CMT1[redacted]ual Computing

I.2 Graphics Systems

Xianfang Sun

School of Computer Science & Informatics

Cardiff University

Overview

- Computer Graphics
  - Image Formation
  - Raster graphi
  - Vector graphi
- 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

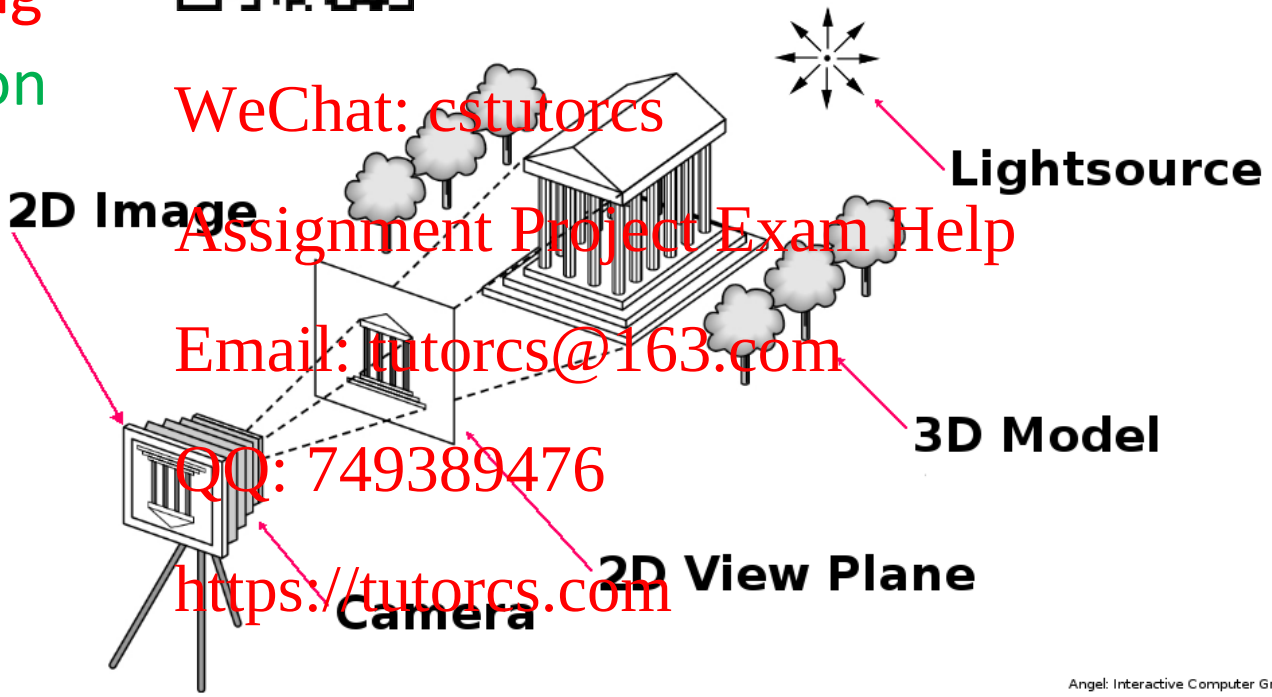
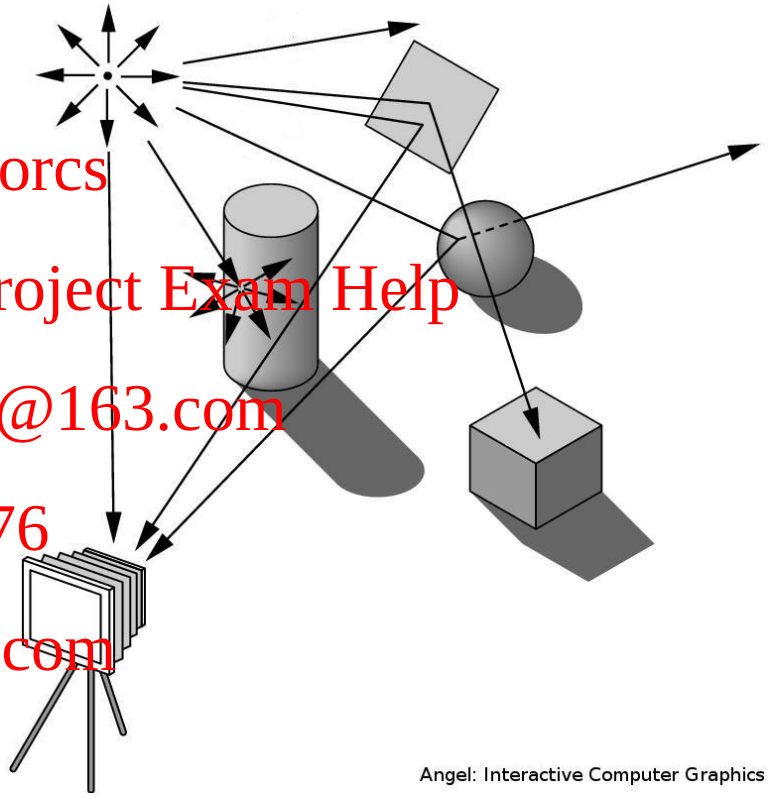


