COSC 414/519I: Computer Graphics

2023W2

Shan Du

Dr. Shan Du

- Ph.D. (The University of British Columbia Vancouver, 2009)
- Spin-off company in Computer Vision
- Associate Editor of IEEE Trans. on Circuits and Systems for Video Technology
- Area Chair and Session Chair of IEEE International Conference on Image Processing (ICIP)

Course Information

- **Instructor**: Dr. Shan Du FIP 324 shan.du@ubc.ca
- **Department**: Computer Science
- **Class Time**: Tuesday and Thursday 3:30PM – 5:00PM
- **Location**: ART 103
- **Office Hour**:
  - Monday 12:00PM – 1:00PM on Canvas
  - Friday 10:00AM – 11:00AM on Canvas
- **Course Website:**

  414: https://canvas.ubc.ca/courses/133338

  519I: https://canvas.ubc.ca/courses/133349

Course Information

- **Teaching Assistant**:
  - Tanmaya Karmarkar, tanmayak@mail.ubc.ca
  - Meghana Khuntia, megh12@student.ubc.ca

TA Office Hours: Please check Canvas

Course Information

- **Prerequisites:**

All of COSC 221, COSC 222 and one of MATH 221, APSC 179

Please note that students who lack the prerequisites should not register for this course and will receive a failing grade if they remain in it. Any exceptions must be brought to the attention of the instructor immediately.

- **Prior Knowledge:**
  - Can program in JavaScript or learn to do so on their own time within the first two weeks of class
  - Have strong math skills, including working with vector and matrix operations

Course Information

- **Outline:**

Topics include raster graphics, graphics architectures, application programming interface, interactive graphics, two- and three-dimensional computer graphics primitives and attributes, transformations, viewing, animation, hidden surface removal, color and shading, and texture. The programming assignments will be designed to use many of the graphics features discussed in class.

Course Information

- **Suggested Textbooks:** Interactive Computer Graphics: A Top-Down Approach with WebGL Seventh/Eighth Edition. Edward Angel and Dave Shreiner. Addison-Wesley (2015/2020).

- **Other References:**
  - WebGL Programming Guide: Interactive 3D Graphics Programming with WebGL. Kouichi Matsuda and Rodger Lea. Addison-Wesley (2013).
  - Computer Graphics from Scratch. Gabriel Gambetta. No Starch Press (2021).
  - Course website on Canvas
  - Lecture Notes (available electronically).

Course Information

Programming Language and Libraries:

We will be using HTML5, JavaScript and WebGL.

Course Information

Learning Outcomes:

Upon successful completion of this course, students will be able to:

- Interpret the core concepts and mathematical foundations of computer graphics
- Understand fundamental computer graphics algorithms and data structures
- Apply basic mathematics and concepts in the development of graphics applications
- Design and develop interactive 2D and 3D programs using WebGL

Course Information

Grading:

Programming Assignments: 40%

Quizzes: 10%

Midterm Exam: 15%

Final Exam: 35%

In-class Participation Bonus: can be used for adding to the Assignments portion (but the maximum mark of Assignments is 40%)

Regulations

- Students MUST achieve a passing grade in overall in order to pass the course.
- Quizzes are closed-book Canvas tests. No course materials, cell phones, or other electronic devices are allowed during the test time. Computers are only used for the tests, not for checking, discussing or searching solutions. Calculator is allowed.

Regulations

- Exams are paper-based, closed-book tests with cheat sheets allowed. You may bring a maximum of 16 (one side) or 8 (both sides) pages (8.5 by 11 inches, photo-reduction allowed) and any scientific calculator you wish (but NO other electronic equipment such as a laptop, portable computer, printing devices, communication devices etc.)

Regulations

- The course website contains the most up-to-date information and important dates for main events such as assignments due dates and tests. Students need to check regularly.
- Attendance is mandatory in lectures.
- **The use of artificial intelligence (AI) assistance for any assessed portions of this course is not permitted.**
- If you feel any mark was unfair or incorrectly recorded, ensure that I am aware of the problem before the last week of classes.

