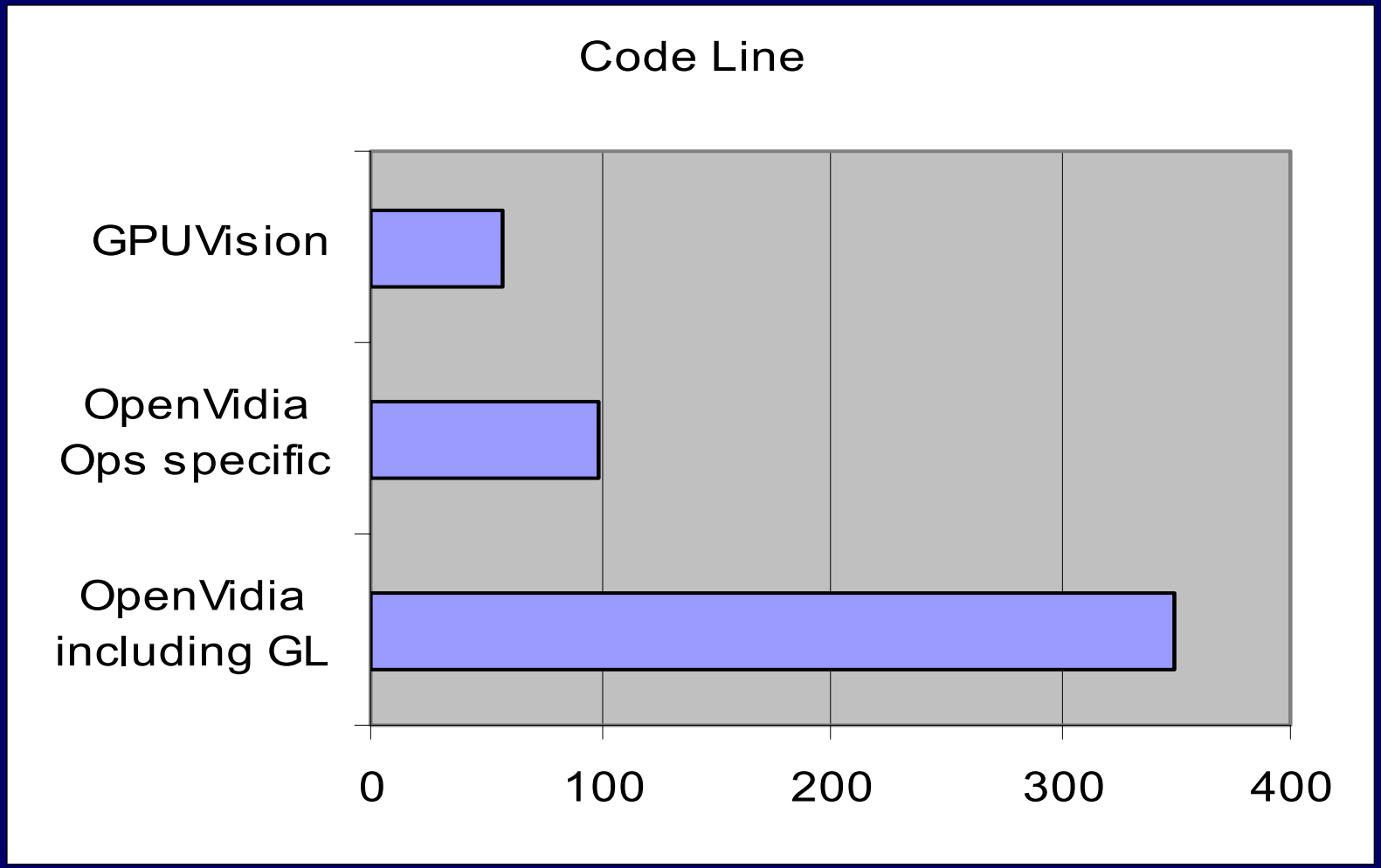
```
static void Display()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    gpuVis = new GPUVision((unsigned int)0,n, m );

    simpMulSparseMtx = new IsoSimplifiedMulSparseMtx(gpuVis->GetContext());

    preProcessing();

    texIDx=gpuVis->UploadToTexture(data_x, n, m, GL_RGBA);
    texIDa=gpuVis->UploadToTexture(data_a, n, m, GL_RGBA);
    texIDremov=gpuVis->UploadToTexture(data_remov, 1, 1, GL_RGBA);

    static float theta = 0;

    simpMulSparseMtx->execute(gpuVis, texIDa, texIDx, texIDremov);

    printf("########Result ##########\n");
    simpMulSparseMtx->showResultMtx(gpuVis, 10, 10);;

    glutSwapBuffers();
    glutPostRedisplay();

    system("pause");
    exit(0);
}
```

GPUVision CodeCount



- In case of Sparse matrix multiplication.

GPUVision vs OpenVidia



In case of Canny Edge Detector

Support Filters

- Add, Subtract, Multiply, Threshold
  - Can take in a number to add/sub/mult or can do pointwise add/sub/mult of two textures
  - Not for performance but for ease
    - Could easily combine these into custom filters
    - Good for proof of concept of filter chains and working with the GPU
- Very very fast…. but, not very useful by themselves..

Kernel Based Filters

- Allow kernels of any shape in ConvolutionFilter
  - Generate the CG code on the fly
  - Can either pass in the array into the cg code (useful if it may change on the fly)
  - Or can write it into the code
    - Currently only do this for small kernels
  - Ex: 150 → 153 fps for putting in code
- Single Filters
  - Blur, Gaussian of Laplacian, Gabor, Packed Convolution, Convolution
- Composite Filters
  - Gaussian and Laplacian Pyramids, Sharpen, Laplacian

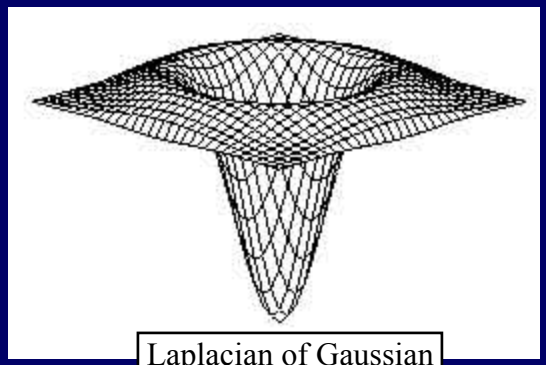
Blur & Kernels in General

- 2 Vectors vs 1 square?
  - For small kernels (around 5 or less) squares are more efficient
  - Otherwise 2 Vectors
  - Code versitle enough to handle anything

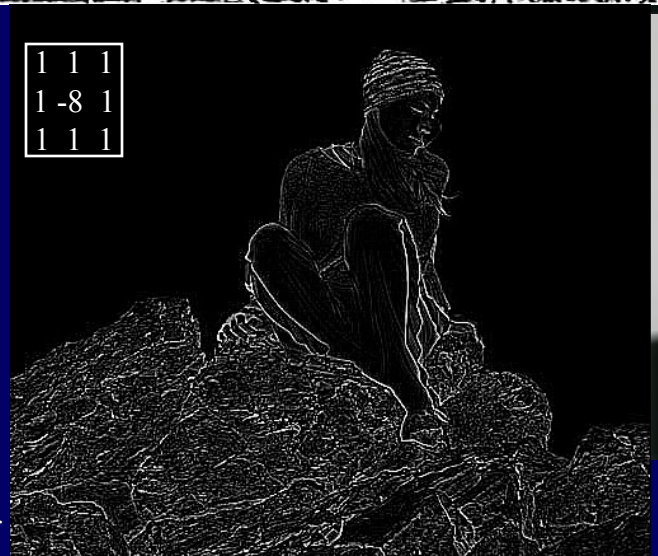
```
_gauss5x5h = new ConvolutionFilter(_gauss5xKernel, 5,1,3,context);
_gauss5x5v = new ConvolutionFilter(_gauss5xKernel, 1,5,3,context);
_gauss3x3  = new ConvolutionFilter(_gauss3x3Kernel,3,3,3,context);
```

