CS 677 Final Project

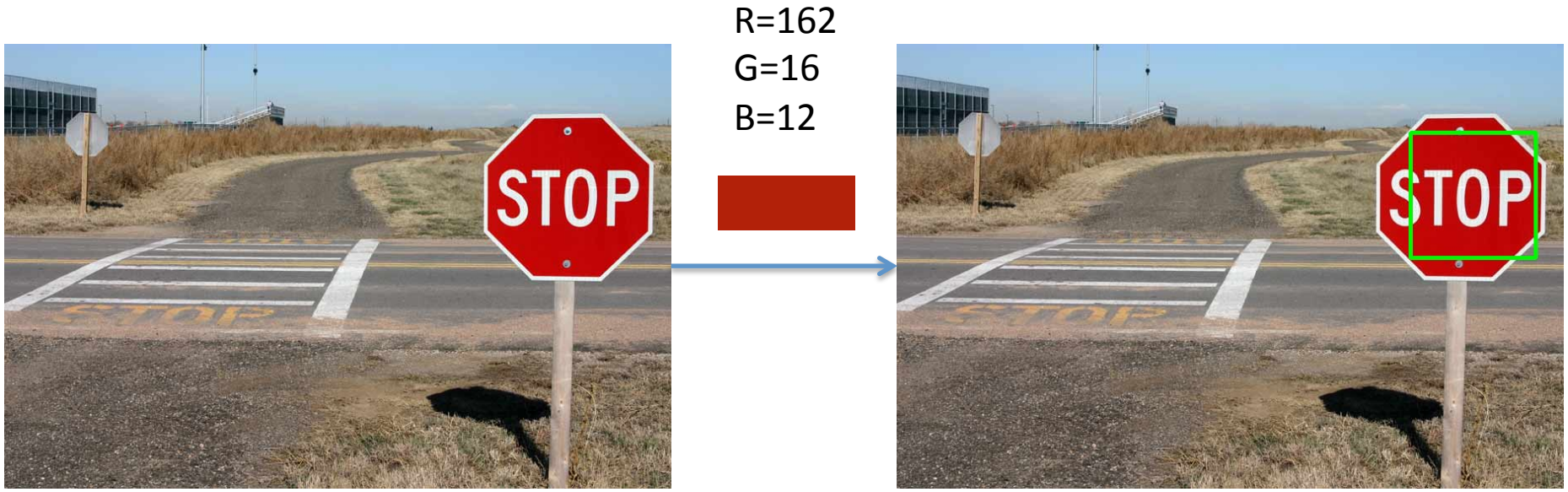
Integral Image and Detection

Tianyu Luo

Problem Description

- General Purpose

Detect largest area with a given color.



Computation

- Two major steps.

1. Computing Integral Image

The area of an arbitrary rectangle can be obtained by using 4 numbers.

2. Find the largest rectangle

These two parts can use GPU to accelerate computing speed.

Integral Image on GPU

- 2D Prefix Sum

Apply prefix sum on each row first and then apply it on each col.

- Consider 1D Prefix Sum on CPU -> O(N)

  1D Prefix Sum on GPU -> O(logN)

- For Integral Image

  O(M*N) -> O(logM*logN)

Find the max on GPU

- Consider finding on CPU -> O(N)
- Consider finding on GPU -> O(logN), use reduction.

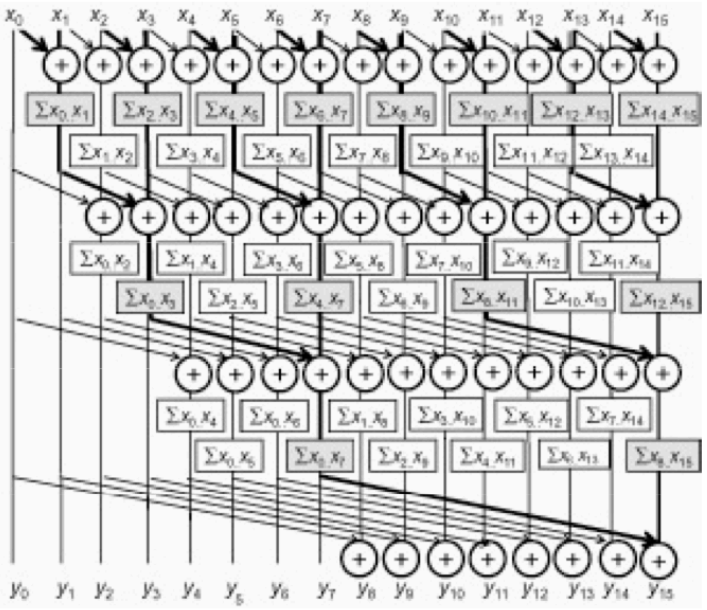
Steps

1, Read source image and given color. Then, convert the image to a matched image.

matched image, int[M*N], 1 is given color, 0 is not.

