

# Обзор библиотек обработки изображений на GPU



---

Арсаяев Марат

*Video Group*  
*CS MSU Graphics & Media Lab*



# Содержание

---

- **Введение**
- Nvidia Performance Primitives
- GPU4VISION
- GpuCV
- OpenVIDIA
- Другие



# Введение

---

- С развитием мощностей, компьютерное зрение и обработка изображений получили широкое применение
- Аппаратная реализация позволяет использовать более качественные и сложные алгоритмы
- Доступно большое количество библиотек примитивов, позволяющих использовать возможности видеокарты

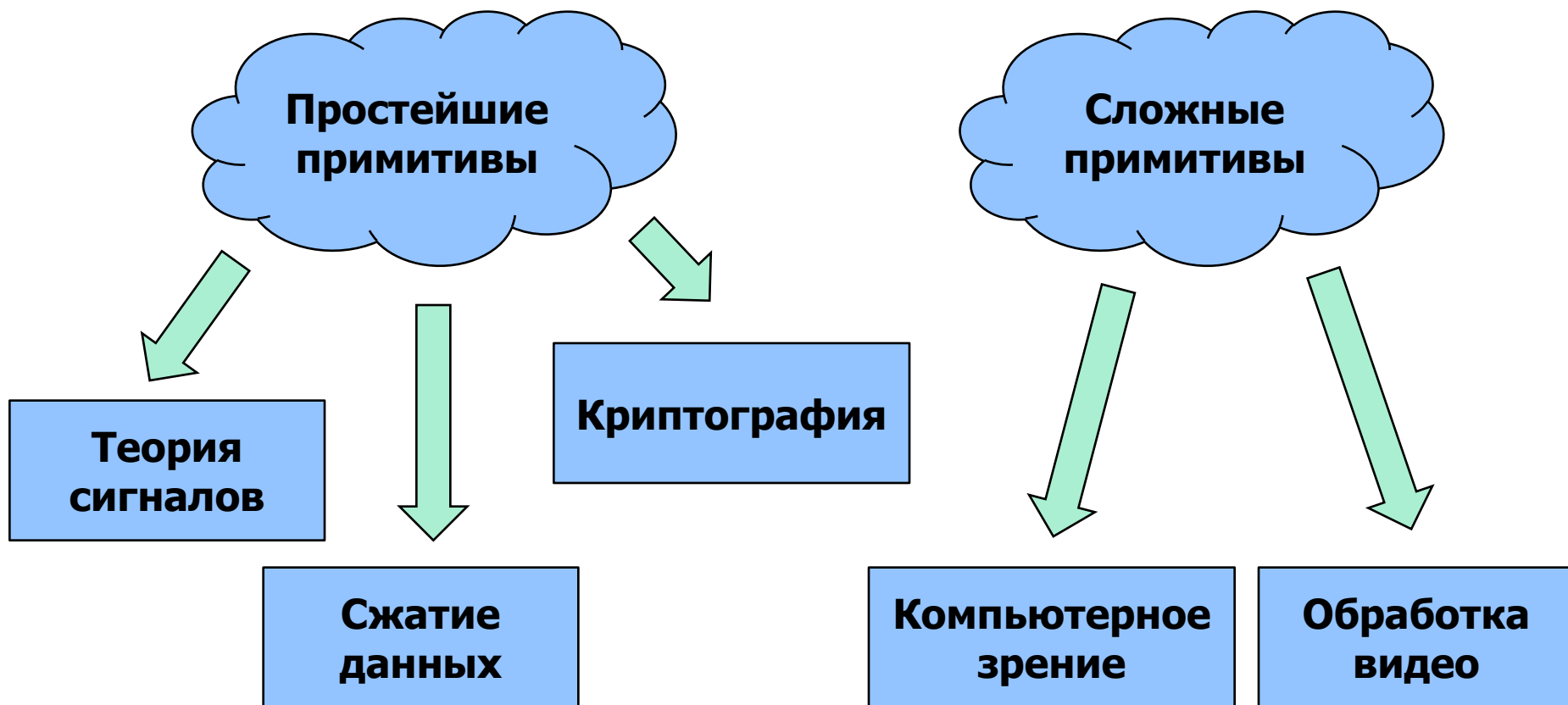


# Содержание

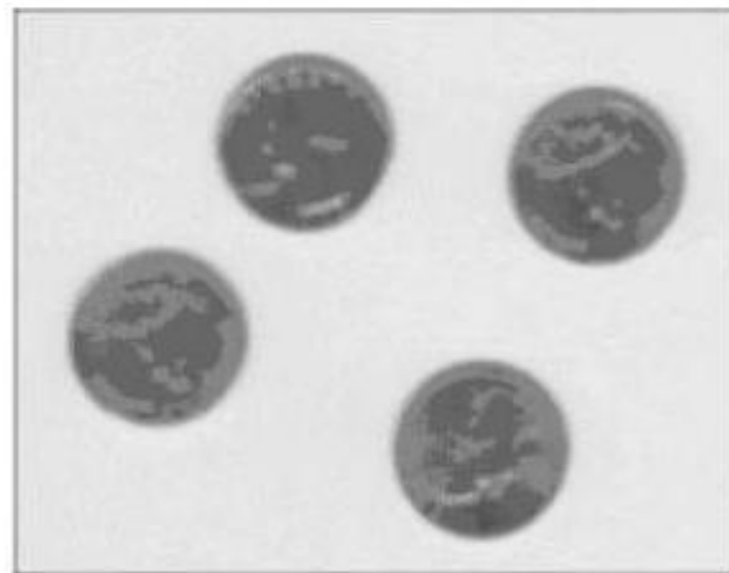
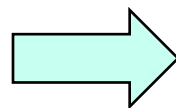
---