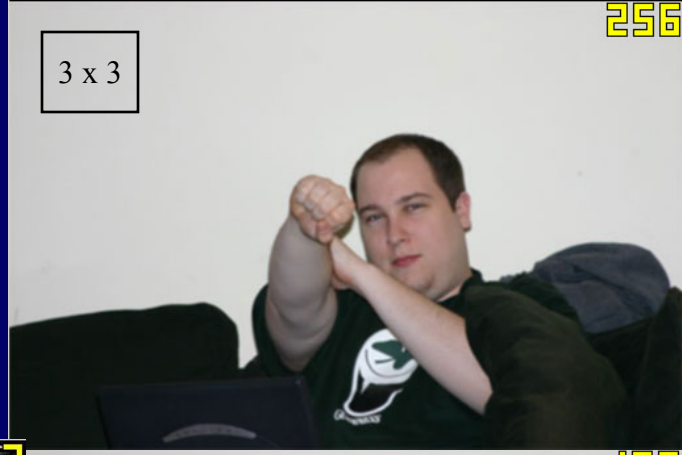
  - params are

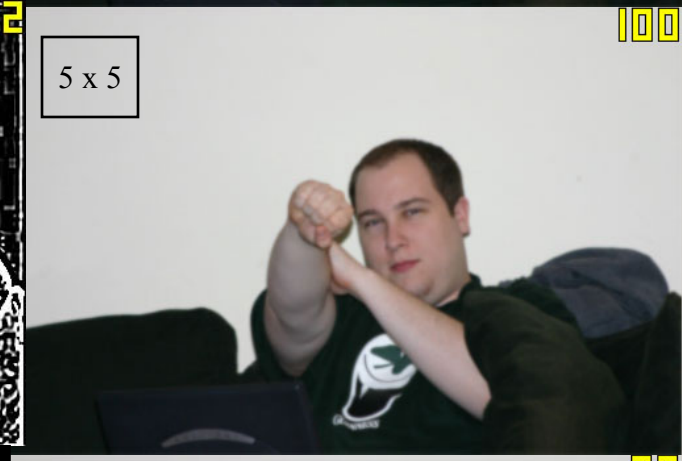
```
Kernel(float*), width, height, channels, CGContext
```

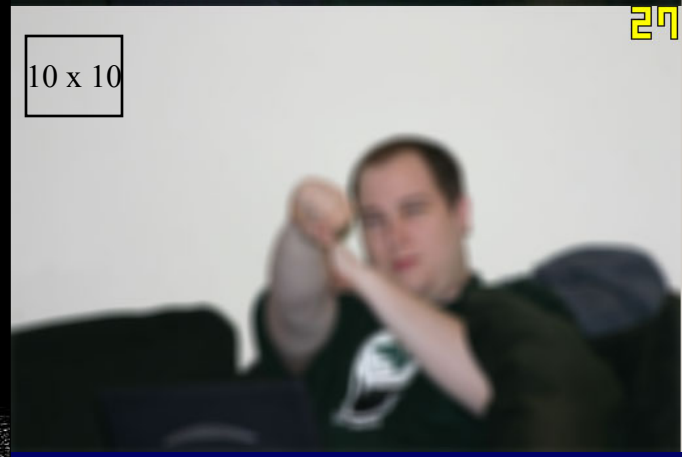


Laplacian of Gaussian
(Mexican Hat)



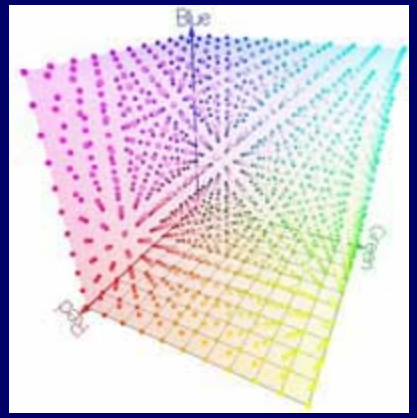


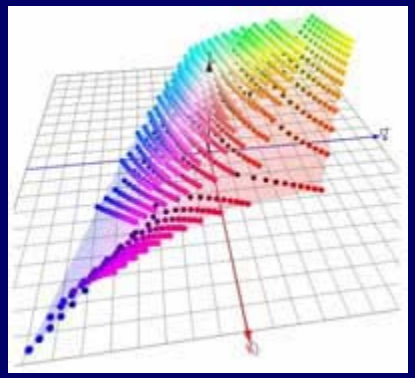




Packing

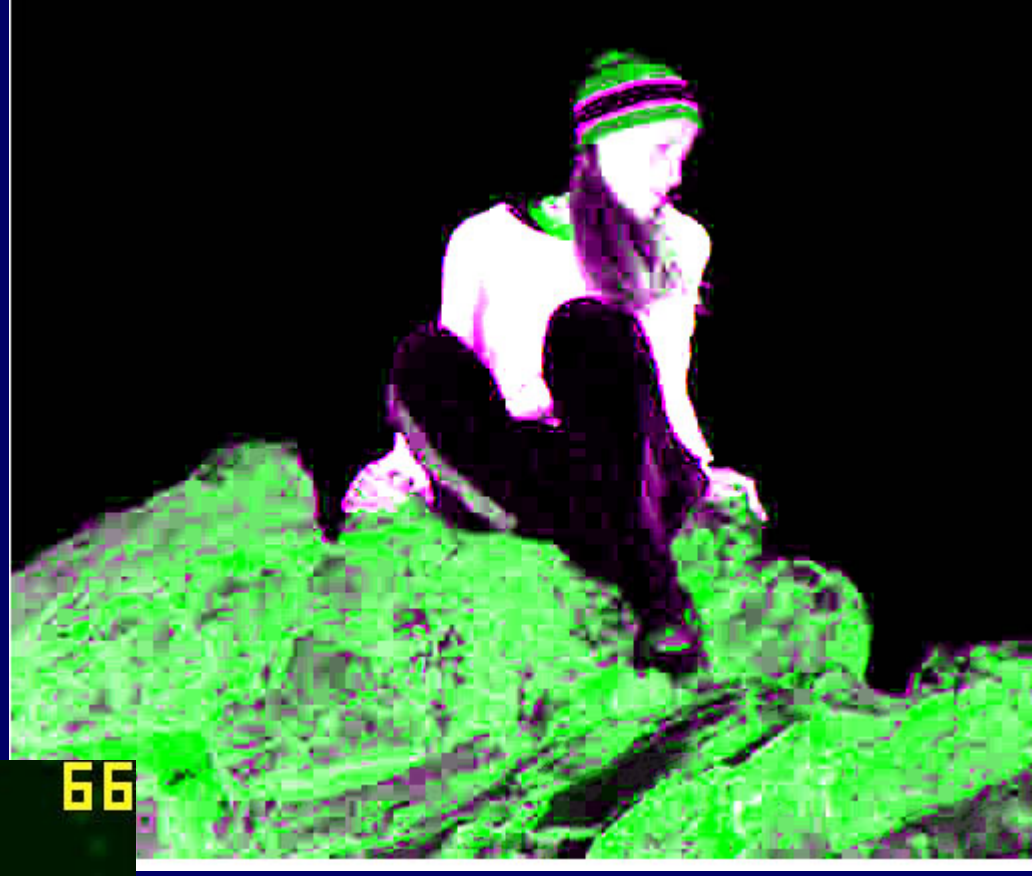
- Convert RGB → XYZ → CIE LAB
- Take AB from LAB color channel and can use all for channels for neighboring pixel
  - Image is ½ the size
  - Can process 2 pixels values per packed pixel
  - For 5x5 goes from 25 texture lookups per pxl to 7.5 per pxl
- Pack 4 into 1 you reduce that to 2¼ tex lookups per pixel





