CSC251
Basics of Computer Graphics
Module: Introduction

P. J. Narayanan

Spring 2019

What is Computer Graphics?

- Techniques and tools to generate realistic images on the computer
- How?
  - Create **representations** and “models” of the world
  - Create **algorithms** to produce ultra-realistic images
  - Do these **fast**

... as a Coomputational Process

- To generate realistic images on the computer
- How?
  - Create **representations** and “models” of the world
  - Create **algorithms** to produce ultra-realistic images
  - Do these **fast**
- What is the **Computational Process**?

  Abstract – Represent – Process – Reproduce
- Computer Revolution: Applying this successfully to different application areas!

Digitial or Computer Revolution

- Changed the world greatly in the past 20-30 years!
- How? "**Digitize**" different things/concepts/ideas/...
  – Digital preseravation, replication, etc., are very cheap
- Initially: Ease tediums or difficulty of activities
  – Aircraft design, payroll, Efficient book-keeping, etc.
- Later: Improve and transform the process
  – Electronic accounts to networked banks to online banking to virtual money to ...
- Enablers: Digital Representation, Efficient Processing and Manipulation, Quick Communication
- And ... **reversing** the digitization process

Some Computational Processes

- Music: Digitize using microphones and analog-to-digital conversion, process to remove noise, store/transmit as MP3 files, playback using D-to-A and speakers
  – Similarly, Video, Skype, etc.
- Weather prediction: Capture parameters from locations, apply metereological models, process at different levels of detail, predict
  – Drug design, molecular dynamics, more science
- Computer games: World and its rules set by designer, some aspects controlled by players, interaction with objects according to rules, show results to players
  – Several simulations, Virtual Reality, etc.

What's this?

Producing Realistic Images

- Represent physical world using basic “primitives”
  - Basic geometric shapes and objects
  - Efficiency vs utility: Use simple and useful primitives
  - Break up complex scenes into available components
  - Efficiency: **Smallness in size** and **ease of operation**

- Process of image generation from the representation
  - Ape the best that we know: Human eyes
  - (Digital) Cameras approximate the eye in our world
  - Pin-hole camera model approximates the eye conceptually
  - Mathematics of pin-hole cameras known
  - Apply pin-hole camera computationally

- Transform a primitive to a camera image correctly
  - Apply pin-hole camera model on the primitive
  - A series of computations to map primitive to image
  - Paint the picture based on physical properties of objects

