Visualizing Millions of Datapoints with GPUs

using both the client and the server

Thibaud Hottelier (in thibaudhottelier)
thibaud@graphistry.com

graphistry

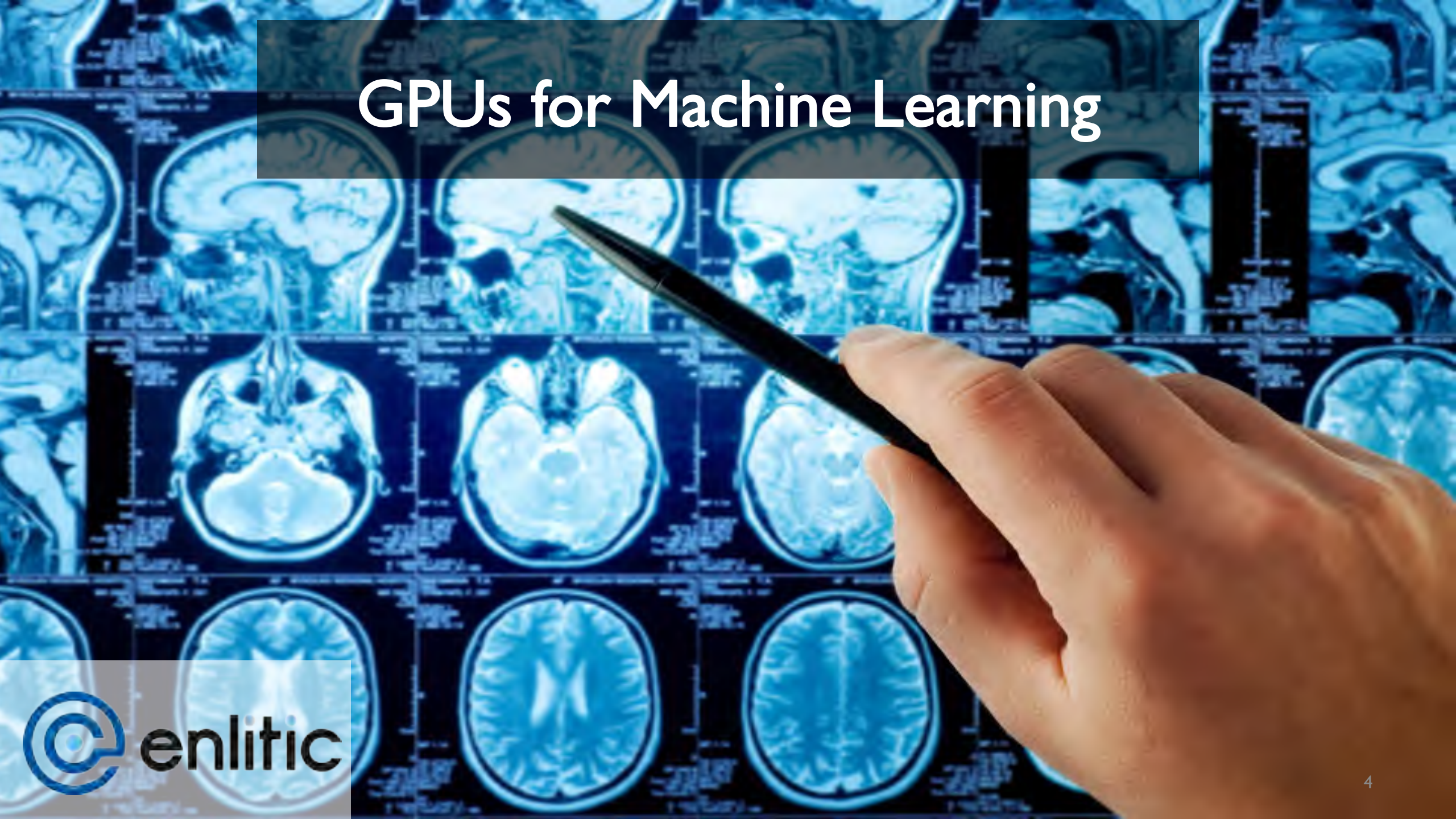
GRID + SHIELD


BRUTAL LEGEND
Vanishing of Ethan Carter
MONSTERS
AMERICAN NIGHTMARE
DIRT 3
GRID 2
THE WITCHER
BORDERLANDS
GRID
DEAD ISLAND
SAINTS ROW
DEAD RISING 2
LEGO BATMAN 2
BROTHERS
REPTO 95
LEGO BATMAN
THE WALKING DEAD
GAS GUZZLERS
NVIDIA.
GEFORCE NOW
THE NEW WAY TO GAME.
$7.99/MONTH
FIRST 3 MONTHS FREE!*
LEARN MORE