Image Formation

- 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 determining how light interacts with elements

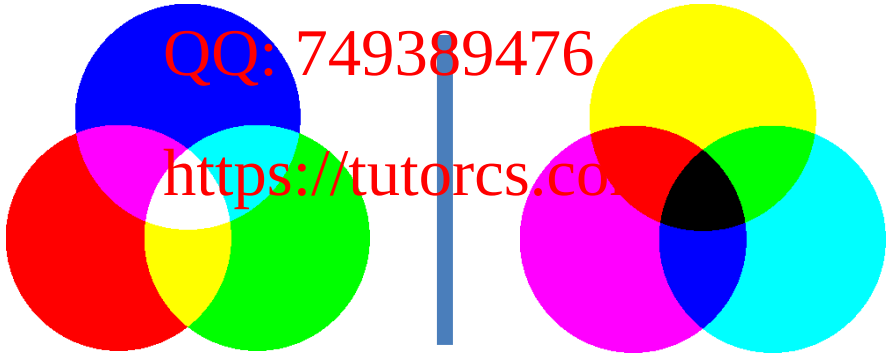
UNC Course notes: Computer Vision

Additive and Subtractive Colour

- Additive colour
  - Form a colour by adding amounts of three primaries
  - (CRTs, projectors, 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)

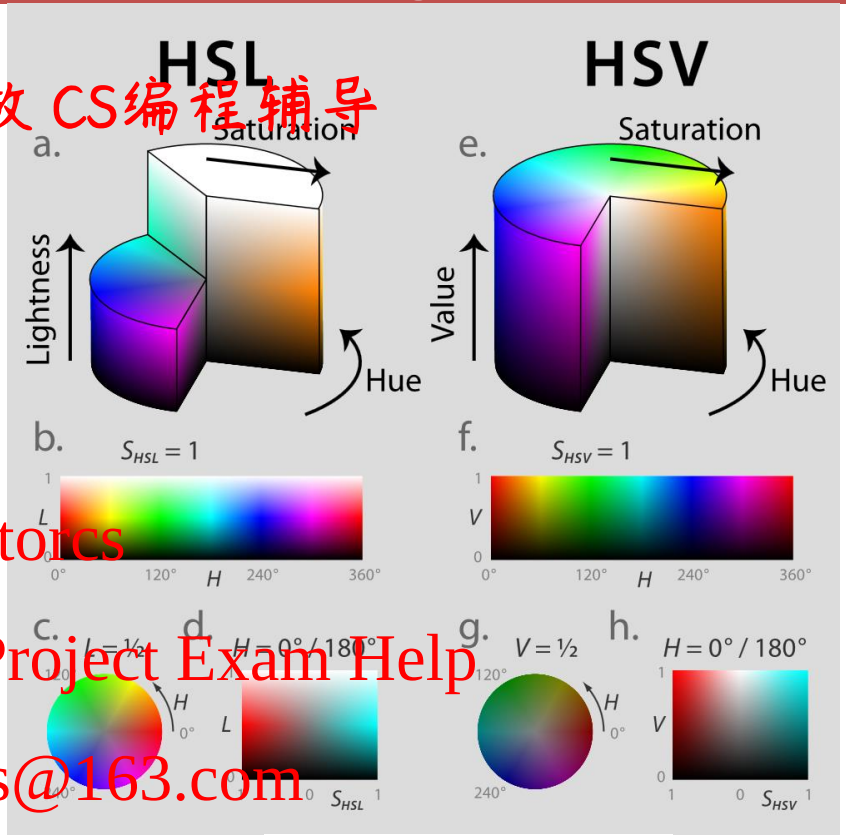


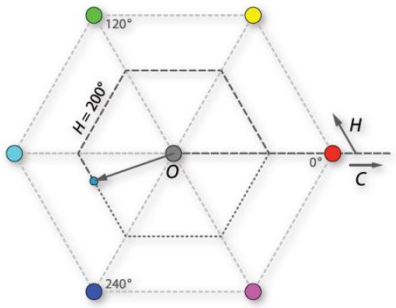
Subtractive CMY(K)

