# Lecture 10: Supercomputing for Robotics

Wei Qi Yan

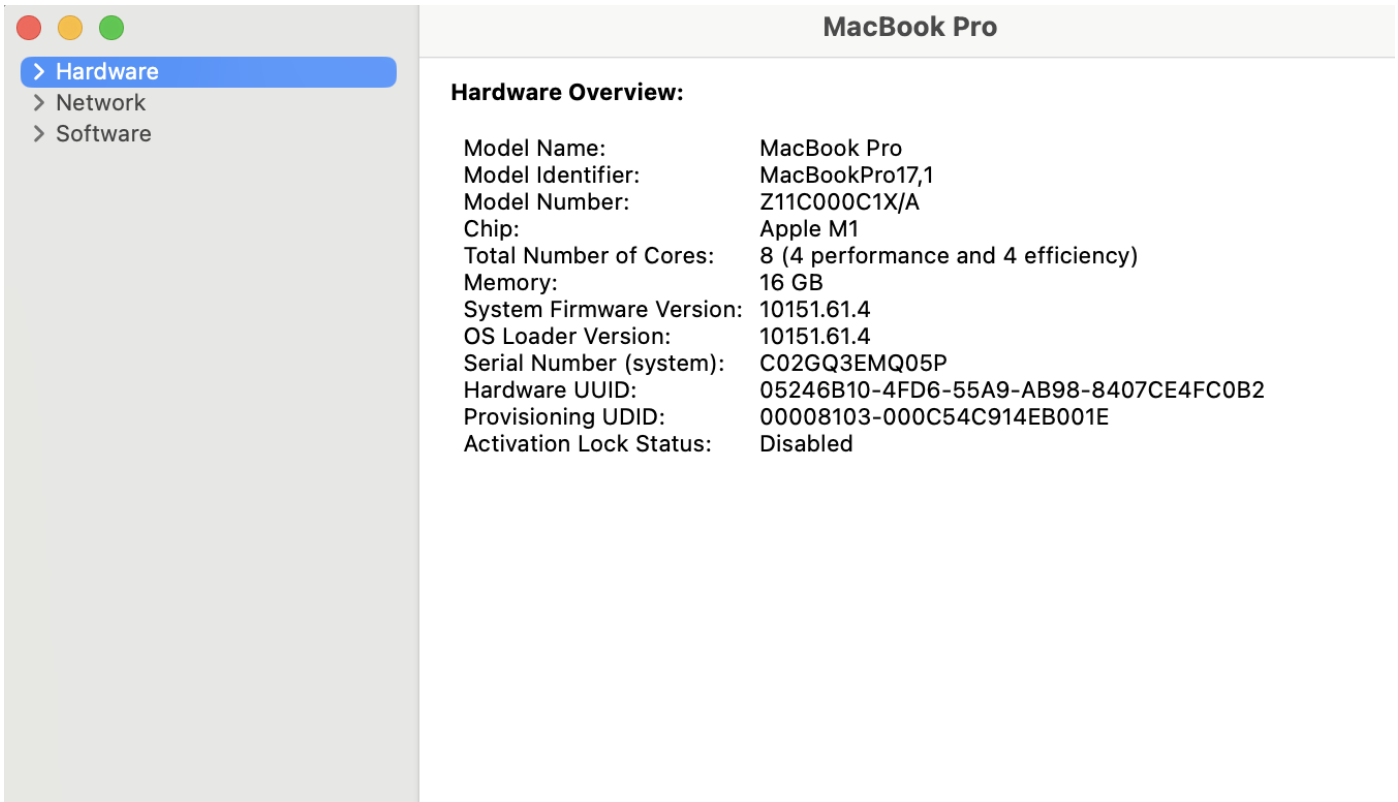
Auckland University of Technology

# Table of Contents

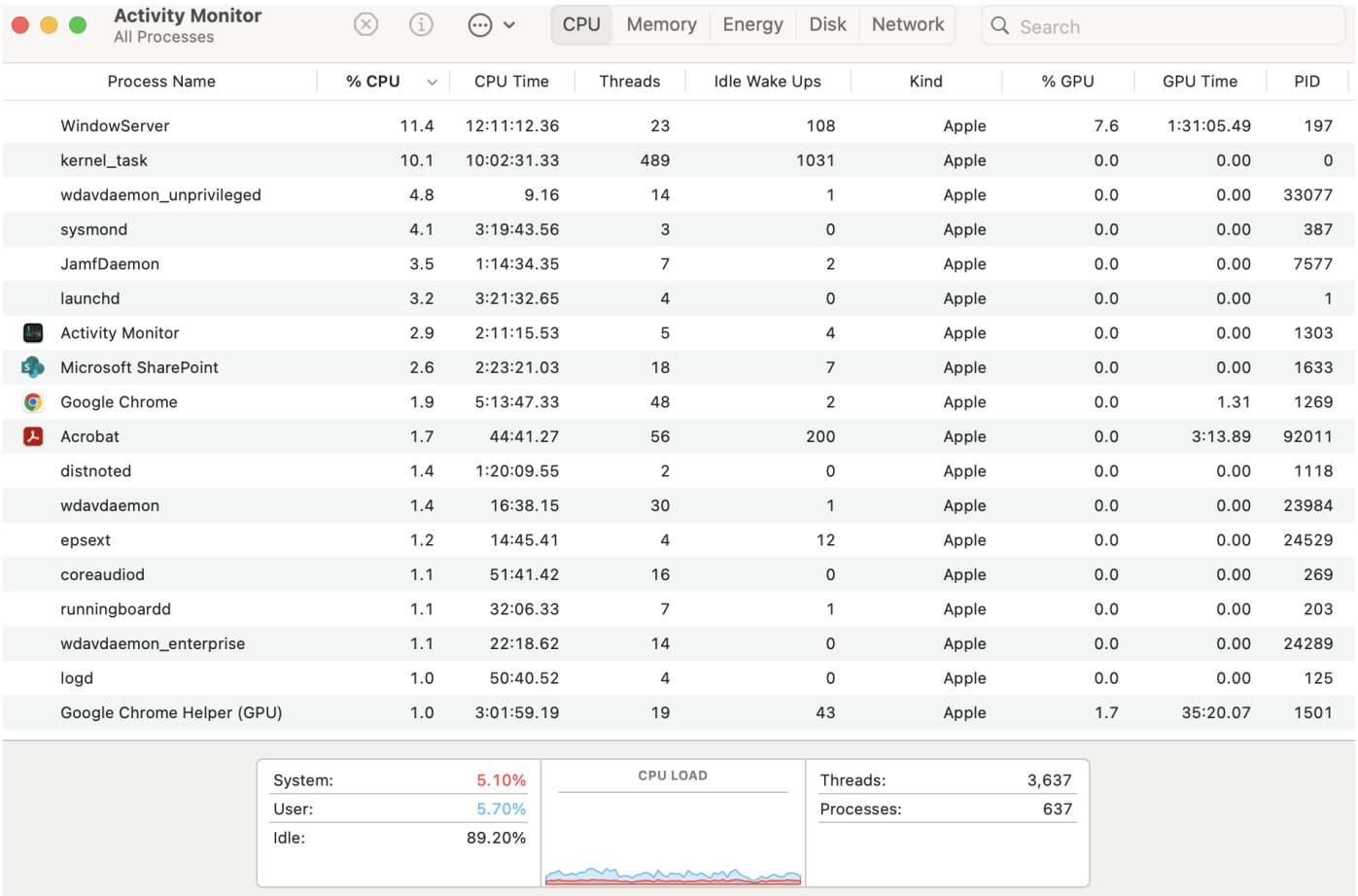
## Challenges of supercomputing

- Challenges of modern computing
- Multicore computing
- Multithread computing
- Graphics Processing Unit (GPU)
- Mobile computing for robotics
- Cloud computing for robotics
- .....