Regulations

- **Late Penalty:** Late assignments will be deducted 10% per day up to 3 days (after which they will receive 0 marks).

- **Plagiarism:** is forbidden (0 mark).

Regulations

Missed Tests:

- There will be no deferred quizzes/midterm exam. If you miss any of the quizzes with a valid excuse (according the UBC Okanagan's policy on excused absences from examinations), duly documented and reported to the instructor within one week of the exam, then the weight of that quiz will be transferred to other quizzes.

Regulations

Missed Tests:

- For midterm exam, the grade will be deferred to the final exam. No midterm make-up or retake will be done.
- For final exam, the student may retake a make-up final exam with the permission of the Dean’s office. Note that a make-up exam may have a question format different from the original regular exam.

Course Contents and Schedule

Week	Contents
1	Introduction
	Image Formation and Graphics Pipeline
2	Graphics Programming and API 1
	Graphics Programming and API 2
3	Graphics Programming and API 3
	Interaction and Animation, Quiz 1
4	Geometric Objects
	Coordinate Systems and Frames

Course Contents and Schedule

5	**Transformations**
	Transformations in WebGL
6	Viewing
	Projections 1, Quiz 2
7	Midterm Break
8	Midterm Exam
	Projections 2
9	Hidden Surface Removal
	Light Sources

Course Contents and Schedule

10	Reflection
	Normal Vectors and Shading 1, Quiz 3
11	Shading 2
	Lighting Calculation
12	Buffer and Texture Mapping 1
	Texture Mapping 2
13	Texture Mapping 3
	Texture Mapping 4, Quiz 4
14	Complex Objects
	Final Exam Review

Acknowledgment

- The course materials are based on:
  - The recommended textbooks/references
  - Some online resources and similar courses offered in other top-ranked universities
  - Some published papers and datasets

Introduction

- The term *Computer Graphics* describes any use of computers to create images.

Graphics Areas

- Modeling: deals with the mathematical specification of shape and appearance properties in a way that can be stored in the computer.
  - For example, a coffee mug might be described as a set of ordered 3D points along with some interpolation rule to connect the points and a reflection model that describes how light interacts with the mug.

Graphics Areas

- Rendering: a term inherited from art and deals with the creation of shaded images from 3D computer models.
- Animation: a technique to create an illusion of motion through sequences of images.

Graphics Areas

- Visualization: attempts to give users insight via visual display.
- 3D scanning: uses range-finding technology to create measured 3D models.

Major Applications

- Video games: increasingly use sophisticated 3D models and rendering algorithms.
- Cartoon: often rendered directly from 3D models.
- Film special effects: use almost all types of computer graphics technology.
- CAD/CAM: computer-aided design and computer-aided manufacturing.

Major Applications

- Simulation: can be thought of as accurate video gaming. For example, a flight simulator uses sophisticated 3D graphics to simulate the experience of flying an airplane.
- Medical imaging: create meaningful images of scanned patient data.
- Information visualization: create images of data that do not necessarily have a “natural” visual depiction.

Graphics APIs

- Application programming interface (API) provides an application program access to graphics libraries, such as drawing an image in a window.
- Most APIs have a user-interface toolkit of some kind that uses callbacks.
- Callbacks refer to the process of using function pointers or virtual functions to pass a reference to a function.

Graphics APIs

- Java packages
- Direct3D and OpenGL
- WebGL

WebGL

- WebGL is a technology that enables drawing, displaying and interacting with sophisticated interactive three-dimensional computer graphics (“3D graphics”) from within web browsers.
  - Combined with HTML5 and JavaScript
  - Default built-in of browsers

WebGL

- Traditionally, a stand-alone application using
  - C/C++ or other programming language
  - Computer graphics library, OpenGL or Direct 3D
  - Device dependent
- WebGL, a standard web page using
  - HTML5
  - JavaScript
  - Without installing special plug-ins or libraries (browser’s default built-in)
  - Device independent

Advantages of Using WebGL

- You can start developing 3D graphics applications using only a text editor and browser.
- You can easily publish the 3D graphics applications using standard web technologies, making them available to your friends or other web users.
- You can leverage the full functionality of the browser.
- A lot of material of WebGL is available.

