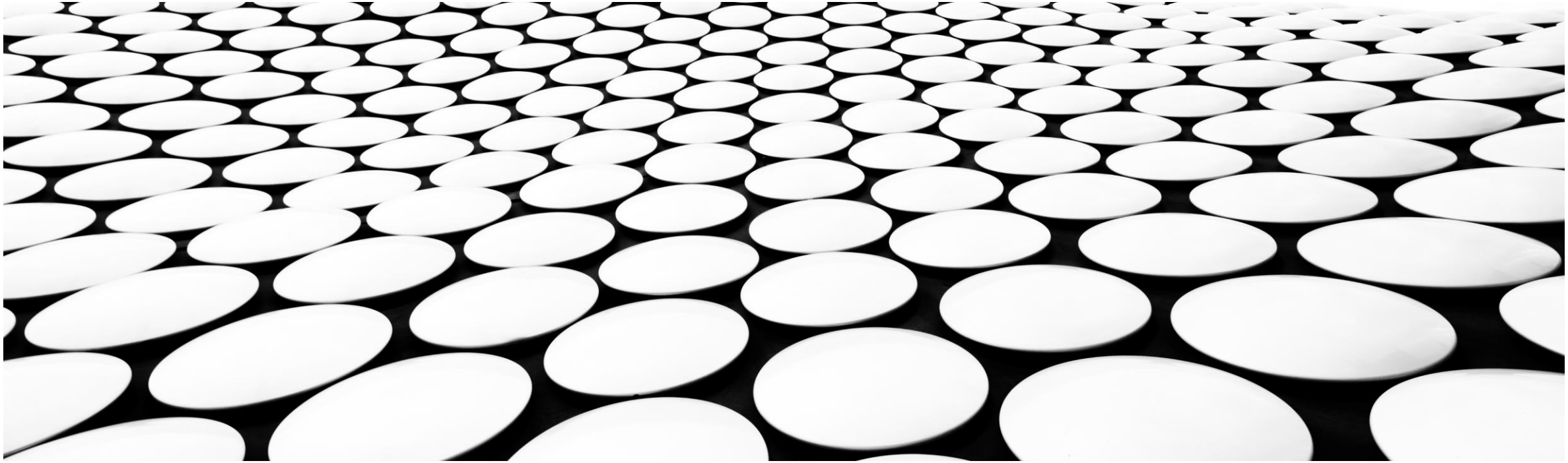
Introduction And Course Objectives

COEN-317: Distributed Systems

Robert Bruce
Department of Computer Science and Engineering
Santa Clara University



What is this class all about?

This is a project-based class.

- Students will work in teams of three or four students.
- Students will develop a distributed computing pipeline managed by a load-balancer queue.
- For assessment purposes, we will also have a midterm and final exam in this class.

In this course I will be discussing techniques for:

- Building distributed systems to achieve high performance, parallel processing.
- Increasing resiliency and security in distributed systems.

Programming environment

We will be using lots of open-source software in the class including:

- GNU/Linux (or OSX) and build tools.
- Postgres (relational database management software).
- Apache (web server).
- OpenCV (computer vision library).
- dLib (machine learning library).

I expect you to be familiar with (and preferably have experience with):

- Command line tools such as make, g++, gcc, vim, etc.
- Creating and modifying Makefiles.
- Relational databases.
- Programming languages: C/C++.
- POSIX Threads (pthreads).
- Inter-process communication (IPC): sockets, pipes, shared memory, and signals.
- Atomic process control: binary semaphores (mutex) and counting semaphores.

Who am I?

Relevant experience:

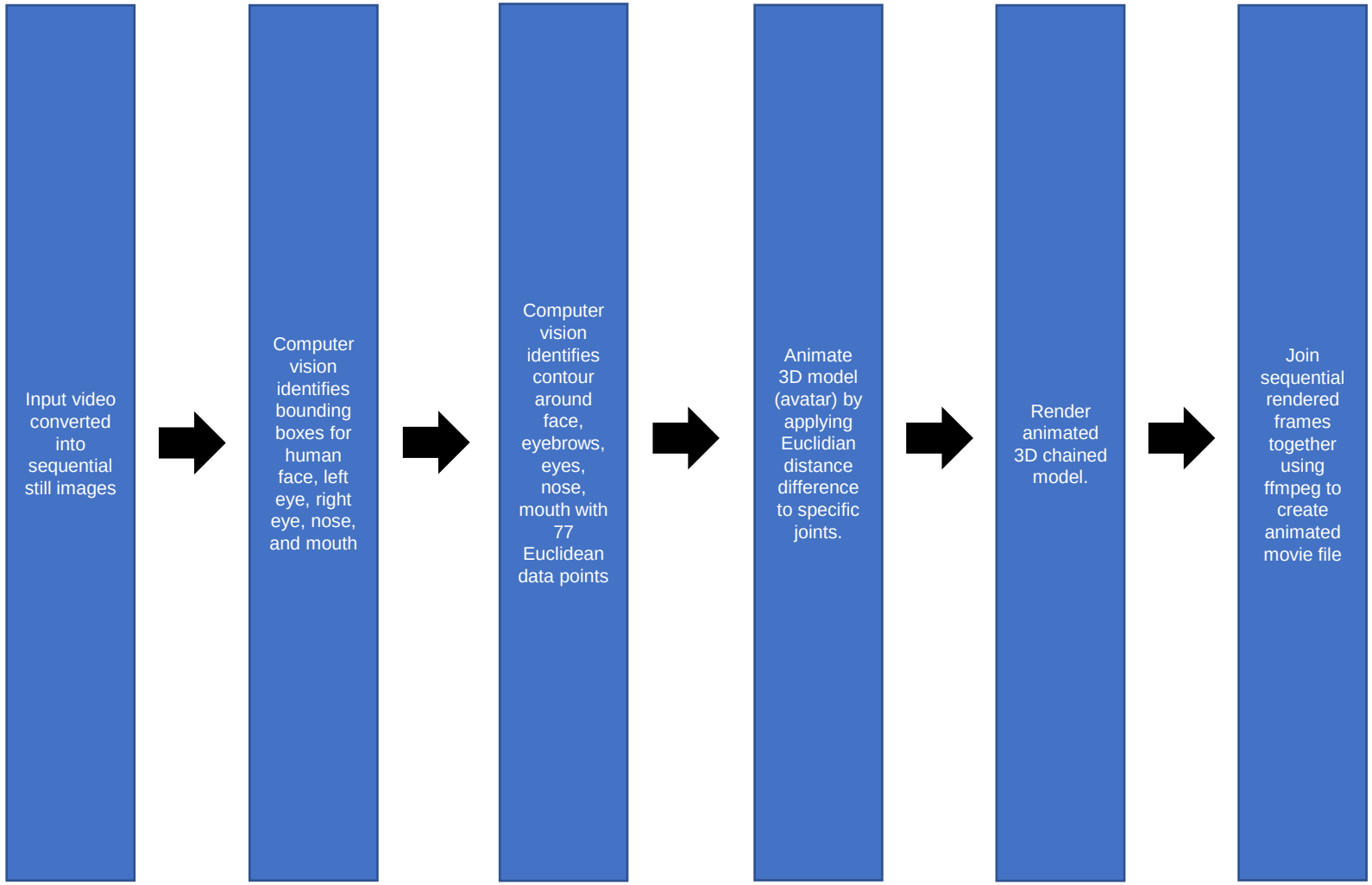
- Distributed Movie Making (seed-funded startup): Developed a scalable, automated, high performance, Linux-based 3D graphics and motion capture pipeline in C/C++ to generate 3D animated avatars from input selfie videos. Created a socket-based render queue and load balancer to distribute, manage, and monitor jobs.
- NASA Ames Research Center: Wrote software for capturing and transcoding stereoscopic video images in a cube-satellite (CubeSat). CubeSat was jettisoned into Low Earth Orbit (LEO) on January 2018.
- Industrial Light + Magic: Wrote high-speed I/O software for this top-100 supercomputing facility in the 1990's.

Practical example for Distributed Computing: a job queue

Challenge:

Creating a rendered, animated avatar from an input selfie video *as quickly as possible*.

Practical example for Distributed Computing: a job queue



Practical example for Distributed Computing: a job queue

Success in COEN-317

Learn how to divide a large project into a series of small tasks

- In my experience, this is a vital skill to develop!
- Breaking a large problem into tasks alleviates feelings of being overwhelmed.
- This process will also help you identify components that can be processed in parallel.

Manage your time well

- Pace yourself: If you sprint a marathon you'll be exhausted before the finish line.
- Binge working will most likely lead to unsatisfactory results. I have seen it with teams before.

Take advantage of my office hours

- Office hours is an opportunity to engage and open dialog with me. Stop by!
- Share your tech knowledge and career dreams with me!