HSL and HSV/HSB Colour Spaces

- User-oriented colour spaces
- More intuitive for interactive colour picking
- Dimensions no lon ries
  - Hue (H): base colo
  - 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





Luminance and Colour Images

- Luminance image
  - Monochromatic
  - Values are gre
  - Analogous to d 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 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

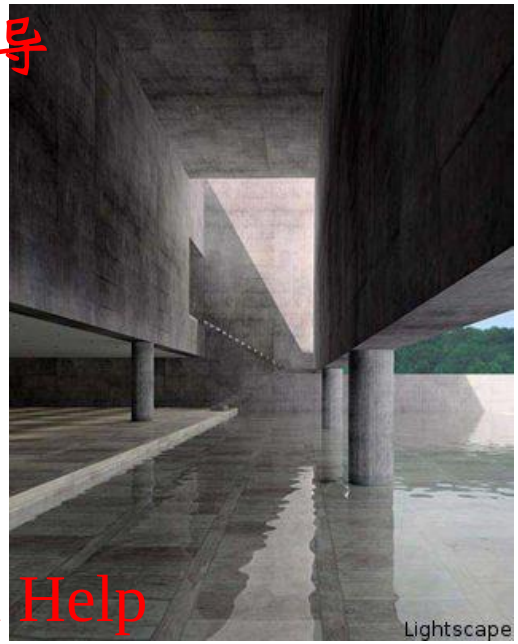
- Basic elements of 3D graphics API:
  - *Objects*: lines, polygons, . . . given by positions, etc.
    - *Material*: pro[illegible]f the material an object is made of, in particular hov[illegible]eflected by the object
  - *Viewer*: virtual ca[illegible]en by viewing transformations
  - *Light sources*: 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/defines 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*)

