CMT107 Visual Computing

I.1 Introduction

Xianfang Sun

School of Computer Science & Informatics
Cardiff University

Overview

- Module Logistics
- Introduction to Visual Computing
- Applications

Prerequisites

- Mathematics
  - Basic Linear Algebra
  - Trigonometry
- Programming
  - Basic Data structures/Programming knowledge
  - Language: Java
  - Graphics API: OpenGL (No prior knowledge required)

Assessment

- Coursework: (30%)
  - Hand out: Week 5
  - Hand in: Week 10
- Written Examination: (70%)
  - Duration: 2 hours

Learning Outcomes

➢ Knowledge / Understanding

- Understand the concepts, techniques and underpinning technologies associated with Visual Computing.
- Critically analyse the present capabilities and limitations of Visual Computing algorithms and techniques.
- Demonstrate an understanding of the present state-of-the-art associated with specific aspects of Visual Computing.
- Design and implement simple algorithms to exercise and test elements of Visual Computing.
- Demonstrate an understanding of the underlying mathematical techniques.
- Understand the computational effort required to perform operations associated with various algorithms.

➢ Skills

- Programming of simple visual computing algorithms, including data handling.
- Critical evaluation of the claims associated with new algorithms and methods.
- Understanding of the computational burdens associated with different processing techniques and be able to select appropriate methods depending on the intended application and context.

Syllabus

- Introduction to Visual Computing
  - Concepts and Applications
  - Mathematics Review
- Computer Graphics:
  - Graphics systems
  - Graphics Programming and API
  - Transformations
  - Lighting and Shading
  - Texture mapping
  - Ray Tracing
- Geometric Modelling
  - Curves and Surfaces
  - Hierarchical Modelling
  - Geometric Operations
  - Boundary Representation (B-rep)
  - Mesh Representation
  - Constructive Solid Geometry (CSG)
- Image Processing
  - Image Representation
  - Image Filtering and Restoration
  - Mathematical Morphology
  - Image feature detection
- Computer Vision
  - Camera Models and Calibration
  - 3D Computer Vision
  - Object Recognition

Textbooks

- Main Textbooks
  - P. Shirley, M. Ashikhmin, and S. Marschner, Fundamentals of Computer Graphics, 3rd ed., A K Peters, 2009
  - M. Sonka, V. Hlavac, and R. Boyle, Image Processing, Analysis, and Machine Vision, Thomson, 2008
  - F. Nielsen, Visual Computing: Geometry, Graphics, and Vision, Charles River Media, Inc., 2005

- Recommended Readings
  - D. Hearn, M.P. Baker, and W.R. Carithers. Computer Graphics with OpenGL, 4th Edition. Pearson Prentice Hall, 2011.
  - D. Shreiner, M. Woo, J. Neider, T. Davis. OpenGL Programming Guide: The Official Guide to Learning, 7th Edition. Addison Wesley, 2010.
  - R.C. Gonzalez and R.E. Woods, Digital Image Processing, 3rd ed., Pearson, 2008
  - D.A. Forsyth, J. Ponce, S. Mukherjee, and A.K. Bhattacharjee, Computer Vision: A Modern Approach, 2nd ed., Pearson, 2012
  - G.E. Farin and D. Hansford, The Essentials of CAGD, A K Peters, 2000

Websites

- Module website:

  https://learningcentral.cf.ac.uk/

- Graphics resources:

  http://www.opengl.org/
  http://jogamp.org/jogl/www/

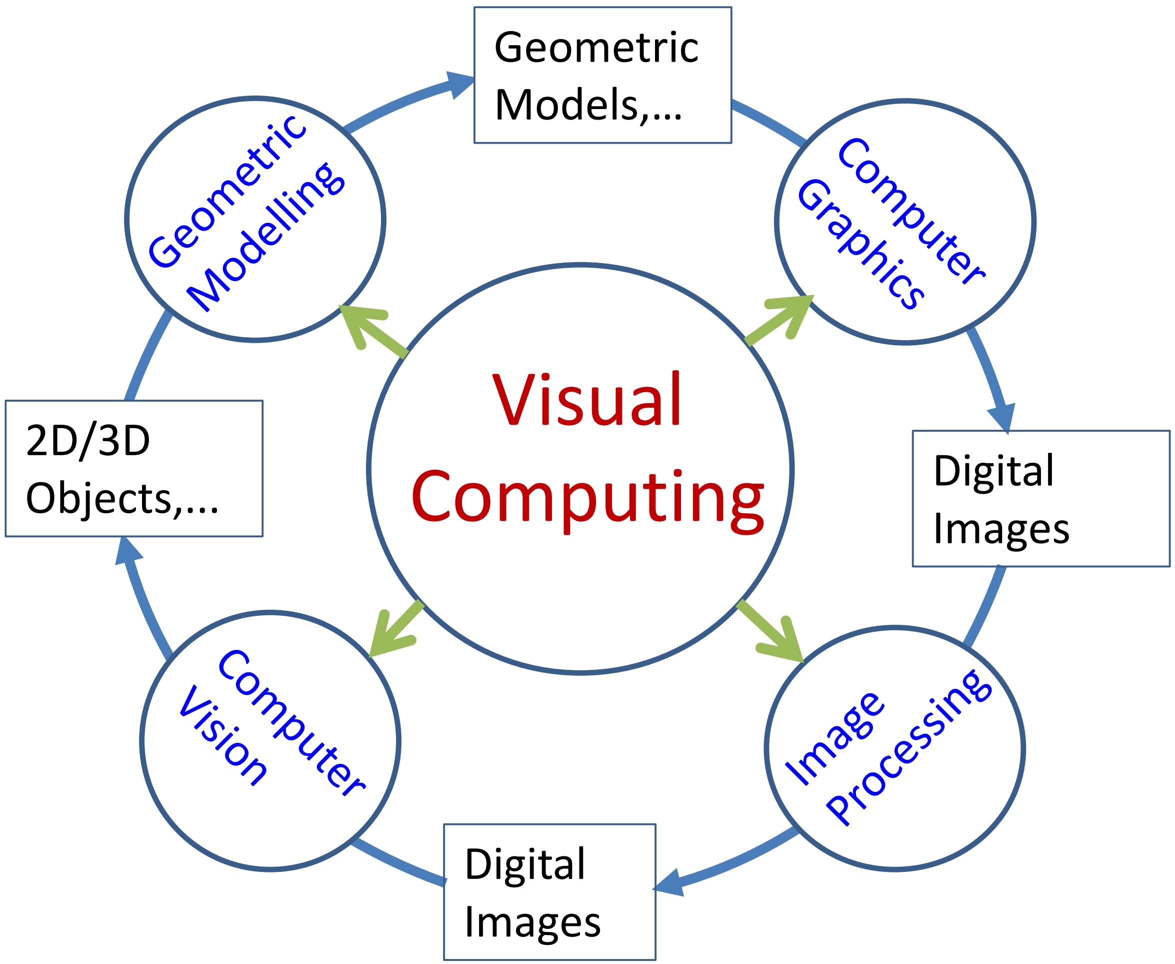
  http://www.siggraph.org/

Top Conferences

- SIGGRAPH: ACM SIGGRAPH Conference (since 1974)
  - **SIGGRAPH 2019: Technical Papers Preview**
  - **SIGGRAPH 2019: Emerging Technologies Preview**
  - **SIGGRAPH 2019: Computer Animation Festival Electronic Theater**
- I3DG: ACM-SIGGRAPH Interactive 3D Graphics (since 1987)
- CVPR: IEEE Conf on Comp Vision and Pattern Recognition (since 1988)
- ICCV: Intl Conf on Computer Vision (since 1987)

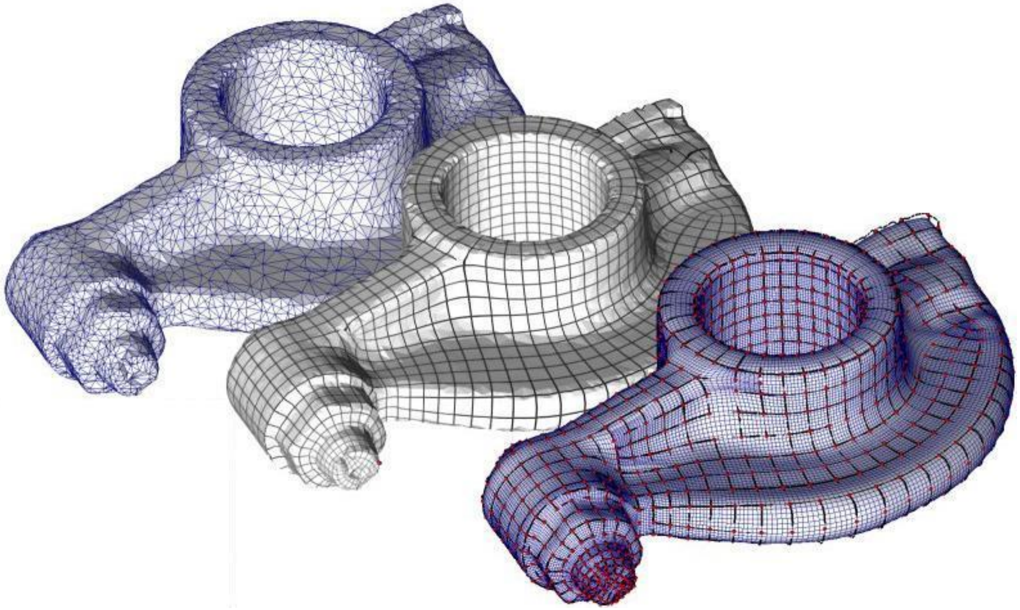
Visual Completing

- Visual Computing is a broad area of acquiring, creating, processing, analysing, and synthesising visual data by means of computers.