- Введение
- **Nvidia Performance Primitives**
- GPU4VISION
- GpuCV
- OpenVIDIA
- Другие

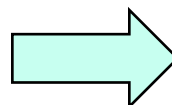
# Примитивы



# Пример Сжатие данных



# Пример Улучшение качества



# Nvidia Performance Primitives

## Реализованные функции



- Арифметические операции (суммирование, умножение)
- Преобразование цветовых форматов
- Вычисление стандартных разниц изображений (SAD, SSD, MAD)
- Сравнение и трешхолдинг изображений
- 1D фильтры (линейный фильтр, Window Sum)
- 2D фильтры (морфология, медианный фильтр, настраиваемые линейные фильтры)



# Nvidia Performance Primitives

## Пример



```
// allocate source image
int sp;
Ipp8u * pSI = ippiMalloc_8u_C1(w, h, &sp);
// fill with some image content
testPattern_8u_C1(pSI, sp, w, h);

// allocated destination image
int dp;
Ipp8u * pDI = ippiMalloc_8u_C1(w, h, &dp);
// Filter mask and anchor
IppiSize mask = {5, 5};
IppiPoint anchor = {0, 0};
IppiSize ROI = {w - mask.width + 1,
                h - mask.height + 1};
// run box filter
ippiFilterBox_8u_C1R(pSI, sp, pDI, dp,
                    ROI, mask, anchor);
```

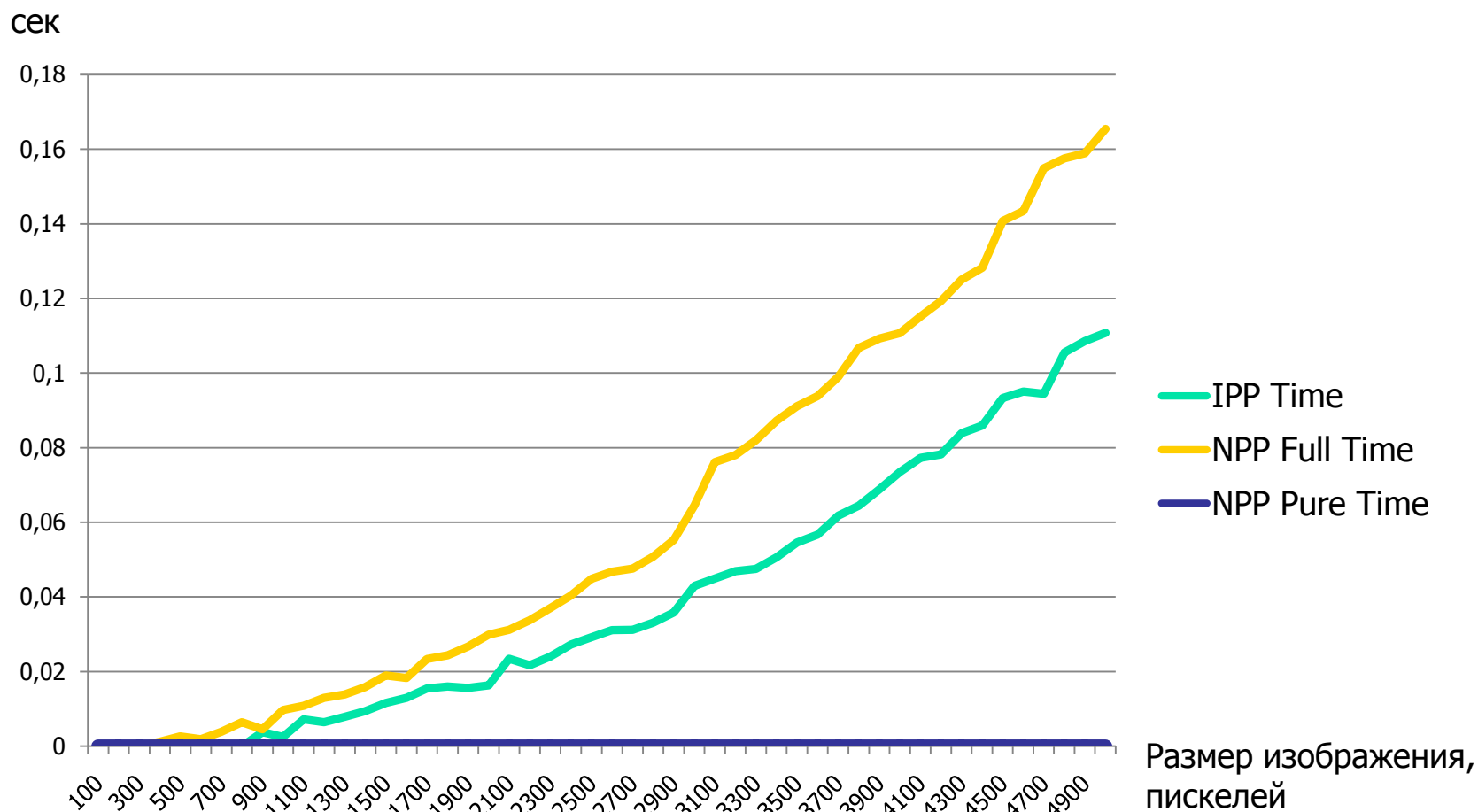
```
// allocate host source image
int hp;
Ipp8u * pHI = ippiMalloc_8u_C1(w, h, &hp);
// fill with some image content
testPattern_8u_C1(pHI, hp, w, h);
// allocated device source image
int sp;
Npp8u * pSI = nppiMalloc_8u_C1(w, h, &sp);
// copy test image up to device
cudaMemcpy2D(pSI, sp, pHI, hp, w, h,
             cudaMemcpyHostToDevice);
// allocate device result image
int dp;
Npp8u * pDI = nppiMalloc_8u_C1(w, h, &dp);
// Filter mask and anchor
NppiSize mask = {5, 5};
NppiPoint anchor = {0, 0};
NppiSize ROI = {w - mask.width + 1,
                h - mask.height + 1};
// run box filter
nppiFilterBox_8u_C1R(pSI, sp, pDI, dp,
                    ROI, mask, anchor);
```