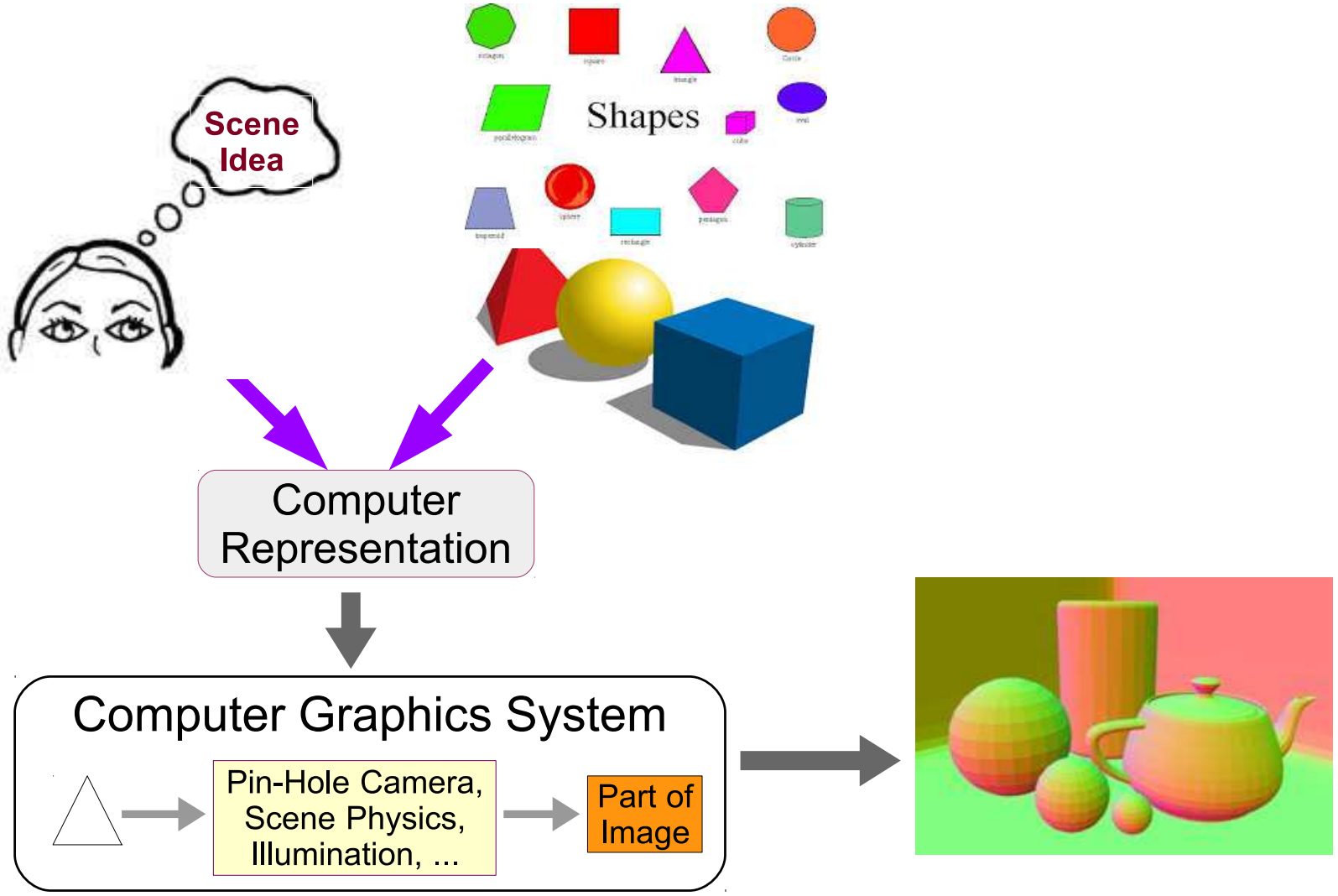
- Apply the same to the scene consisting of primitives
  - Evaluate how multiple primitives interact or interfere
  - Paint the physically correct picture of the whole scene

- Do all this efficiently
  - Millions of primitives. High resolution images
  - Complex objects with fine structure and properties
  - Update image fast for application. Real-time for games!

Some Glimpses through Videos

Show videos

Graphics Process



Computer Graphics: Components

- **Mathematics/Geometry:**
  - Geometry: Euclidean and Projective
  - Linear Algebra, Matrices, etc.
- **Physics:**
  - Illumination, shadows, ...
  - Mechanics, dynamics, etc., for the domain
- **Computer Science:**
  - Data structures: Scene graph, KD-Trees, BVH Trees, etc.
  - Algorithms: Line drawing, visibility determination, etc.
  - Efficiency: Saving a few cycles from everything
- **Creativity:** Lots and lots of it!!

Some Questions and Concerns

- What geometric primitives to use?
  – Geometric shapes exist. Economy & Utility important
- How do we represent them on the computer?
- How do we create representations of real-world objects?
- How do we manipulate the computer-resident world?
  – Change shape, size, appearance, etc.
- How do we produce images or views of scenes?
  – Realism is important
- How do we do all these efficiently and quickly?

Application Areas

- User interfaces
- Computer aided design (Civil/Mech/VLSI)
- Visualization of scientific & engineering data
- Art
- Virtual Reality
- Entertainment: Great computer games!
- Special effects in movies. **Whole movies themselves!!**
- ...

Quick History

- Whirlwind Computer (1950) from MIT had computer driven CRTs for output.
- SAGE air-defense system (mid 50s) had CRT, lightpen for target identification.
- Ivan Sutherland's Sketchpad (1963): Early interactive graphics system.
- CAD/CAM industry saw the potential of computer graphics in drafting and drawing.
- GE's DAC system (1964), Digitek system, etc.
- Systems were prohibitively expensive and difficult to use.