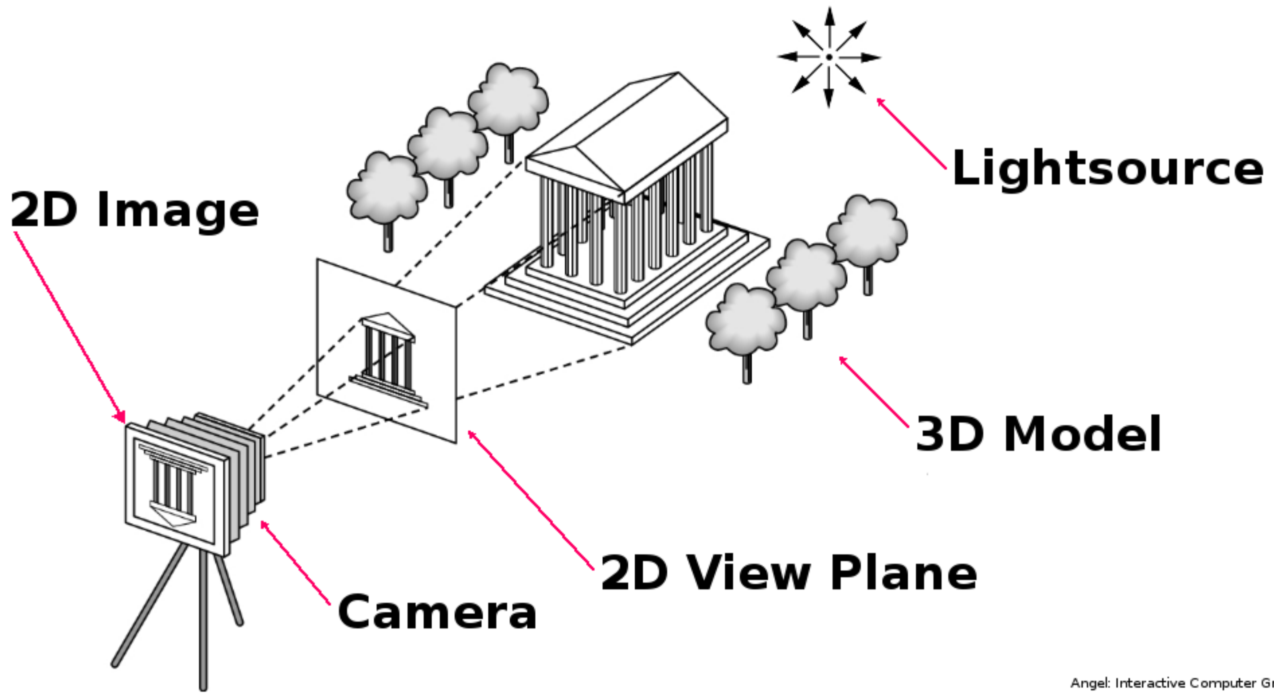
Geometric Modelling

- Geometric modelling is a subject of studying methods and algorithms for the mathematical description of shapes.
  - The shapes studied in geometric modelling are mostly two- or three-dimensional, although many of its tools and principles can be applied to sets of any finite dimension.



Computer Graphics

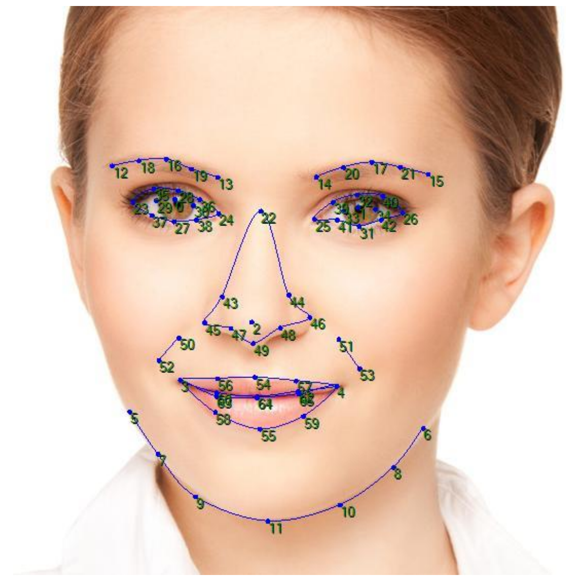
- Computer graphics is the art and science of representing and manipulating information using images generated through computation.
  - Imaging: capturing and manipulating 2D images
  - Modelling : representing and manipulating 3D objects
  - Rendering : creating 2D images from 3D models
  - Animation: simulating image changes over time with object motion



Angel: Interactive Computer Graphics

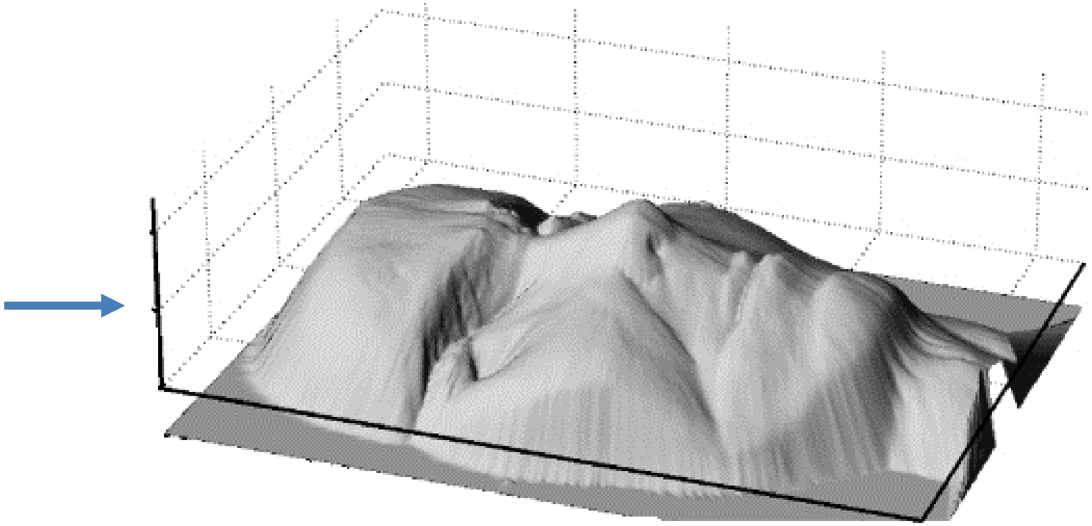
Image Processing

- Image processing is any form of signal processing for which the input is an image, such as a photograph or video frame, and the output may be either an image or a set of characteristics or parameters related to the image.



Computer Vision

- Computer vision is a field that includes methods for processing, analysing, and understanding images in order to produce numerical or symbolic information, e.g., in the forms of decisions.
  - The boundary between image processing and computer vision is blurred sometimes.

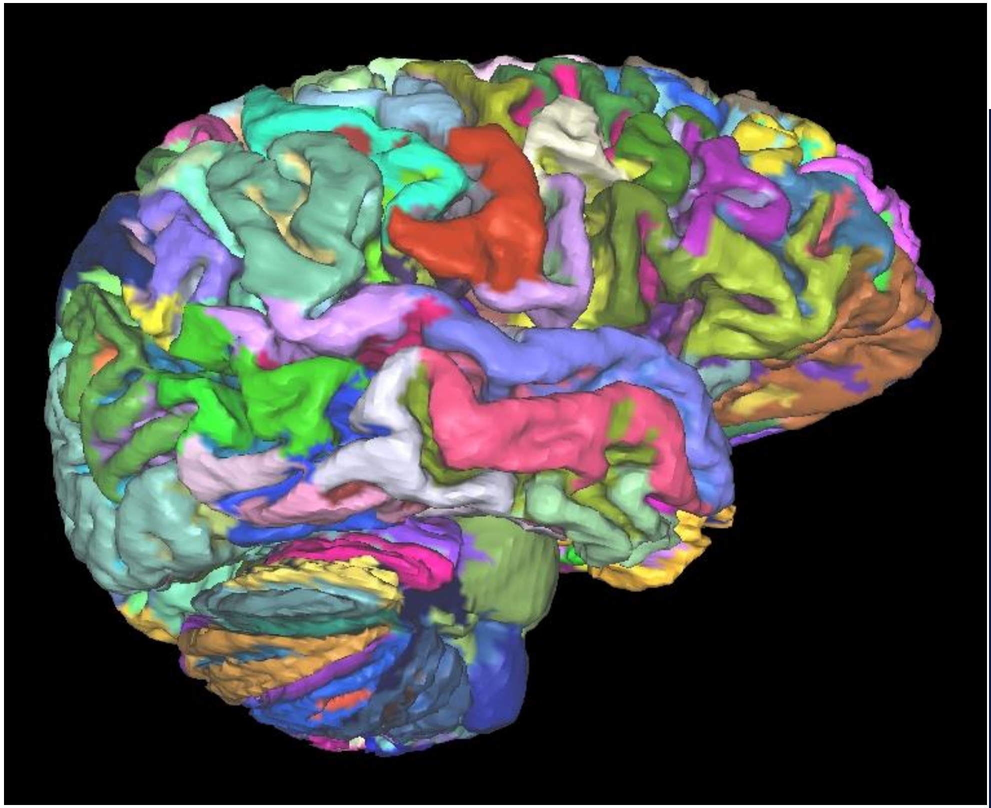


Applications

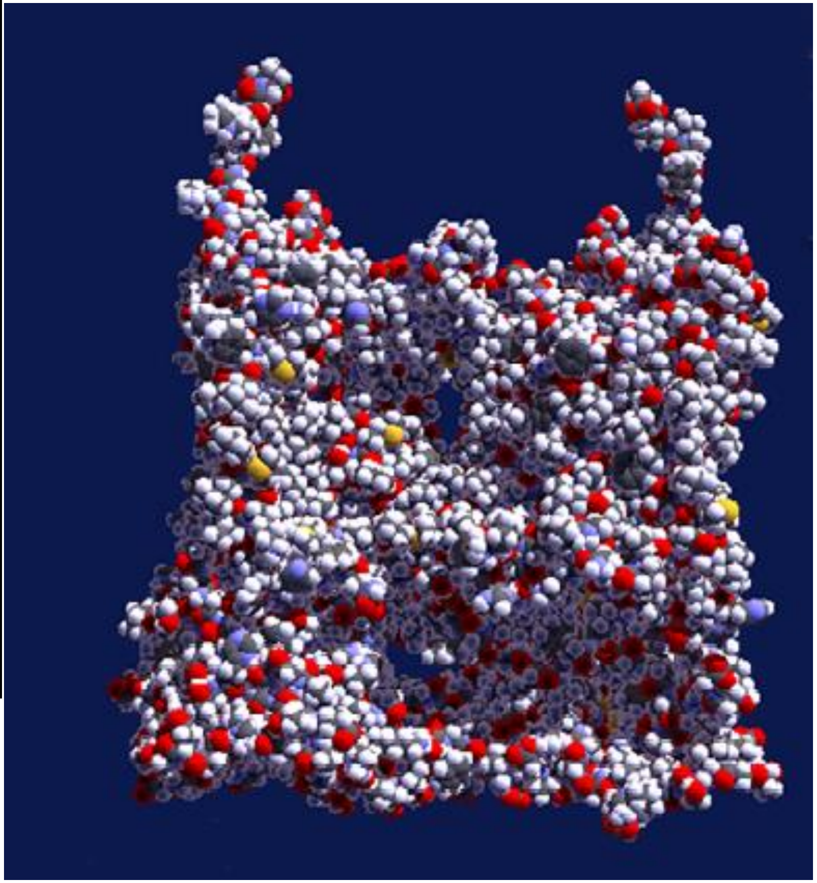
- Graphics generation
  - Visualisation of data (accurate non-realistic images)
  - Photo-realistic images (inaccurate)
  - Non-photo-realistic images, paintings
- Dynamic graphics: simulation and animation
  - Visualisation and simulation of processes
  - Realism and virtual environments
- Entertainment
  - Games, film special effects
- Industrial applications
  - Visual navigation, surveillance, biometric identification, …
- Design
  - Creating, modelling, editing and representing objects
- User interfaces
  - Suitable interactive environments