## My Mac Laptop

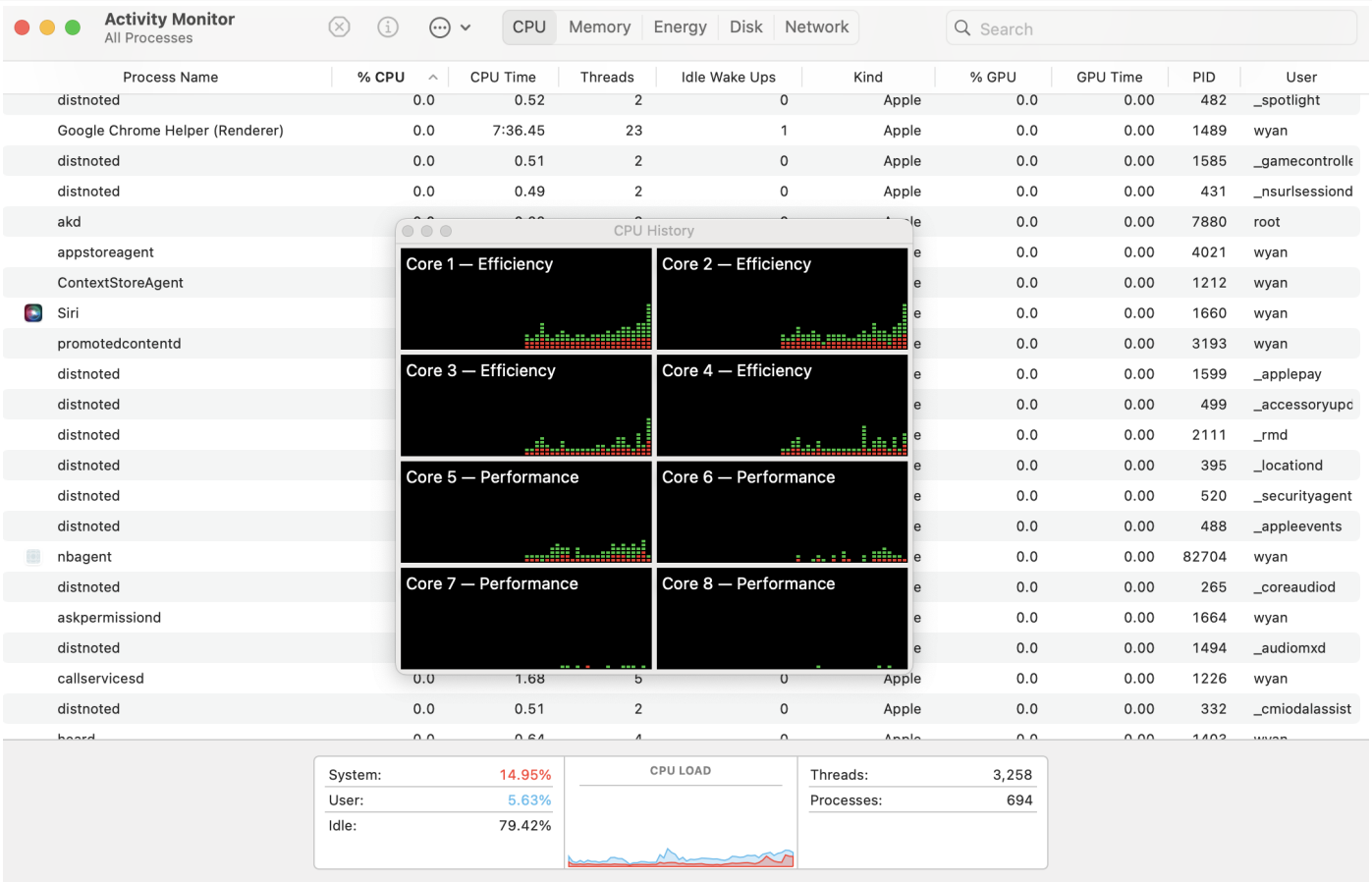


## My Mac Laptop

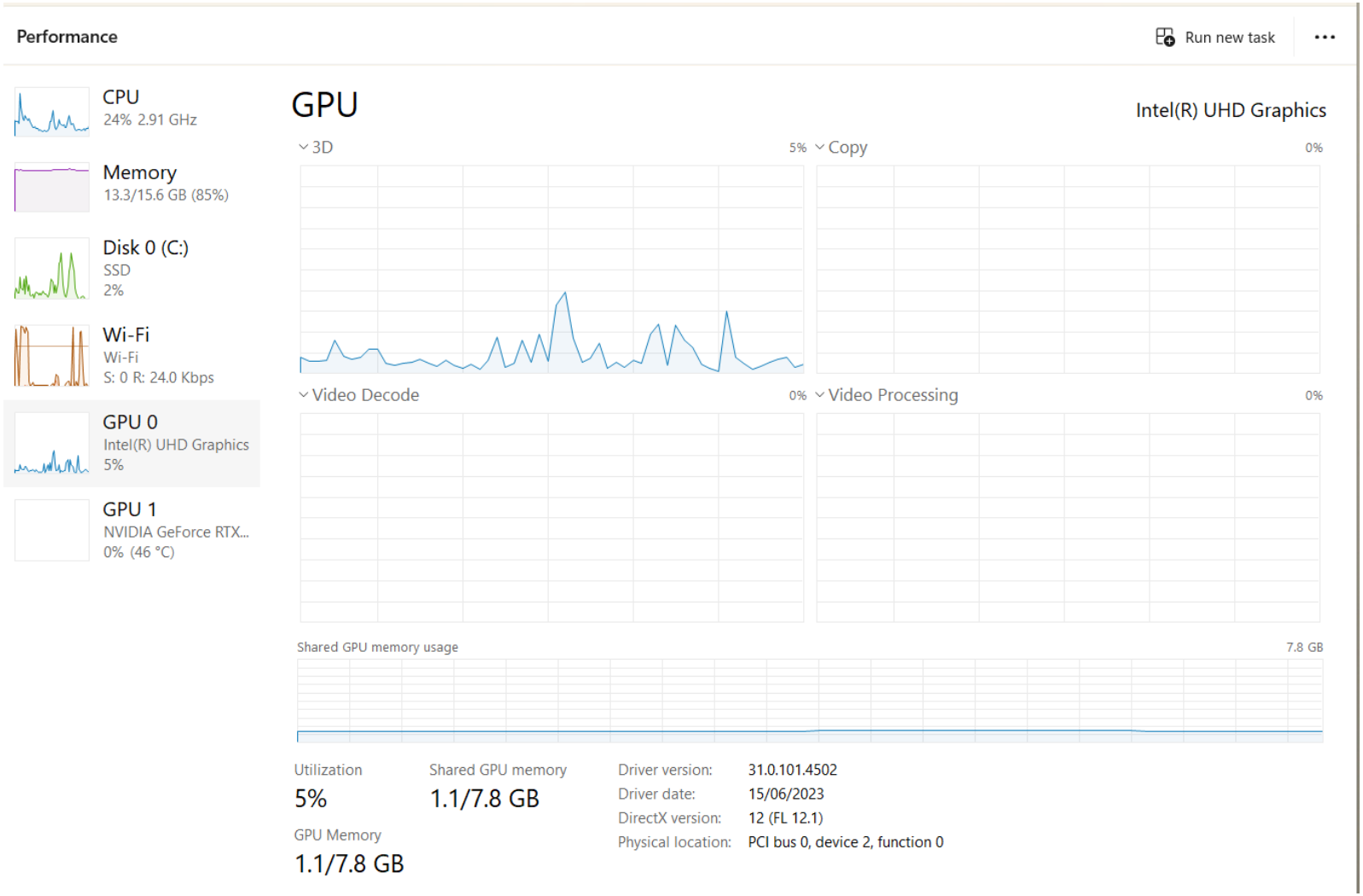



| Process Name | % CPU | CPU Time | Threads | Idle Wake Ups | Kind | % GPU | GPU Time | PID |
|---|---|---|---|---|---|---|---|---|
| WindowServer | 11.4 | 12:11:12.36 | 23 | 108 | Apple | 7.6 | 1:31:05.49 | 197 |
| kernel_task | 10.1 | 10:02:31.33 | 489 | 1031 | Apple | 0.0 | 0.00 | 0 |
| wdavdaemon_unprivileged | 4.8 | 9.16 | 14 | 1 | Apple | 0.0 | 0.00 | 33077 |
| sysmond | 4.1 | 3:19:43.56 | 3 | 0 | Apple | 0.0 | 0.00 | 387 |
| JamfDaemon | 3.5 | 1:14:34.35 | 7 | 2 | Apple | 0.0 | 0.00 | 7577 |
| launchd | 3.2 | 3:21:32.65 | 4 | 0 | Apple | 0.0 | 0.00 | 1 |
| Activity Monitor | 2.9 | 2:11:15.53 | 5 | 4 | Apple | 0.0 | 0.00 | 1303 |
| Microsoft SharePoint | 2.6 | 2:23:21.03 | 18 | 7 | Apple | 0.0 | 0.00 | 1633 |
| Google Chrome | 1.9 | 5:13:47.33 | 48 | 2 | Apple | 0.0 | 1.31 | 1269 |
| Acrobat | 1.7 | 44:41.27 | 56 | 200 | Apple | 0.0 | 3:13.89 | 92011 |
| distnoted | 1.4 | 1:20:09.55 | 2 | 0 | Apple | 0.0 | 0.00 | 1118 |
| wdavdaemon | 1.4 | 16:38.15 | 30 | 1 | Apple | 0.0 | 0.00 | 23984 |
| epsext | 1.2 | 14:45.41 | 4 | 12 | Apple | 0.0 | 0.00 | 24529 |
| coreaudiod | 1.1 | 51:41.42 | 16 | 0 | Apple | 0.0 | 0.00 | 269 |
| runningboardd | 1.1 | 32:06.33 | 7 | 1 | Apple | 0.0 | 0.00 | 203 |