(3*5)/2 = (width*height)/numPxls

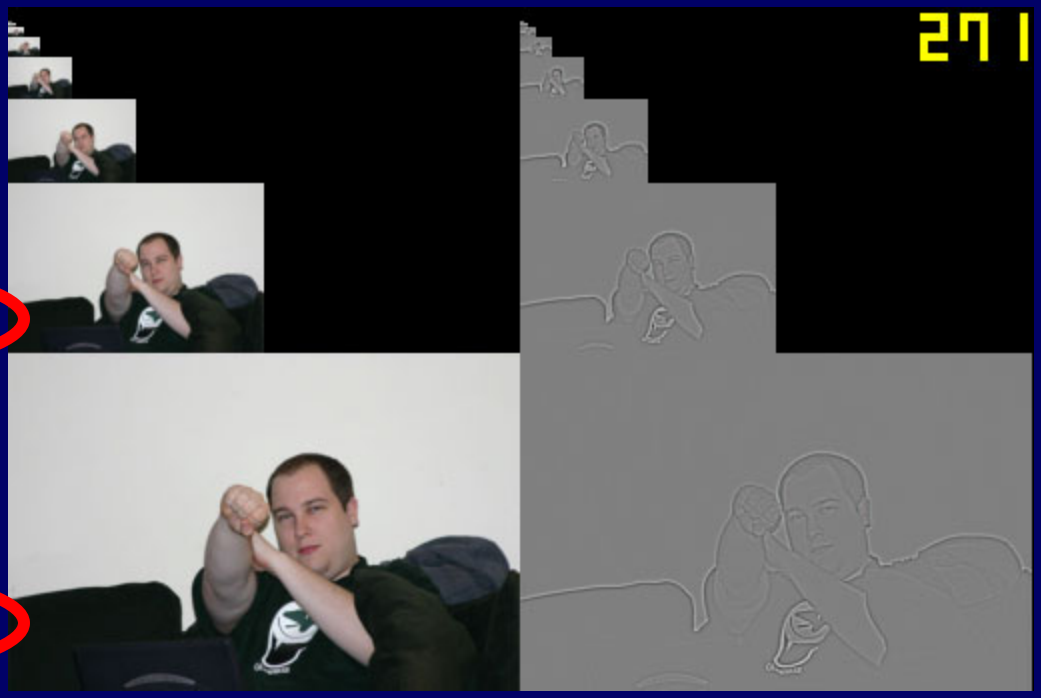
5x5 Gaus blur	FPS	% Packed Faster
1 blur Pack	186	
1 blur Normal	101	184%
2 blur Pack	117	
2 blur Normal	53	220%
3 blur Pack	85	
3 blur Normal	36	236%
4 blur Pack	67	
4 blur Normal	27	248%



- These are AB Channels
- Times for AB counts packing and unpacking images

Gaussian and Laplacian Pyramids

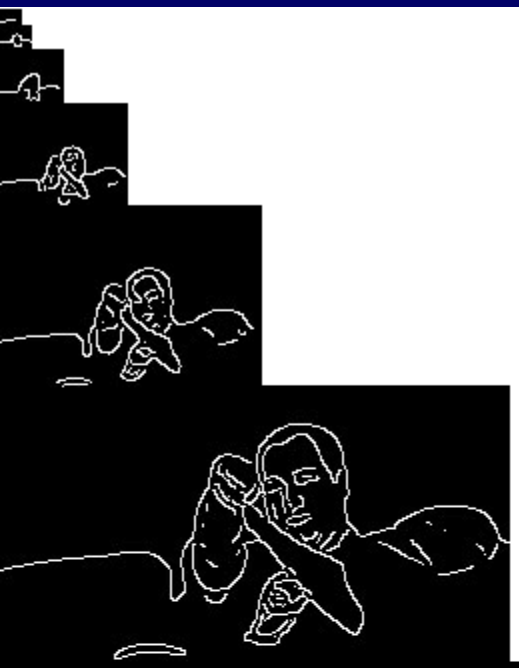
Gaussian	GPU	MipMap	CPU
create RT	.277	--	--
create	.00265	--	.01375
ul+create	.0086	.14	.01375
ul+cr+dl	.01485	.15	.01375
Laplacian & Gaussian			
create	.0054	--	.094
ul+create	.0104	--	.094
ul+cr+dl	.01859	--	.094

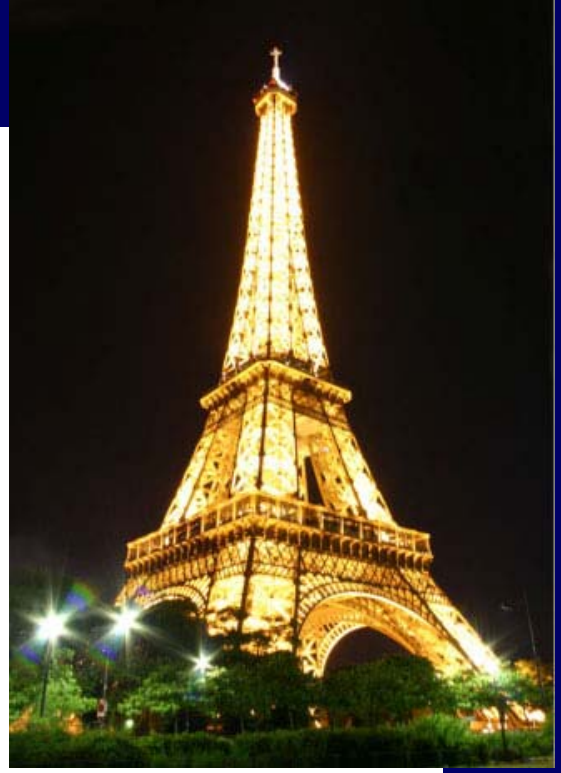


- Close to Matlab Gaus Pyramid Performance!
  - Demolished the Matlab Laplacian performance but…
    - Could not find efficient Laplacian Matlab code
- Once RTs are made, demolish regular mipmapping
- Am downloading ALL levels of the pyramid