- Special display processors or image generators were used for high-end graphics.
- Workstations by Silicon Graphics: early eighties.
- Graphics was expensive, escoteric, and hence rare!
- **A parallel:** Computing became "popular" only after mass-produced personal computers became a reality in mid 80s. Before that, bulky, expensive, and rare devices.
- **Circle of Computing Revolution:** *More users* lead to *greater revenues/returns* which affords *more research* which result in *better/cheaper computers* which in turn bring *yet more users*. And this continues!!

Popular Graphics

- Graphics became "popular" only after mass-produced *Graphics Processing Units* (**GPU**s) or *graphics accelerators* came into existence.
- Graphics Accelerators: on board hardware to speed up graphics computations.
- Accelerators were expensive until end nineties!
- Very high end performance is available economically today. Getting part of the CPU chip these days.
- **Computer Games** provide the fuel for fast growth

Graphics Programming

- Device dependent graphics in early days.
- 3D Core Graphics system was specified in SIGGRAPH 77. (Special Interest Group on Graphics)
- GKS (Graphics Kernel System): 2D standard. ANSI standard in 1985.
- GKS-3D: 1988.
- PHIGS: Programmer's Hierarchical Interactive Graphics System. (ANSI 1988)

- **OpenGL**: current ANSI standard.
  - Evolved from SGI's GL (graphics library).
  - Window system independent programming.
  - GLUT (utility toolkit) for the rest.
  - Popular. Many accelerators support it.
- DirectDraw/Direct3D: Microsoft's attempt at it!
- **WebGL:** OpenGL to be used for web programming that is now gaining popularity
- **OpenGL ES:** Slightly reduced version for mobile devices, which will be the prime computing platform
- Desirable: High level toolkits.

Course Content

- 2D & 3D Graphics: Concepts, Mathematics, Hierarchical Modelling, Algorithms. Practice in OpenGL.
- Representation: Lines & Curves, Surfaces, Solids.
- Drawing algorithms: Primitives, visibility, efficienty
- Lighting and Shading: Simluating the physics of image generation
- Ray Tracing: If we get time

Background Required

- Good programming skills in C/C++.
- Geometry: Points, vectors, matrices, transformations, etc.
- Data structures.
- Java for Web or Mobile graphics
- **Good imagination. Ability to visualize in 3D**

About This Semester

- Dr Avinash Sharma will be the main instructor
- Parikshit Sakurikar will be a co-instructor
- This is a 3-credit, half-course. Lots of programming work
- We target for 15 lectures, 2 exams
- Saturday classes: lecture, tutorial, or nothing! Start with a tutorial on OpenGL3.0 at **10AM on January 5th, 2019**
- You will get instructions today about it. **Be on Moodle!!**
- An assignment on a 2D game will be given in a week
- We want to complete the course in the first week of April
- Will need an out-of-turn “final” exam early in April

Text Books and Reference

- **Computer Graphics with OpenGL** by Hearn and Baker, Third edition. Indian Edition available.

Additional books:

- Computer Graphics: Principles & Practice by Foley, van Dam, Feiner, Hughes. Indian Edition available.
- Interactive Computer Graphics: A Top-Down Approach Using OpenGL, Fifth edition by Edward Angel.
- OpenGL Programming Guide by Neider, et. al.

Course Management

- Homework assignments, Programming assignments, lab test, mid-term test, final exam
- Weightages of different components:
  50-60% for the two exams.
  30-40% for programming assignments
  10% for the rest (Written assignments, etc.)
- This course involves a lot of fun programming! Enjoy it!
- Several students do much more in assignments than asked for. Let your creative juices flow!!

About This Semester

- Dr Avinash Sharma will be the main instructor
- Parikshit Sakurikar will be a co-instructor
- This is a 3-credit, half-course. Lots of programming work
- We target for 15 lectures, 2 exams
- Saturday classes: lecture, tutorial, or nothing! Start with a tutorial on OpenGL3.0 at **10AM on January 5th, 2019**
- You will get instructions today about it. **Be on Moodle!!**
- An assignment on a 2D game will be given in a week
- We want to complete the course in the first week of April
- Will need an out-of-turn “final” exam early in April