| wdavdaemon_enterprise | 1.1 | 22:18.62 | 14 | 0 | Apple | 0.0 | 0.00 | 24289 |
| logd | 1.0 | 50:40.52 | 4 | 0 | Apple | 0.0 | 0.00 | 125 |
| Google Chrome Helper (GPU) | 1.0 | 3:01:59.19 | 19 | 43 | Apple | 1.7 | 35:20.07 | 1501 |

## My Mac Laptop



## My GPU Laptop



## Challenges of Supercomputing

- *Parallel computing*: Simultaneous use of multiple processors
- *Multi-core processors*: Multiple processors (cores) on a single chip
- *Cluster computing*: Hierarchical combination of commodity units to build parallel system
- *Supercomputing*: Historically vector computers, now parallel vector
- *High performance computing*: Solving problems via supercomputers + fast networks + visualization

## Multithreading

- In sequential computing, the components of a program are executed step-by-step to produce correct results.
- Parallelism is a condition wherein multiple tasks or distributed parts of a task run independently and simultaneously on multiple processors.
- A process is a program in execution with its own address space, memory, data stack, etc. under one operating system.
- The multiple threads execute within the same process and share the same context.

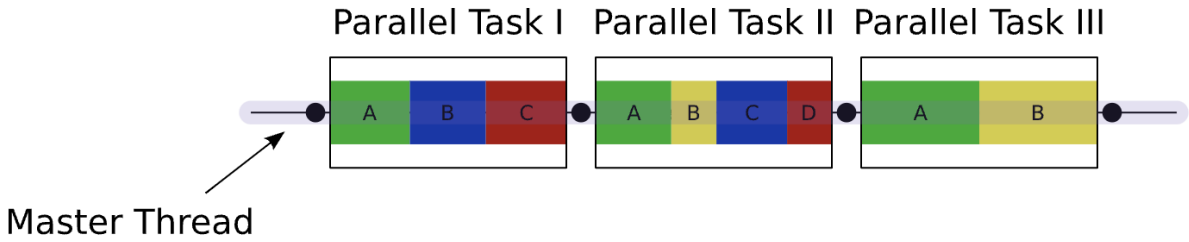
https://www.dataquest.io/blog/multithreading-in-python/