© 2009 NVIDIA CORPORATION



GPU Technology Conference, Nvidia, 2009

# Сравнение FilterBox\_8u\_C4R

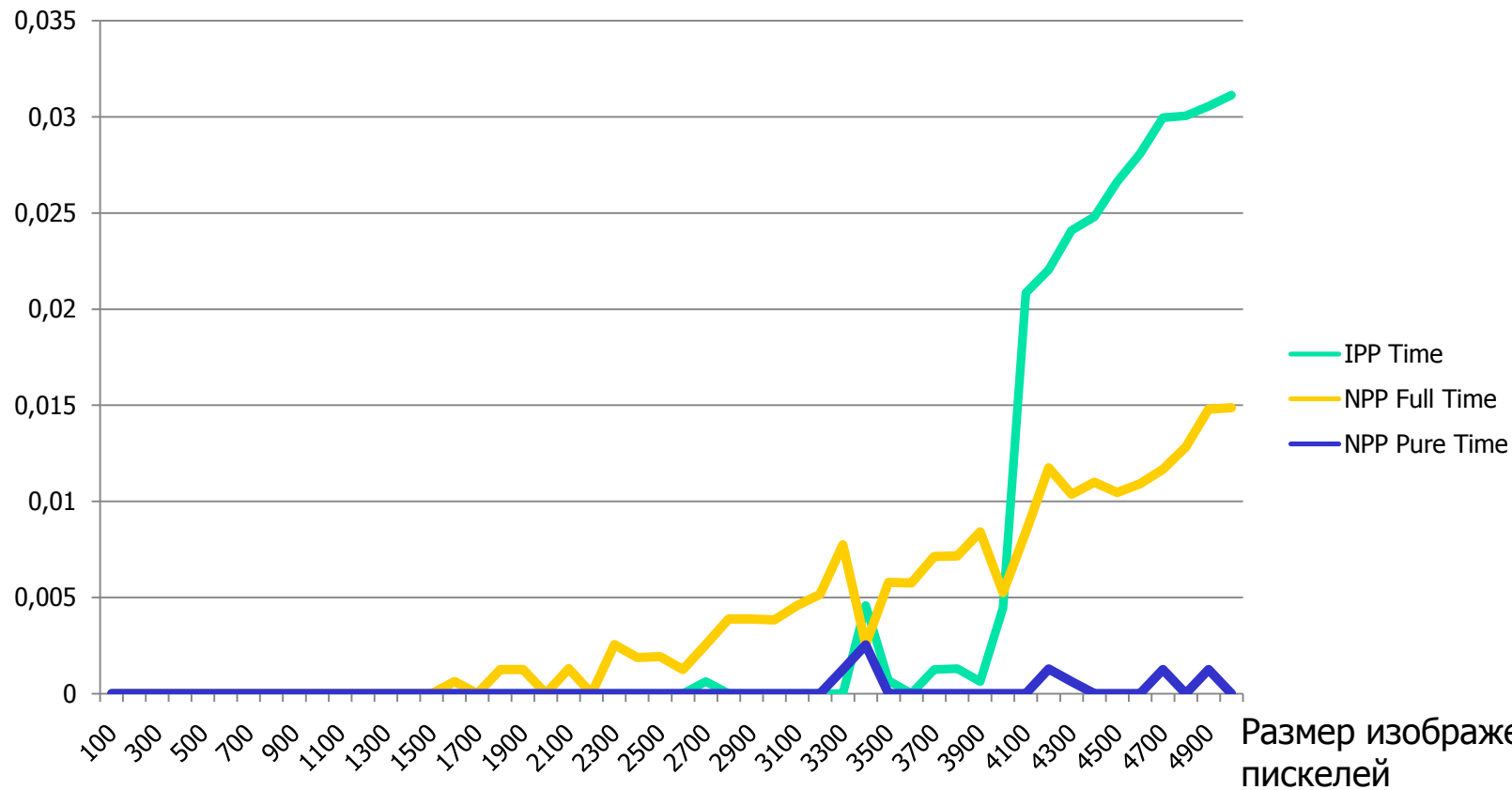


8800 GTS (640mb), Core2Duo 6550, 4GB RAM

# Сравнение Mean\_StdDev\_8u\_C1R



сек



8800 GTS (640mb), Core2Duo 6550, 4GB RAM

# Nvidia Performance Primitives: Резюме



- Реализовано небольшое количество примитивов
- Много времени тратится на копирование данных
- Подходит для многократной обработки данных на NPP



# Содержание

---

- Введение
- Nvidia Performance Primitives
- **GPU4VISION**
- GpuCV
- OpenVIDIA
- Другие



# GPU4VISION

---

- Выкладывают части своих проектов:
  - FlowLib (Anisotropic Huber-L1 OpticalFlow)
  - VMFLib (Image Denoising & Segmentation)
- Алгоритмы заточены под realtime
- Реализованы интерфейсы к камерам и сенсорам



# Содержание

---

- Введение
- Nvidia Performance Primitives
- GPU4VISION
- **GpuCV**
- OpenVIDIA
- Другие