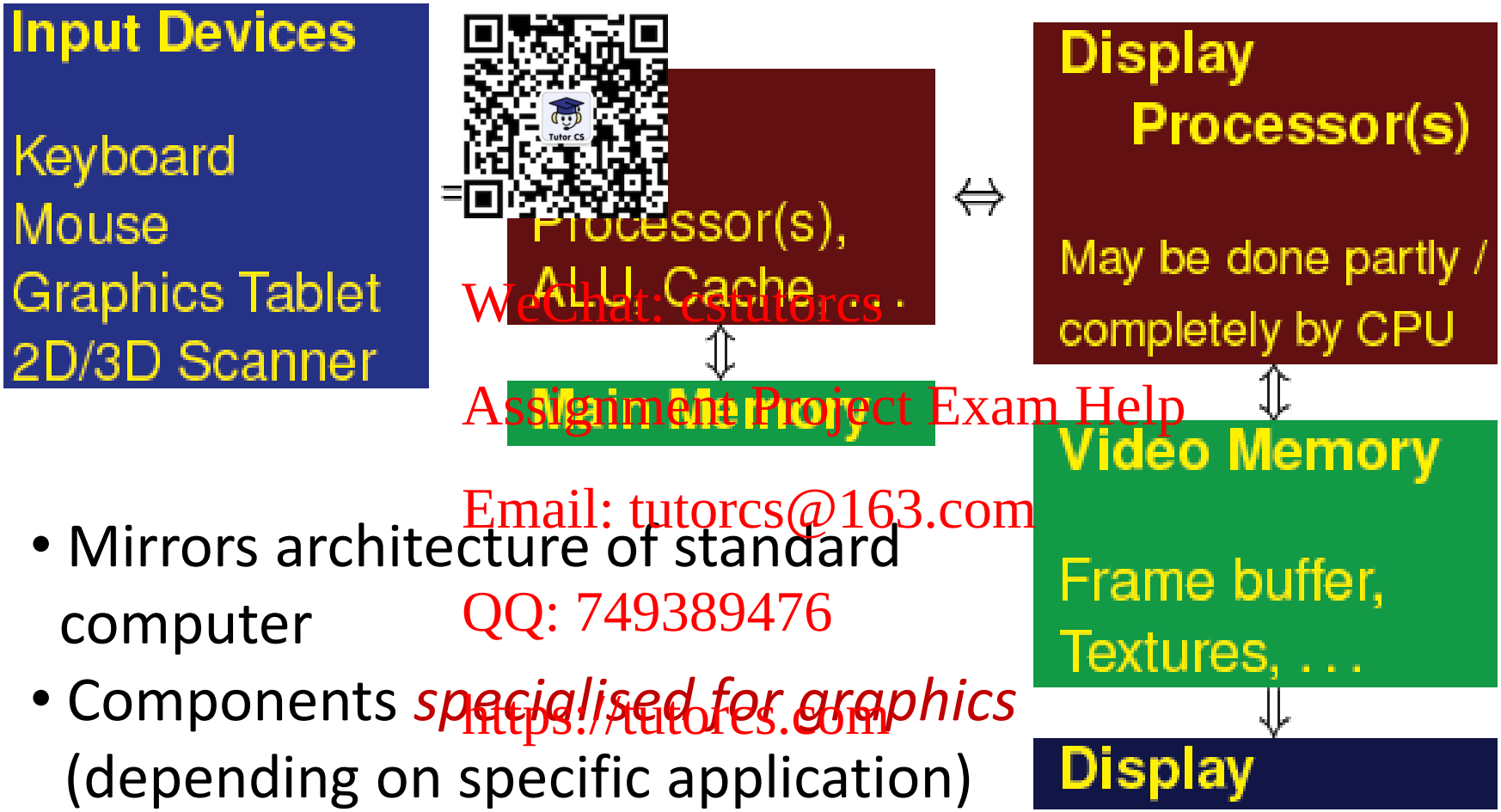
Realism vs Real-Time Graphics

- Realism:
  - make images look as real as possible
    - Realistic shapes
    - Realistic illumin
    - 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

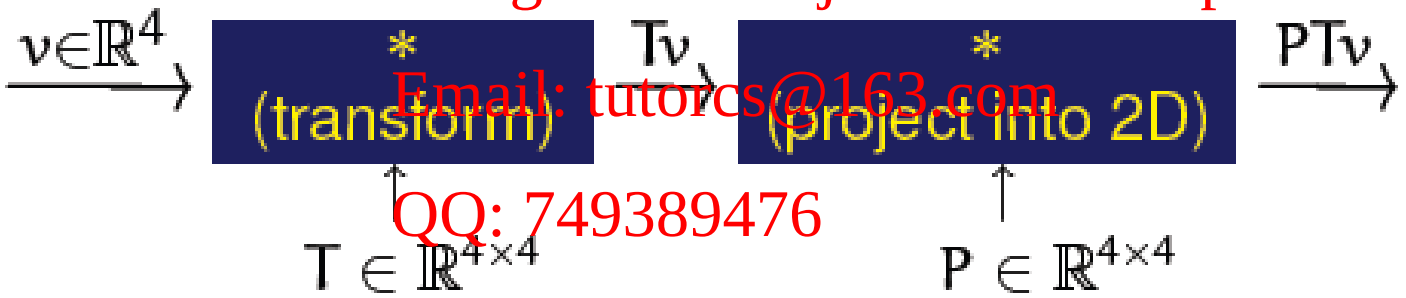
- Task of the display processor:

  Relieve the host (CPU) from expensive graphics computations by 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 or[illegible]m of primitives)
  - Back-end sub-sy[illegible] handle rasterisation (e.g. parallel processing on raster)
  - Pipelining and/or parallel processing used for both
- Special processing unit for individual graphics operations