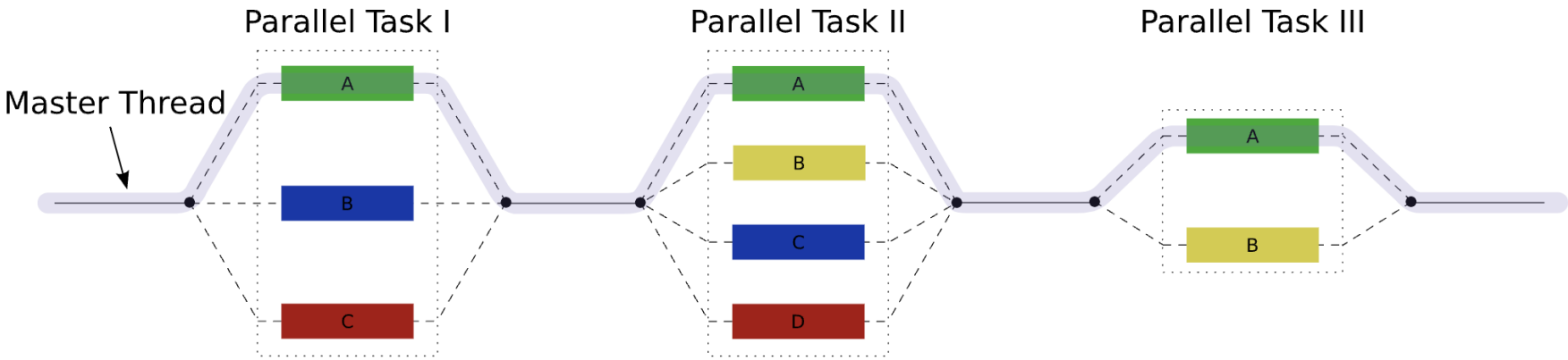
## Multithreading in Python

```python
import threading
import time

def calc_square(numbers):
    for n in numbers:
        print(f'\n{n} ^ 2 = {n*n}')
        time.sleep(0.1)

def calc_cube(numbers):
    for n in numbers:
        print(f'\n{n} ^ 3 = {n*n*n}')
        time.sleep(0.1)

numbers = [2, 3, 5, 8]

start = time.time()

square_thread = threading.Thread(target=calc_square, args=(numbers,))
cube_thread = threading.Thread(target=calc_cube, args=(numbers,))

square_thread.start()
cube_thread.start()

square_thread.join()
cube_thread.join()

end = time.time()

print('Execution Time: {}'.format(end-start))
```

https://www.dataquest.io/blog/multithreading-in-python/

## Fork–Join Model




https://en.wikipedia.org/wiki/Fork%E2%80%93join_model

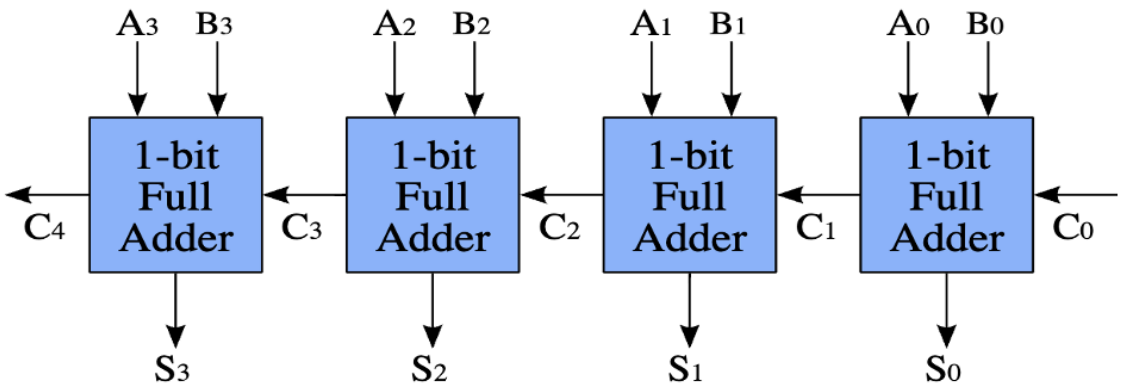
## Parallel Computing for Matrix Multiplication in Python

```python
import numpy as np
from multiprocessing import Pool

# Define the matrix multiplication function
def matrix_multiply(args):
  A, B = args
  return np.dot(A, B)

# Create two random matrices of size 1000x1000
A = np.random.rand(1000, 1000)
B = np.random.rand(1000, 1000)

# Split the matrices into 4 parts
A_parts = np.array_split(A, 4, axis=1)
B_parts = np.array_split(B, 4)

# Create a multiprocessing pool with 4 workers
pool = Pool(4)

# Map the matrix multiplication function to the 4 parts of the matrices
C_parts = pool.map(matrix_multiply,
  [(A_part, B_part) for A_part, B_part in zip(A_parts, B_parts)])

# Concatenate the parts of the result matrix
C = np.concatenate(C_parts, axis=1)

print(C)
```

https://www.geeksforgeeks.org/parallel-programming-with-numpy-and-scipy/

## Adder and Matrix Multiplication

```
P-MATRIX-MULTIPLY(A, B, C, n)
1   parallel for i = 1 to n                 // compute entries in each of n rows
2       parallel for j = 1 to n             // compute n entries in row i
3           for k = 1 to n
4               c_ij = c_ij + a_ik · b_kj
```