# GpuCV

---

- Задача – реализовать примитивы OpenCV на GPU
- Предоставляет CPU, OpenGL, CUDA реализации примитивов
- Открытый код и кроссплатформенность
- Три варианта выбора реализации:
  - Принудительный выбор
  - Выбор наиболее производительной архитектуры
  - Выбор по статистике





# GpuCV

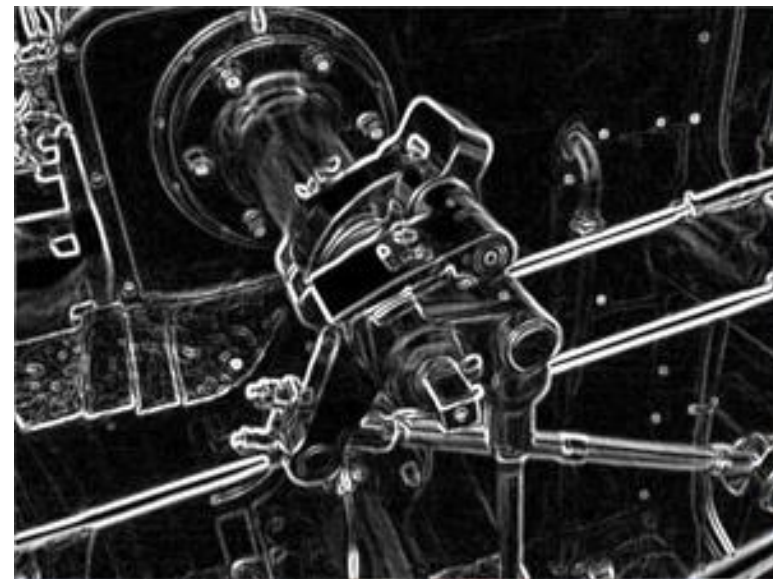
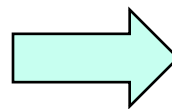
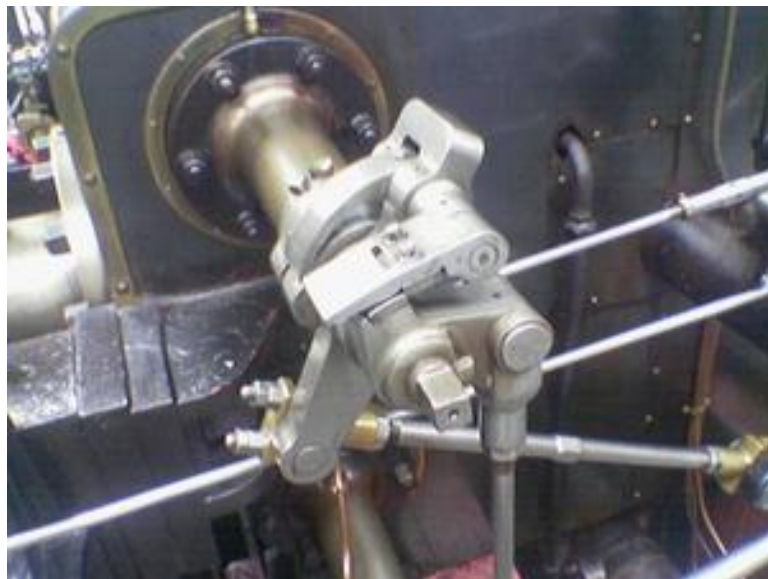
---

В состав входят:

- преобразования цветовых форматов
- вычисление краев объектов, морфология
- дискретные преобразования (FFT, DCT)
- геометрические преобразования
- гистограммы

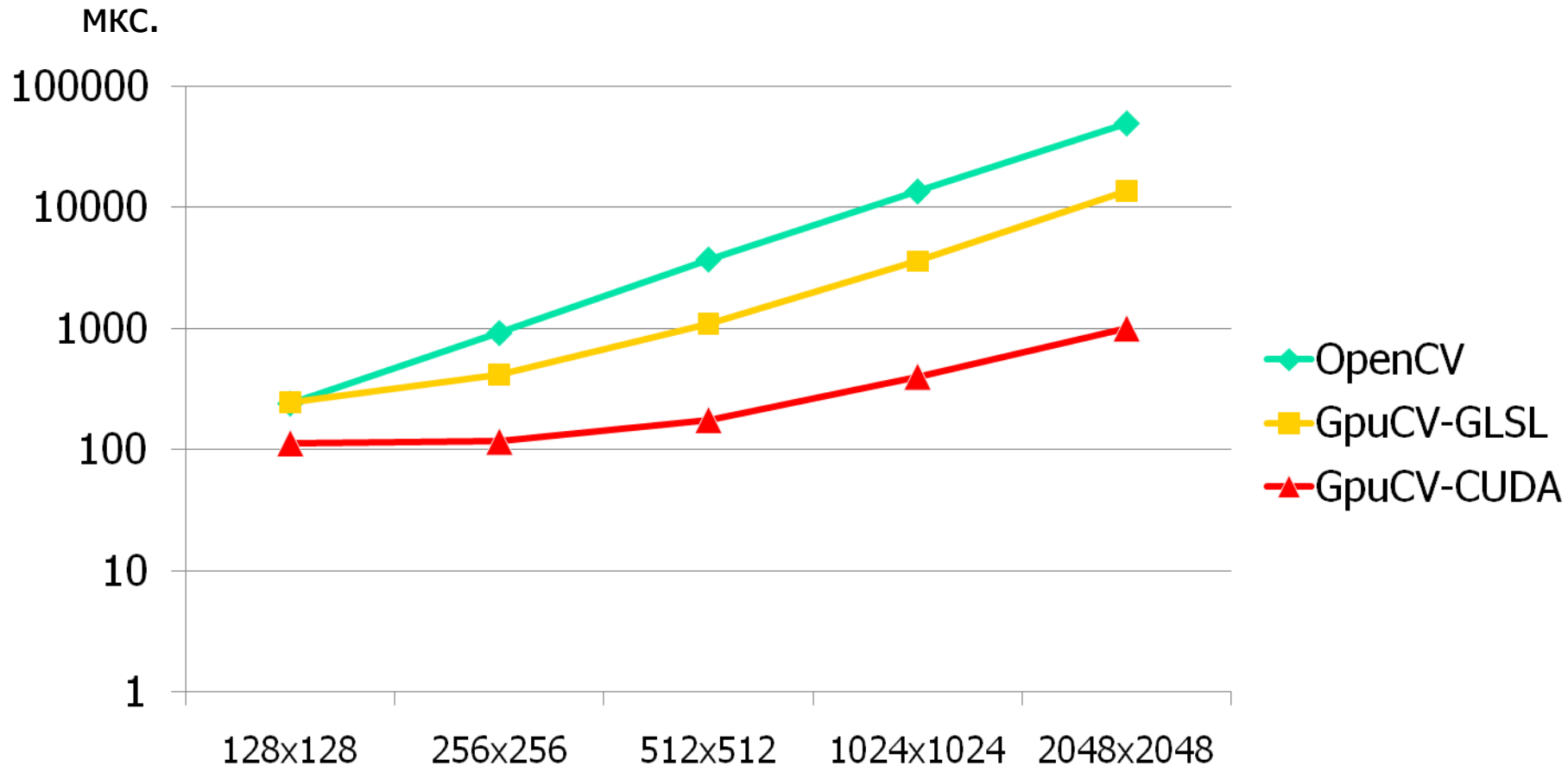
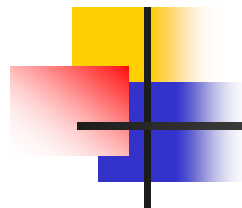
# ГриCV

