COMPUTER GRAPHICS
T.A.: Gur Harary
- Email: gur@tx.technion.ac.il
- Office: Meyer 1051
- Office hours: Wednesday 15:30-17:30

OSG consultant: Daniel Barsky
- Email: danielbr@tx.technion.ac.il
Administrative Information

- Sites:
  - http://webee.technion.ac.il/~cgcourse/
  - http://moodle.technion.ac.il/
Grades

- Final exam (55%)
- Homework (45%)
  - Dry exercises 10%
  - Wet exercises 35%
- Submission in PAIRS
- Basic programming skills are desirable
Postponements?

No
Postponements?

No
Postponements?

No
Tutorials schedule

- OSG – 4 classes
- Theory – 6 classes
- Solution of problems – 4 classes
Books

- **Computer Graphics with OpenGL**
  - Donald Hearn and M. Pauline Baker

- **Computer Graphics: Principles and Practice**
  - James Foley, Andries van Dam, Steven Feiner and John Hughes

- **Fundamentals of Computer Graphics**
  - Peter Shirley
Books - OSG

- OSG Reference Manual v2.2
  - [http://www.openscenegraph.org/documentation/OpenSceneGraphReferenceDocs/index.html](http://www.openscenegraph.org/documentation/OpenSceneGraphReferenceDocs/index.html)

- OSG Quick Start Guide
  - [http://osgbooks.com/books/osg_qs.html](http://osgbooks.com/books/osg_qs.html)

- Wiki
Project?

http://webee.technion.ac.il/labs/cgm/
OBJECT ORIENTED PROGRAMMING
OOP Concepts

- Modular code.
  - Partitioning the program so that the data is hidden within modules.

- Well defined interface.
  - Defines functionality. Each module has specific functionality.
  - The user is unaware of internal implementation, which can be changed without affecting the external interface.
OOP Concepts

- Data abstraction.
  - Internal data structure is hidden.
  - It is better, if only the module for which certain data belongs, can change it directly.

- Reuse.
  - Classes and inheritance.
Objects

- **State** - object's data
  - Better if controlled only by the object’s methods (data abstraction).

- **Behavior** - object’s methods
  - Part of the interface, if can be invoked by other objects.
  - Part of implementation, if can be invoked only by the object itself.

- **Identity** - unique name
  - A reference is given when object created
Classes

- **Class**
  - Abstraction of object - structure for object definition.
  - Represents common characteristics of all its objects.
  - A new object is created, when a new instance of the class is created.
class Sphere {
    public:
    Sphere(); // Constructor with default values
    Sphere(float r, float c[]); // Constructor with values
    ~Sphere(); // Destructor
    void render(); // renders the sphere
Class definition example

Boolean pick(int x, int y);  // picks the sphere
void SetRadius(float r);      // set the radius
void SetCenter(float c[]);    // set the center
float GetRadius();            // get radius
float[] GetCenter();          // get center

private:
    float radius;                // radius of the sphere
    float center[3];             // center of the sphere
};
Creation of a new instance

Sphere *mySphere;  // pointer to Sphere Object
float center[3] = { 0.0, 0.0, 0.0};
mySphere = new Sphere(3.0,center);  // Create new instance
mySphere->render();  // renders the sphere
Inheritance

class hollowSphere : public Sphere {
    public:
        void showEquator();
        // show equator during render
        float thickness;
}

OPEN SCENE GRAPH
Open Scene Graph

- Graphics toolkit for the development of high-performance graphics applications such as flight simulators, games, virtual reality and scientific visualization.
- Providing an object-oriented framework on top of OpenGL.
Open Scene Graph

3D Application

Scene graph middleware (OpenSceneGraph)

Low-level rendering API (OpenGL)
Open Scene Graph
Open Scene Graph
Open Scene Graph
Scene graph

- A hierarchical tree data structure that organizes spatial data for efficient rendering
Scene graph

- The graph consist the following:
  - Root
  - Nodes
  - Children/parents
  - Branches / Edges
  - Leaves
Scene graph

- Leaves contain geometry
- Other nodes are groups, transformations, and properties
Scene graph
Scene graph

Scene

Road

Translation

Tricycle
Scene graph

Scene
- Road
- Translation
  - Tricycle
    - Translation
      - Front
        - Translation
          - Wheel
        - Handle bars
      - Translation
        - Seat
      - Translation
        - Back
          - Translation
            - Wheel
          - Translation
            - Wheel
          - Translation
            - Wheel
Scene graph

- Scene
  - Road
  - Translation
    - Tricycle
      - Translation
        - Front
          - Translation
            - Wheel
        - Handle bars
        - Seat
      - Back
        - Translation
          - Wheel
Scene graph

Scene

Road
Translation
Tricycle

Translation
Front
Translation
Handle bars
Scale
Wheel

Translation
Seat
Translation
Back
Translation
OSG - Naming Conventions

- Namespaces
  - osg, osgSim, osgFX

- Classes
  - MatrixTransform, NodeVisitor, Optimizer

- Class methods
  - addDrawable(),getNumChildren()

- Class member variables
OSG - Naming Conventions

- Templates
  - `ref_ptr<>`, `graph_array<>`, `observer_ptr<>`

- Statics
  - `s_applicationUsage`, `s_ArrayNames()`

- Globals
  - `g_NotifyLevel`, `g_readerWriter_BMP_Proxy`
Scene Graph Classes

- Node – base class
- Group – base class with children
- Geode – Geometry Node class
  - osg::Drawable
- MatrixTransform – contains a trans. matrix
- LOD – Level of Details
- Switch – enable/disable children
Geometry Classes

- **Drawable** – store geometric data
  - **ShapeDrawable** (Drawable)
    - predefined shapes
  - **Geometry** (Drawable)
    - Vertex array, texture coordinates, color, normal.
- **PrimitiveSet**
  - Triangles, quads ...
- **Array classes**
  - Vertices, colors, normals, texture coordinates
- **Vector classes**
Installing OSG

- **Download for VS8 (2005)**
    - openscenegraph-all-2.8.2-win32-x86-vc8osp1-Debug.zip

- **Download for VS9 (2008)**
    - openscenegraph-all-2.8.2-win32-x86-vc90-Debug.zip

- **Download for VS10 (2010)**
    - OpenSceneGraph-3.0.1-VS10.0.30319-x86-debug-12741.7z
Installing OSG

- Unzip the zip file to C:\OSG

- Some notes:
  - Make sure you downloaded a zip file that is consistent with your VS version.
  - 64 bit computers can use x86 binaries – use one of the links from the previous slide.
Create new project

![Visual Studio Start Page](image)
Create new project

Choose Win32 console application and name
Create new project

- Download Class1.cpp from the course page
- Remove original main and add Class1.cpp
Include directories
Define LIBs

Additional Library Directories
Specifies one or more additional paths to search for libraries; configuration specific; use semi-colon delimiter.
Define LIBs
Define LIBs

- osgXXXd.lib files are for debug mode
- For release mode use osgXXX.lib files (without the “d”)

Compile and run

- Build – F7, Run – Ctrl + F5, Debug – F5
HELP!!!

- XXX.dll not found

- Correct solution:
  - Go to “My Computer” → System Properties → Advanced system settings → Environment variables
  - Add C:\OSG\bin to system::PATH variable

- Another solution:
  - Copy all dlls from C:\OSG\bin to home project directory.
HELP!!!

- XXX.dll not found
- Correct solution:
HELP!!

- MSVCRTD.lib(crtexew.obj) : error LNK2019
- Solution:
  change the “SubSystem" in your Linker settings from "Windows" to "Console"