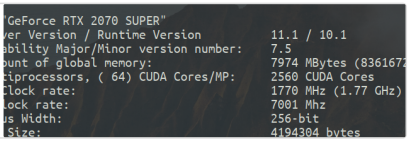
LAB4

Created	@Sep 27, 2020 4:14 PM
Tags	

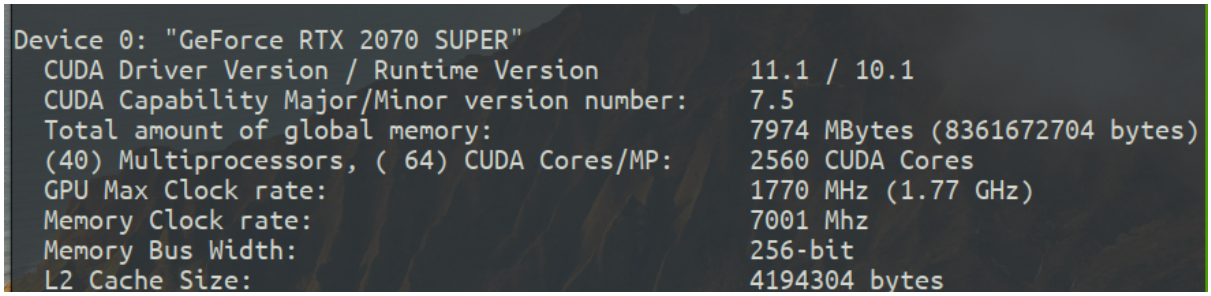
LAB4

1. Device Info

https://www.notion.so/zhiquanw/LAB4-4e618363f27b4faa9c52fcbc4e2de4e2



1. Device Info



```
Device 0: "GeForce RTX 2070 SUPER"
  CUDA Driver Version / Runtime Version          11.1 / 10.1
  CUDA Capability Major/Minor version number:    7.5
  Total amount of global memory:                 7974 MBytes (8361672704 bytes)
  (40) Multiprocessors, ( 64) CUDA Cores/MP:     2560 CUDA Cores
  GPU Max Clock rate:                            1770 MHz (1.77 GHz)
  Memory Clock rate:                             7001 Mhz
  Memory Bus Width:                              256-bit
  L2 Cache Size:                                 4194304 bytes
```

2. Original Video

▼ video

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/0bf6d752-4478-428e-b2f5-b084279f8176/IMG_7237.mp4

3. Convert to images with ffmpeg

```
ffmpeg -i input.flv -vf fps=60 out%d.png
```

1906 images generated.

4. Parameters

- Image Number: 1906
- Image Size: 1920 * 1080 * 3
- Circular Buffer Size : 300 (300 images in buffer at most)
- grid size: 64
- blur radius: 50