Visualisation of Data

accurate non-realistic images



http://stubber.math-inf.uni-greifswald.de/~linsen/research/index_en.html



From MIT

Photo-realistic Image

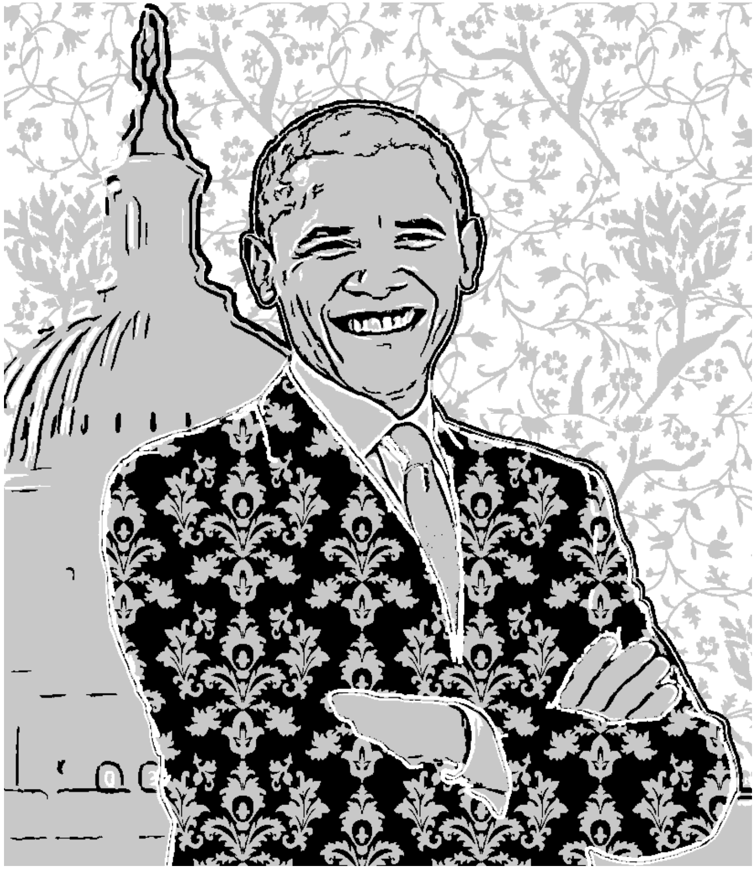
inaccurate

Boreal by Norbert Kern

Pebbles by Jonathan Hunt

Generated using POV-**Ray**

Non-photo-realistic Image



by Paul Rosin

http://marctenbosch.com/npr_shading/fruitbowl.png

Simulation and Animation

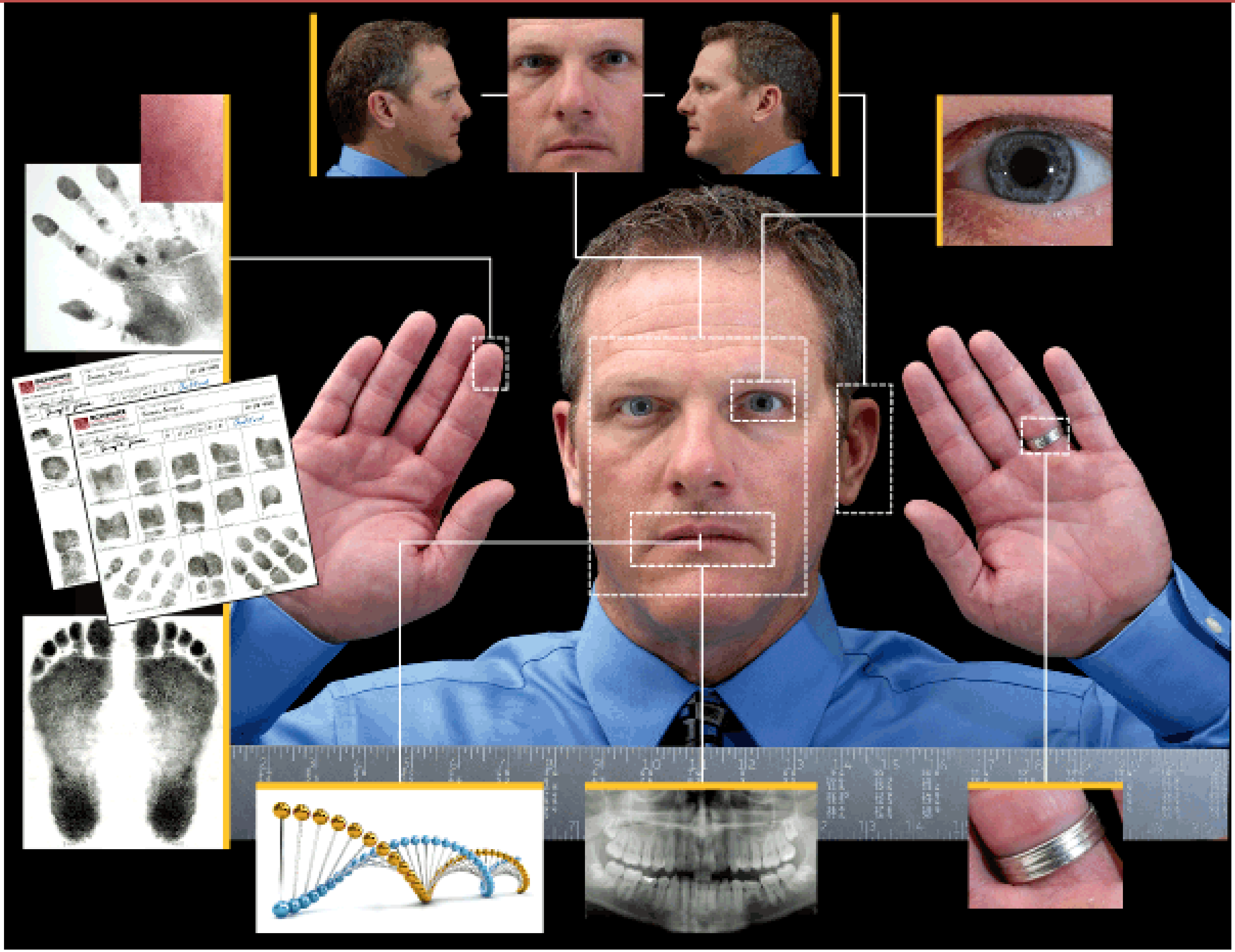
From MIT

Entertainment

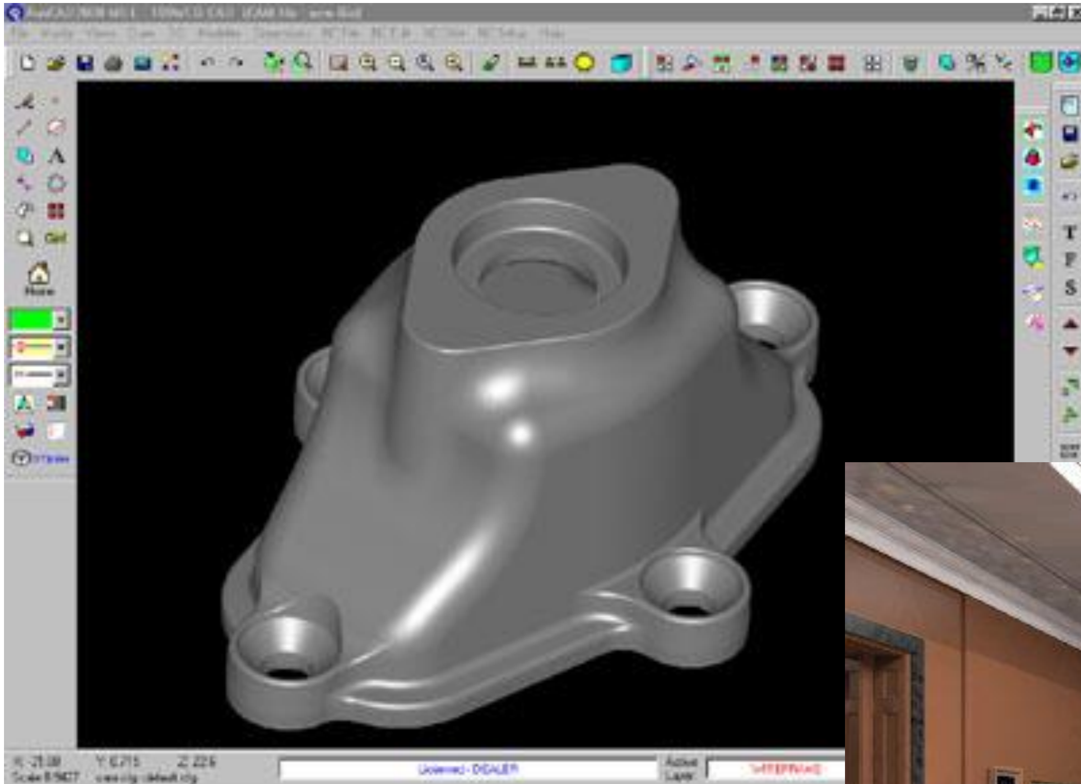
Surveillance

Autonomous Driving & Robot Navigation

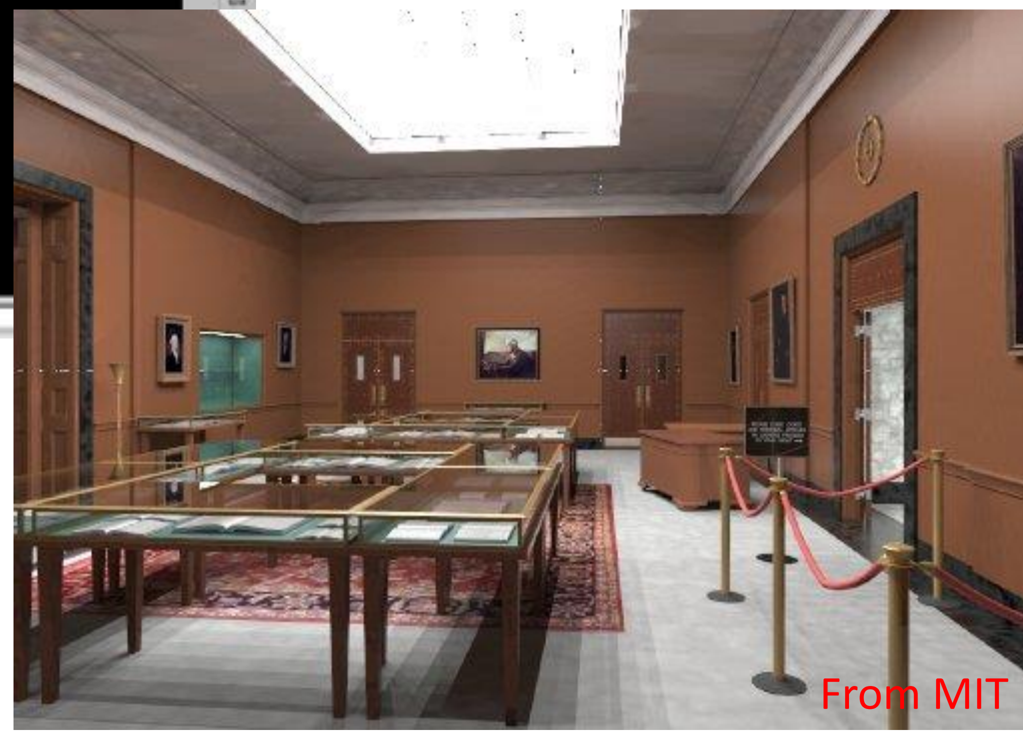
Biometric Identification



Design

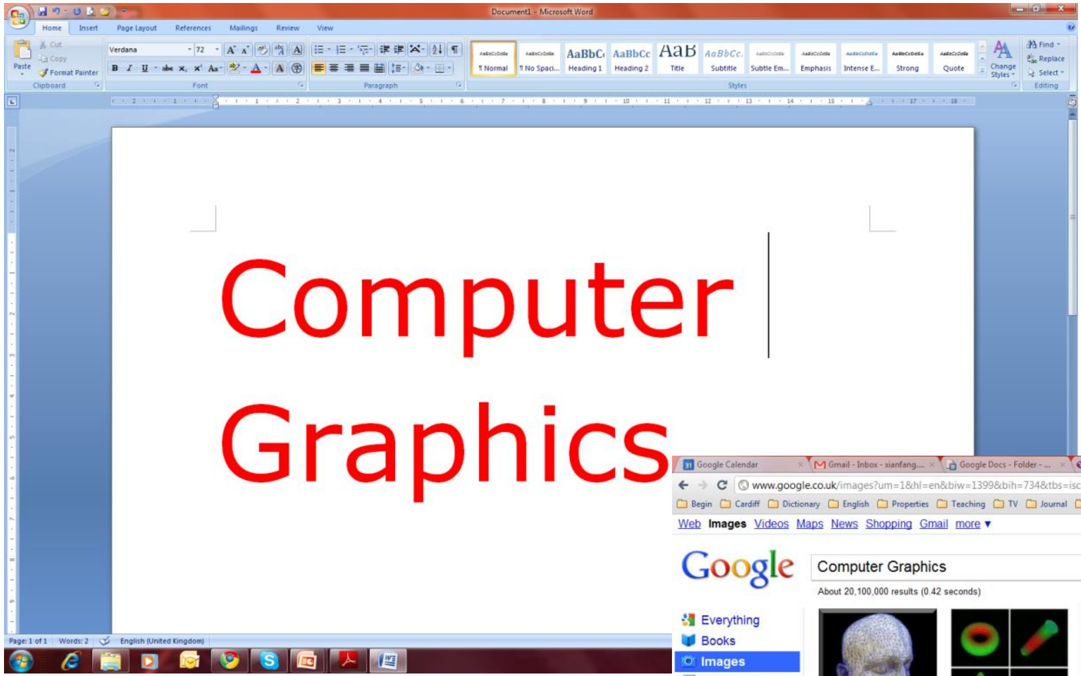


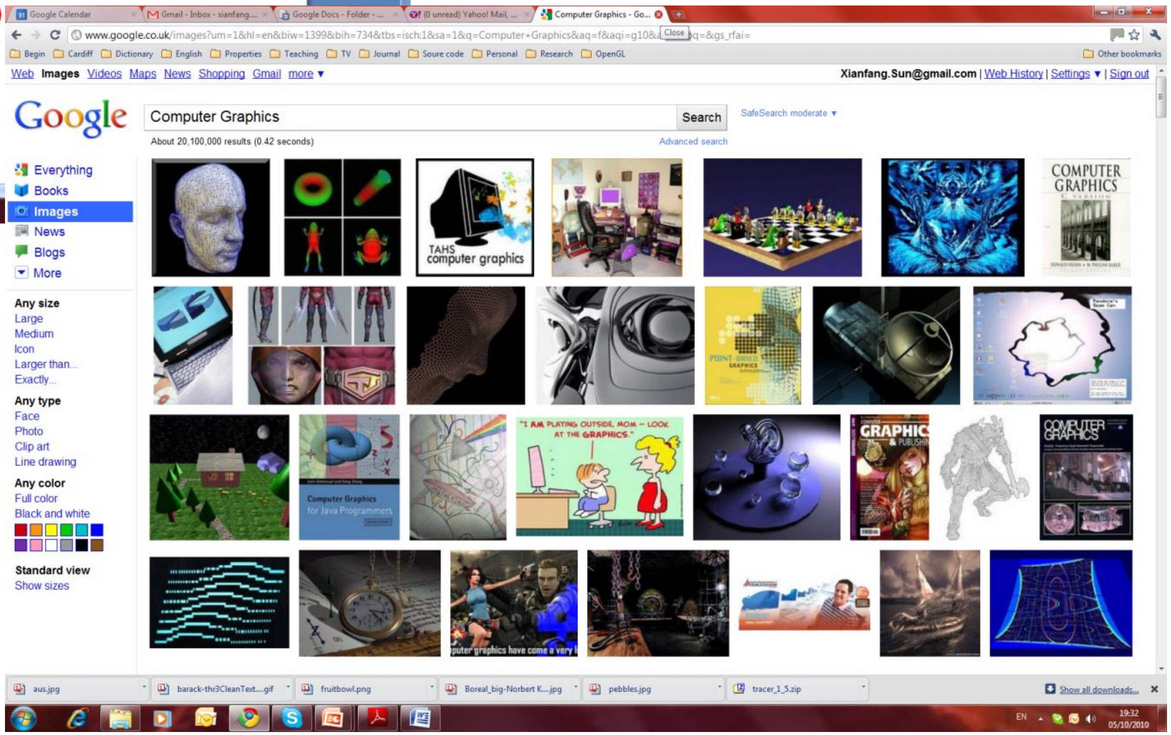
freebyte.com



From MIT

User Interface





Summary

- What is Visual computing?
- What is geometric modelling, computer graphics, image processing, and computer vision?
- Describe the relationship among the above-mentioned fields.
- List some examples of the applications of visual computing.

CMT107 Visual Computing

I.2 Graphics Systems

Xianfang Sun

School of Computer Science & Informatics
Cardiff University

Overview

- Computer Graphics
  - Image Formation
  - Raster graphics
  - Vector graphics
- Object oriented modelling
  - Modelling and Rendering
  - Realism vs real-time graphics
- A typical graphics system
  - Display Processor
- 3D Graphics Pipeline

Computer Graphics

➢ Computer graphics: Creating and manipulating visual content.