GPUs for Live VR Streaming

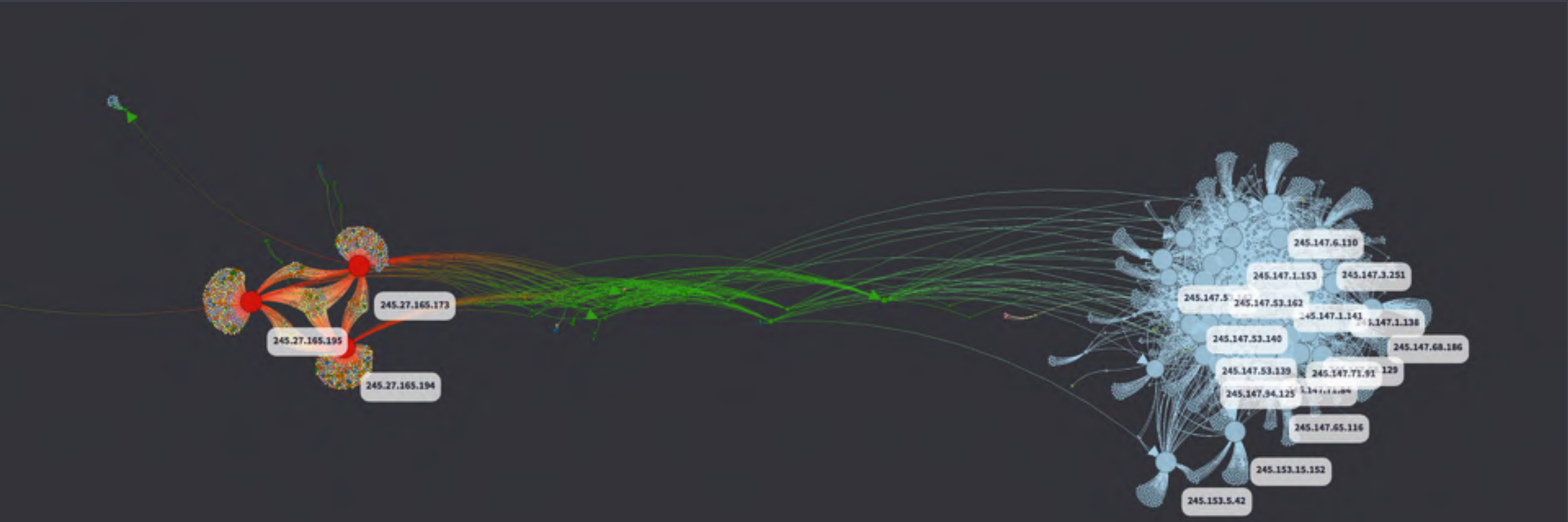
otoy

GPUs for Machine Learning



enlitic

GPUs to Map Cyber Attacks



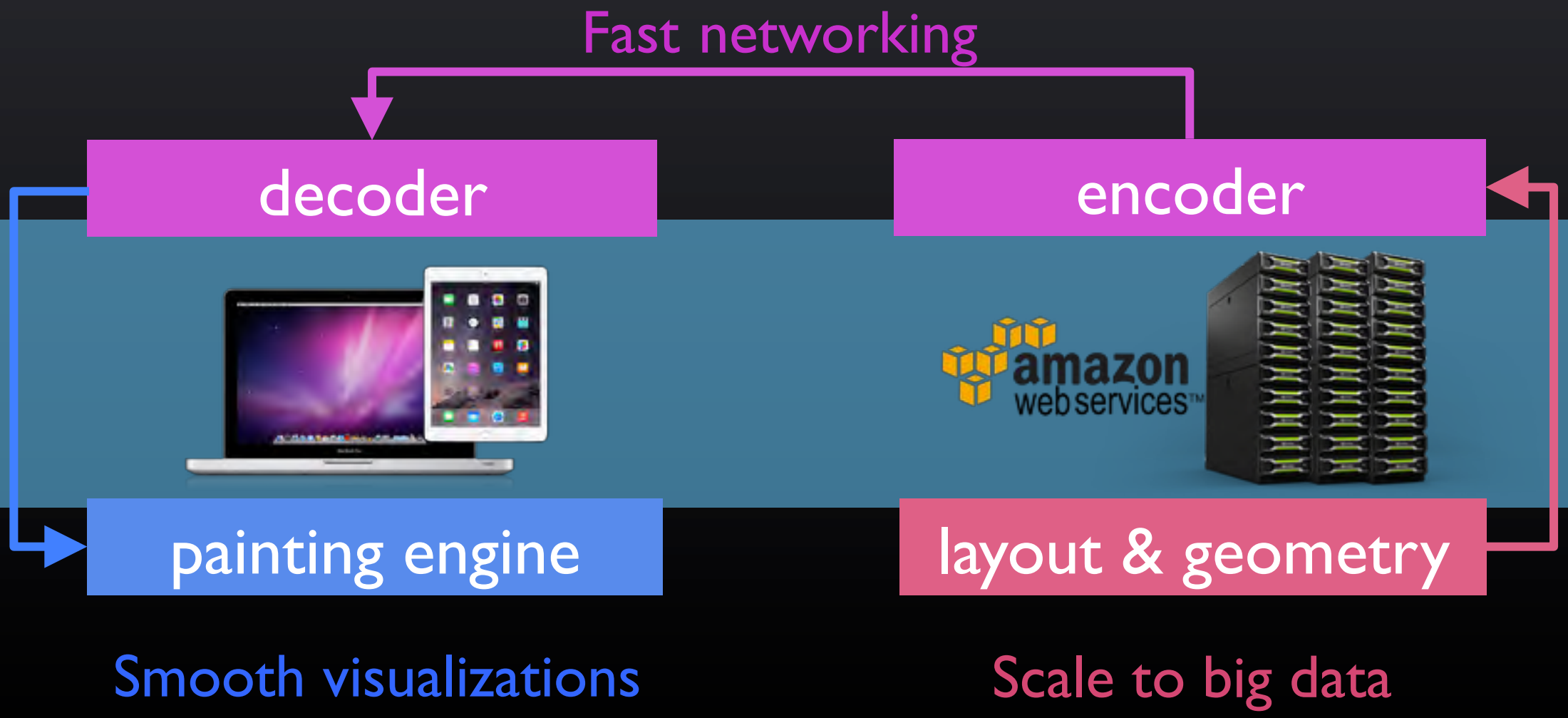
1. **Distributed Rendering: Dual GPUs**

2. **GPU Programming with Node**

   General Programming with GPUs

   Using node-opencl and cl.js for easy acceleration

Dual GPU Architecture



Why Not All Clientside?