5. Implementation

Kernel

```
__global__ void motion_blur_kernel(const unsigned int n, const unsigned char *d_images, unsigned char *d_results,
                                   const unsigned int start, const unsigned int proc_num) {
    unsigned int cuda_start_idx;
    unsigned int row_idx = blockIdx.x * blockDim.x + threadIdx.x;
    unsigned int col_idx = blockIdx.y * blockDim.y + threadIdx.y;

    unsigned int pixel_offset = CHANNEL * (row_idx * WIDTH + col_idx);
    for (unsigned int i = 0; i < proc_num; ++i) {
        cuda_start_idx = (start + i) % MAX_IMAGE_IN_DEVICE;

        unsigned int last_image_idx = cuda_start_idx + n;
        unsigned int target_image_idx = cuda_start_idx * IMAGE_SIZE;
        unsigned int target_image_offset = target_image_idx + pixel_offset;
        float r = 0;
        float g = 0;
        float b = 0;
        int cuda_copy_idx;
        float start_weight, delta_h;
        if (MODE == 0) {
            start_weight = 1.0f / (float)n;
            delta_h = 0.0f;
        } else if (MODE == 1) {
            start_weight = 2.0f / (float)n;
            delta_h = - 2.0f / (float)n / (float)n;
        }
        for (unsigned int j = cuda_start_idx; j < last_image_idx; ++j) {
            float pos = (float)j - (float)cuda_start_idx;
            cuda_copy_idx = j % MAX_IMAGE_IN_DEVICE;
            unsigned int blur_image_offset = cuda_copy_idx * IMAGE_SIZE;
            unsigned int offset = pixel_offset + blur_image_offset;
            float weight = start_weight +  pos* delta_h;
            if (MODE==2){
                weight = 1.0f/sqrt(2.0f*3.141592654f) * exp(-(1.0f/2.0f)*(4*pos/n*pos/n));
            }
            if (offset < MAX_IMAGE_IN_DEVICE * IMAGE_SIZE - 2) {
                r += weight * (float) d_images[offset];
                r = min(r,255.0f);
                g += weight * (float) d_images[offset + 1];
                g = min(g,255.0f);
                b += weight * (float) d_images[offset + 2];
                b = min(b,255.0f);
            }

        }
        d_results[target_image_offset] = (unsigned char) min((unsigned char) r, 255);
        d_results[target_image_offset + 1] = (unsigned char) min((unsigned char) g, 255);
        d_results[target_image_offset + 2] = (unsigned char) min((unsigned char) b, 255);
    }
}
```

6. Results (r=15)

Rect Blur Example

▼ video

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/56a44610-3ebb-41cd-8d21-4afa6aa989dd/rect_15.mp4

Tri Blur Example

▼ video

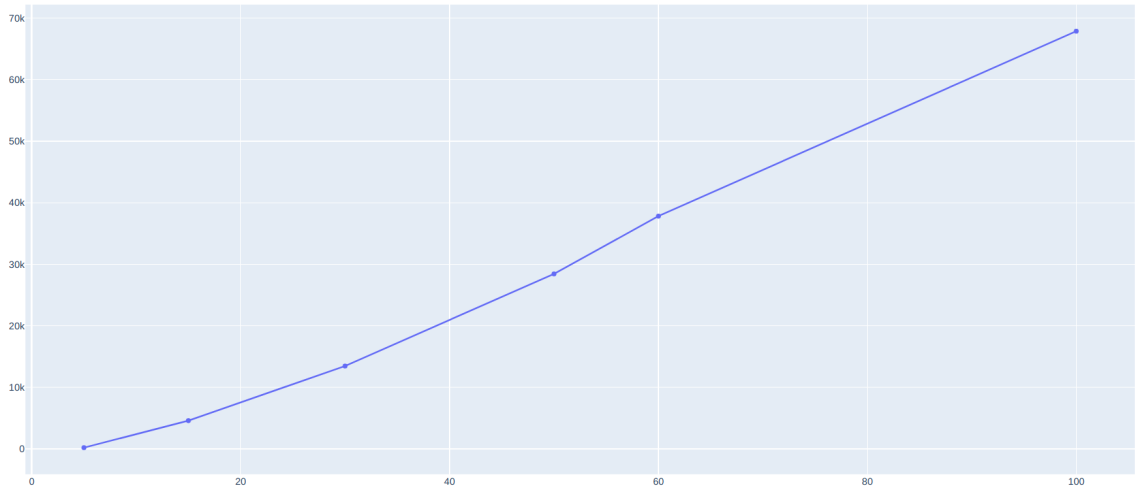
https://s3-us-west-2.amazonaws.com/secure.notion-static.com/eeba218f-02b3-45d2-9e8e-9780352fb3a1/tri_15.mp4

Gaussian Blur Example

▼ video

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/4af06acb-5340-43ad-9b84-6bad6155da4d/gaussain_15.mp4

7. Performance (input & output excluded) in ms



8. Conclusion

The computing duration increase linearly as the task increases linearly.