Canny

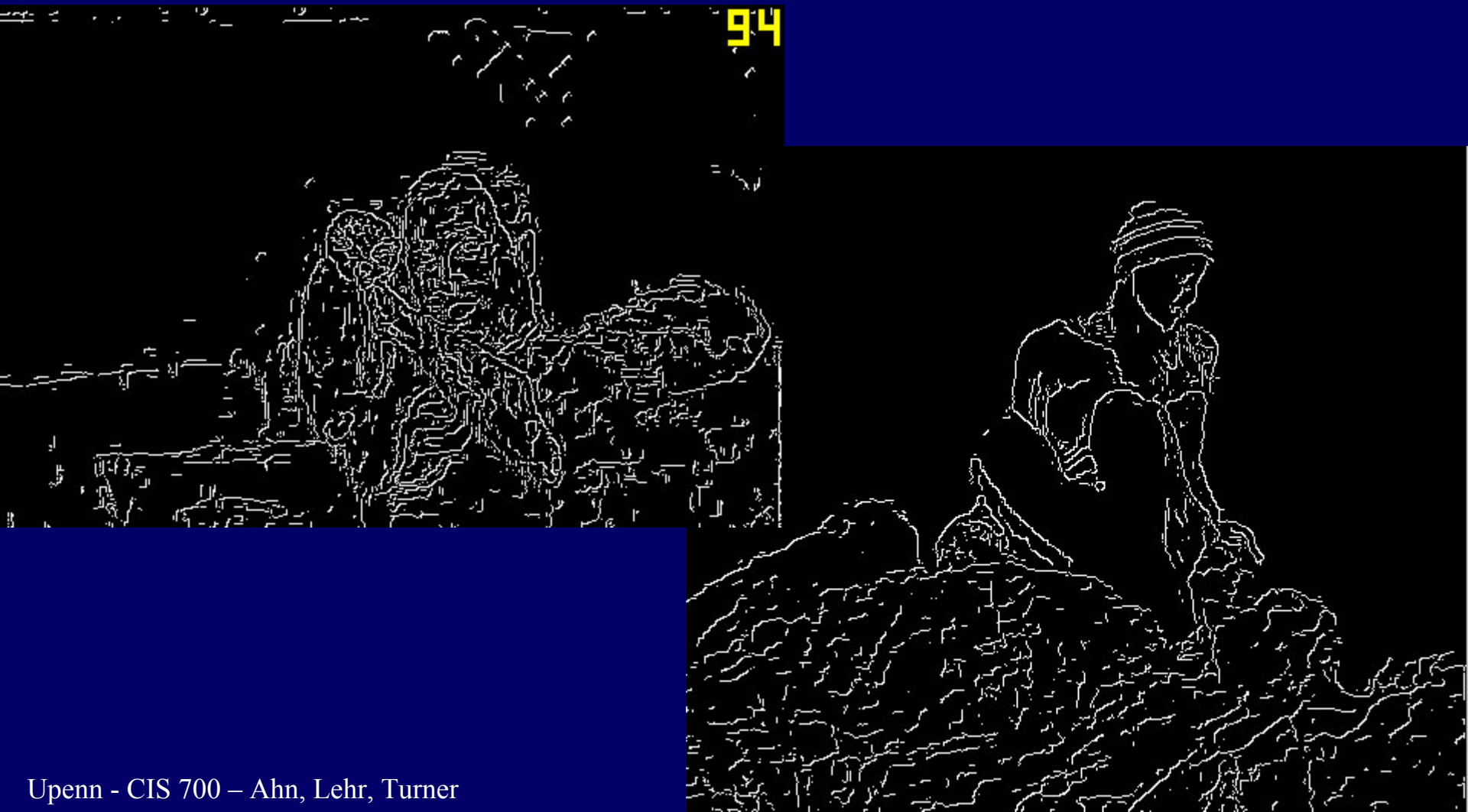
- Described earlier
  - Greyscale (Y from YUV colorspace)
  - Blur Image using 5x vertical and horizontal
  - Find X and Y magnitudes
    - Can find magnitude and orientation of edges
    - Threshold
  - NonMaxSupression
    - Convert multi-pixel line into single pixel
      - Only shrinks within an area – 5 pxl difference will create 2 lines



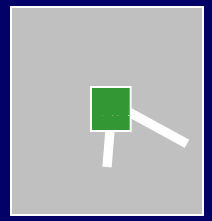


98

Packed AB Canny



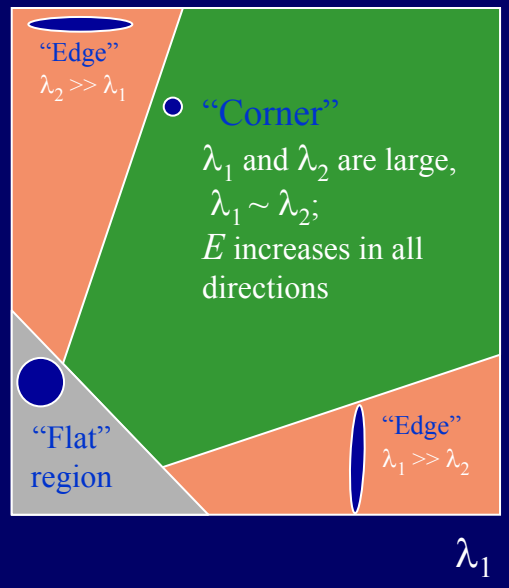
Harris Corner Detector



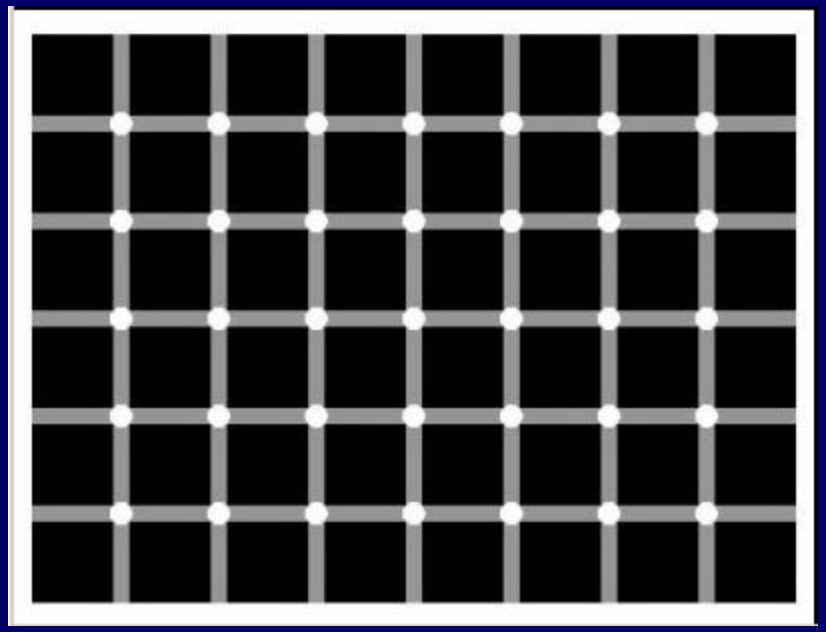
- Find corners in a scene
- Solve following equation

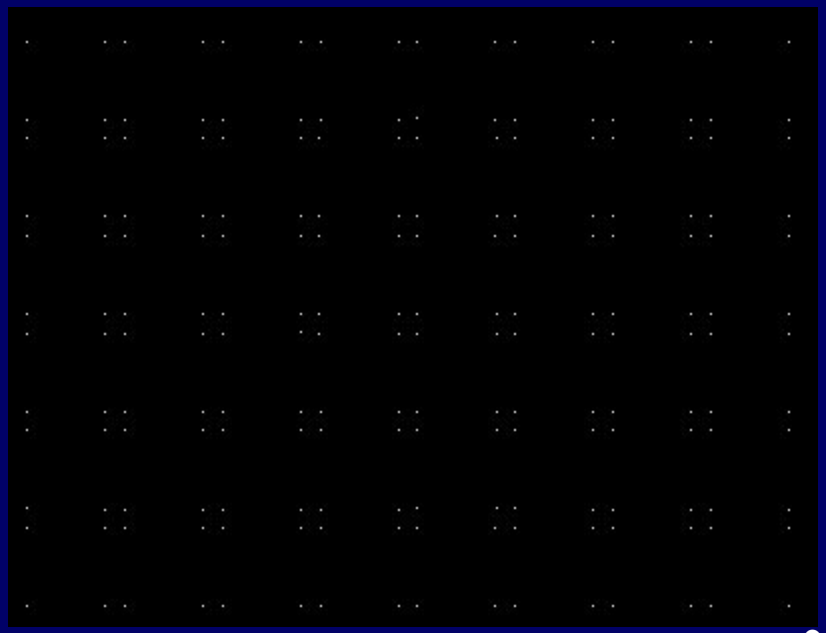
$$M = \int_W g\,(g^T) w\,dx = \sum_{i,j} \begin{bmatrix} g_x(i,j)g_x(i,j) & g_y(i,j)g_x(i,j) \\ g_x(i,j)g_y(i,j) & g_y(i,j)g_y(i,j) \end{bmatrix}$$

- If falls in green area then corner
- Basically looking for x and y magnitude in a window to be large
- If det(M) > threshold
- Need to find the local maximum in a window so we don't get many points for the same corner!