We have WebGL, but…

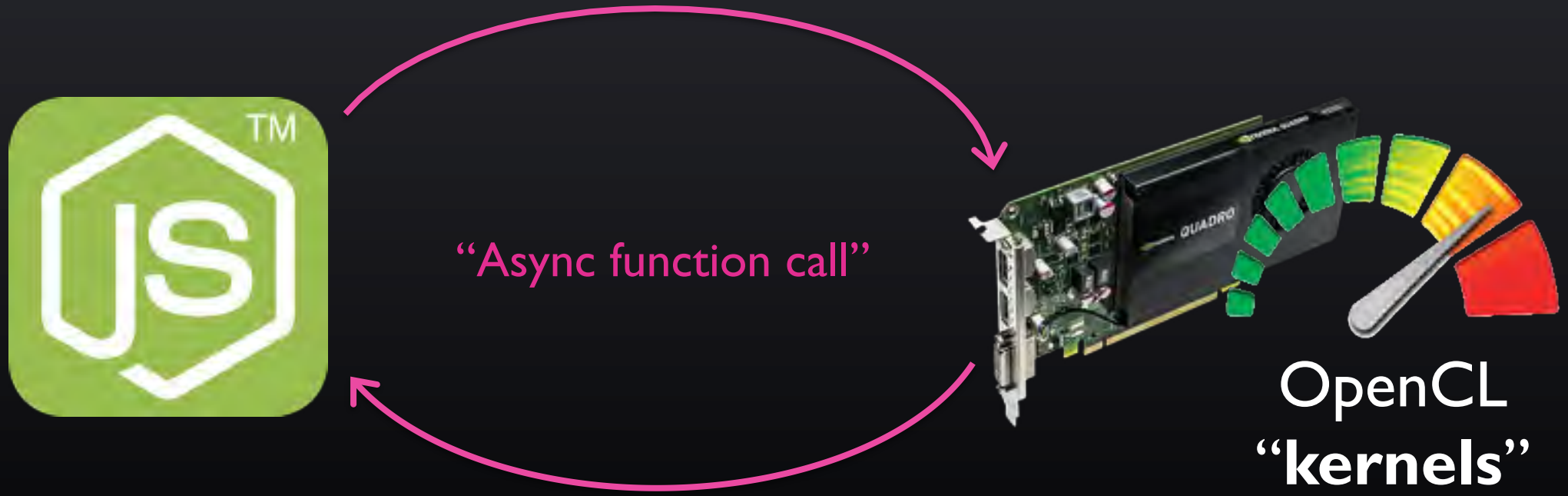
- 1GB+/minute input data
- Client provides very restricted access to GPU features
  - Not even WebGL2 provides core performance features, e.g., memory barriers.

Why Not All Serverside?


Low-level
C APIs


See other
great talks!