## Оператор Собеля



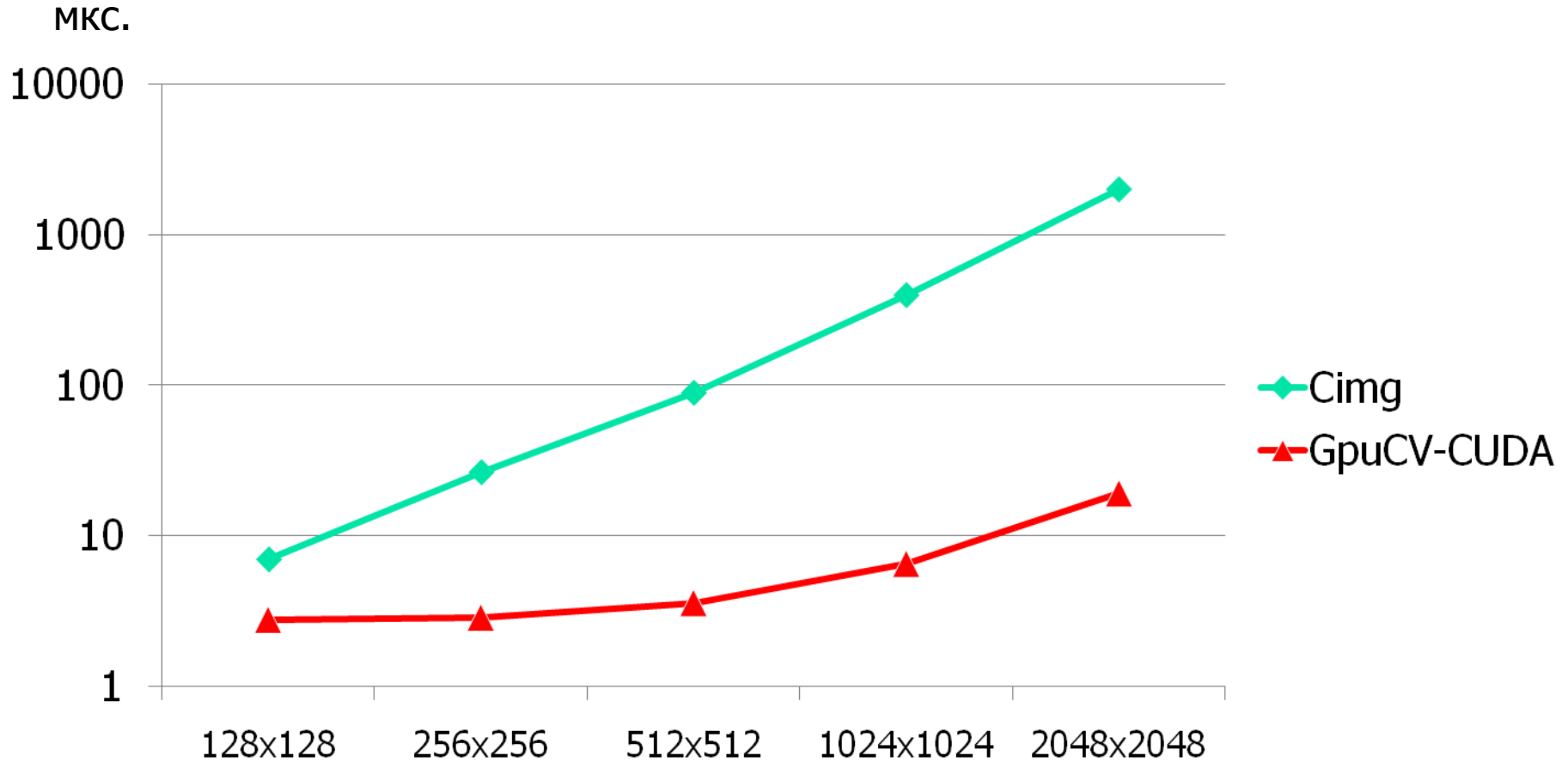
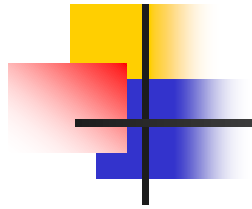
# ГриCV