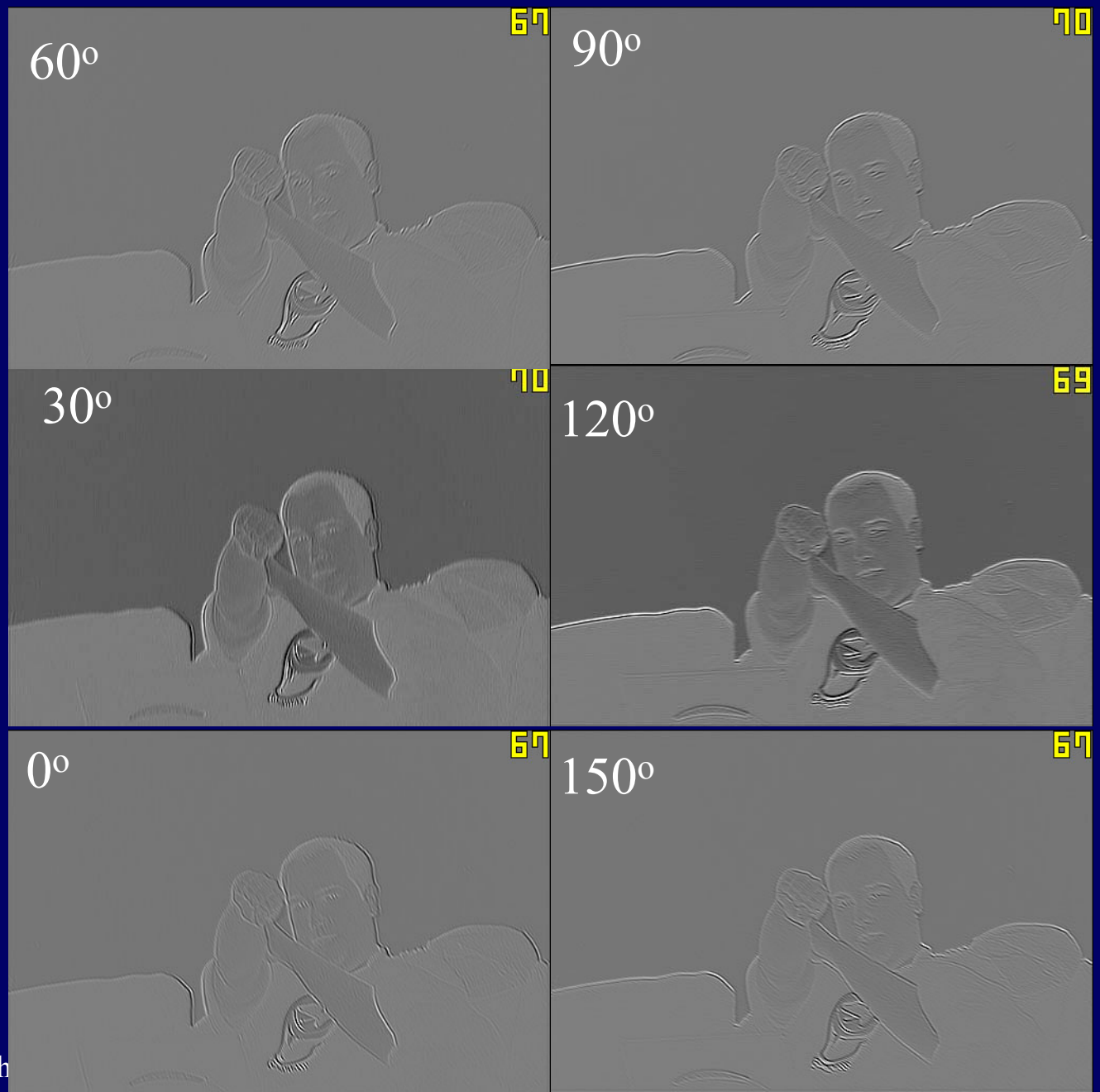

93





Kernel Sharpen

Gabor Filters - Demo



Matrix Operation

- Matrix – Vector
  - Sparse matrix-vector multiplication
    - 13 ms on GeForce FX5200
    - 9 ms using Matlab on 3 Ghz AMD 64bit.
  - Maximum element search
- Vector – Vector
  - Dot product
  - Pointwise operations
  - Element manipulation operations.
- Vector - Scalar

Conjugate Gradient

- 7 frame/sec for 640 x 512 image
  - 327680 x 327680 matrix operation.
  - Quadro FX 3400
  - Matlab Conjugate Gradient (AMD 64bit 3GHz)
    - 20 iters for 7.4 secs = 2.7 iters/sec
    - They are using up-to-date library.
    - Without Convergence
- 152 lines using GPUVision functions
  - Without GPUVision?
    - I can't do without that. ☺
    - 9 RTs + 3 textures

Image segmentation - Performance

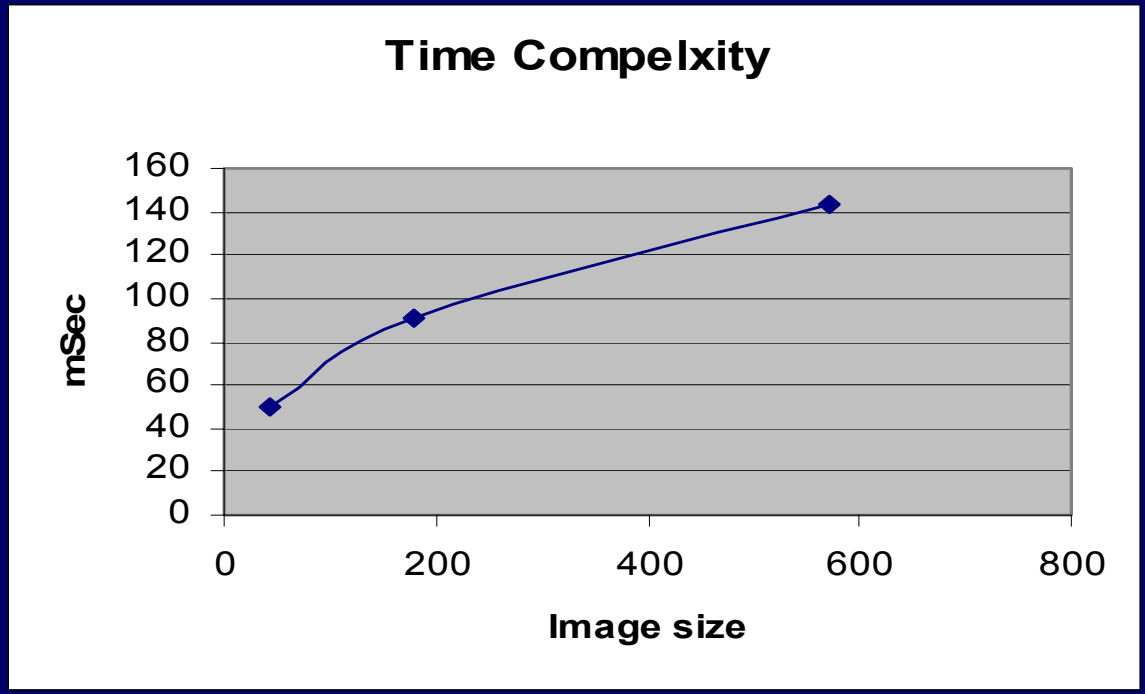


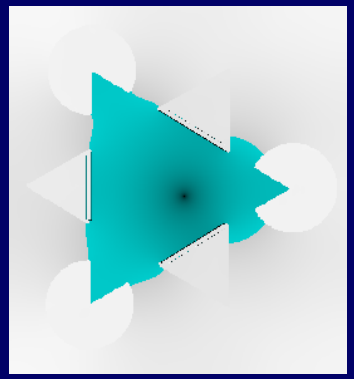
Image	Dim	frs
50x40	2000	20
200x160	32000	11
640x512	327680	7