Always Bet on JS



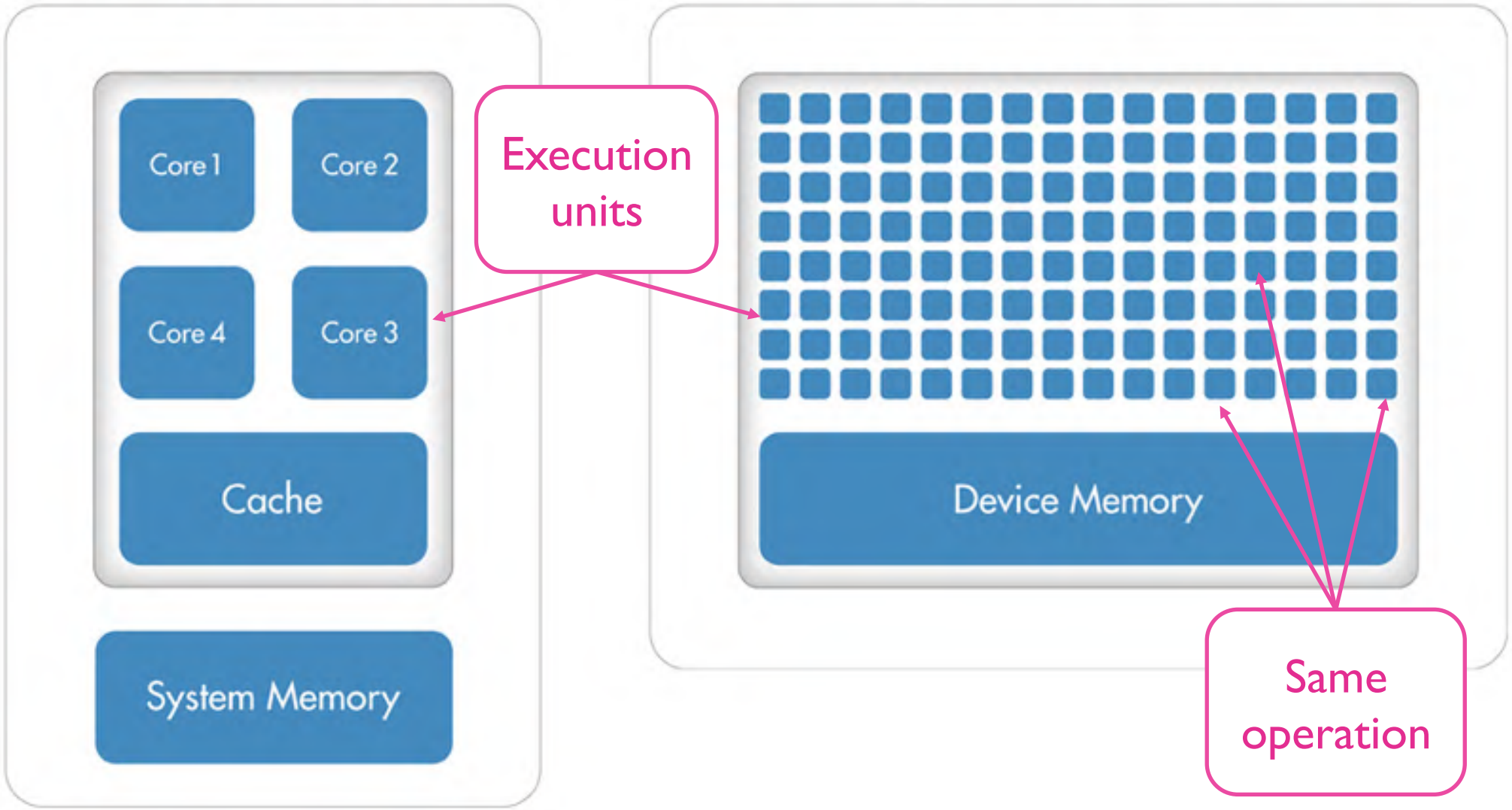
JS community has (inadvertently) made GPU programming awesome!