- Imaging
- Modelling
- Rendering
- Animation

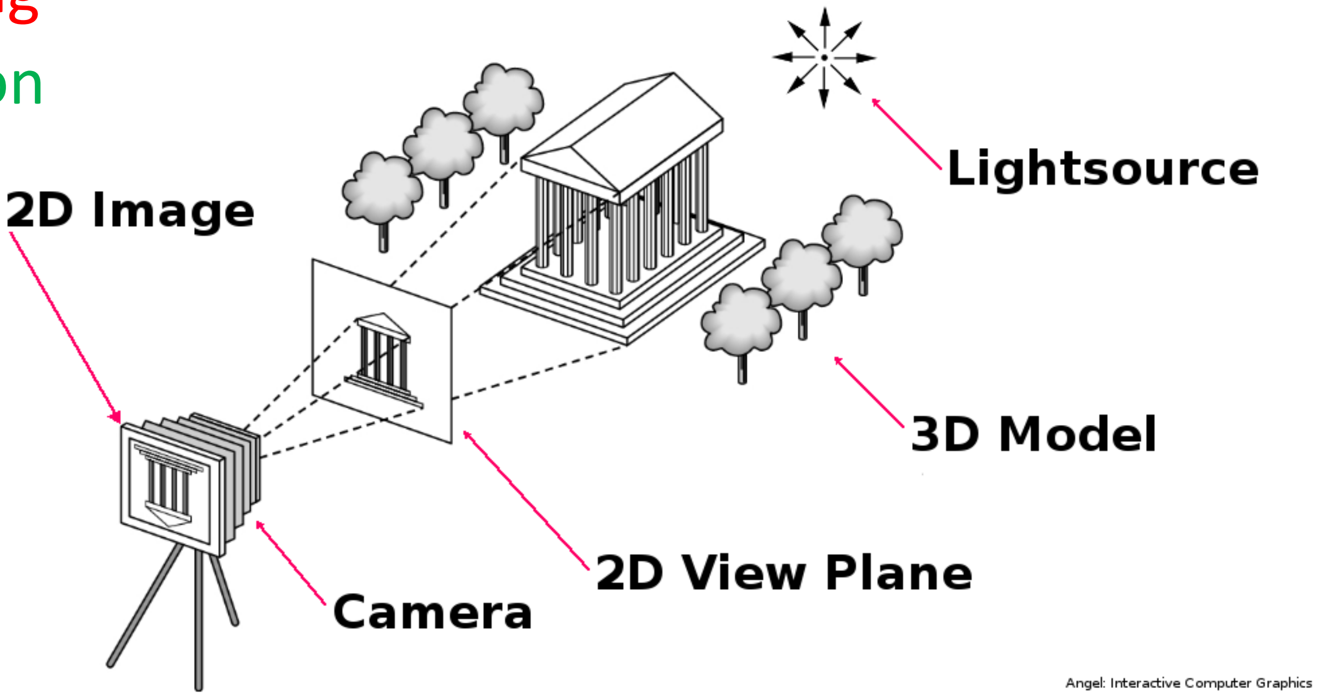
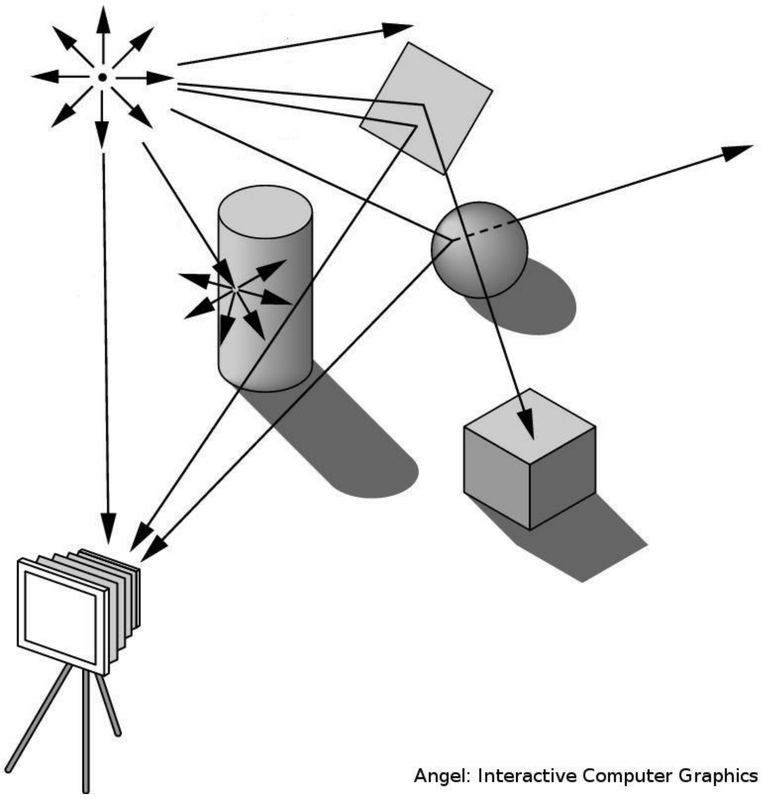


Image Formation

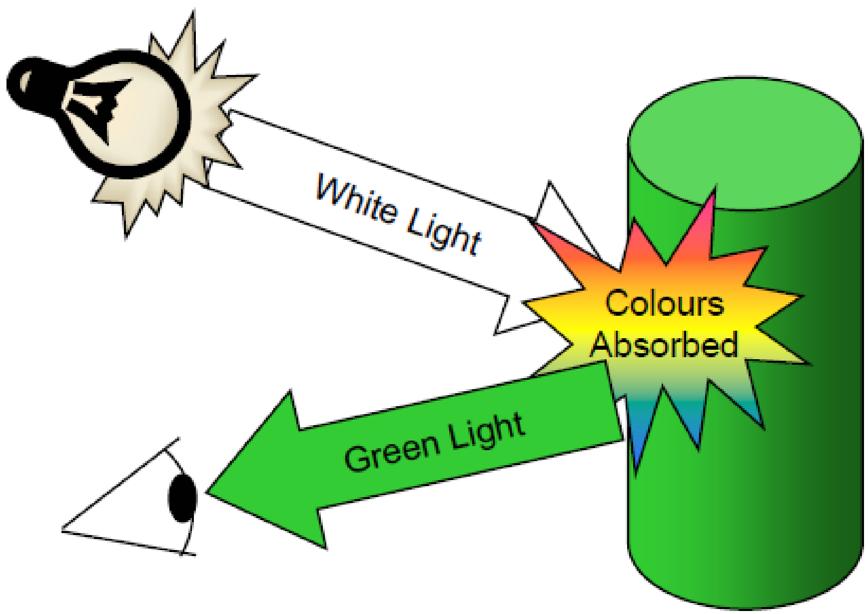
- Rendering is about forming (2D) images from 3D models
  - Analogous to physical imaging systems (cameras, microscopes, telescopes, human visual system)
- Involved elements:
  - Objects
  - Viewer / camera
  - Light sources
- Images are represented by colours



Angel: Interactive Computer Graphics

Colour

- Colour is the result of interaction between physical light in the environment and our visual system
- Attributes determine how light interacts with elements

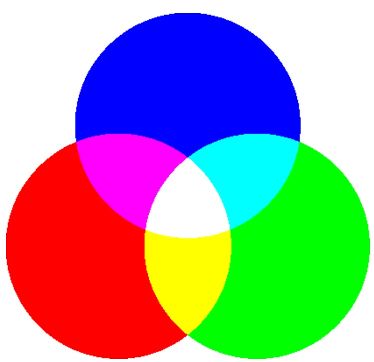


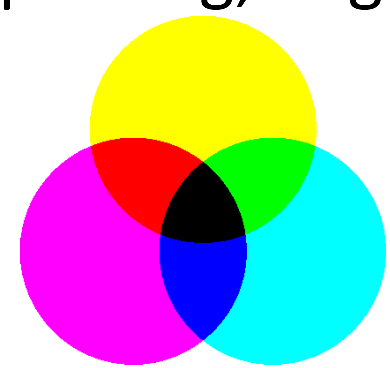
UNC Course notes: Computer Vision

Additive and Subtractive Colour

- Additive colour
  - Form a colour by adding amounts of three primaries
  - (CRTs, projection systems, positive film)
  - Primaries are Red (R), Green (G), Blue (B)
  - Sometimes alpha (A) value for transparency
- Subtractive colour
  - Form a colour by filtering white light with Cyan (C), Magenta (M), Yellow (Y) (and Black (K)) filters
  - (light-material interactions, printing, negative film)

Additive
RGB(A)



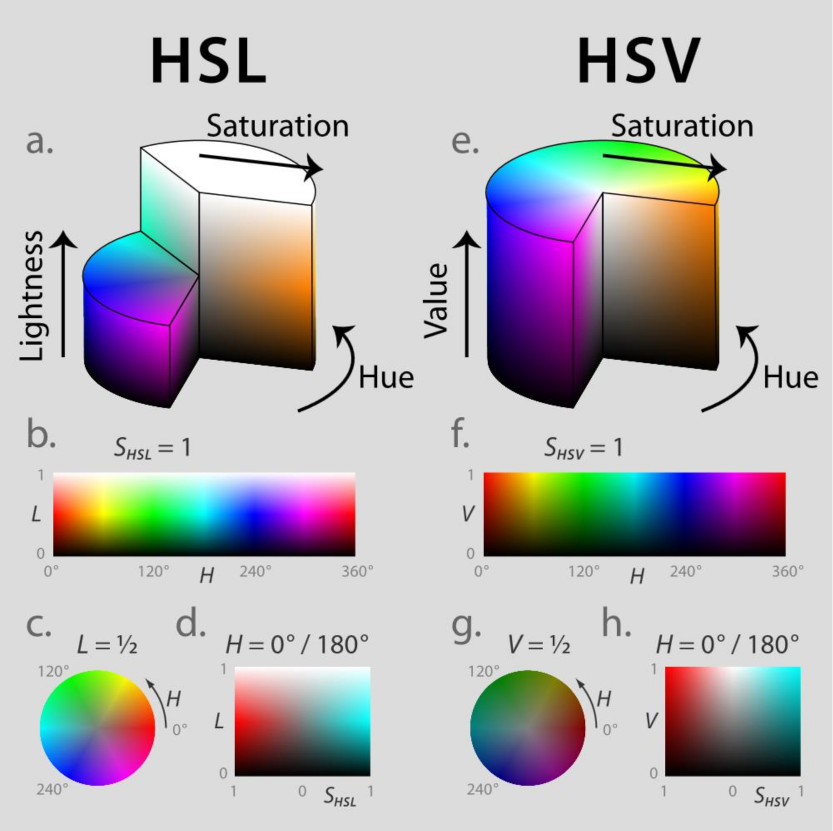


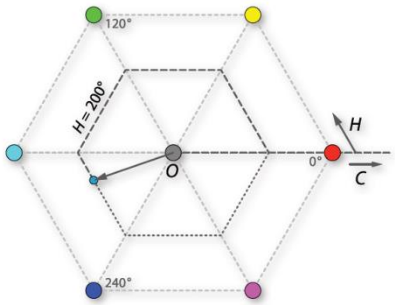
Subtractive
CMY(K)

HSL and HSV/HSB Colour Spaces

- User-oriented colour spaces
- More intuitive for interactive colour picking
- Dimensions no longer primaries
  - Hue (H): base colour
  - Saturation (S): purity of colour
  - Lightness / Luminance (L)

    Value (V) / Brightness (B)
- The lightness of a pure colour is equal to the lightness of a medium grey
- The brightness of a pure colour is equal to the brightness of white