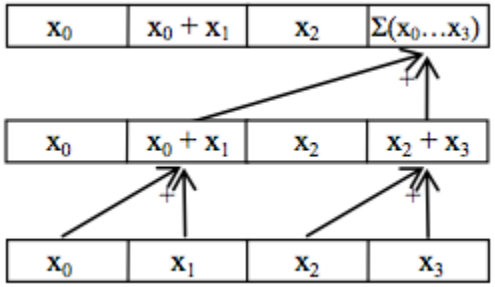
2, Computing integral image.

computeOnDeviceNaive()

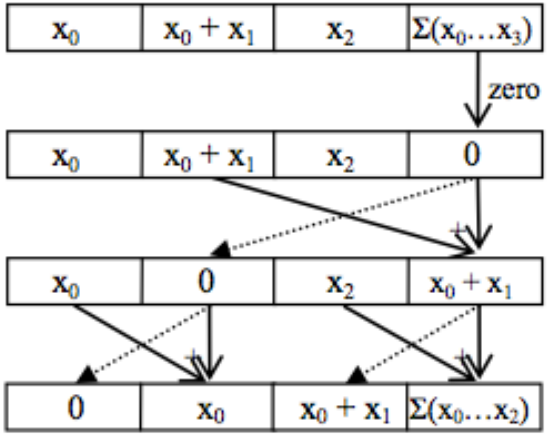


2, Computing integral image.

computeOnDevice()