## Оператор Собеля



# GpuCV

## Deriche Filter





# Содержание

---

- Введение
- Nvidia Performance Primitives
- GPU4VISION
- GpuCV
- **OpenVIDIA**
- Другие



# OpenVidia

---

- Open-source проект
- Реализует новейшие алгоритмы компьютерного зрения
- Содержит эффективно реализованные и отпрофилированные алгоритмы
- На сайте проекта собирается информация об алгоритмах компьютерного зрения



# OpenVidia

---

Включает в себя:

- Optical Flow
- Feature Detection & Tracking
- настраиваемую свертку небольших радиусов
- детектор краев и медианный фильтр
- построение гистограмм
- геометрические и афинные преобразования

# OpenVidia: Feature Detection & Tracking





# Пример вызова свертки

```
libopenvidiaCUDA::onvImage onvImgF( width, height, 1,  
    ONV_FLOAT, src_data );  
libopenvidiaCUDA::convolutionSeparable cs;  
cs.init( width, height, 3 );  
float k[7] = { 0.0132999f, 0.1080111f, 0.2403623f, 0.0f,  
    0.2403623f, 0.1080111f, 0.0132999f };  
cs.setKernel( k, k );  
...  
cs.convolve( (float*) onvImgF.d_ptr, (float*)  
    onvImgTmp.d_ptr );  
cs.convolveCols( (float*) onvImgTmp.d_ptr, (float*)  
    onvImgF.d_ptr );
```

# CUDA VisionWorkbench

CUDA VisionWorkbench v1.3.0.9    Cuda v 3.00    BEOWULF-VISTA64    Microsoft Windows NT 6.0.6002 Service Pack 2    2 Procs, 2 Threads    NvDrv: 195.62, 5059340

Session Time: 91 s    Set Cycle # 1 20 100 500 1 Counter 1

Data    Kernel Config    Constants    Help

Test Pattern Image    Image from File    Topmost Form    Enable Histo Dialog

1024 W 1024 H    Auto-Range on open

GPU / CUDA 0 GeForce 8800 GTS 512    2-CPU / 2 Threaded C++    16 Bit Compute

Cap 1.1, SMs 16, MHz 1625, Gm-MBs 536, Regs 8192, Sh-Bytes 16384    # of Threads 2    8 Bit Compute

Current View: Smile.jpg    Input    Output    2000 W 996 H

Thumb Mag 2x    Thumb Mag 4x    Thumb Mag 8x    Thumb Mag 16x

3x3 Convo    3x3 Median    Rms33 Corr    3x3 Min    3x3 Max    Sobel Mag    Mult 2D    Mult In x b    Avg In | Out    Input Histo

5x5 Convo    5x5 Median    Rms55 Corr    5x5 Min    5x5 Max    Sobel Dir    Subt 2D    Subt In - a    Copy Out > In    Thres <->

7x7 Convo    7x7 Median    Rms77 Corr

Series -> 6 ->  
>Mult2D >Subt2D >Median77 >SobelMag >TileMin33 >TileMax33

Input Detail (128x128, 8x8)    Output Detail (128x128, 8x8)

61456   61472   61824   62224	62144   61760   61280   60720	00000   00000   00000   00000	00000   00000   00000   00000
61776   61488   61856   62576	62808   61952   61264   61056	00000   00000   00000   00000	00000   00000   00000   00000
61840   61520   61968   62624	62384   61568   61120   61488	00000   00000   00000   00000	00000   00000   00000   00000
61024   60880   61248   61808	61904   61696   61616   61840	00000   00000   00000   00000	00000   00000   00000   00000
60736   60400   60288   60688	61424   62112   62272   61856	00000   00000   00000   00000	00000   00000   00000   00000
61344   60752   60208   60304	61136   62112   62288   61504	00000   00000   00000   00000	00000   00000   00000   00000
61760   61632   61232   60944	61120   61520   61536   61024	00000   00000   00000   00000	00000   00000   00000   00000
61296   61744   61824   61616	61424   61136   60880   60960	00000   00000   00000   00000	00000   00000   00000   00000

GPU Timings (seconds) for Sobel Mag, Ret = 1992000:

0 App. Call/Logic	0.000034 s
1 Host->GPU Copy	0.002922 s
2 CUDA Compute	0.001001 s ( 1,990.6 MPix/Sec)
3 GPU->Host Copy	0.002499 s
4 Return from DLL	0.000091 s

Total Time = 0.006456 s

Main Display (16 bpp Input Image)

# CUDA VisionWorkbench





# Содержание

---

- Введение
- Nvidia Performance Primitives
- GPU4VISION
- GpuCV
- OpenVIDIA
- **Другие**