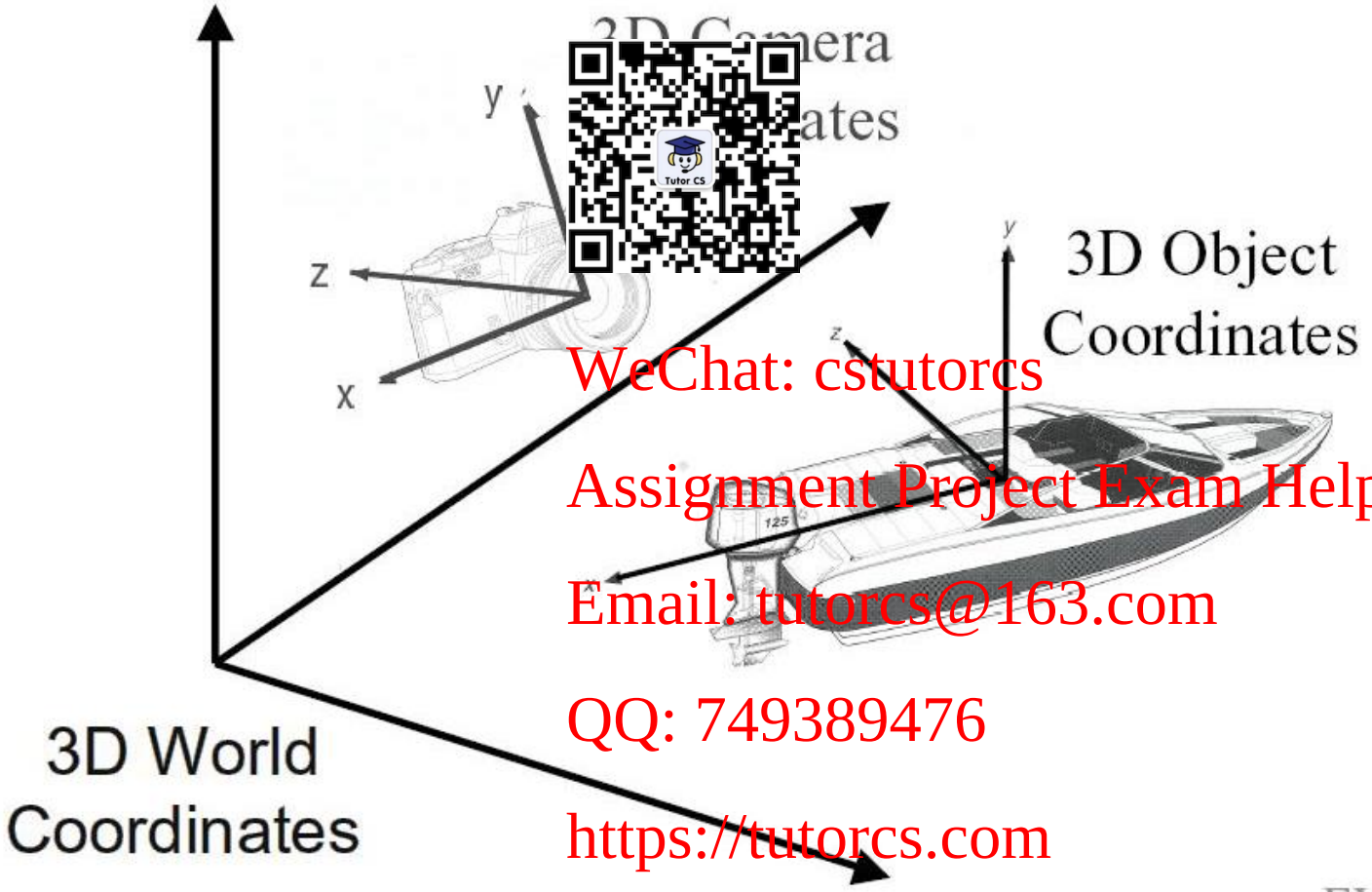


(vertices given by 4 numbers define geometry and are modified by linear transformations / matrices, this will become clearer later)

Graphics Pipeline Tasks

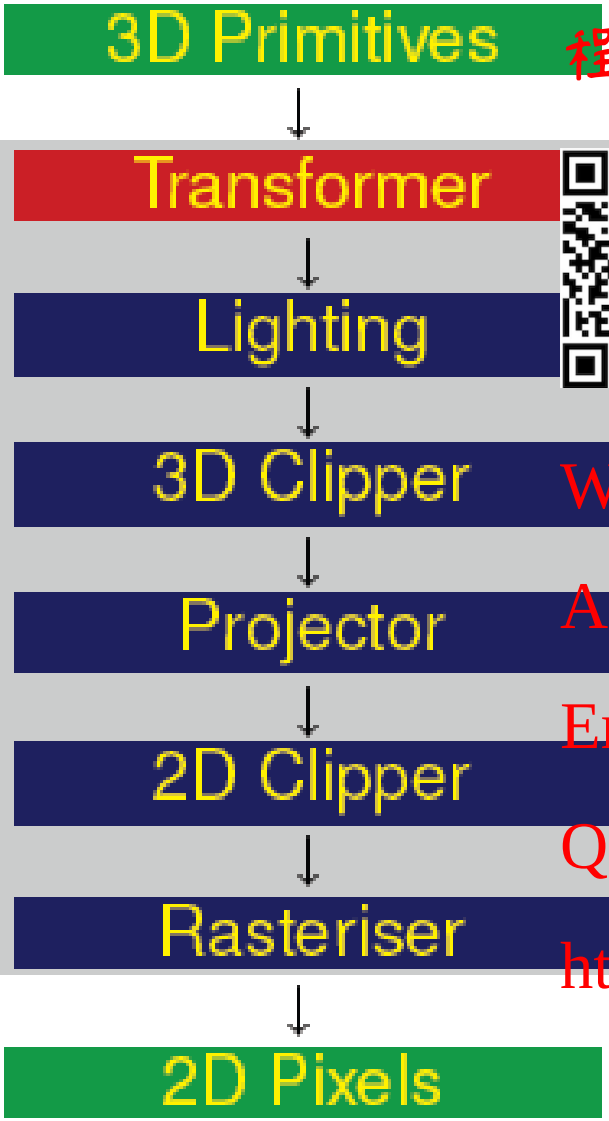
- Input of graphics pipeline provided by host / user code:
  - 3D models (e.g. triangular meshes)
    - Transformations applied to models (e.g. rotations)
    - Material properties (e.g. colour)
  - Light sources
  - Camera
- Output of graphics pipeline:
  - 2D pixels in a raster
- What operations does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

