UNIVERSITÄT TÜBINGEN
PROF. DR.-ING. HENDRIK P.A. LENSCH
LEHRSTUHL FÜR COMPUTERGRAFIK
LUKAS RUPPERT (LUKAS.RUPPERT@UNI-TUEBINGEN.DE)
FAEZEH ZAKERI (FAEZEH-SADAT.ZAKERI@UNI-TUEBINGEN.DE)

MARCH 6, 2024

MASSIVELY PARALLEL COMPUTING
ASSIGNMENT 1

Instructions

Download the framework `exercise01.zip` from the ILIAS course web page.
Present your results to the exercise instructors to get a grading on this exercise sheet.

1.1 Run a Simple Program (10)

- Compile and run the pre-written CUDA test program: `deviceQuery`
- Show its output to an exercise instructor.

1.2 Move Data between Host and GPU (20)

- Start from the `cudaMallocAndMemcpy` template.
- Part 1: Allocate memory for pointers d_a and d_b on the device.
- Part 2: Copy h_a on the host to d_a on the device.
- Part 3: Do a device to device copy from d_a to d_b.
- Part 4: Copy d_b on the device back to h_a on the host.
- Part 5: Free d_a and d_b on the host.
- Bonus: Experiment with `cudaMallocHost(...)` in place of `malloc(...)` for allocating h_a.

1.3 Launching Kernels (20)

- Start from the `myFirstKernel` template.
- Part 1: Allocate device memory for the result of the kernel using pointer d_a.
- Part 2: Configure and launch the kernel using a 1D grid of 1D thread blocks.
- Part 3: Have each thread set an element of d_a as follows:

```
unsigned int idx = blockIdx.x*blockDim.x + threadIdx.x;
d_a[idx] = 1000*blockIdx.x + threadIdx.x;
```

- Part 4: Copy the result in d_a back to the host pointer h_a.
- Part 5: Verify that the result is correct.

1.4 Image Difference (30)

- Start from the `gamma` template.
- Compute the absolute difference between two images independently for all three colors
- Output is again an RGB image
- Execute your program with the given image pair: vase.ppm and vase_blur.ppm

1.5 Dot Product (20)

- Start from the `dotProduct` template.
- Compute the dot product of two large scalar vectors (i.e. the result is one scalar value at the end)
- Rather than computing the dot product in one go, we will accumulate intermediate results per thread and then sum up the results of all threads on the CPU.
- Given n threads and m elements in the vector, each thread has to iterate m/n times (+1, if necessary)
- Execute for vector sizes 10.000, 1.000.000, 10.000.000, 100.000.000
- Similar to the already given CPU version, the dot product should be executed multiple times
- At how many dimensions does the GPU start being faster than the CPU?
  - make sure to compile in Release mode
  - use the shell `time` command to determine the execution time