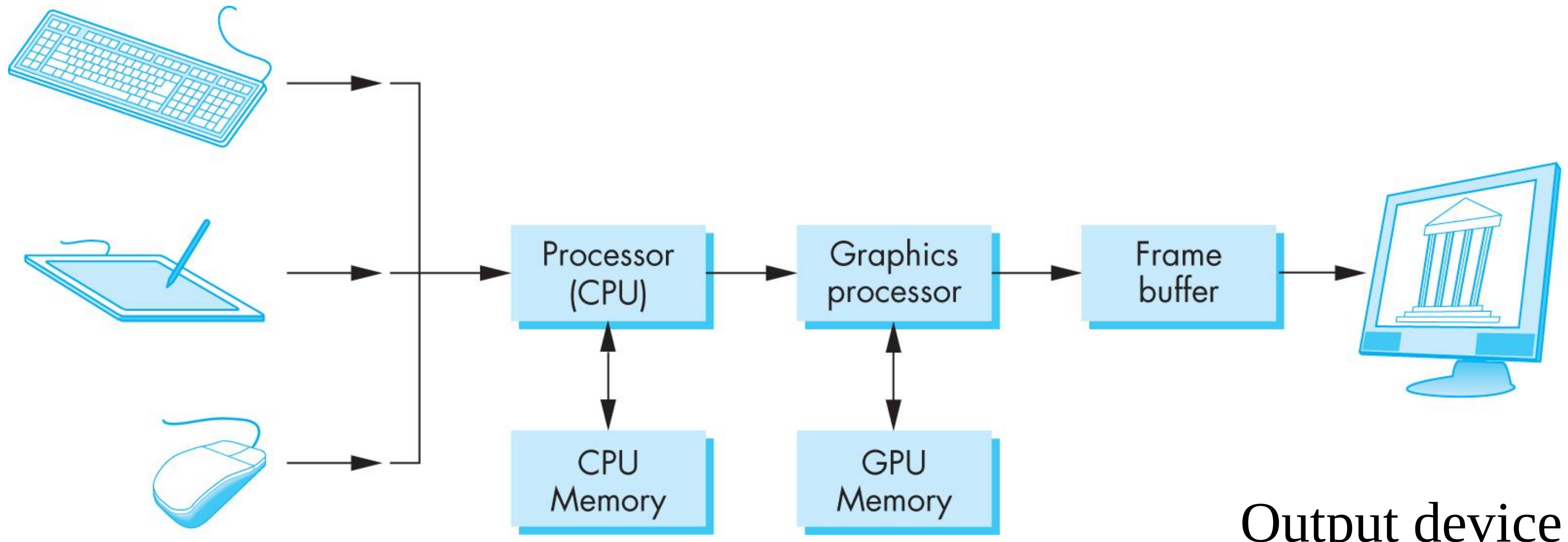
Sample WebGL Applications

- Published by Google:

https://webglsamples.org/

https://github.com/WebGLSamples/WebGLSamples.github.io

Basic Graphics System



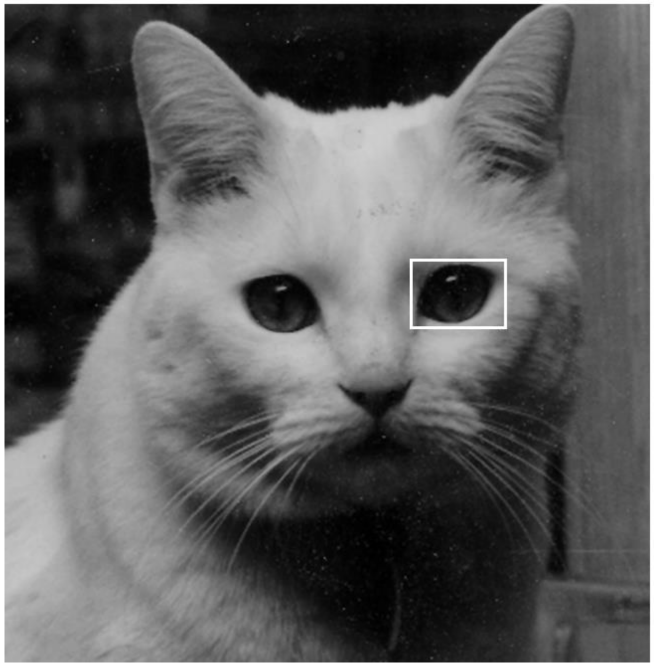
Output device

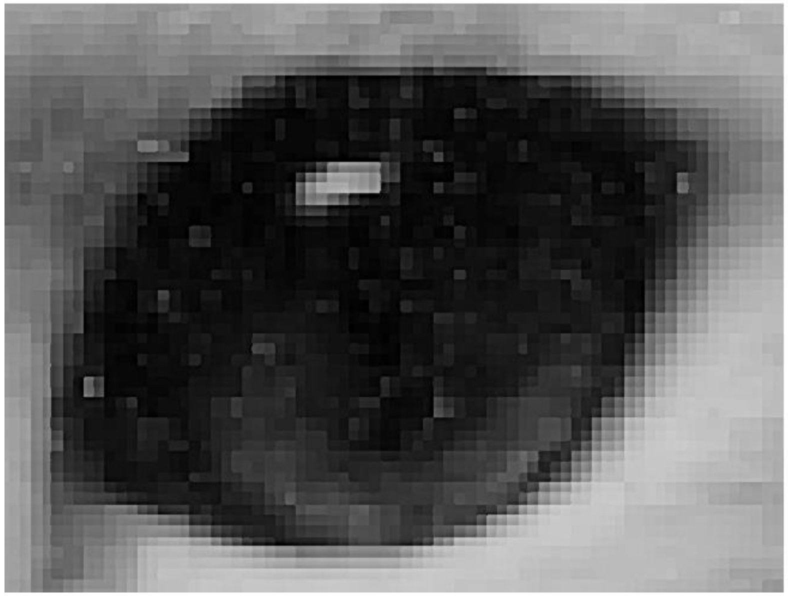
Input devices

Image formed in frame buffer

Raster Graphics

- Image produced as an array (*the raster*) of picture elements (*pixels*) in the framebuffer





