Advaned Programming fo HPC - Report labwork 6

NGUYEN TAT HUNG

November 5, 2021

Implementation

```
__device__ int brightness(){

}
__device__ int blending(){

}

__device__ int binarization(int input, int threshold){
    if(input < threshold){
        return 0;
    }else{
        return 255;
    }
}

__global__ void labwork6_a(uchar3 *input, uchar3 *output, int imgWidth, int imgHeight){
    int col = threadIdx.x + blockIdx.x * blockDim.x;
    int row = threadIdx.y + blockIdx.y * blockDim.y;
    if(col > imgHeight || row > imgWidth){
        return;
    }
    int tid = col + row * imgWidth;

    int threshold = 128;

    output[tid].x = binarization(input[tid].x, threshold);
    output[tid].z = output[tid].y = output[tid].x;
}

void Labwork::labwork6_GPU() {
    // Calculate number of pixels
    int pixelCount = inputImage->width * inputImage->height;

    outputImage = static_cast<char *>(malloc(pixelCount * 3));
    // Allocate CUDA memory
    uchar3 *devInput;
    uchar3 *devOutput;
    uchar3 *devGray;

    cudaMalloc(&devInput, pixelCount*3); // Perfect version
    cudaMalloc(&devOutput, pixelCount*3); // Perfect version
    cudaMalloc(&devGray, pixelCount*3); // Perfect version
    // Copy CUDA Memory from CPU to GPU
    cudaMemcpy(devInput, inputImage->buffer, pixelCount*3, cudaMemcpyHostToDevice); // Per

    // Processing
    dim3 blockSize = dim3(16, 16);
    dim3 gridSize = ((int) ((inputImage->width + blockSize.x - 1)/blockSize.x), (int)((inpu

    grayscale_2d<<<gridSize, blockSize>>>(devInput, devOutput, inputImage->width, inputIma
    //labwork6_a<<<gridSize, blockSize>>>(devGray, devOutput, inputImage->width, inputImage

    // Copy CUDA Memory from GPU to CPU
    cudaMemcpy(outputImage, devOutput, pixelCount*3, cudaMemcpyDeviceToHost); // Perfect v

    // Cleaning

    cudaFree(devInput);
    cudaFree(devOutput);
    cudaFree(devGray);
}
```

Result

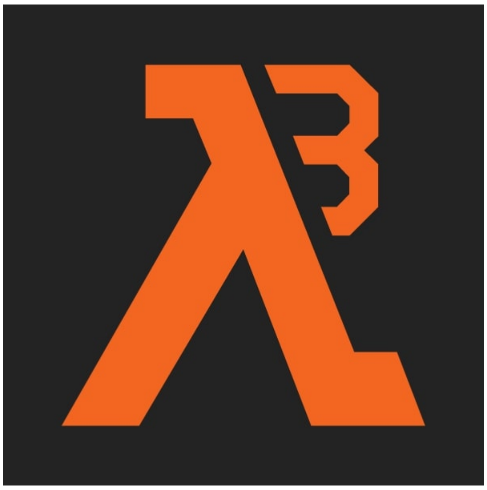

Figure 1: Original input image

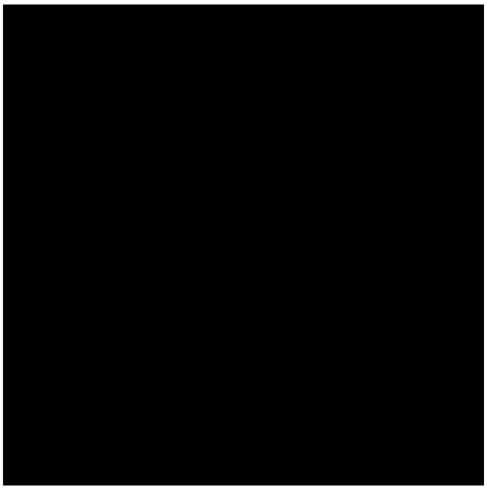

Figure 2: Output image