CS418
Discussion Section
(V)
MP2 & Picking

Presented by : Wei-Wen Feng
2/25/2009

MP2 : Flight Simulator

- Due on March 2 (Mon)
  - Terrain Texturing / Lighting
  - Camera Control (Flight Simulator)
  - Bonus (20%)
    - Multiple Object rendering (Model Transformation)
    - Object picking/control

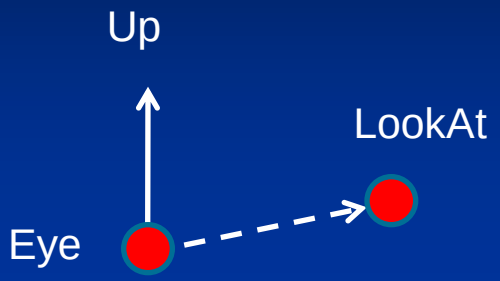
Flight Control

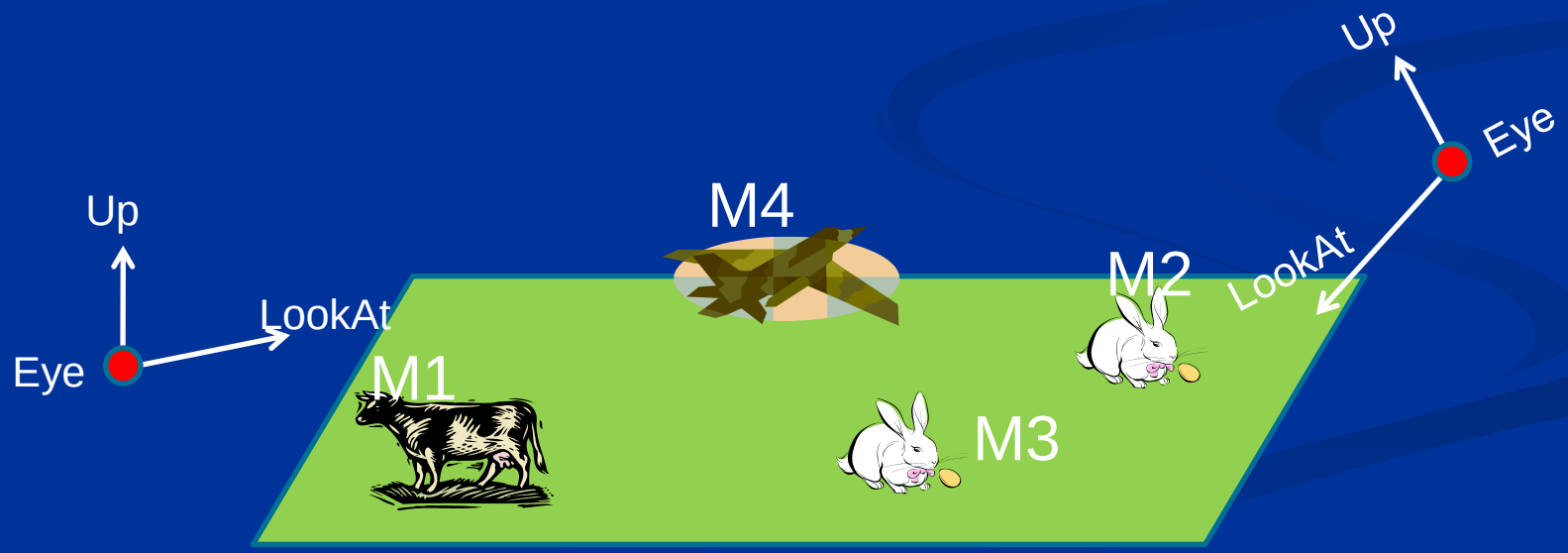
- Move your camera based on user keyboard input
- Intuitive Way
  - Update gluLookAt parameters
  - Easier to think, but more work (keep track your own matrix transformation)
  - Recommended if you don't want to mess with OpenGL transformation.

- Less intuitive Way
  - Update OpenGL transformation
  - Easier to implement, but difficult to make it correct
  - Recommended if you are absolutely sure what to do.