Framebuffer

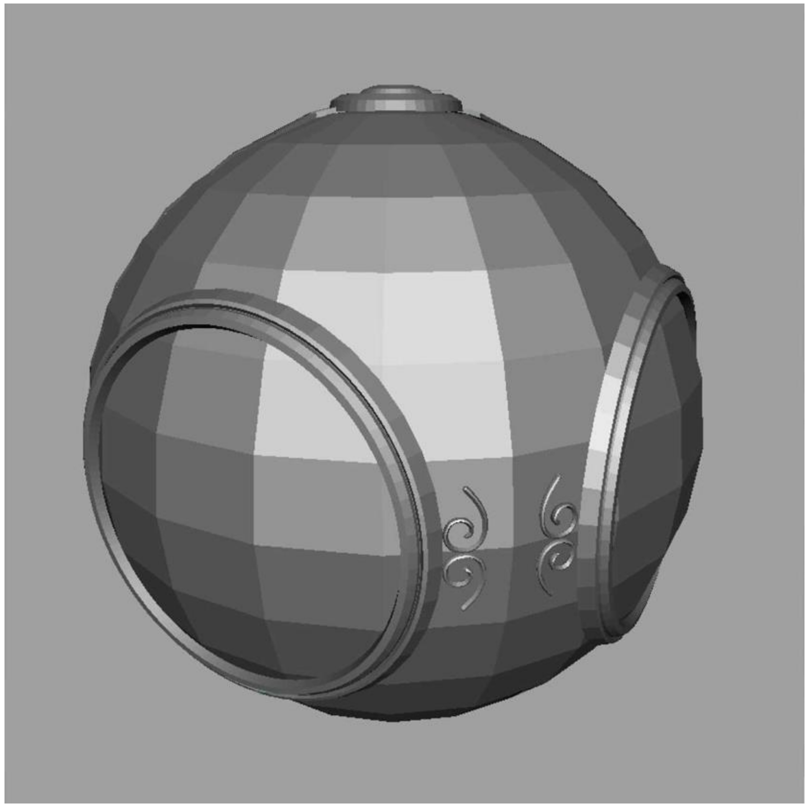
- The framebuffer can be viewed as the core element of a graphics system.
- Its resolution – the number of pixels in the framebuffer – determines the detail that you can see in the image.
- The depth, or precision, of the framebuffer, defined as the number of bits that are used for each pixel, determines properties such as how many colors can be represented on a given system.

Framebuffer

- 1-bit: two colors
- 8-bit: 2^8=256 colors
- Full-color system (true color, or RGB color system): 8-bit for R/G/B, totally 24-bit
- High dynamic range (HDR) system: 12-bit = $2^{12} = 4096$ colors

Raster Graphics

- Allows us to go from lines and wire frame images to filled polygons



Rasterization

- Take specifications of graphical primitives (lines, circles, and polygons) generated by application programs and to assign values to the pixels in the framebuffer that best represent these entities.
- For example, a triangle is specified by its three vertices, the graphics system must generate a set of pixels that appear as line segments to the viewer.

Rasterization

- The conversion of geometric entities to pixel colors and locations in the framebuffer is known as rasterization or scan conversion.

CPU and GPU

- In a simple system, there may be only one processor, the central processing unit (CPU), which must perform both the normal processing and the graphical processing.
- In early graphics systems, the framebuffer was part of the standard memory that could be directly addressed by the CPU.
- Today, virtually all graphics systems are characterized by special-purpose graphics processing units (GPUs) to carry out specific graphics functions.

Output

- In a raster system, the graphics system takes pixels from the framebuffer and displays them as points on the surface of the display in one or two fundamental ways.
- In a non-interlaced system, the pixels are displayed row by row, or scan line by scan line, at the refresh rate.
- In an interlaced display, odd rows and even rows are refreshed alternately.

Output

- Interlaced displays are used in commercial television.
- In an interlaced display operating at 60Hz, the screen is redrawn in its entirely only 30 times per second, although the visual system is tricked into thinking the refresh rate is 60Hz rather than 30Hz.

3D Display

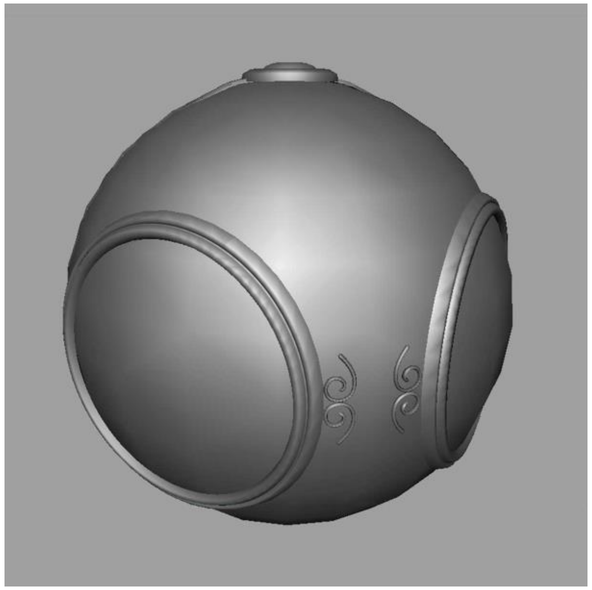
- Stereo (3D) television displays use alternate refresh cycles to switch the display between an image for the left eye and an image for the right eye. The viewer wears special glasses that are coupled to the refresh cycle.
- 3D movie projectors produce images with different polarizations. The viewer wears polarized glasses so that each eye sees only one of the two projected images.

