

## 14

## Summary

**R. Mukundan** ([mukundan@canterbury.ac.nz](mailto:mukundan@canterbury.ac.nz))

Department of Computer Science and Software Engineering  
University of Canterbury, New Zealand.



# Final Exam

- Online exam via Learn.
  - 15<sup>th</sup> June 2020, 9am. Duration: 2 Hours
- Max Marks: 50 (Contribution to the final grade: 50%)
- Answers manually graded
- The exam may be available for a set period of time from the start time. Once you start the exam, the duration will commence.

- A calculator will be required.
- Preparation:

Note: Past year's exams were of 3 hours duration


 [Lec13\\_WebGL.pdf](#)


 [WebGL-Files.zip](#)

## 14. Summary

 [Summary.pdf](#)

 [Exam Paper 2017](#)

 [Exam Paper 2018](#)

 [Exam Paper 2019](#)

# Equations Provided in Exam

- The following formulae will be provided if required for answering any of the questions:
  - Vector cross product
  - Rotation matrices
  - Shadow transformation matrix
  - Equation for  $t$  at the point of intersection between
    - a ray and a plane
    - a ray and a sphere
  - Equation of the reflected ray
  - Bernstein polynomials for generating Bezier curves.



# Equations **NOT** Provided in Exam

- A vector from one point to another
- Vector dot product, computation of angle
- Magnitude of a vector, normalization of a vector.
- Linear interpolation equation
- Homogeneous to Cartesian coordinate conversion
- Translation, scale transformation matrices
- Ambient, diffuse and specular reflections from a polygonal vertex
- Half-way vector used in computing specular reflections
- Equation of a ray

# 1 OpenGL Basics

You may be asked to write small OpenGL code segments (see past year exams) containing some basic functions. You must be able to use any of the following functions with relevant parameters in the code:

`glBegin()`, `glEnd()`

`glVertex3f()`, `glVertex4f()`

`glTexCoord2f()`

`glNormal3f()`

**Model definitions**

`glPushMatrix()`, `glPopMatrix()`

`glTranslatef()`, `glRotatef()`,

`glScalef()`

**Transformations**

# 1 OpenGL Basics

You must be familiar with the following OpenGL functions, and be able to explain their applications. You must be able to use any of the following functions with relevant parameters in a small code segment:

`gluLookAt()`

**View, Projection**

`glFrustum()`

`gluPerspective()`

`glLightfv()`

**Lighting**

`glMaterialfv()`

`glTexParameterf()`

# 1 OpenGL Basics

- Representation of R, G, B, C, M, Y, W, K colours using normalized floating point values.
- Right-handed reference frame for Cartesian coordinates.
- Camera fixed coordinate frame and view frustum

## **Not important:**

- Event processing in OpenGL

## 2 Transformations

- Sequence of transformations
  - Order in which OpenGL transformation functions are called.
  - Order in which transformation matrices are multiplied.
- Rotations about pivot points
- OpenGL transformation stack
  - Specifying transformations inside nested `glPushMatrix()` – `glPopMatrix()` blocks.
- General structure of transformation matrices:
  - Identifying transformations from a given matrix





## 3 Illumination Model

- OpenGL illumination model
  - Light-material interaction
  - Ambient, diffuse, specular reflections and their properties
- Spotlights
- Computation of the surface normal vector



## 4 Texture Mapping

- Assigning texture coordinates to vertices (see past year exam papers)
- Texture wrap modes
- Minification and magnification filtering (concepts only)
- Mipmaps (general concepts)



## 5 Object Modelling

- Extruded surfaces
  - The process of constructing a surface from a base polygon
- Triangle strips and quad strips
- Surfaces of revolution
  - The process of constructing a surface from a base curve
- You will not be asked to write any OpenGL code for constructing sweep surfaces using rotational transforms.

## 6 Mathematical Aspects

- Homogeneous coordinates
- Vector operations
  - Dot products, Orthogonality of vectors, Angle between vectors
  - Cross products, Surface normal computation
  - Projection of a vector
- Matrices
  - Transformation matrices
- Linear interpolation
- You should be able to multiply a vector by a transformation matrix.



## 7 Mathematics of Lighting and Viewing

- The lighting equation
  - Vectors used in the lighting equation
  - Ambient, diffuse and specular terms
  - Half-way vector
  - spotlight

# 7 Mathematics of Lighting and Viewing

- The eye coordinate frame.

`gluLookAt()`

- Changes in the display with `gluLookAt` parameters
- The view transformation matrix and the projection matrices will not be required for answering any questions.
- View volumes
  - `glFrustum(L, R, B, T, N, F)`
  - `gluPerspective(aspect, fov, N, F)`
  - Conversions between parameters of the above two frustum definitions
- Eye coordinates and clip coordinates

## 8 Ray Tracing

- Global illumination models
- Ray definition (parametric equation)
- Equations for computing reflected rays, ray-plane intersections and ray-sphere intersections will be given.
  - You must be able to compute the points of intersection on a plane or a sphere using the equations.
- Equations of planes and spheres will also be given.
- The following topics are **not** covered in the exam
  - Equations of cones, cylinders etc.
  - Anti-aliasing
  - Texturing in a ray tracing application

## 9 OpenGL-4

- Vertex and Fragment shaders (General concepts, structure, applications)
- Vertex buffer objects (General concepts)
- You are not required to remember any OpenGL-4, GLM functions.
- Mechanisms for passing values between application and shaders, and between shaders.
- Typical outputs of a vector and a fragment shader
- You may be given a shader code and asked to describe the implemented method.