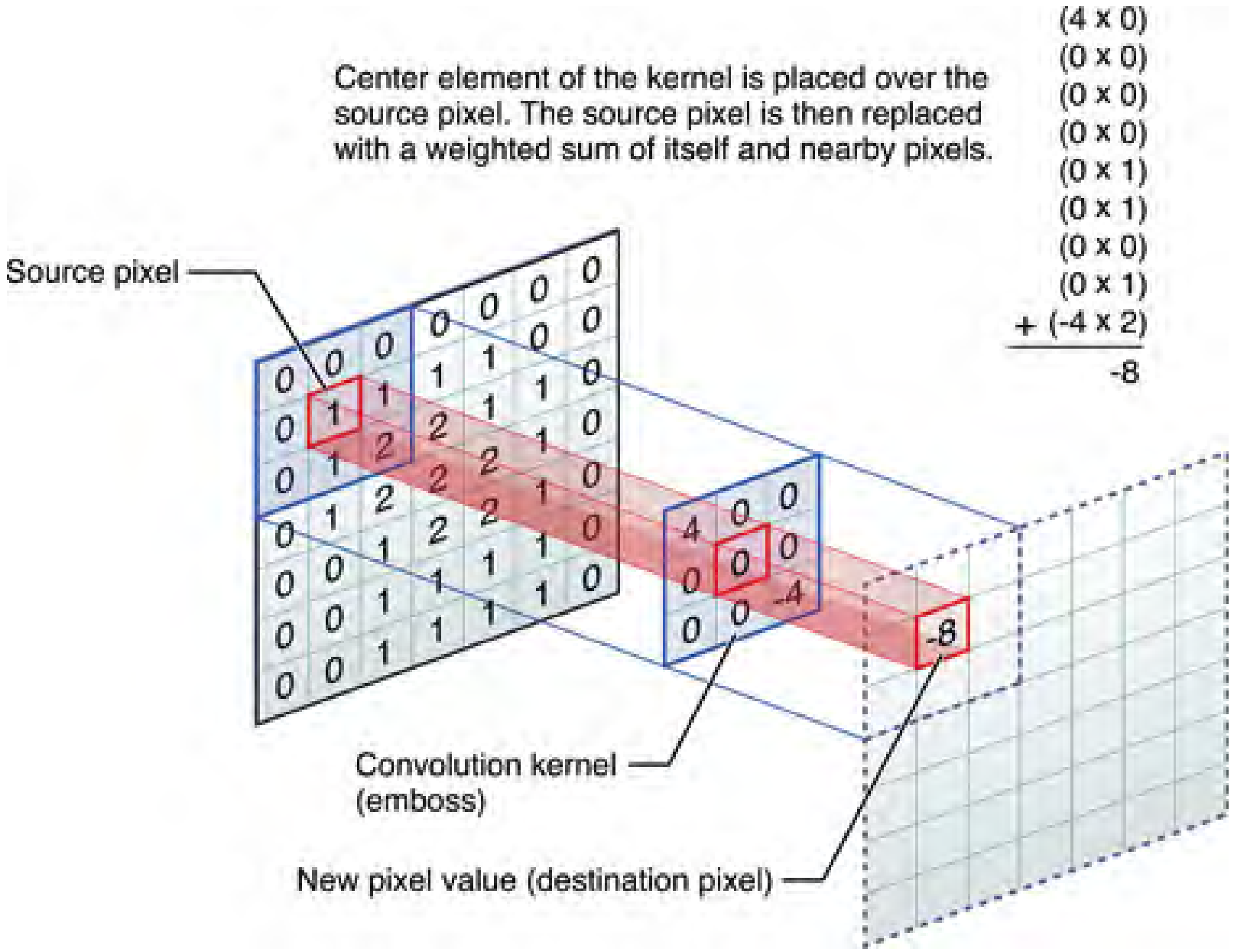
CPU

Few but Flexible

GPU

Many but Rigid





Source: Apple iOS Developer Library

```
int centerIdx = get_global_id(0);
int sum = 0;

for (int dx = -1; dx <= 1; dx++) {
    for (int dy = -1; dy <= 1; dy++) {
        int idx = centerIdx + (4 * width * dy) + (4 * dx);
        idx = isValidImageIndex(idx, numElements) ? idx : centerIdx;
        int scaleFromMask = mask[(dy+1)*3 + (dx+1)];
        sum += ((int) imageData[idx]) * scaleFromMask;
    }
}

if (isTransparencyNotColor(centerIdx)) {
    sum = (uchar) 255;
}

newImageData[centerIdx] = make8Bit(sum);
```