Flight Control

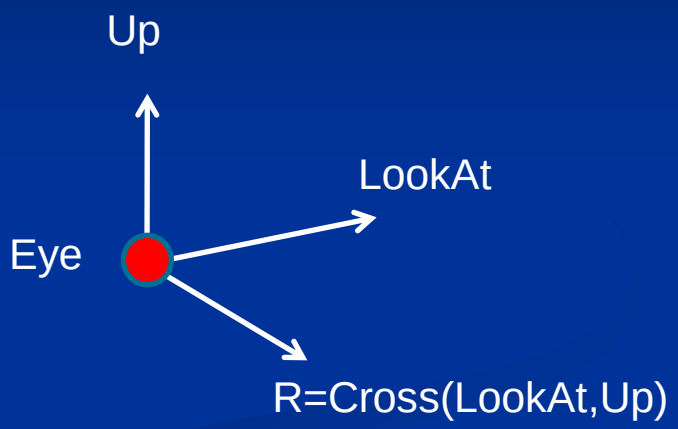
- gluLookAt way :
  - Eye position
  - Look At point (direction)
  - Up vector





Flight Control

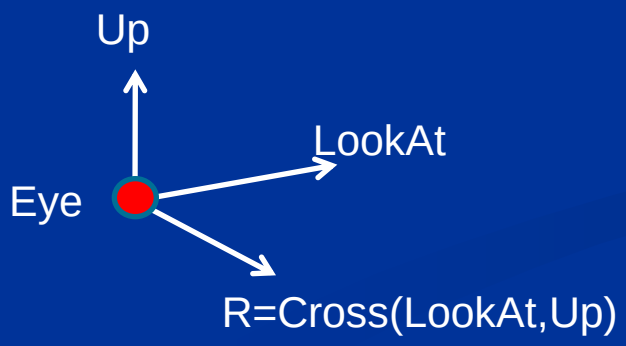
- Initilize Eye position and Up,LookAt, R vector.



- Move forward :
  - Offset Eye position in LookAt direction
- Tilt Up/Down
  - Rotate LookAt,Up about R axis.
- Turn Left/Right
  - Rotate LookAt,R about Up axis.

Flight Control

- Every time you press arrow keys, update Up,LookAt, R vector accordingly.
- Every time period (Ex : 1/30 sec), move Eye position.
- In display function, set look at function :
  - gluLookAt(Eye, Eye+LookAt, Up);



Flight Control

- Arrow Key Called-back function
  - glutSpecialFunc instead of glutKeyboardFunc
  - Refer to OpenGL doc. for its parameters.
- Reset OpenGL matrix before calling gluLookAt.
- You may use the formula in lecture slides to generate rotation matrix (axis-angle).

Flight Control

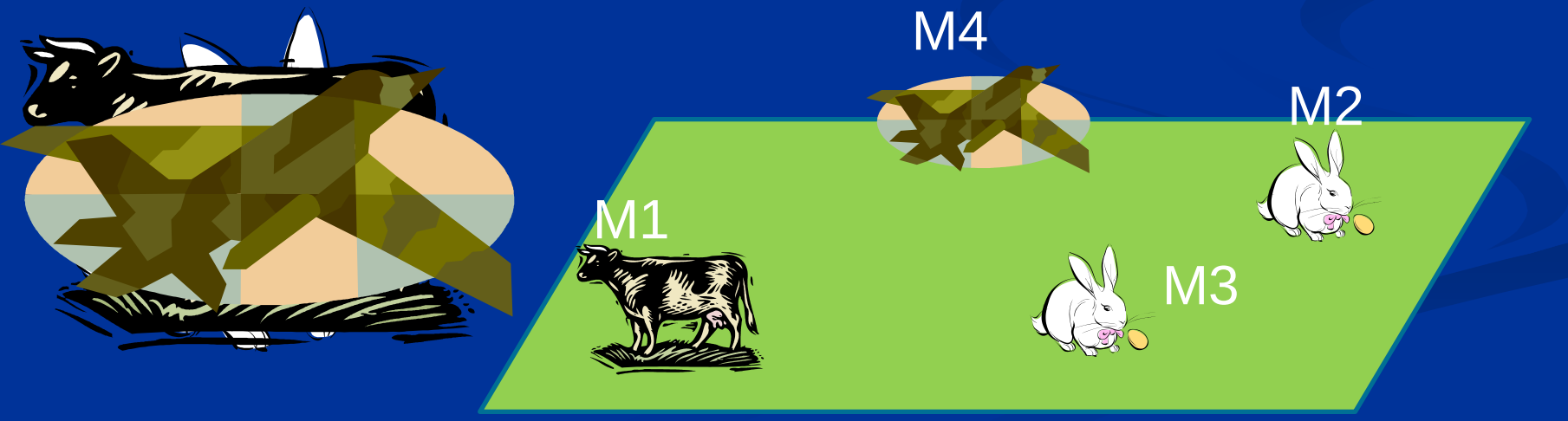
- Less Intuitive way
  - Moving camera is equivalent to moving every object in the world towards a stationary camera
  - Using a fixed gluLookAt, but call OpenGL transformation command instead.
  - Where should you put glTranslate/glRotate to simulate a flight simulator ?
    - Before or after gluLookAt ?
    - Pre-multiply or Post-multiply ?