https://www.geeksforgeeks.org/parallel-programming-with-numpy-and-scipy/

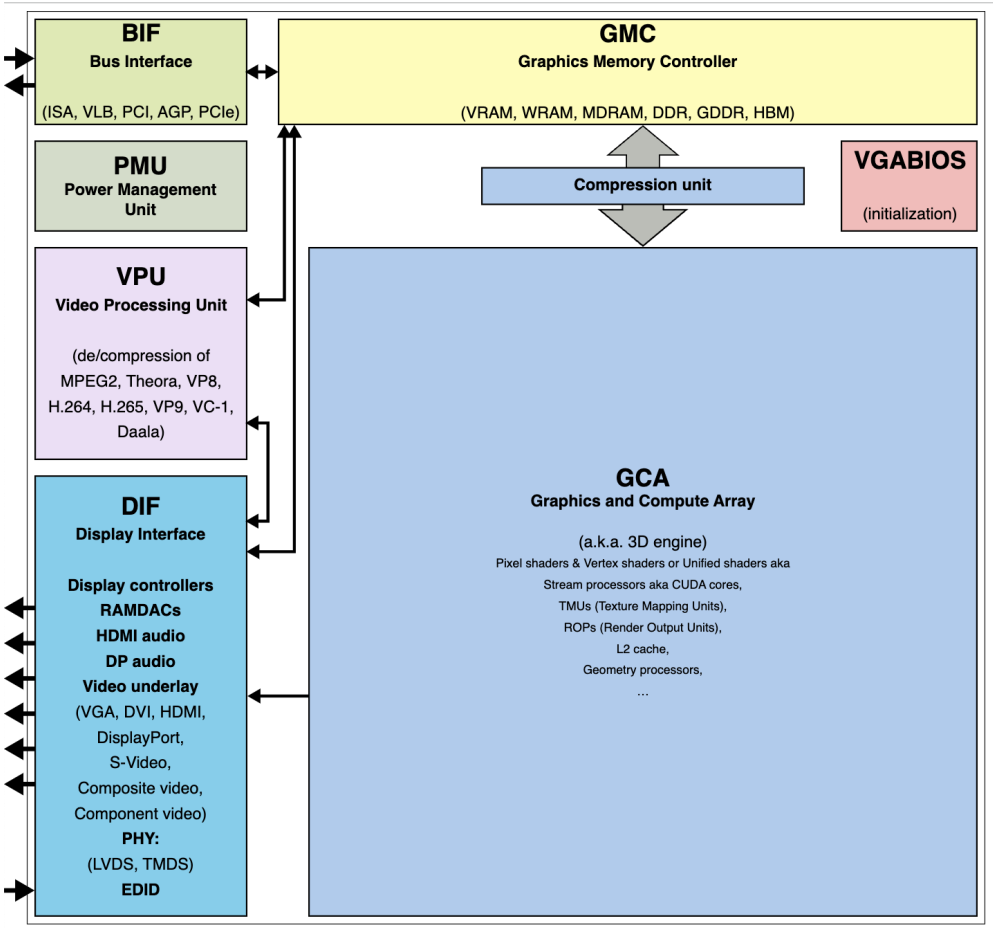
## GPU: Graphics Processing Unit

- A GPU is a specialized electronic circuit to accelerate the creation of images in a frame buffer for output to a display device.
- Modern GPUs are very efficient at manipulating computer graphics and image processing.
- In a personal computer, a GPU can be presented on a video card or embedded on the motherboard.

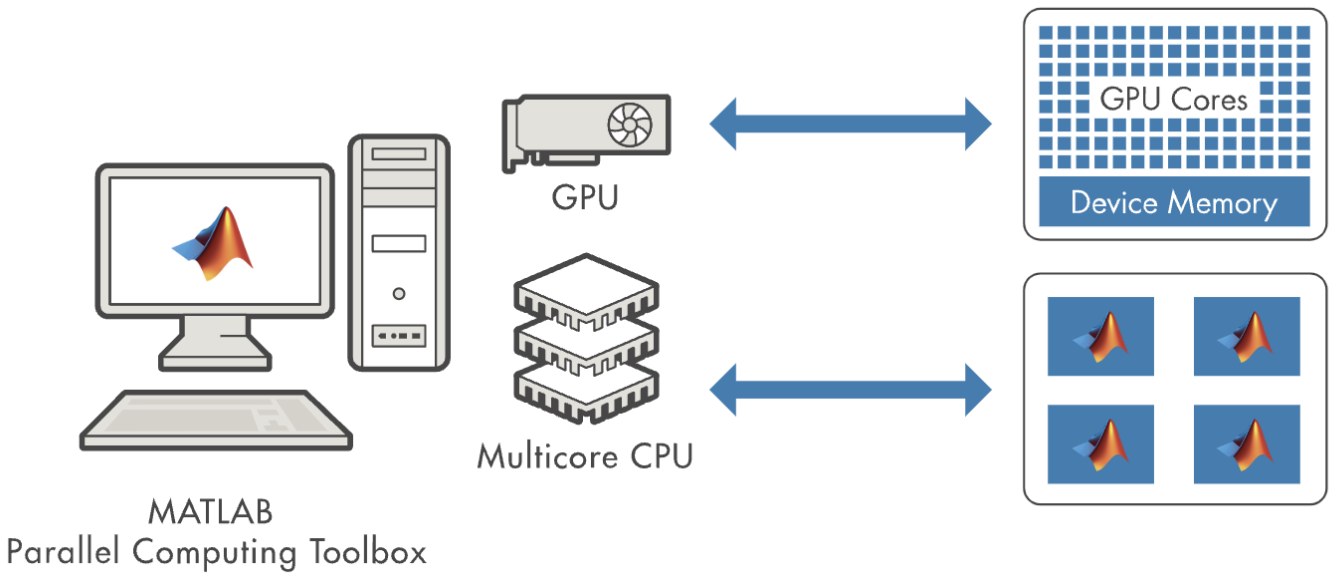
## GPU: Graphics Processing Unit

https://youtu.be/-P28LKWTzrI?si=-FFv9_0Vv1MWxgP_

## GPU: Graphics Processing Unit



## MATLAB Parallel Computing



## MATLAB Parallel Computing

- *Worker*: An independent MATLAB session that runs code distributed by the client.
- *Client*: MATLAB session as an interface that distributes jobs to workers.
- *Parfor*: A parallel computing function that distributes independent code segments to workers.
- *Random stream*: A pseudorandom number generator and the sequence of values it generates.
- *Reproducible computation*: A computation that can be exactly replicated even in the presence of random numbers.

## MATLAB Paralell Computing

- Parallel for-loop (parfor) for running task-parallel algorithms on multiple processors
- Support for CUDA-enabled NVIDIA GPUs
- Ability to run workers locally on a multicore desktop
- Computer cluster and grid support (with MATLAB Distributed Computing Server)
- Interactive and batch execution of parallel applications
- Distributed arrays and single program multiple data (SPMD) constructed for large dataset handling and data-parallel algorithms

## Google Colab

Google Colaboratory, or "Colab" allows us to write and execute Python codes in a browser with:

- Zero configuration required;
- Free access to GPUs;
- Easy sharing;
- With Google Colab, we harness the full power of popular Python libraries to analyze and visualize data.
- Colab notebooks execute code on Google's cloud servers, including GPUs, TPUs, and NPUs.
- All we need is a browser.

## Google Colab

Google Colab is used extensively in the machine learning community with applications including:

- Getting started with TensorFlow
- Experimenting with GPUs & TPUs
- Disseminating AI research
- Creating tutorials
- .....

## Google Colab

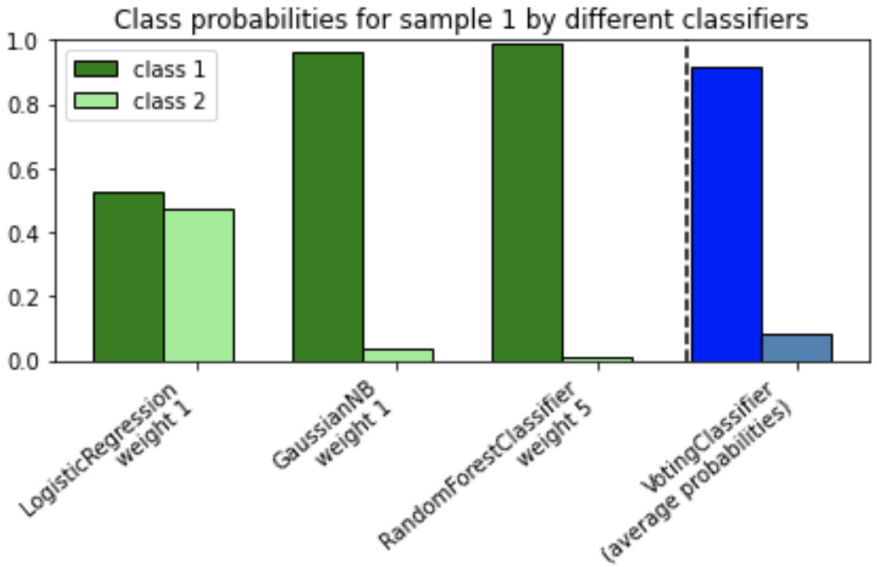
Untitled ☆

File Edit View Insert Runtime Tools Help Last saved at 10:41 PM

+ Code + Text

```
plt.show()
```

```
Automatically created module for IPython interactive environment
```



