Lab 7

Introduction to Programming Laboratory

Outline

- Survey
- CUDA Reminders
- Shared Memory & Occupancy
- 2D Memory & Kernel Launch
- HW3 updates
- Task
- Contact

Survey

問卷 統計

CUDA Reminders

Caller & callee locations

Specifier	Caller	Callee
`__host__`	host	host
`__global__`	host	device
`__device__`	device	device

host = CPU, device = GPU

Error handling

All CUDA API calls return a `cudaError_t` value.

Remember to check them!

You can use `cudaGetLastError`, `cudaPeekAtLastError`, `cudaGetErrorName`, `cudaGetErrorString`

Introductory matrial

CUDA C/C++ Basics

— tutorial @SC11 by Cyril Zeller, NVIDIA

Shared Memory & Occupancy

With the help of compiler...

- Use the `-Xptxas=-v` flag to see how much resource your kernel function uses
- `gmem`: global memory
- `smem`: shared memory
- `registers`: registers, typically for storing local variables

Shared Memory

- All threads within the same block share the **shared memory**
- Global memory read/write is expensive, so when a region of global memory is frequently used (read/write) by threads within a block, considering putting it in shared memory
- Cooperate the threads to put the desired data in shared memory
- Use `__syncthreads` to synchronize the threads

Occupancy

The number of active blocks are limited by:

- shared memory usage
- register usage
- max threads / threads per block

Occupancy

We can use ask CUDA to suggest grid and block size that achieves maximum potential occupancy for a device function using **`cudaOccupancyMaxPotentialBlockSize`**. (This does not directly translates to maximum performance)

See also: `samples/0_Simple/simpleOccupancy`

2D Memory & Kernel Launch

Why?

Ease programming, nearby indices in a matrix tend to share memory better

API functions

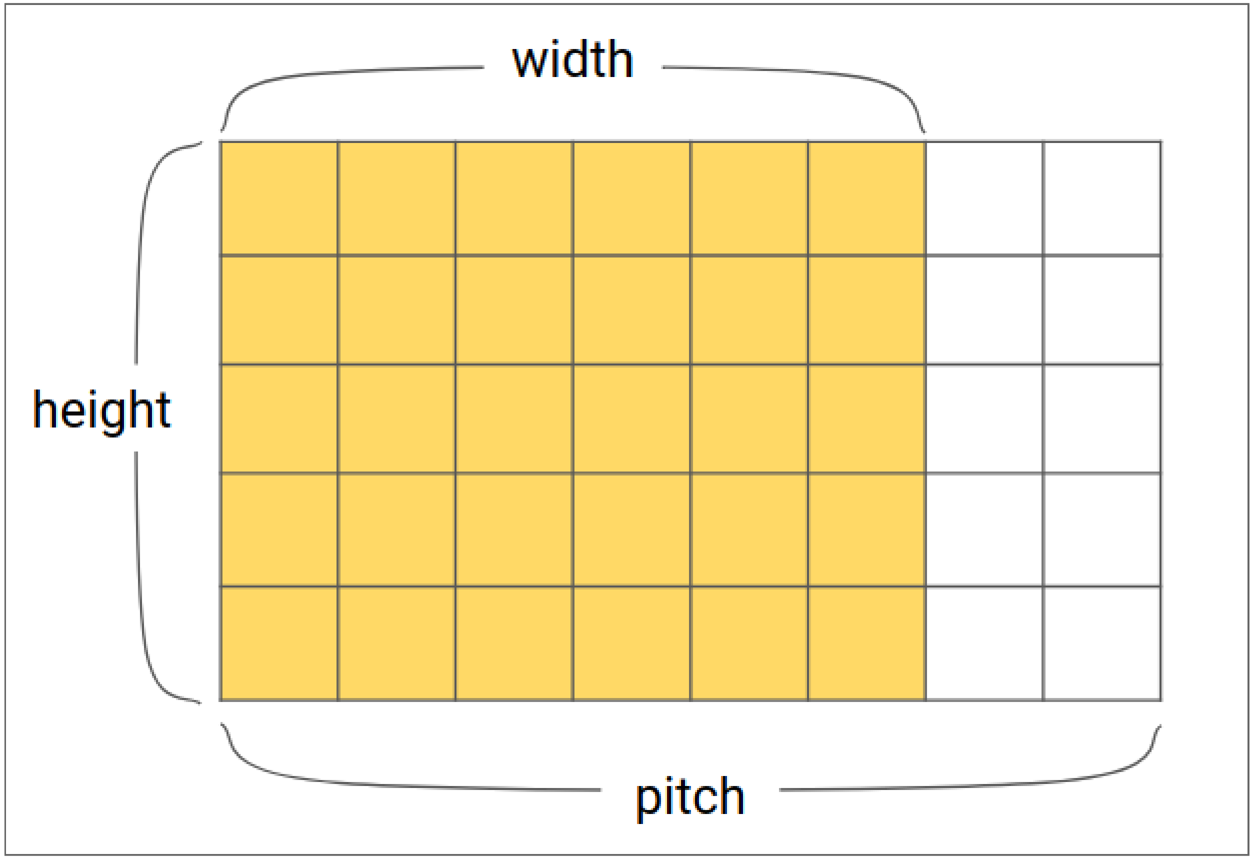
- `__host__ cudaError_t cudaMallocPitch ( void** devPtr, size_t* pitch, size_t width, size_t height )`
- `__host__ cudaError_t cudaMemcpy2D ( void* dst, size_t dpitch, const void* src, size_t spitch, size_t width, size_t height, cudaMemcpyKind kind )`

Concepts

suppose we have an 2D data region with
$(0, 0) \le (x, y) < (xMax, yMax)$

- `width`: distance (size in bytes) between (0, y) and (xMax, y)
- `height`: equals yMax, e.g. number of rows
- `pitch`: distance (size in bytes) between (x, y) and (x, y + 1)

cudaMallocPitch



HW3 updates

Contact

Email	**afg984@gmail.com**
GitHub	**afq984**
LinkedIn	**afq984**
Steam	**afg984**

for other social media, perform the search yourself :)