Code sample

```
while(r_result>threshold){
    vecDP->execute(rho_1, r_1, r_1);
    if(DEBUG) vecDP->showResultMtx(rho_1, 1, 1);

    vecPntDiv->execute(beta_1, rho_1, rho_2);

    vecSclMul->execute(temp, beta_1, p_1);

    vecPlus->execute(p, r_1, temp);


    mulMtx->execute(q, mtxId, p, dMaxPos);

    vecDP->execute(temp, p, q);

    vecPntDiv->execute(alpha, rho_1, temp);

    vecSclMul->execute(temp, alpha, p);

    vecPlus->execute(x, temp);

    vecSclMul->execute(temp, alpha, q);

    vecMinus->execute(r, r_1, temp);

    r->getResult(r_result);

    swapGPUVision(&r_1,&r);
    swapGPUVision(&rho_2,&rho_1);
    swapGPUVision(&p_1,&p);
}
```

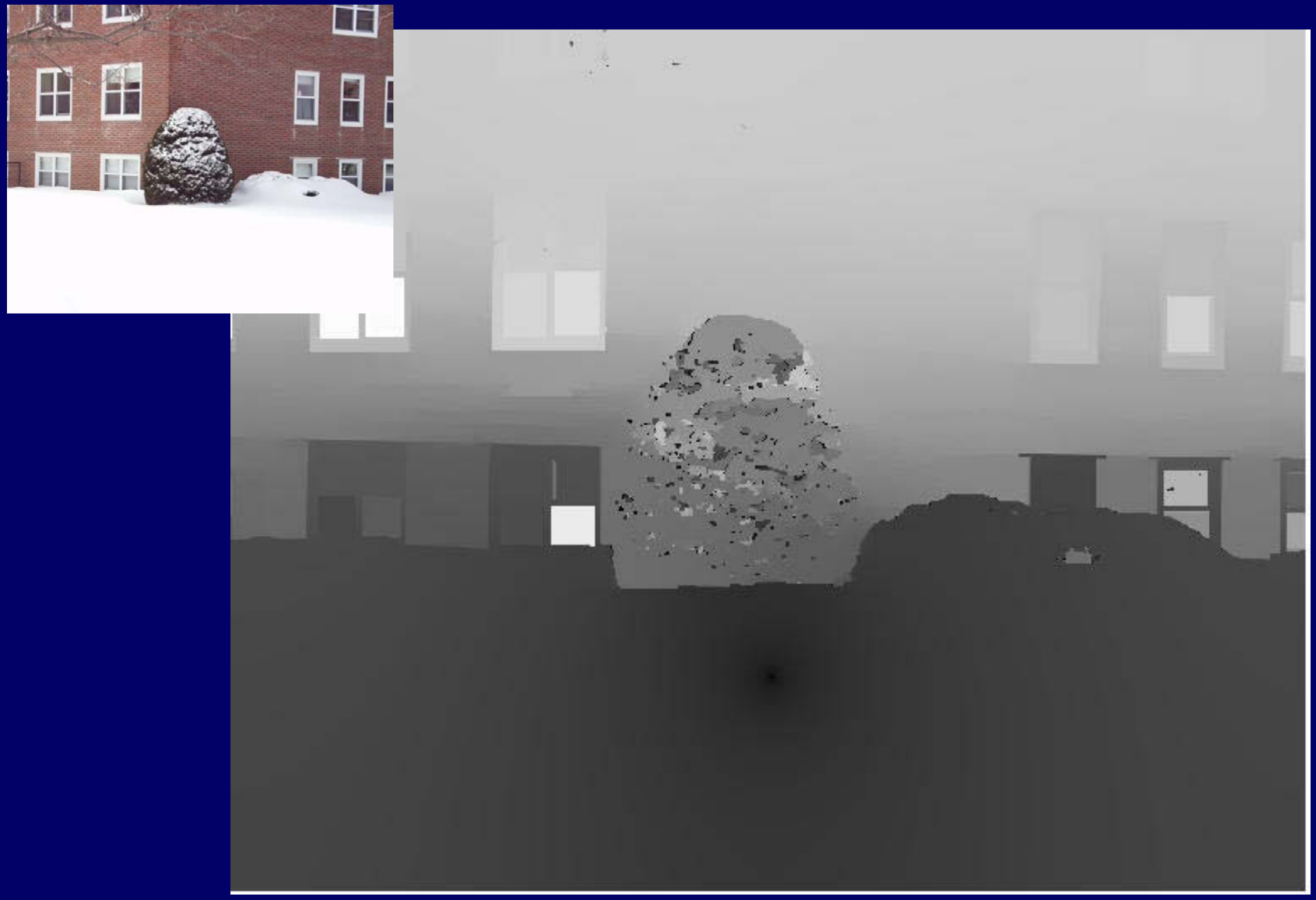
Image Segmentation

- Gestalt Theory
  - Can you see a triangle?

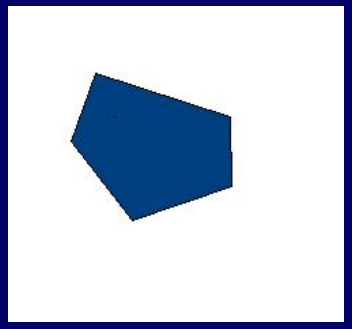


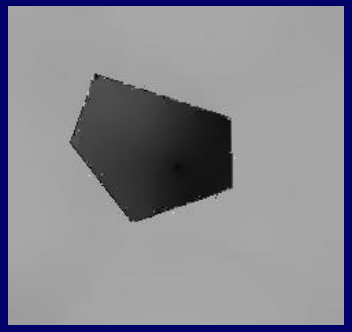
- Isoperimetric Graph Partitioning
  - Ideas from electronic circuits
  - Set a ground node (like a GND in circuit)
  - Calculate energy map (equivalent voltage)
  - Interactive