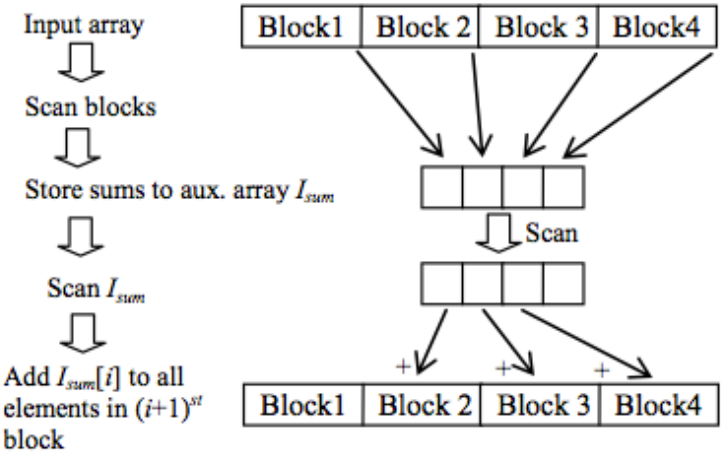


Reduce Phase



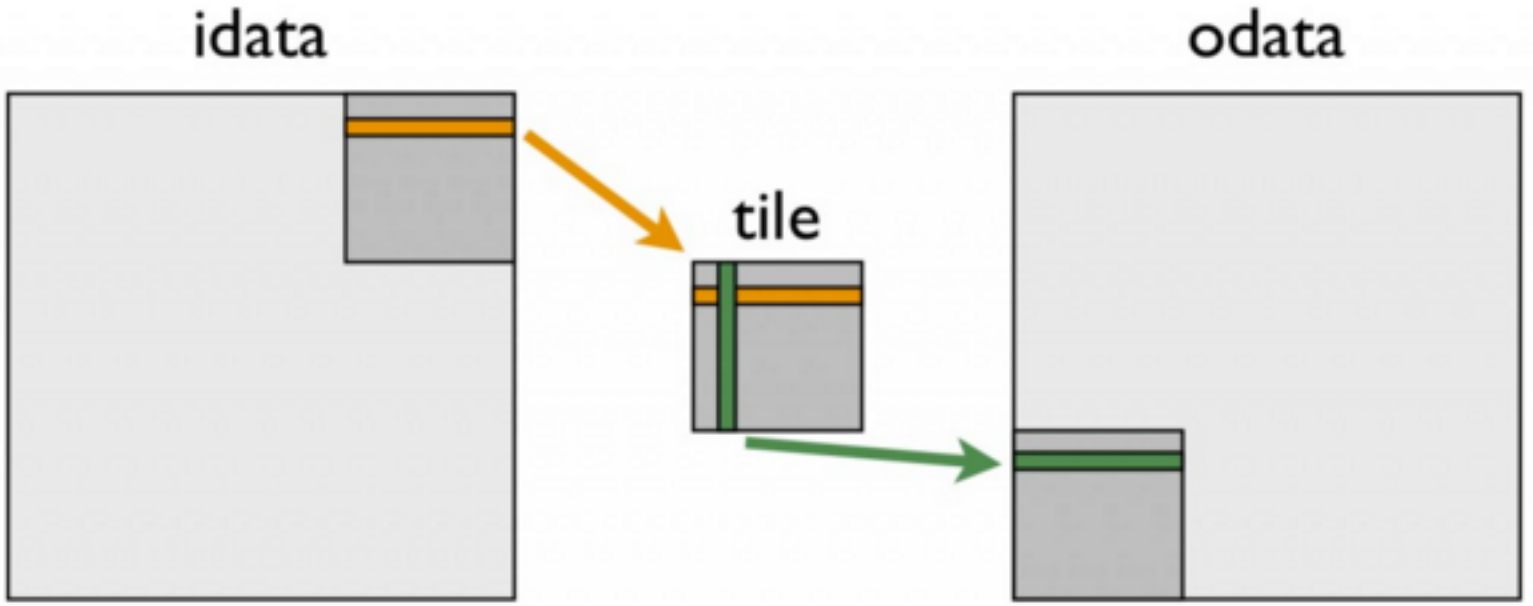
Down Sweep Phase

scanBlocks()



2, Computing integral image.

transpose()



- 3, If use CPU, the finding part looks like

```
for each (x, y)
    for each (w, h)
        area = integral_img(x, y, w, h)
        Largest = max(area, Largest)
```

In GPU, I use

```
for each (w, h)                          -> on CPU
    area = findLargest(any (x,y ))       -> on CPU
    Largest = max(area, Largest)
```

findLargest(any(x, y))

Reduction tree with interleaved addressing.

findLargestInBlocks()

CPU find largest in blocks. This can be paralleled in GPU.

- Scan Part

```
dim3 threadsPerBlock(SECTION_SIZE);
dim3 blocks((int)ceil((float)N/SECTION_SIZE));
```

- Transpose Part

```
dim3 block_dim(BLOCK_DIM,BLOCK_DIM);
dim3 grid_dim((int)ceil((float)N/BLOCK_DIM), (int)ceil((float)M/BLOCK_DIM));
```

- Finding Largest

```
dim3 threadsPerBlock(256);
dim3 blocks((int)ceil((float)width*height/256));
```

- Scan Part

```
__shared__ int partialSum[SECTION_SIZE];
```

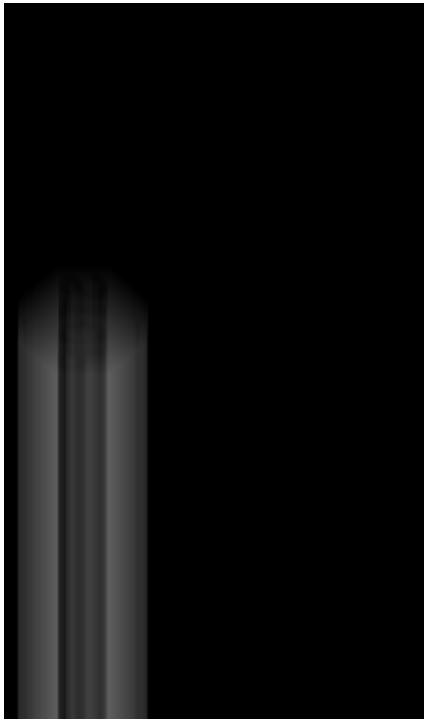
- Transpose Part

```
__shared__ int tile[BLOCK_DIM][BLOCK_DIM];
```

- Finding Largest

```
__shared__ float sdata[256];
__shared__ float sIndices[256];
```


STOP



Result