Coordinate Systems



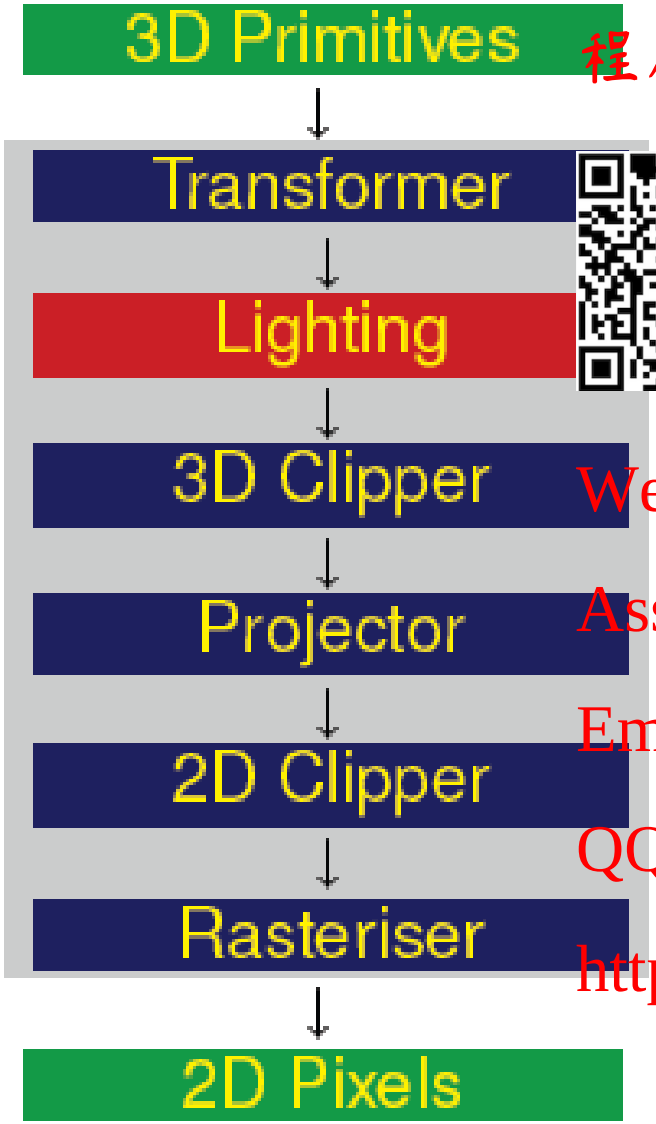
FVFHP Figure 6.1

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

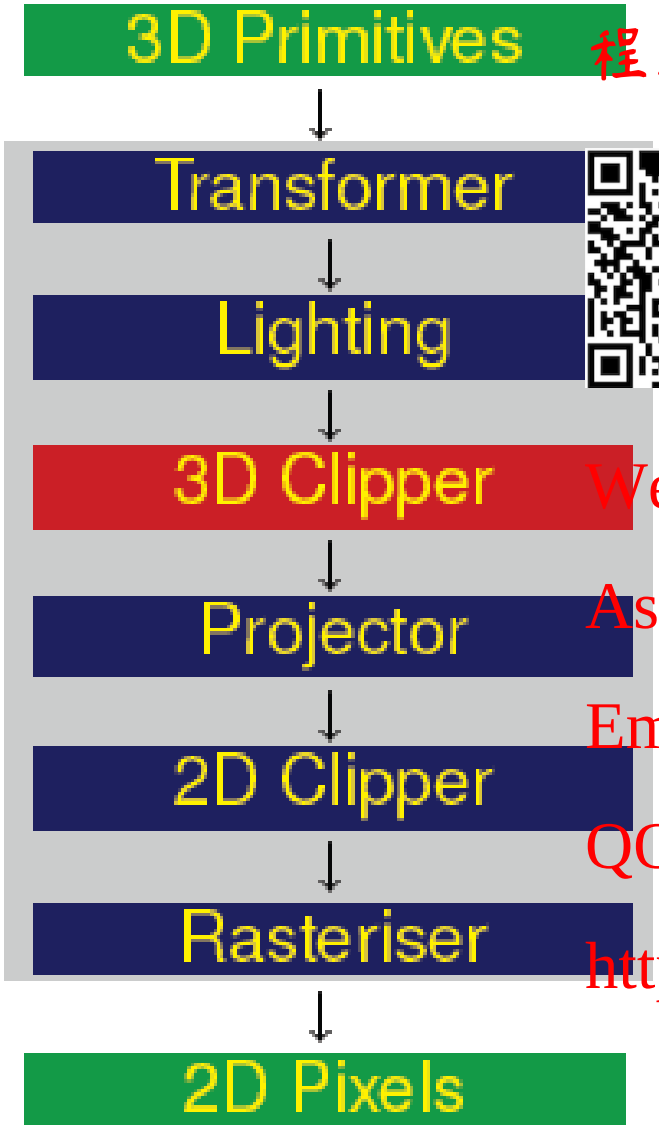
3D Graphics Pipeline



Evaluate illumination model