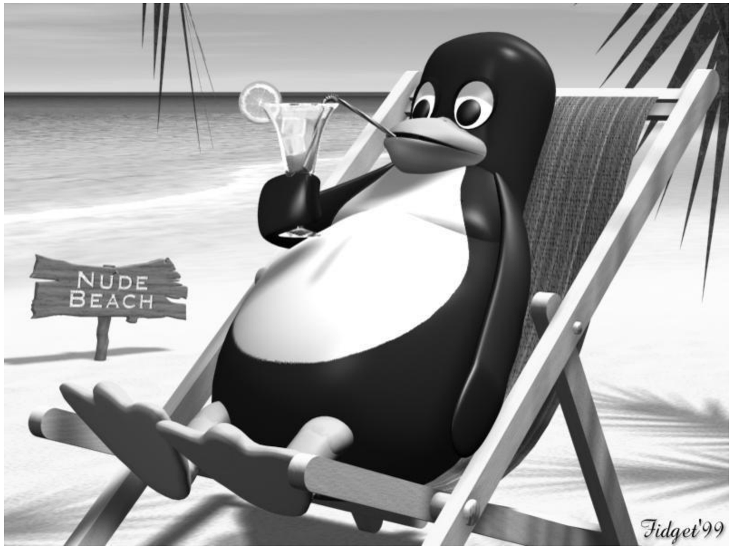




Ethz Course notes: Visual Computing

Luminance and Colour Images

- Luminance image
  - Monochromatic
  - Values are grey levels
  - Analogous to black and white film or television



- Colour image
  - Has perceptional attributes of hue, saturation, and lightness (HSL/HSB/HSV colour model)

Raster Graphics

- An image is a continuous function *f* on a *rectangular area* $A \subset \Re^2$
  - For each point $(x, y) \in A$ we have a colour value $f(x, y)$
- A raster image is a discrete function *F* on a "rectangular set" $R \subset N_0 \times N_0$ of discrete pixels (picture elements)
  - For each pixel $(u, v) \in R$ we have a colour value $F(u, v)$
- Generate a raster image from a continuous image by setting a proper value $F(u, v)$ for each pixel to *represent* the corresponding subset of the image.

Vector Graphics

- Vector graphics represents images as plotting instructions using the pen-plotter model
  - Like drawing with a pen on a rectangular sheet of paper
  - Instructions to specify movement of a pen (in straight lines, but also circles, polygons, free-form curves, etc.)
  - Pen can be on paper or not while moving
  - Attributes to fill areas with colours, patterns, and to specify line drawing styles, colours, etc. may exist
  - Continuous (non-raster) shapes and canvas
  - Rasterisation etc. is handled by API automatically
- Vector graphics APIs are normally used for 2D drawing
  - Not easily generalised to 3D

Object Oriented Modelling

- Basic elements of 3D graphics API:
  - *Objects*: lines, polygons, . . . given by positions, etc.
    - *Material*: properties of the material an object is made of, in particular how light is reflected by the object
  - *Viewer*: virtual camera given by viewing transformations
  - *Light sources*: defined by location, strength, colour, direction
- API provides methods to create and modify these elements
  - Need suitable data-structures and algorithms to represent and process graphical objects
- The image is generated from this information automatically

Modelling and Rendering

- Separate modelling of a scene from rendering it
- Modeller generates a description of the 3D scene
- Model objects of the scene on a high abstraction level
  - Describe/define properties of 3D scene
  - Designer creates and refines model (a human or program or from measurements)
  - E.g. wire-frame model for designer (faster, more suitable for editing), like a dinosaur
- Renderer *creates images* from it
  - Fast real-time rendering of images (e.g. OpenGL)
  - Computationally more expensive realistic rendering of images (e.g. *POV-Ray*)

Realism vs Real-Time Graphics

- **Realism:**

  make images look as real as possible

  - Realistic shapes
  - Realistic illumination
  - Realistic behaviour and movements

- **Real-time:**

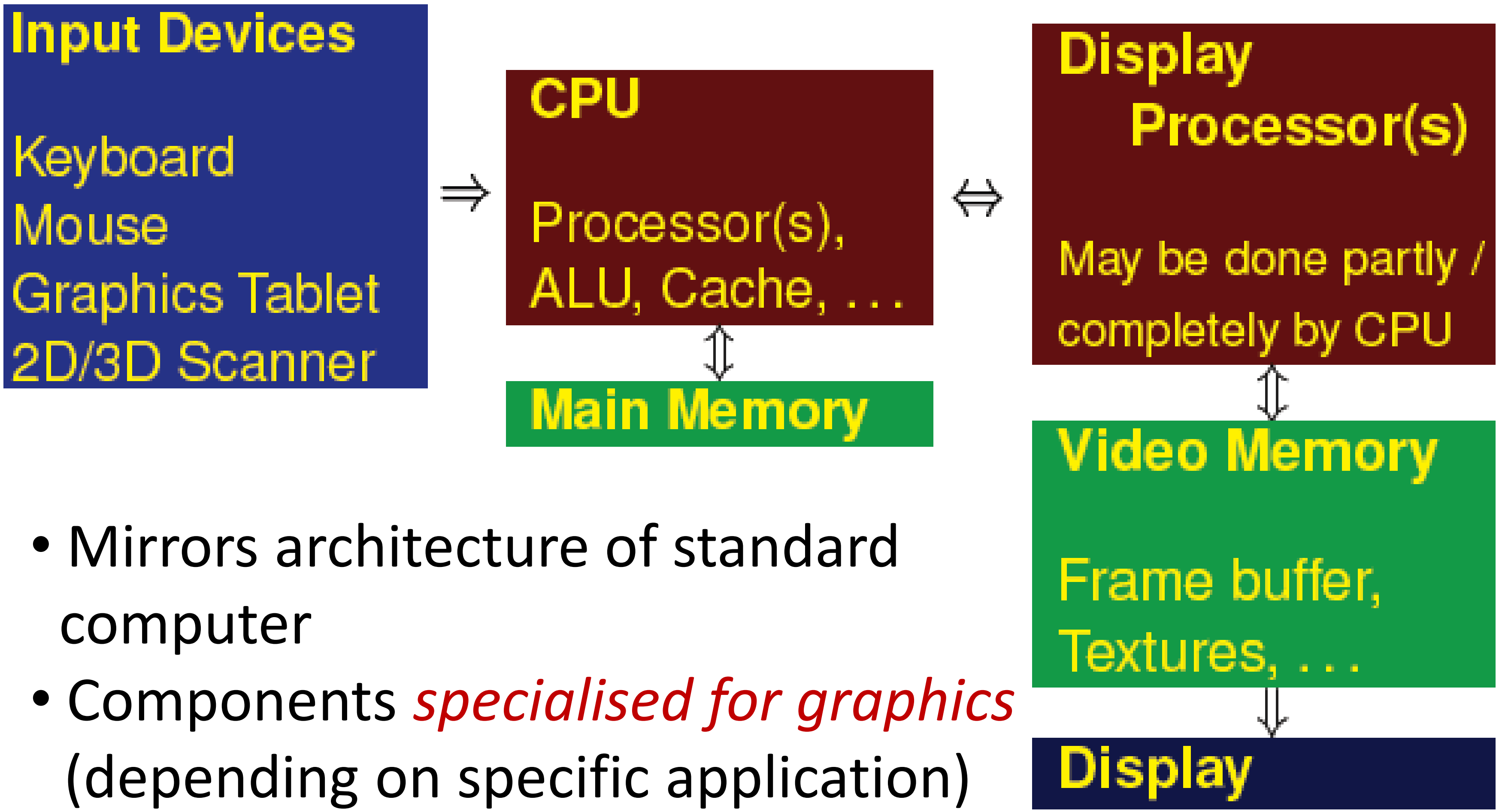
  display images "fast enough"

  - (high number of frames per second)
  - Perceive smooth motion
  - Interact with the environment

- Tension between the two goals

Typical Graphics System

- *Simple model* of a graphics system



- Mirrors architecture of standard computer
- Components *specialised for graphics* (depending on specific application)

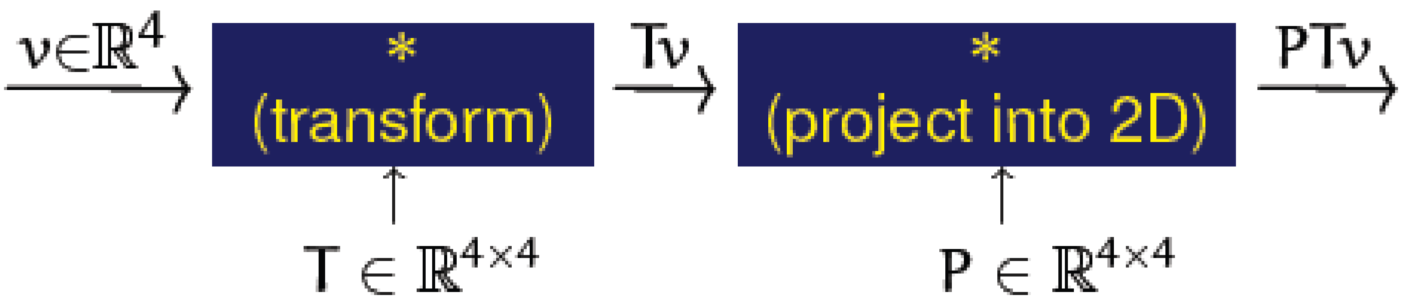
Display Processor

- Task of the display processor:

  Relieve the host (CPU) from expensive graphics computations using specialised hardware
- Initial versions of display processors:
  - Host computes instructions to create image: *display lists*
  - Display processor *executes display lists* in local memory repetitively to refresh image
- Modern display processors: pipeline architecture

Display Processor Pipeline

- Display processors consist of two sub-systems
  - Front-end sub-system to handle geometry (e.g. pipeline on a stream of primitives)
  - Back-end sub-system to handle rasterisation (e.g. parallel processing on raster)
  - Pipelining and/or parallel processing used for both
- Special processing unit for individual graphics operations