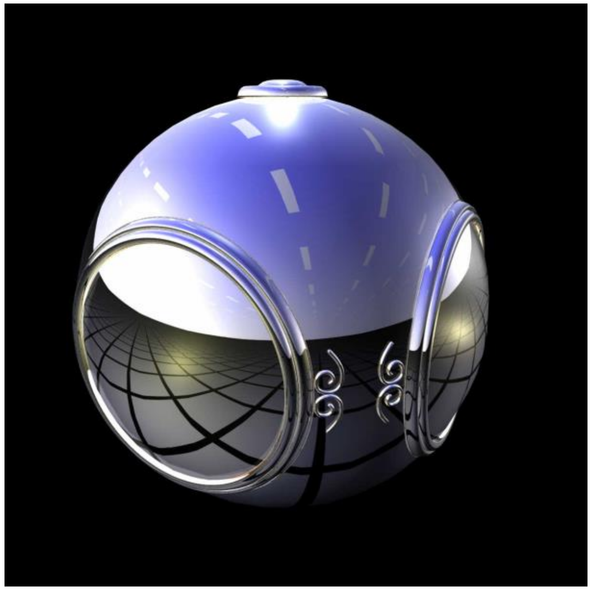
Input

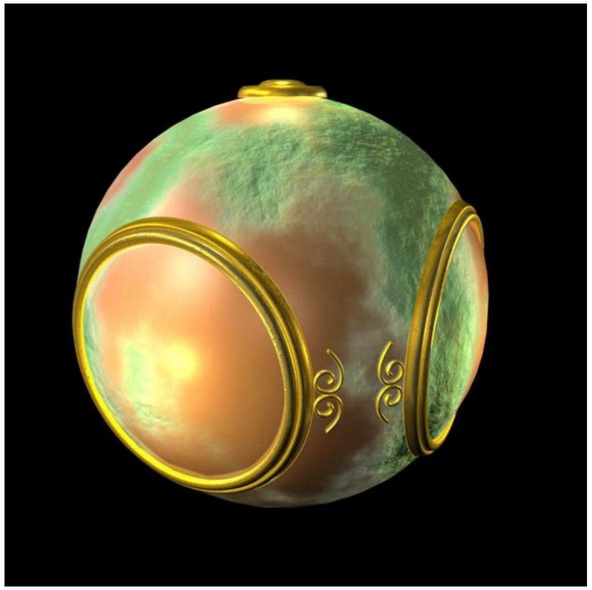
- Keyboard
- Mouse
- Joystick
- Data Tablet
- ……

Computer Graphics: 1980-1990

Realism comes to computer graphics


smooth shading


environment mapping


bump mapping

Computer Graphics: 1990-2000

- OpenGL API
- Completely computer-generated feature-length movies (Toy Story) are successful
- New hardware capabilities
  - Texture mapping
  - Blending
  - Accumulation, stencil buffers

Computer Graphics: 2000-2010

- Photorealism
- Graphics cards for PCs dominate market
  - Nvidia, ATI
- Game boxes and game players determine direction of market
- Computer graphics routine in movie industry: Maya, Lightwave
- Programmable pipelines
- New display technologies

Computer Graphics: 2011-

- Graphics is now ubiquitous
  - Cell phones
  - Embedded
- OpenGL ES and WebGL
- Alternate and Enhanced Reality
- 3D Movies and TV