# 10 Parametric Domains, Bezier Surfaces

- Quad and triangle domains
- Barycentric coordinates
- Bernstein polynomials of degree 2 and 3 will be given.
- Quadratic and cubic Bezier curves
- General properties
- Bi-cubic Bezier surfaces

# 11 Tessellation Shader

- GL\_PATCHES (general properties)
- Tessellation Coordinates
- Tessellation levels
- Tessellation control shader and evaluation shader
  - Usefulness and main applications
  - Built-in variable `gl_in`
- You will not be asked to write any shader code
- You may be given a shader code and asked to describe the implemented method.

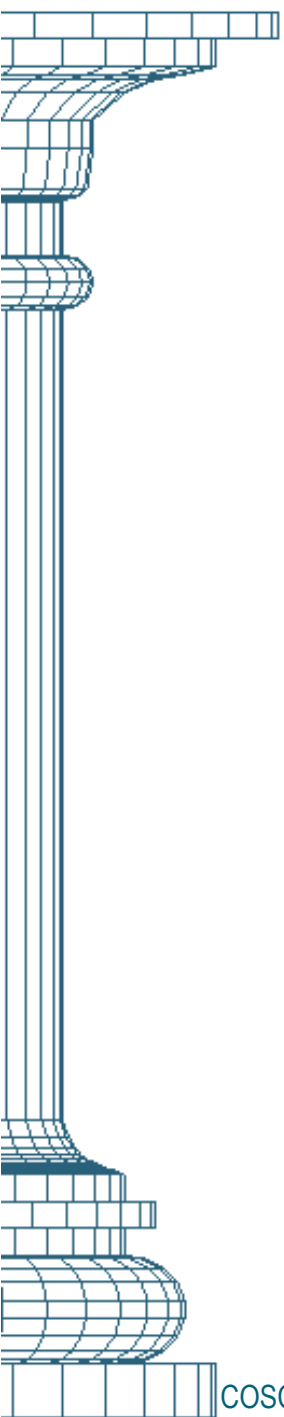
## 12 Geometry Shader

- General concepts – inputs, capabilities
- Usefulness of a geometry shader (sample applications)
- GL\_TRIANGLES\_ADJACENCY primitive
- You will not be asked to write any shader code.
- You may be given a shader code and asked to describe the implemented method.



# 13 WebGL

- Not included in the exam!



Where to next?

# COSC422 Advanced Computer Graphics

COSC422 covers a range of topics that find applications in real-time rendering and animation.

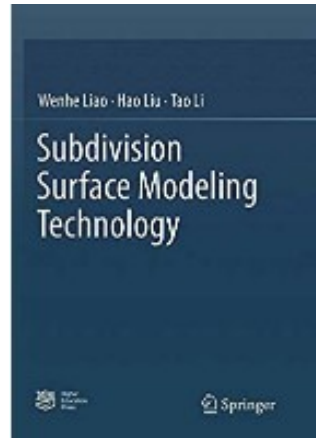
Not offered in 2020.

- Applications of shader stages: Non-photorealistic rendering, Particle systems, Point sprites, Transformation feedback.
- Image Based Rendering (IBR): Imposter rendering, Projective texturing, Shadow mapping
- Hierarchical transformations: Scene graphs
- Motion Data and Skeletal Animation: 3D Character animation
- Quaternions and Inverse Kinematics
- Advanced modelling: Mesh processing algorithms
- Advanced Illumination: Ambient Occlusion, Microfacet models