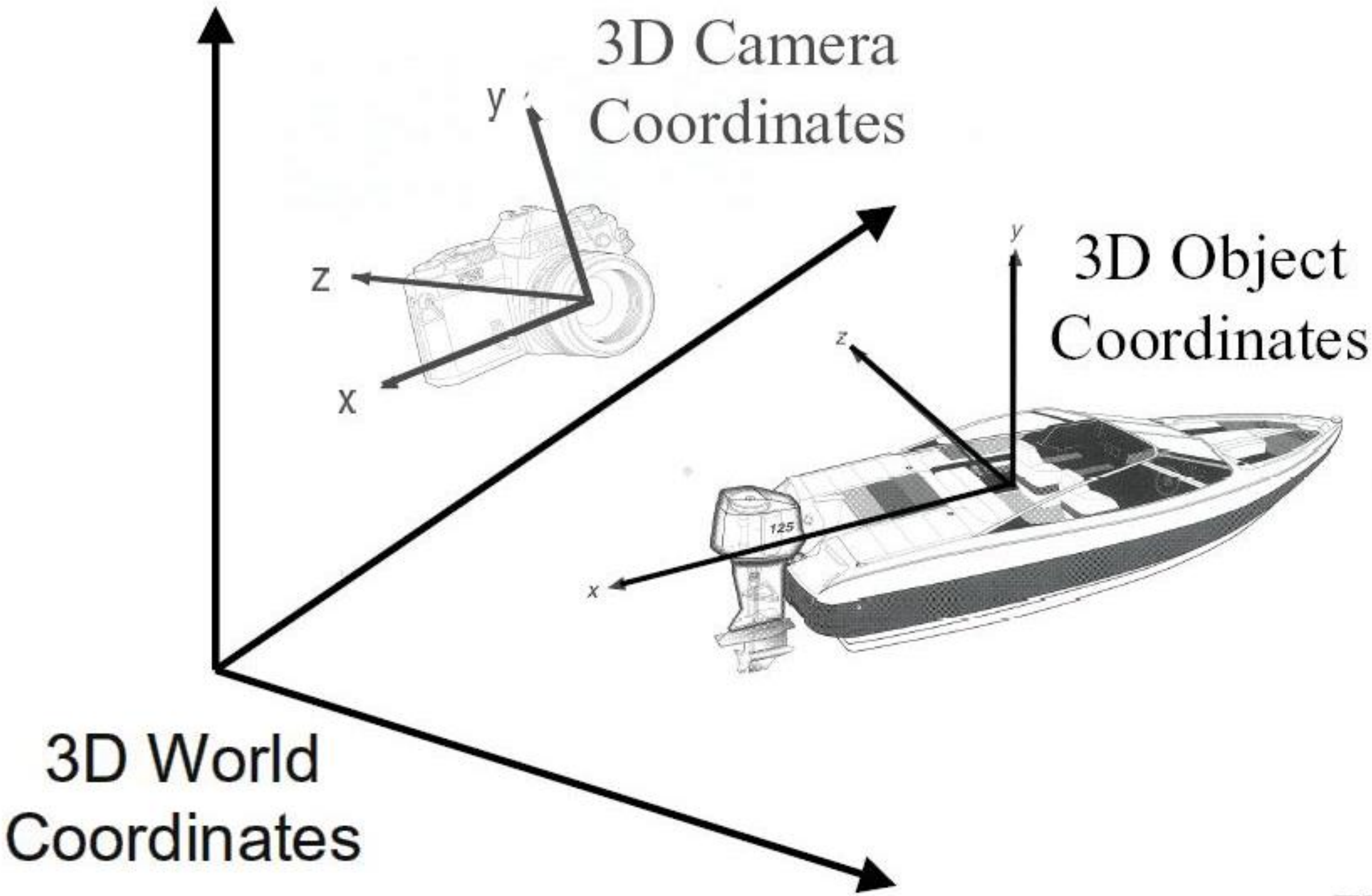


(vertices given by 4 numbers define geometry and are modified by linear transformations / matrices, this will become clearer later)

Graphics Pipeline Tasks

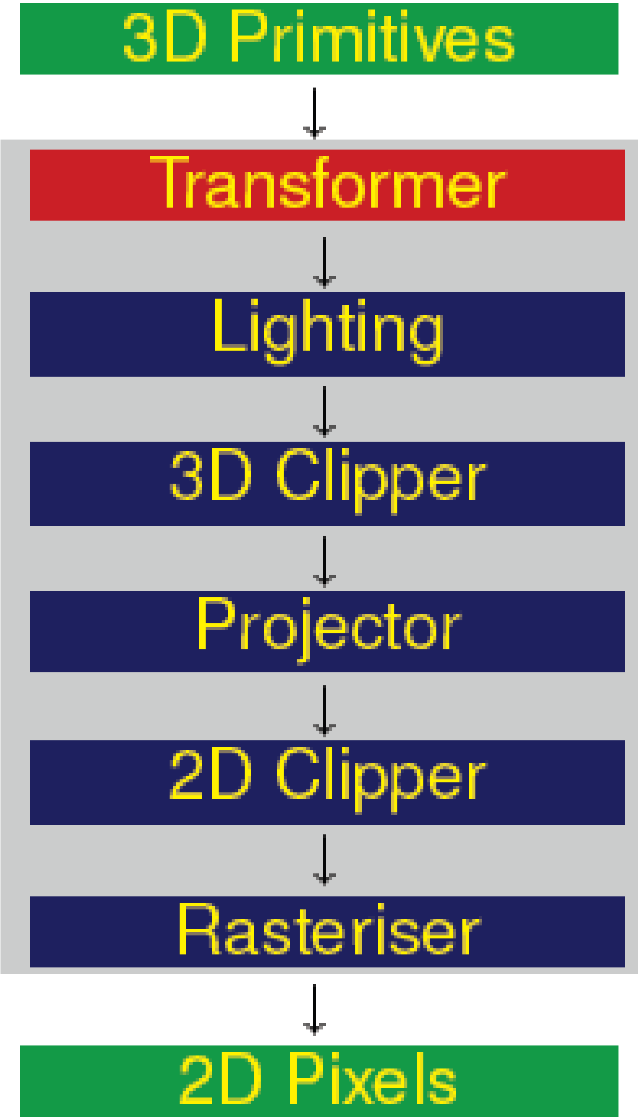
- **Input** of graphics pipeline provided by host / user code:
  - 3D models (e.g. triangular meshes)
    - Transformations applied to models (e.g. rotations)
    - Material properties (e.g. colour)
  - Light sources
  - Camera
- **Output** of graphics pipeline:
  - 2D pixels in a raster
- What operations does the pipeline have to execute?
  - Models (vertices) are transformed into pixels by pipeline
  - The attributes are transformed in the pipeline

Coordinate Systems



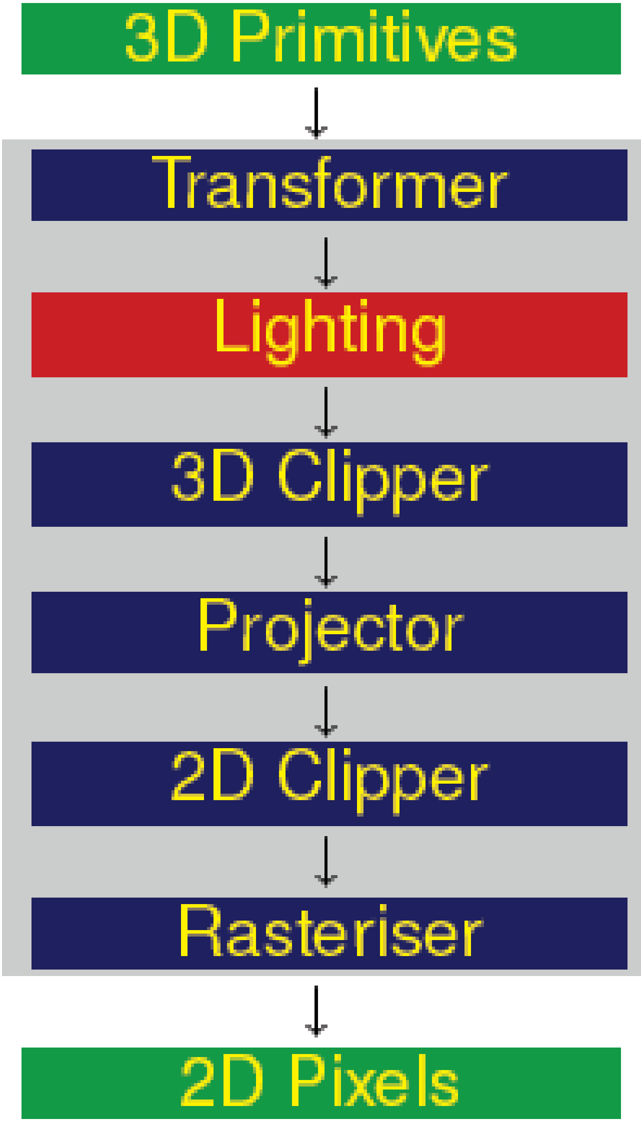
FVFHP Figure 6.1

3D Graphics Pipeline



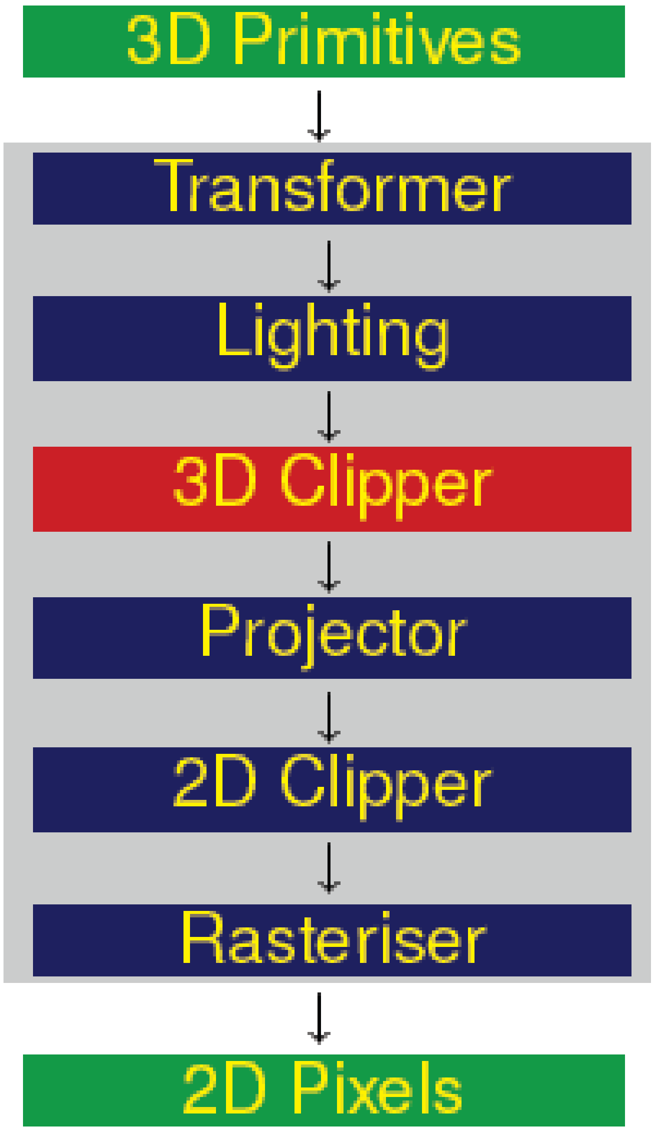
- Modelling transformations:
  - Convert *model coordinates* into *world coordinates*
- Viewing transformations:
  - Convert *world coordinates* into *camera coordinates*
- Multiple transformation matrices can be combined to single matrix
- Parallel realisation allows to multiply vector with matrix in one operation