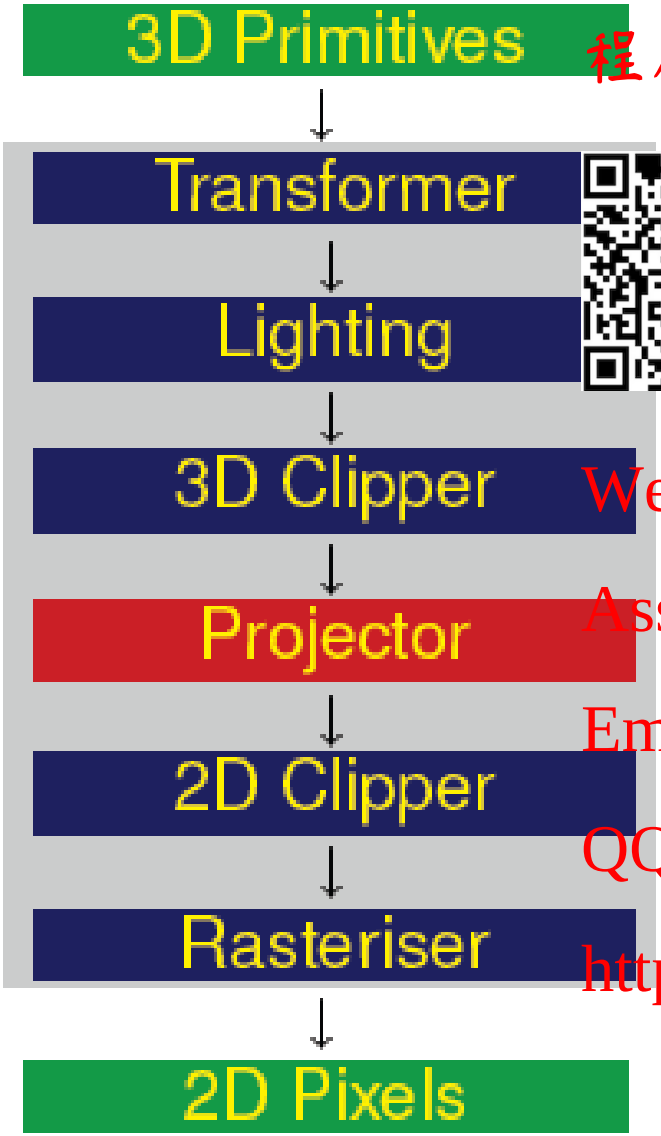
- To determine colour/shades of primitives
  - E.g. compute colour value for polygon vertices based on material properties, light sources, etc.
  - Shading of whole primitive is done during rasterisation based on these values

