Final Project

CS194-15 Fall 2012:
Engineering Parallel Software

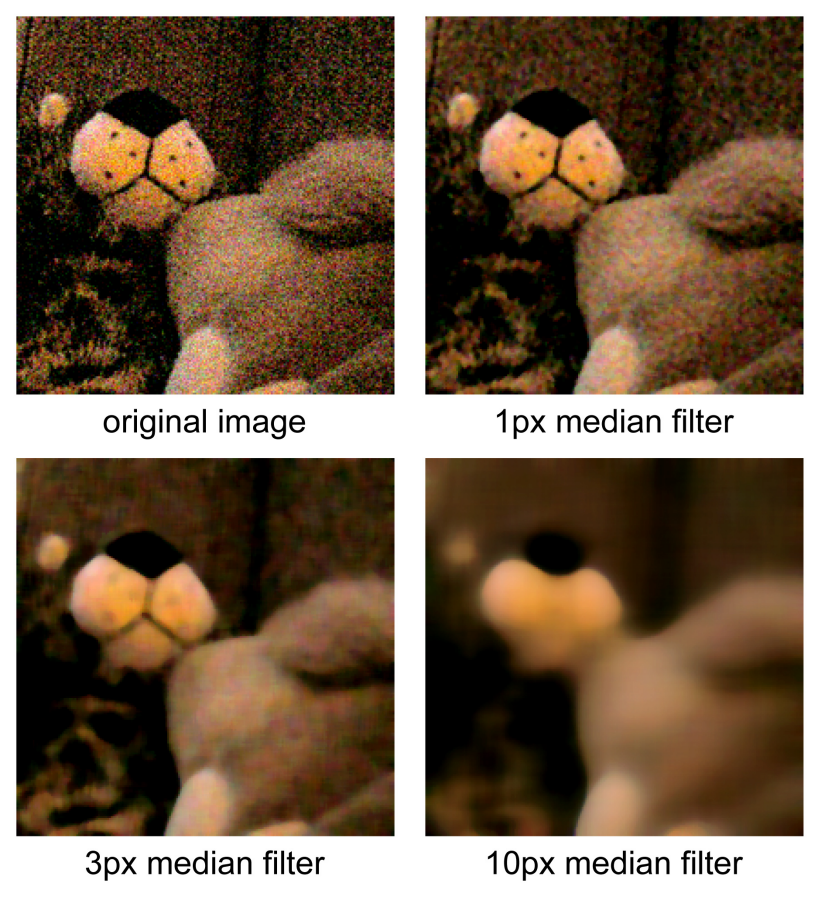
Anting Shen, Matt Miller
Kishan Karunaratne, ByungSoo Ko

2013 Final Project Presentation

University of California, Berkeley

Motivation

- Median filtering is very widely used in digital image processing.
  - Under certain conditions, it preserves edges while removing noise.
- This image shows median filters of varying radii applied to the same noisy photograph.



Problem: The Median Filterd Image

(A)

(B)

(A)The original salt and pepper noise corrupted image.
(B)The median filtered image using a 5 x 5 mask.

Algorithm I: Median Filter

- A median filter in terms of an image A is $\text{Median}(A) = \text{Median}[A\{(x + I, y + j)\}]$, where the coordinate (x+i), (y+j) is defined over the image A and the coordinate i, j is defined over the mask M.

- The median filter is defined as the median of all pixels within a local region of an image. Pixels that are included in the median calculation are specified by a mask