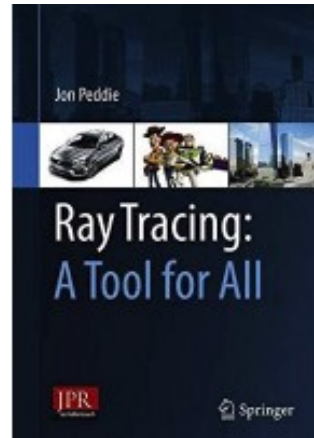
# The present state of CG



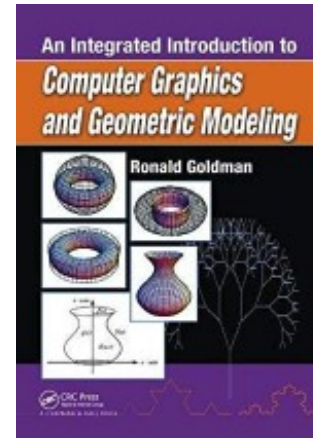
2019



2019



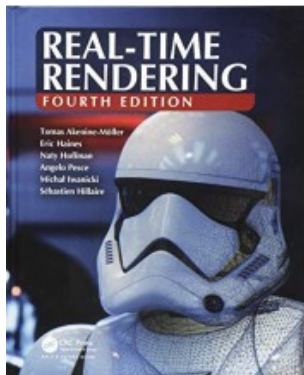
2019



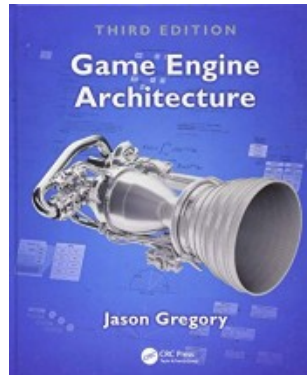
2019



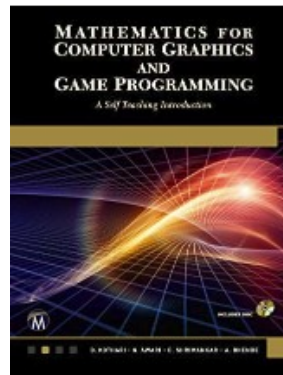
2019



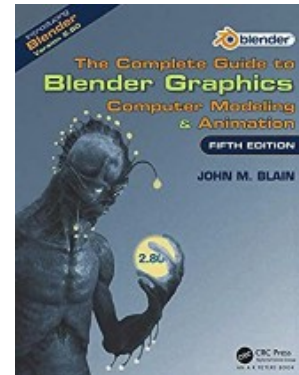
2018



2019



2019



2019



2019



# RTX. IT'S ON.

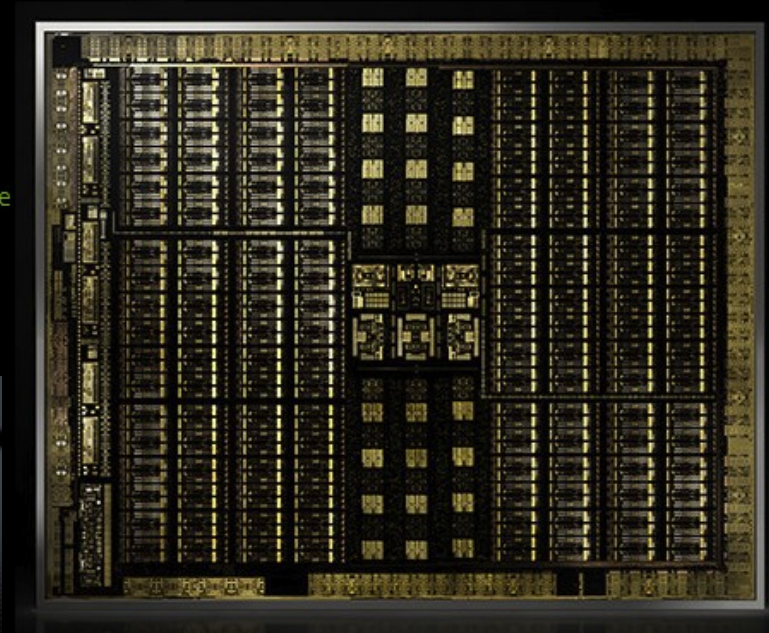
## NVIDIA TURING

The revolutionary NVIDIA Turing™ architecture, combined with our all new GeForce RTX™ platform, fuses together real-time ray tracing, artificial intelligence, and programmable shading to give you a whole new way to experience games.

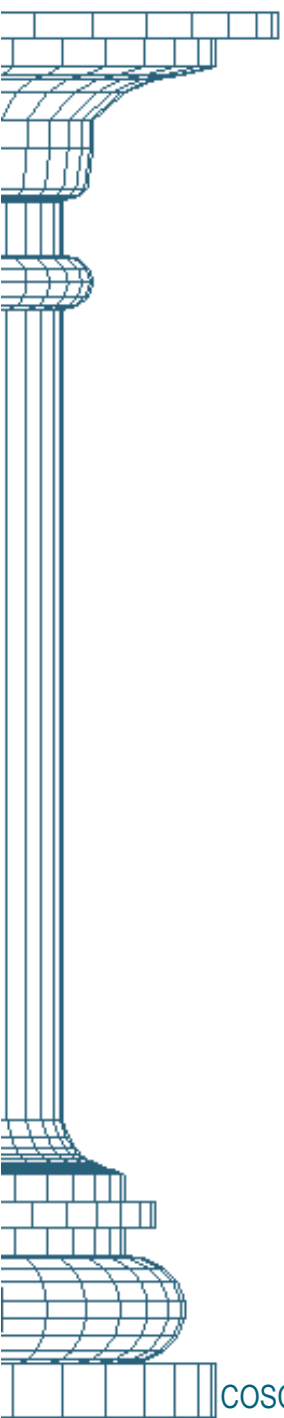
### TURING FEATURES

#### RT CORES

Dedicated ray tracing hardware enables fast real-time ray tracing with physically accurate shadows, reflections, refractions, and global illumination.







# Good Luck!