3D Graphics Pipeline



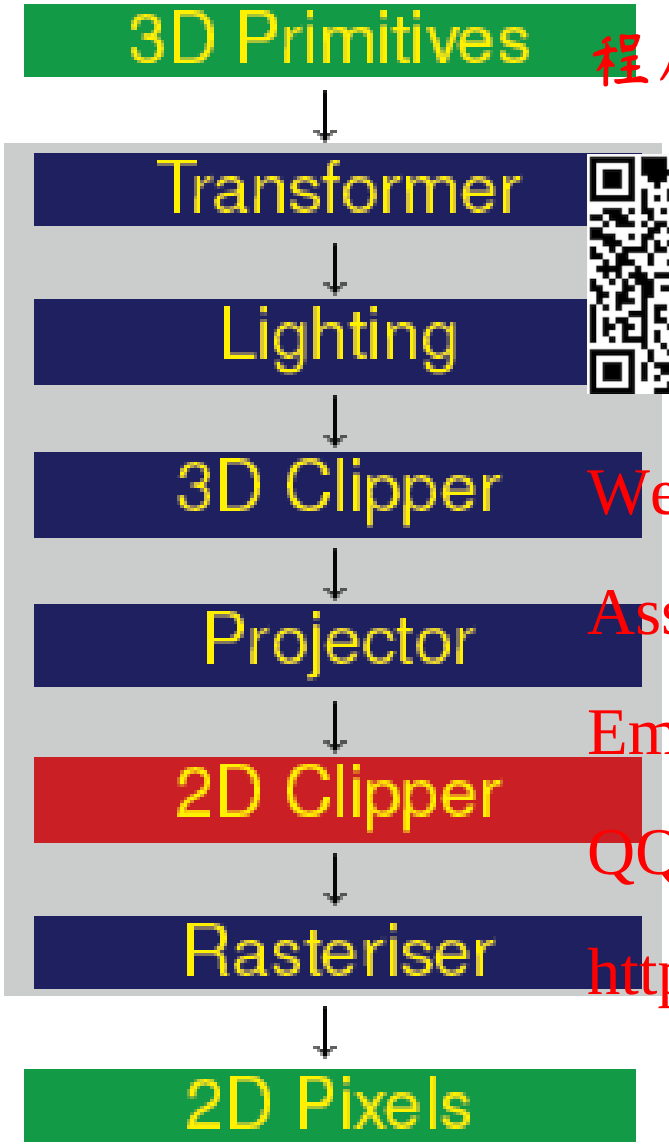
- Clipping selects visible part of the whole scene for displaying
  - 3D clipping selects primitives inside viewing volume (cut off objects at planes)
    - For *perspective* projection: frustum (cut-off pyramid)
    - For *parallel* projection: rectangular parallelepiped
  - May also remove hidden surfaces

3D Graphics Pipeline



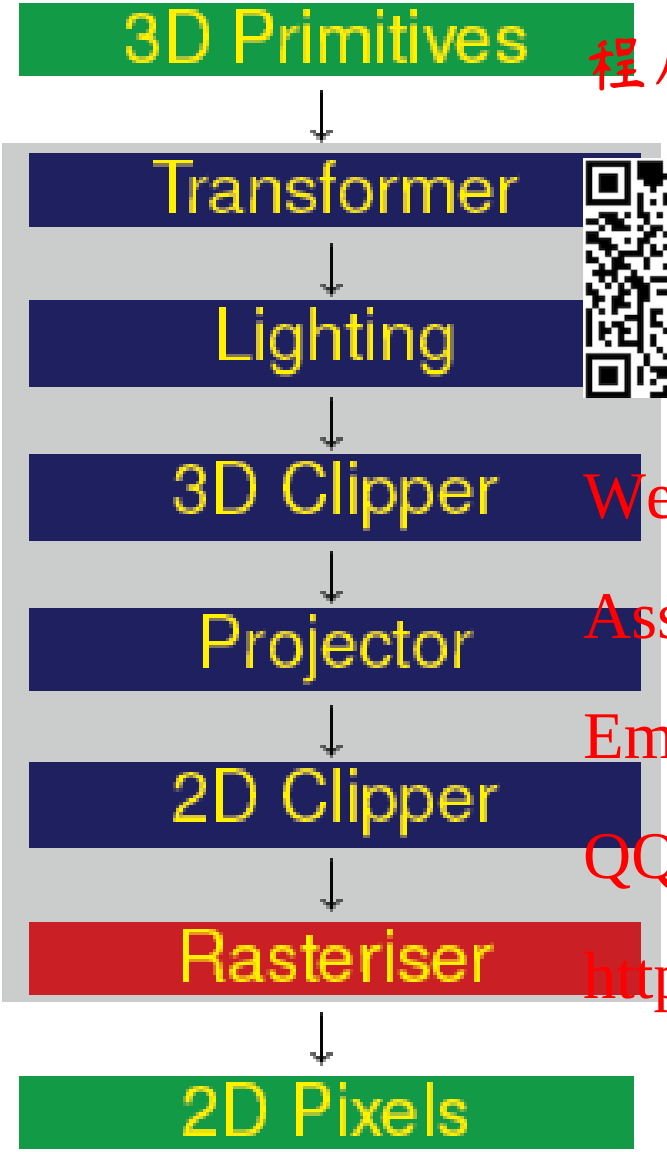
- Project 3D primitives onto plane to give 2D shapes
- Projection matrix specifies type of projection
- Camera properties specify projection in detail

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?