## Questions?

## Questions?

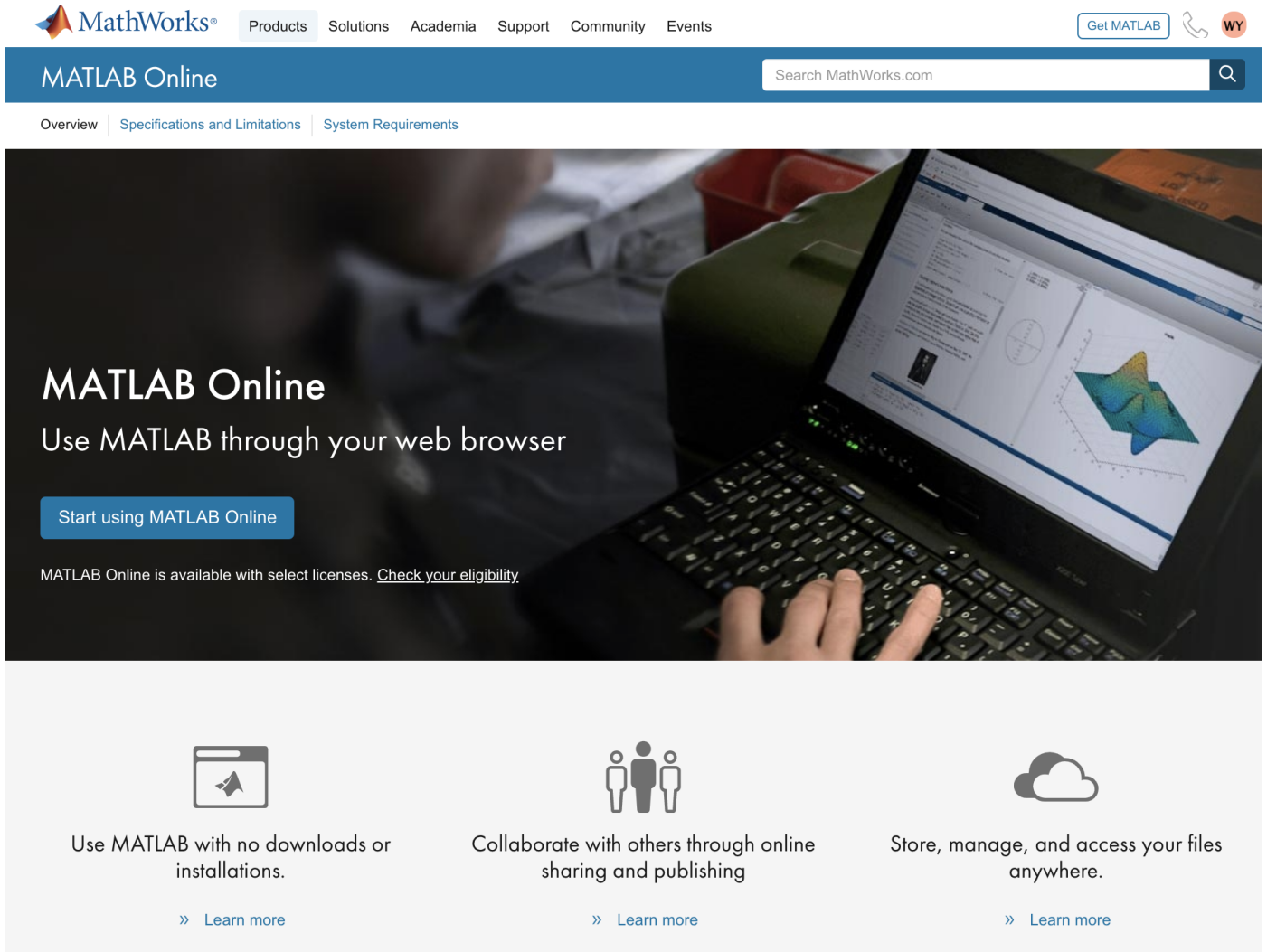
Parallelism is a condition wherein multiple tasks or distributed parts of a task run independently and simultaneously on

1. multiple processors
2. a single processor
3. a single core
4. none of the given options