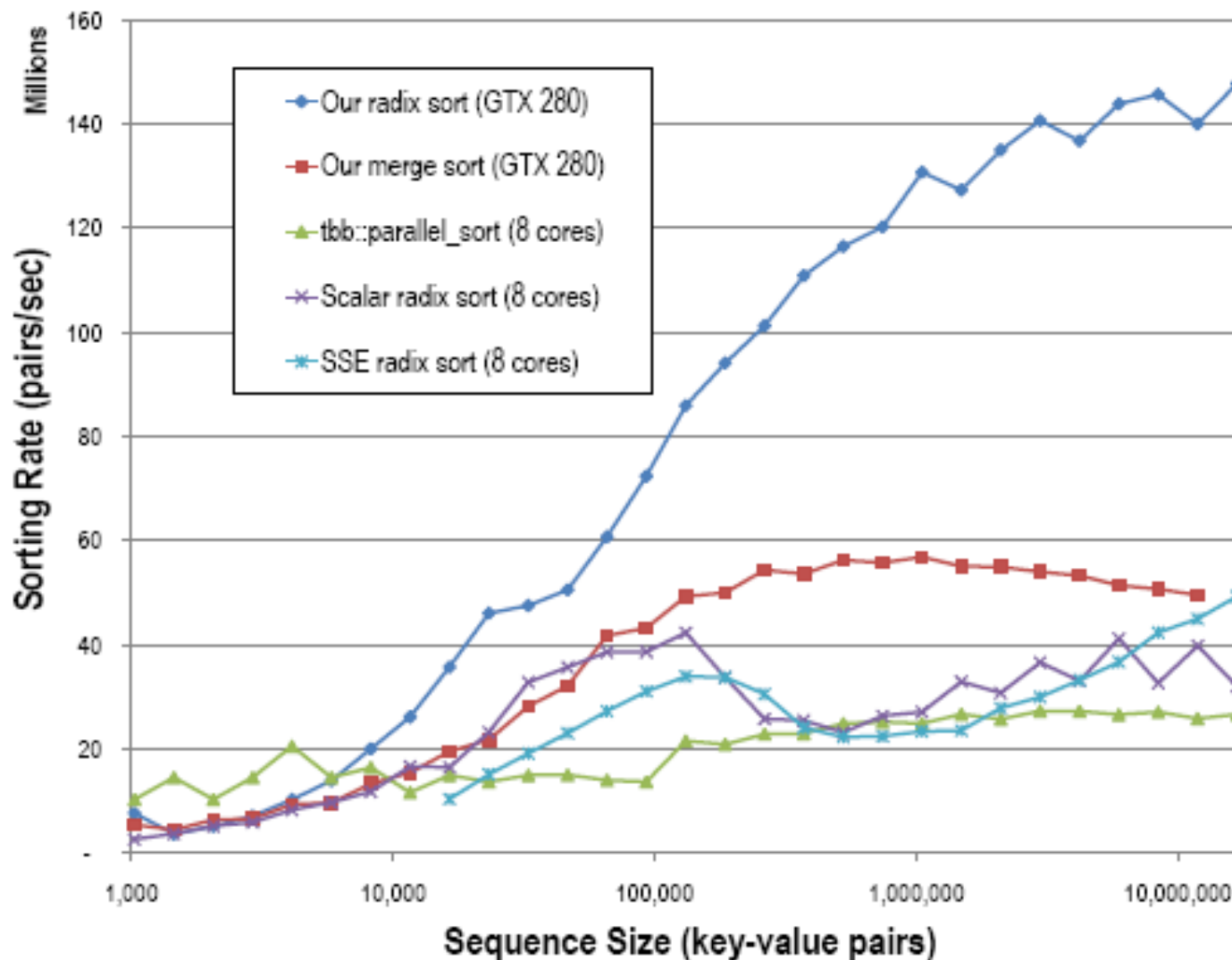
# CUDA Data Parallel Primitives Library



Открытая библиотека алгоритмов-примитивов:

- Префиксная сумма
- Сортировка
- Редукция
- Генерация случайных чисел

# CUDPP: Сортировка

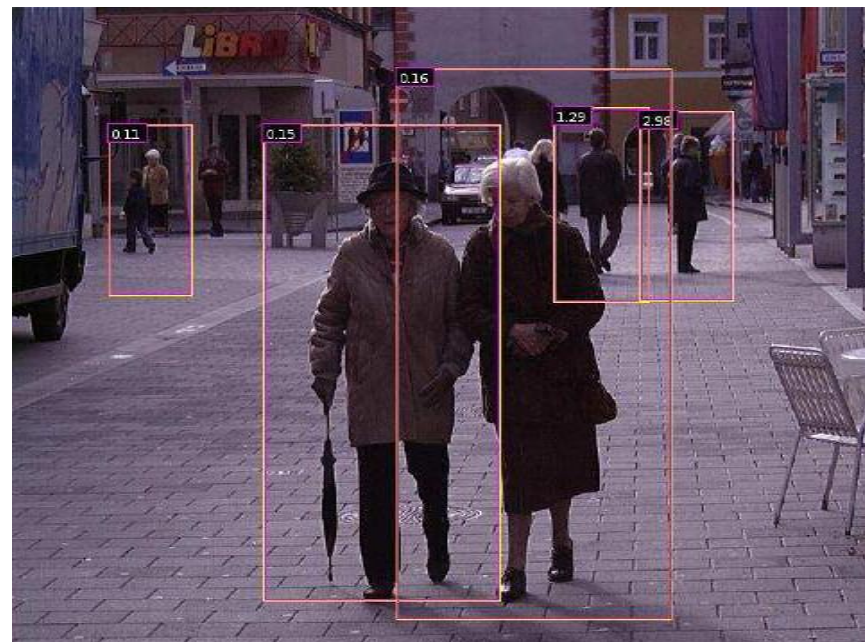
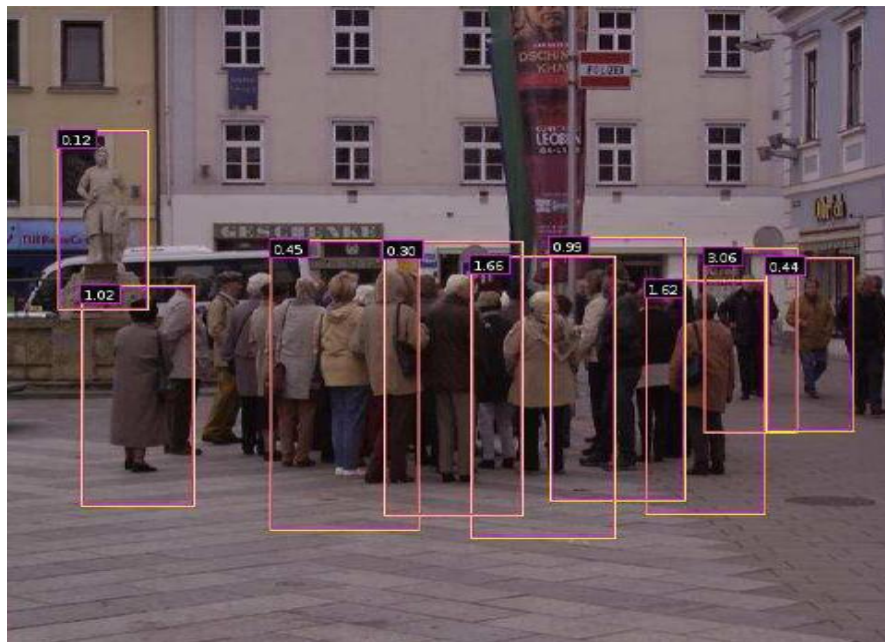