3D Graphics Pipeline



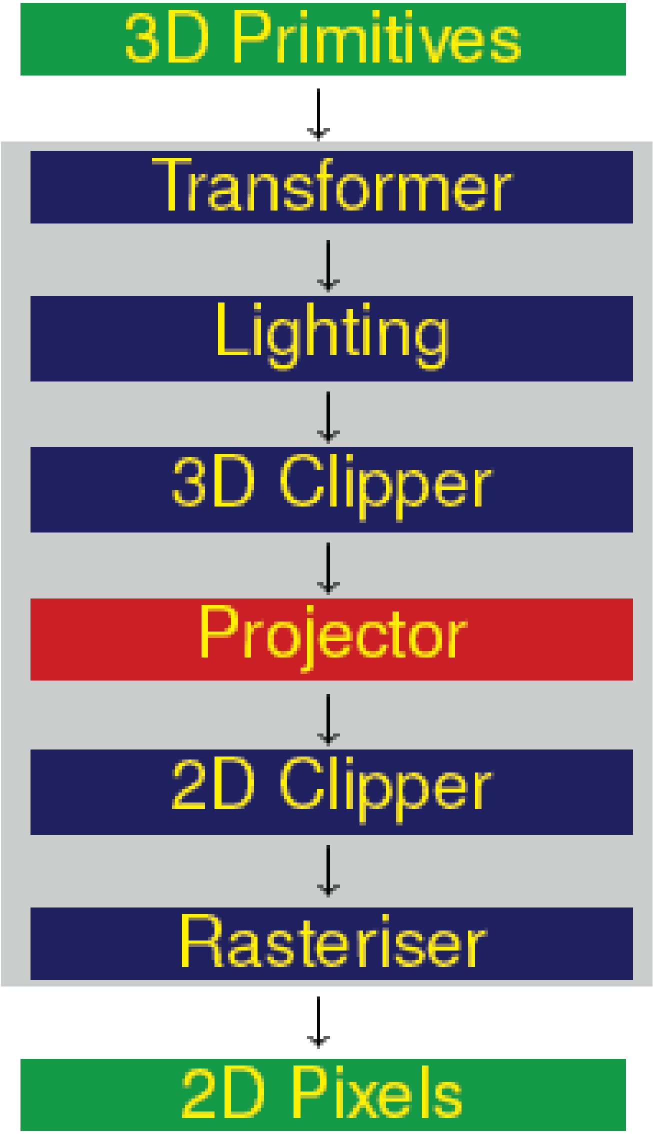
- Evaluate illumination model
  - To determine colour/shades of primitives
    - E.g. compute colour value for polygon vertices based on material properties, light sources, etc.
    - Shading of whole primitive is done during rasterisation based on these values

3D Graphics Pipeline



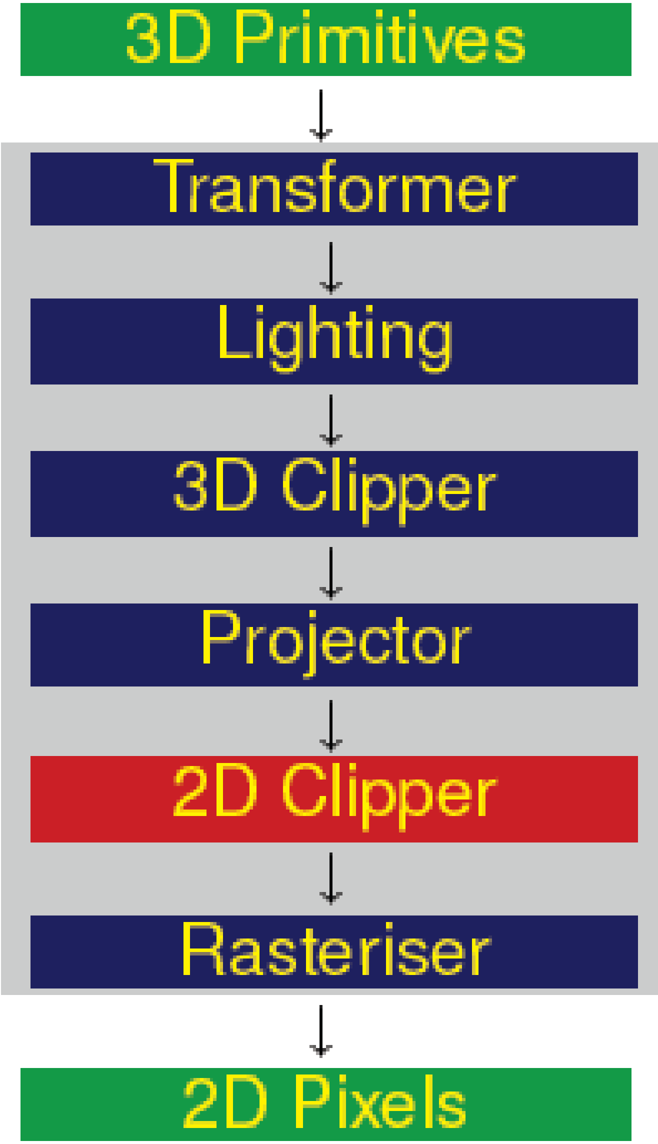
- Clipping selects visible part of the whole scene for displaying
  - 3D clipping selects primitives inside viewing volume (cut off objects at planes)
    - For *perspective* projection: frustum (cut-off pyramid)
    - For *parallel* projection: rectangular parallelepiped
  - May also remove hidden surfaces

3D Graphics Pipeline



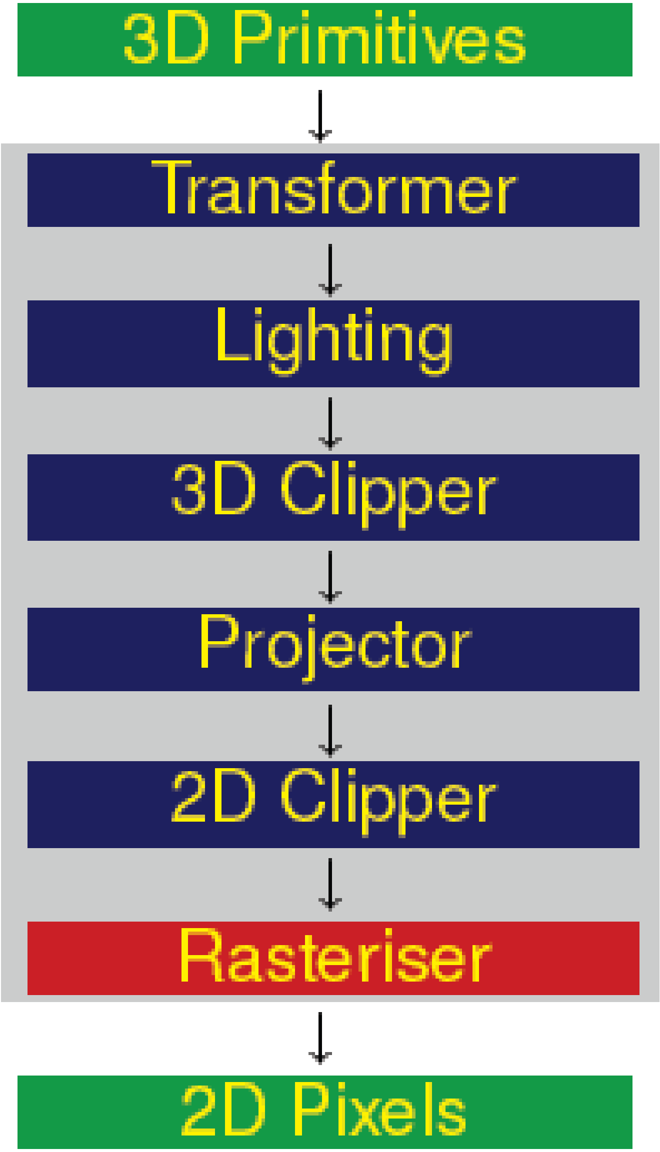
- Project 3D primitives onto plane to give 2D shapes
  - Projection matrix specifies type of projection
  - Camera properties specify projection in detail

3D Graphics Pipeline



- 2D clipping:
  - Cut off (partially or completely) 2D shapes outside a rectangular area
  - Apply viewport transformation (project rectangular area onto raster)
  - May also remove hidden surfaces

3D Graphics Pipeline



- Convert 2D shapes into pixels
  - Scan conversion: draw 2D polygons, lines, points in frame buffer
  - May include shading of lines and polygons

Summary

- What is computer graphics?
- What is rendering? List the elements of rendering.
- What are raster graphics and vector graphics? Explain their major differences.
- What are the functions of the modeller and renderer?
- Describe a simple model of a typical graphics system.
- Describe three major coordinate systems in the graphics pipeline.
- What are the major components of a graphics pipeline and how do they interact?