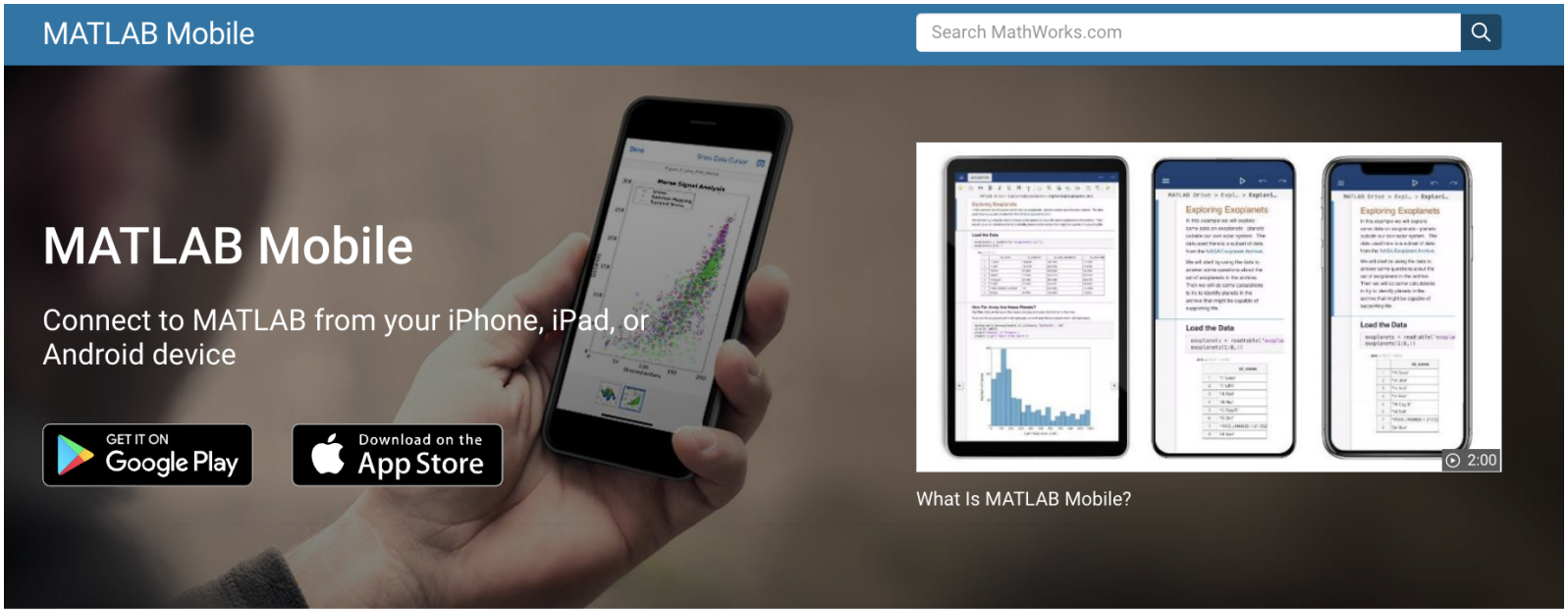
The right answer is:___

## Questions?

# MATLAB Mobile Computing for Robotics



## MATLAB Mobile




**Connect to MATLAB Mobile**

Connect to a MATLAB® session running on MathWorks Cloud.

» Connect to the cloud

**Acquire Sensor Data**

Acquire data from device sensors – like the accelerometer and GPS – and analyze the data in MATLAB.

» Learn more

**Capture Images, Video, and Audio**

Take pictures and record video and audio for further processing and analysis.

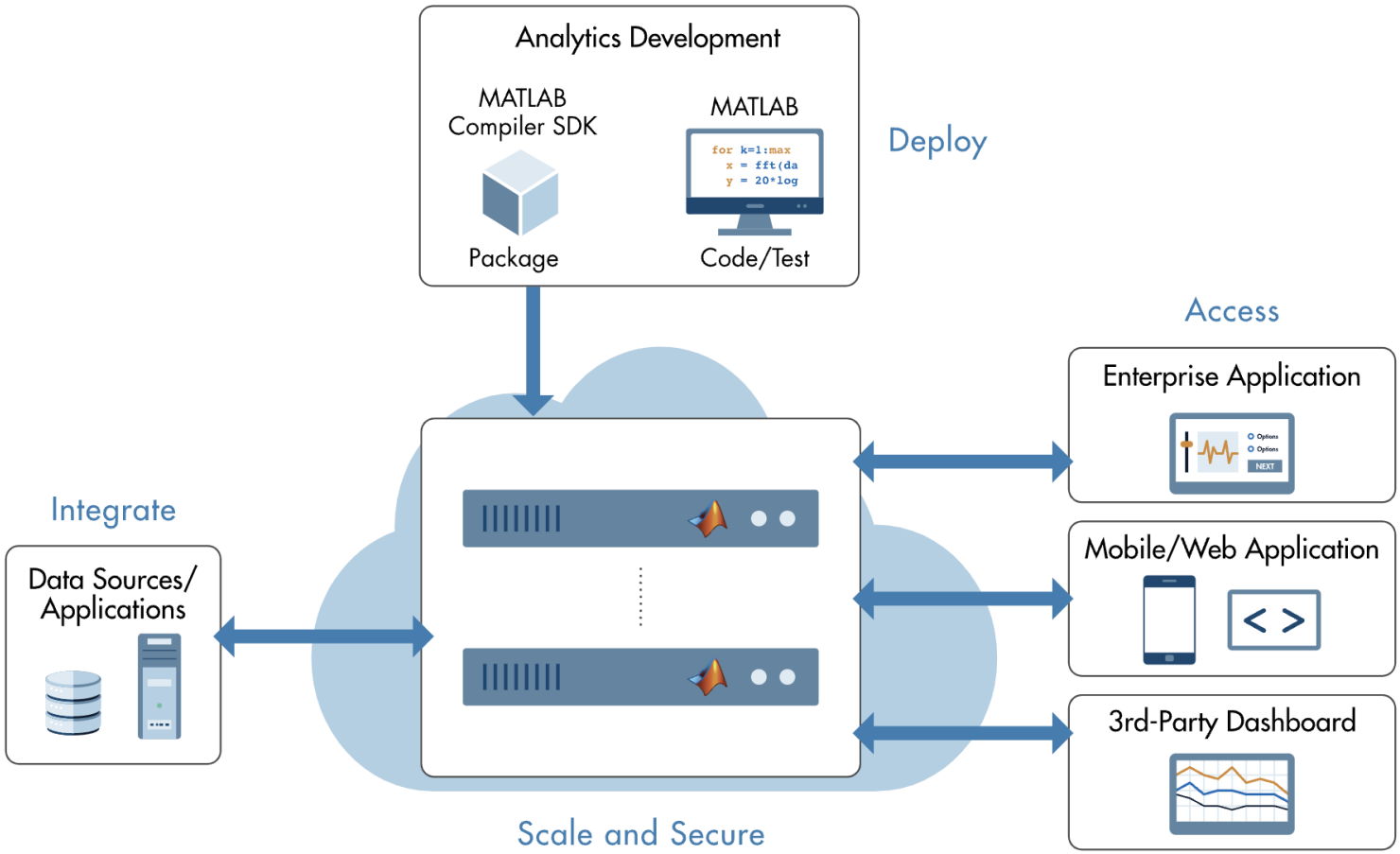
**Learn and Teach**

Learn and teach math, engineering, and other subjects right from your mobile device.