“Designing Efficient Sorting Algorithms for  
Manycore GPUs”. Nadathur Satish., 2009



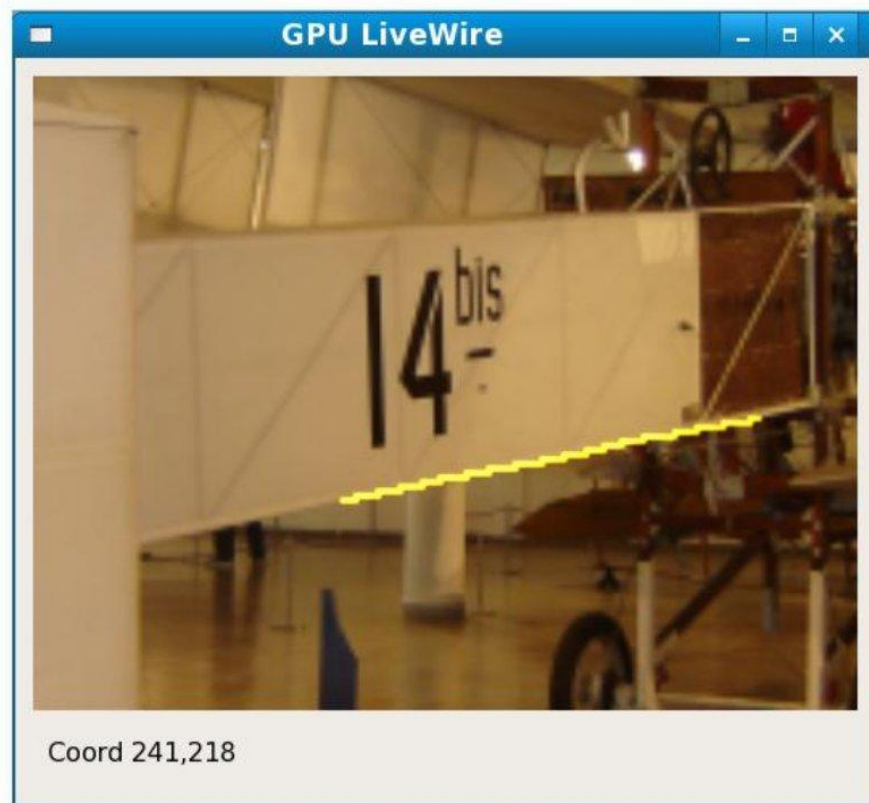
# CUDA-based Pedestrian Detection

Проект с открытым кодом, основанный на ранних разработках MIT



# GPU Wire

Проект интерактивной сегментации с открытым кодом. Предлагает инструмент, схожий с лассо Adobe Photoshop





# Список литературы

1. <http://gpu4vision.icg.tugraz.at/> GPU4VISION project
2. <http://www.youtube.com/user/gpu4vision> GPU4VISION's channel
3. <http://server.cs.ucf.edu/~vision/MinGPU/> MinGPU project
4. <http://openvidia.sourceforge.net/index.php/OpenVIDIA> OpenVIDIA
5. <http://www.nvidia.com/object/npp.html> NVPP project
6. <http://www.itlab.unn.ru/?dir=174> Сравнение NPP и IPP, проект ITLab
7. <https://picoforge.int-evry.fr/cgi-bin/twiki/view/Gpucv/Web/GpuCV> GpuCV
8. <http://code.google.com/p/cudapeddet/> CUDA implementation of pedestrian detection algorithm, MIT Project
9. <http://gpgpu.org/developer/cudpp> CUDA Data Parallel Primitives Library
10. <http://code.google.com/p/gpuwire/> Gpu Wire project

# Лаборатория компьютерной графики и мультимедиа



Видеогруппа это:

- Выпускники в аспирантурах Англии, Франции, Швейцарии (в России в МГУ и ИПМ им. Келдыша)
- Выпускниками защищено 5 диссертаций
- Наиболее популярные в мире сравнения видеокодеков
- Более 3 миллионов скачанных фильтров обработки видео