Algorithm II: Median Filter

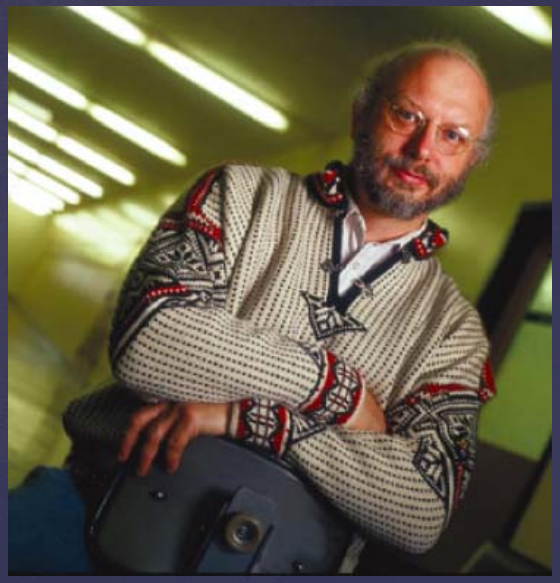
- The program assumes that the original image is a 256 graylevel x IMAGE->Rows x IMAGE ->Cols pixel image stored in the structure IMAGE.
- The program then performs a N x N median filter on the image.
- The size of the filtering operation is determined by the variable N and should be set to an odd number and be less than 12.
- Upon completion of the program, the filtered image is stored in the structure IMAGE1.

Pseudo Code: Median Filter

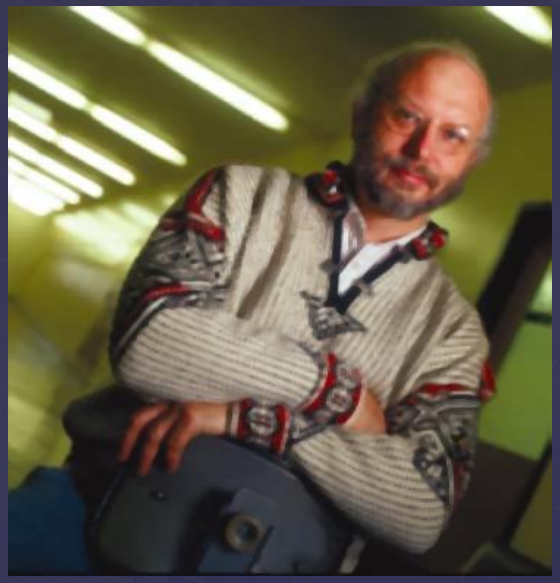
```
Median(struct Image *IMAGE, struct Image
*IMAGE1)
      {
      int X, Y, I, J, Z;
      int N, AR[121], A;
      N=7;
      for(Y=N/2; Y<IMAGE->Rows-N/2; Y++)
        for(X=N/2; X<IMAGE->Cols-N/2; X++)
          {Z=0;
          for (J=-N/2; J<=N/2; J++)
              for (I=-N/2; I<= N/2; I++)
                {
                AR[Z]= *(IMAGE->Data+X+
                I+(long)(Y + J)
                *IMAGE->Cols);
                Z++;
                }
          for (J=1; J<N*N-1; J++)
              {
                                          }
A = AR[J];
I = J – 1;
while (I >= 0 && AR[I]  >A)
      {
      AR[I + 1] = AR[ I ];
      I = I – 1;
      }
      AR[ I + 1 ] = A;
   }
   *(IMAGE1->Data+ X +(long)Y
   *IMAGE->Cols) = AR[N*N/2];
   }
```

Software Architecture

Serial Version



(A)



(B)

(A)The original cpt-kurt.jpg.
(B) The median filtered cpt-kurt.jpg using a 5 x 5 mask.

Runtime: 0.521373 sec

OpenCL

- OpenCL (Open Computing Language) is a low-level API for heterogeneous computing that runs on CUDA-powered GPUs.

- Using the OpenCL API, developers can launch compute kernels written using a limited subset of the C programming language on a GPU

OpenCL Version

CUDA

- CUDA is a parallel computing platform and programming model invented by NVIDIA.
- It enables dramatic increases in computing performance by harnessing the power of the graphics processing unit (GPU).
- CUDA is broadly used for GPU computing
  - Identify hidden plague in arteries
  - Analyze air traffic flow
  - Visualize molecules

CUDA Version

Experimental Results

Method	Runtime
Serial Median Filter	0.521373 sec
OpenCL Median Filter	
CUDA Median Filter	

Theoretical Max Speed up with OpenCL

Theoretical Max Speed up with CUDA

Conclusions