» Learn more

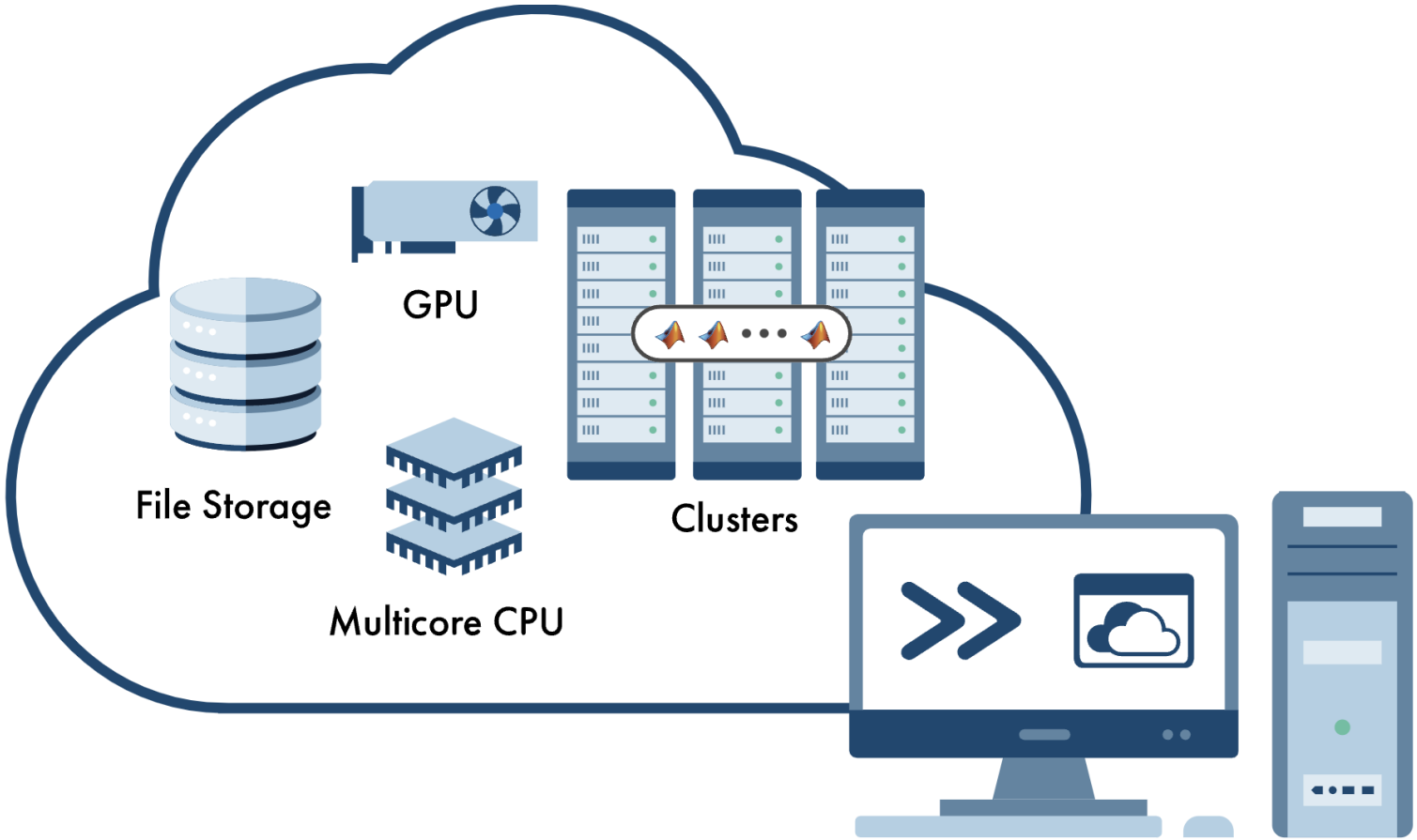
https://au.mathworks.com/products/matlab-mobile.html

## MATLAB Mobile




https://au.mathworks.com/solutions/cloud.html

## MATLAB Mobile




https://au.mathworks.com/solutions/cloud.html

## MATLAB Mobile

- **Connection**: Connect to a MATLAB session running on MathWorks Cloud.
- **Acquisition**: Acquire data from device sensors – like the accelerometer and GPS – and analyze the data in MATLAB.
- **Capturing**: Take pictures and record video / audio for further processing and analysis.
- **Teaching and Learning**: Learn and teach maths, engineering, and other subjects right from mobile device.

https://au.mathworks.com/products/matlab-mobile.html

## Acquire Data from Mobile Sensors

Acquire data from built-in sensors on mobile device and stream sensor data directly to the MathWorks Cloud. The data includes:

- Acceleration on 3 axes
- Angular velocity on 3 axes
- Magnetic field on 3 axes
- Orientation (azimuth, pitch, and roll)
- Position (latitude, longitude, altitude, horizontal accuracy, speed, and course)

https://au.mathworks.com/products/matlab-mobile.html

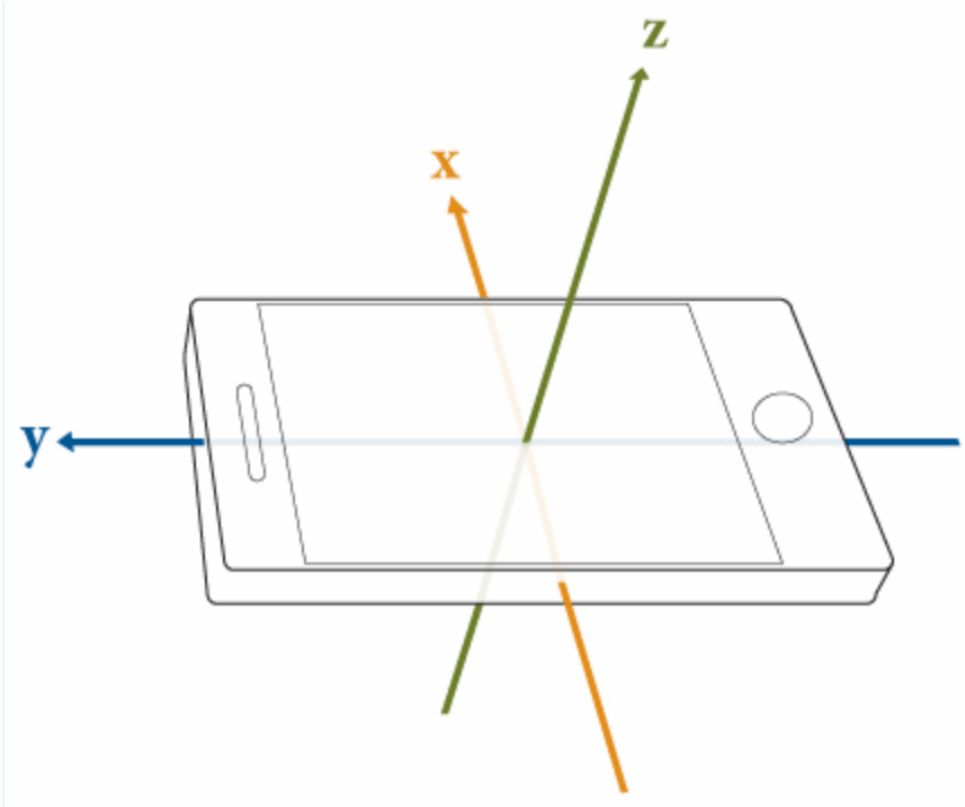
## MATLAB Mobile

- MATLAB Mobile sends all commands that were entered on the device to the Cloud for evaluation.
- Autocomplete in MATLAB Mobile makes typing easier.
- MATLAB Mobile displays thumbnails and larger previews when figures are created or updated with MATLAB commands.
- MATLAB Mobile deletes unwanted commands to improve scrolling performance in history.

https://au.mathworks.com/products/matlab-mobile.html

## MATLAB Mobile: Logging Accelerometer Data

This example shows how to manipulate and visualize data coming from a smartphone or tablet accelerometer.




https://au.mathworks.com/help/matlabmobile/ug/logging-accelerometer-data.html

## Questions?

## Book: Concise Robot Vision

# Concise Robot Vision

June 2025

## Questions?

## Learning Objectives

- Derive solutions for particular robotic vision and visual control tasks characterised by specifics of image data and deep learning algorithms.
- Examine opportunities of using robotic vision as a part of complex robotic systems and applications.