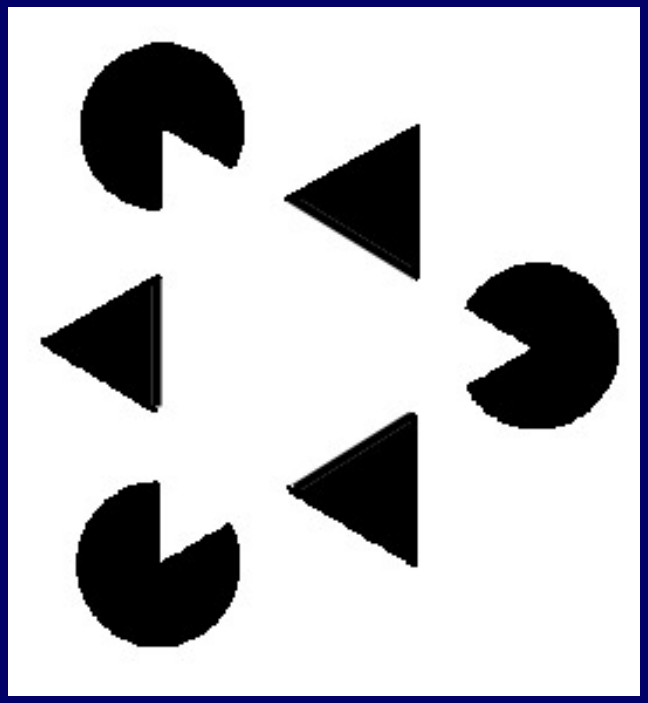
Image Segmentation Examples

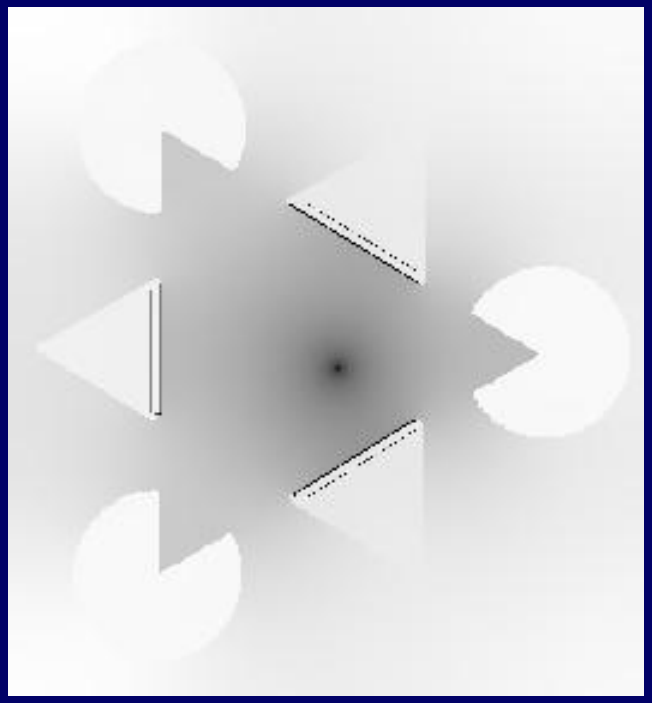


More examples

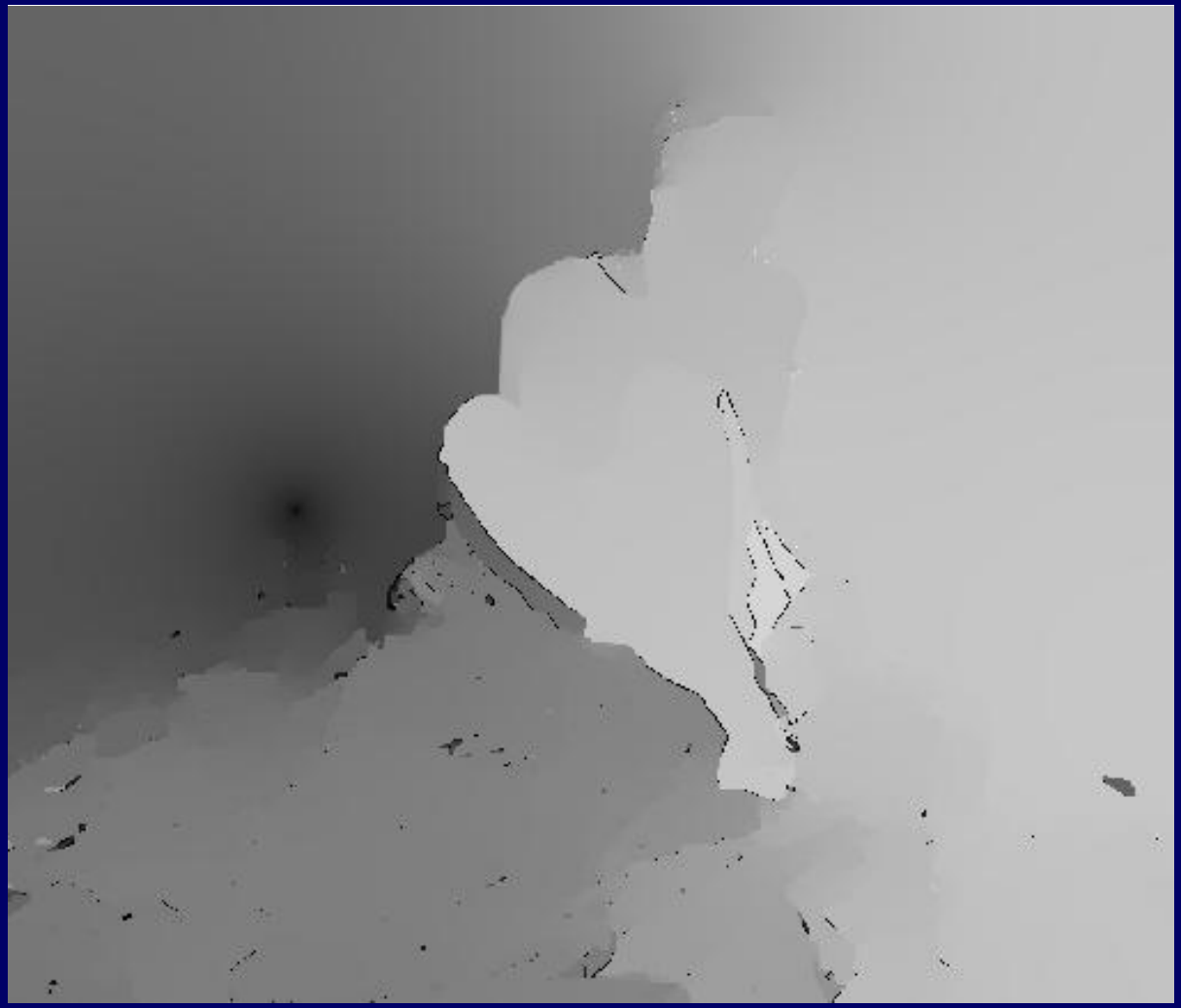




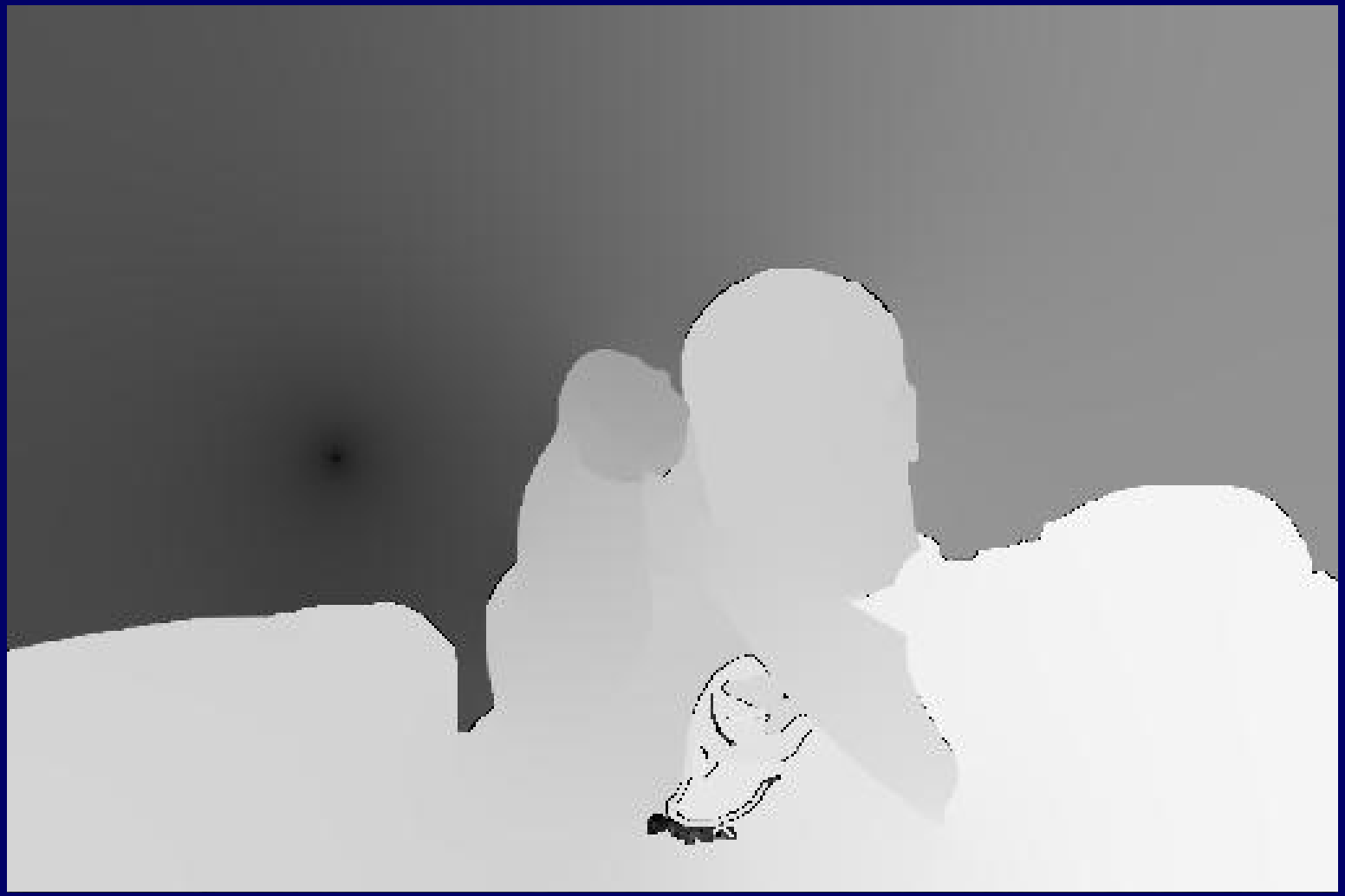




Examples

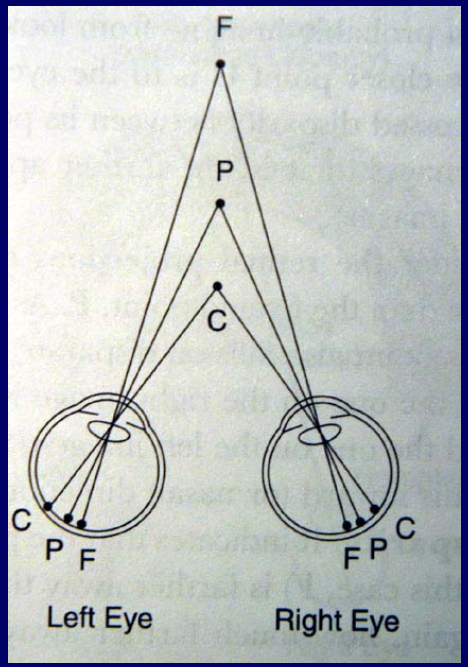


Examples



Disparity Map

- Calculate distance information
  - From stereo calibrated cameras
  - Vertically aligned



Feature point detection (Laplacian filter)

↓

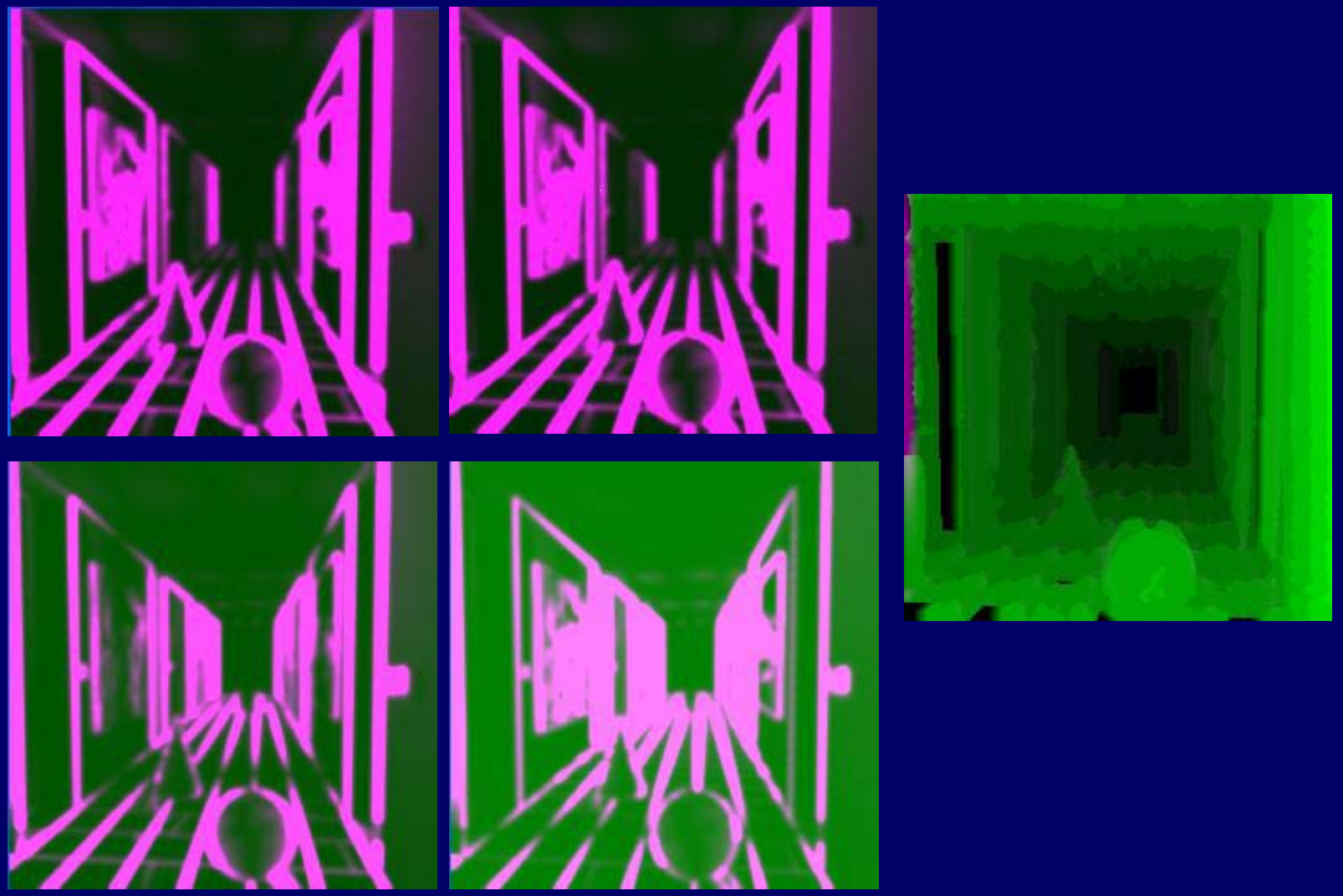
Difference calculation

↓

Search Minimum Difference

Disparity Map

Disparity Map



Problems

- Using AB from LAB color space should result in *different* edges
  - No/little shadows
  - We got *bad* edges
- Flip/Begin was tricky
  - When switching between contexts the Read and Write buffer did not stay consistent.
  - Need to reset Read/Write and wglBindTexImageARB
  - These are less as costly as a context switch but not considerably less

Future

- Program users
  - Uses a GUI and appropriate filters to create effect
  - Integrate into Photoshop (free SDK and implementation description)
    http://download.developer.nvidia.com/developer/SDK/Individual_Samples/featured_samples.html
  - Video for real time computer vision (like OpenVidia)
- More optimized 4 packing and blurring

Future

- Video Support
- Create an OpenSource Library for the community.
- Change GPUVision to allow holding of any number of textures and manages begin/end and flipping of all PingPong Units (move Ping/Pong to lower class)
  - Will hide more of the details from the users creating the Filters

Future

- Debugging methods for GPUVision
  - We already have basic tools
    - Do you remember?

```
if(DEBUG) vecDP->showResultMtx(rho_1, 1, 1);
```

  - Print section
  - Check pixel value
  - Assert
- Render to multi-texture support
- **All these things for OpenGL novice**