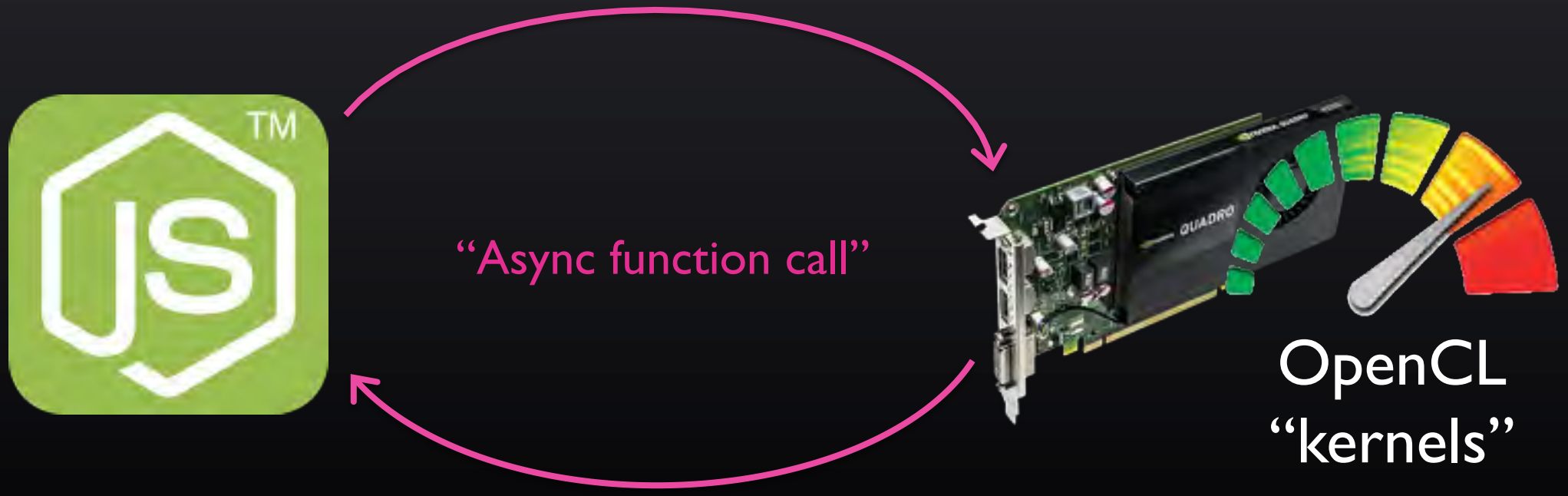
ID (Pixel)

For Each Neighbor

Do Math

Write Result

Using Kernels From JavaScript



Node-opencl

Works on Intel, Nvidia, and AMD hardware

Developed by Mikael Sevenier at AMD.

Provides bindings in node to C++ driver functions.

Handles memory management, concurrency, etc., for you!

https://github.com/mikeseven/node-opencl

```
// Setup Context
var ocl = require('node-opencl');
var platforms = ocl.getPlatformIDs();
var platform = platforms[0];
var devices = ocl.getDeviceIDs(platform, ocl.DEVICE_TYPE_ALL);
var clErrorHandler = function (e) { throw e;};
var context = ocl.createContext([ocl.CONTEXT_PLATFORM, platform],
        device, clErrorHandler, clErrorHandler);
var queue = ocl.createCommandQueue(context, device, 0);
```

CL.js

```
var cl = require('cljs');

var myKernel = cl.createKernel('kernel.cl', 'kernelName');

var inputBuffer  = cl.createBuffer(input);
var outputBuffer = cl.createBuffer(output);

return myKernel.run(
    [256],
    workgroup,
    [inputBuffer, outputBuffer]
).then(function () {
    return outputBuffer.read(Uint8Array);
});
```

Make a kernel

Upload data to GPU

Number of items (eg, pixels)

Kernel arguments

Download result

https://github.com/graphistry/cljs

I Will Do It Live!

Takeaways

Use the very best tool for each task!

OpenCL

Performance (60x with one GPU)

Productivity (95% of code)

Thanks!

Try out

CL.js	`github.com/graphistry/cljs`
PyGraphistry	`github.com/graphistry/pygraphistry`

Help us catch hackers

graphistry

We are hiring
Front-end / Full-stack
UI/UX design

Email us at
thibaud@graphistry.com