MP2 Bonus

- Object Rendering/Selection
  - Rendering multiple objects on the terrain
  - Let user choose which object to be animated
  - Should be performed by mouse-click
- Animate Object
  - Object should move along a velocity vector
- Orientation Manipulation
  - Change direction of velocity vector
  - Choose the UI yourself

Multiple Object Rendering

- Model Transformation
  - Specify scaling, translation for each object
  - Apply different transformation on each mesh
  - Utilize push/pop matrix to backup matrix state



Push/Pop Matrix

- glPushMatrix()
  - Create a copy of top matrix & push it into stack.
- glPopMatrix()
  - Remove the top matrix from stack

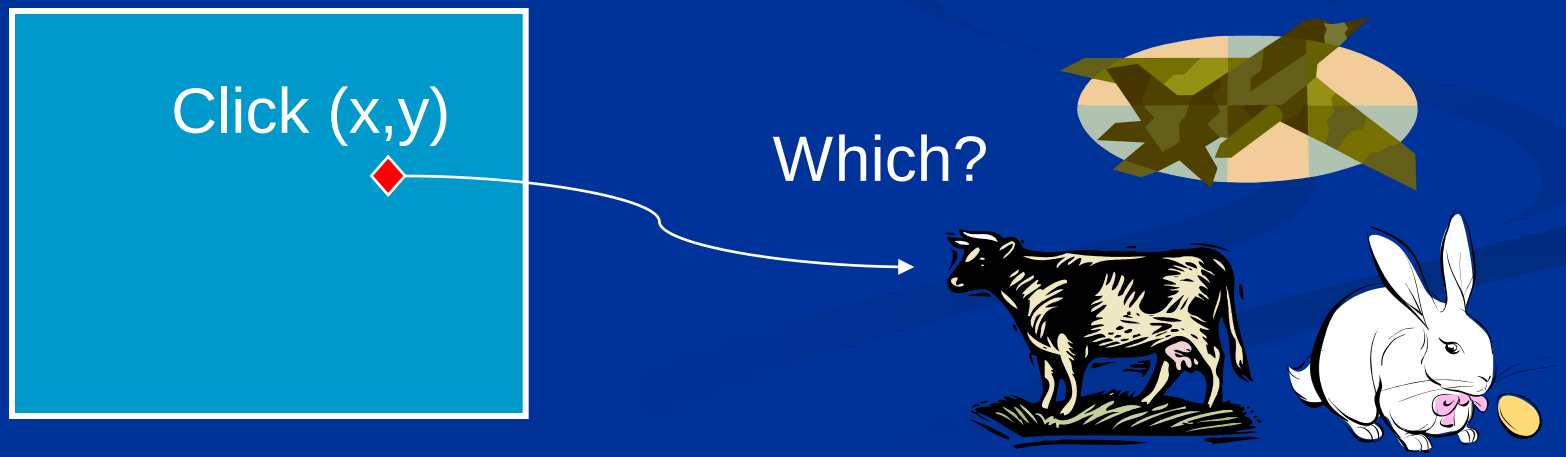
Multiple Object Rendering

Drawing each object :

```
glPushMatrix();
glTranslate()
glScale()
glBegin()
....
glEnd()
glPopMatrix();
```

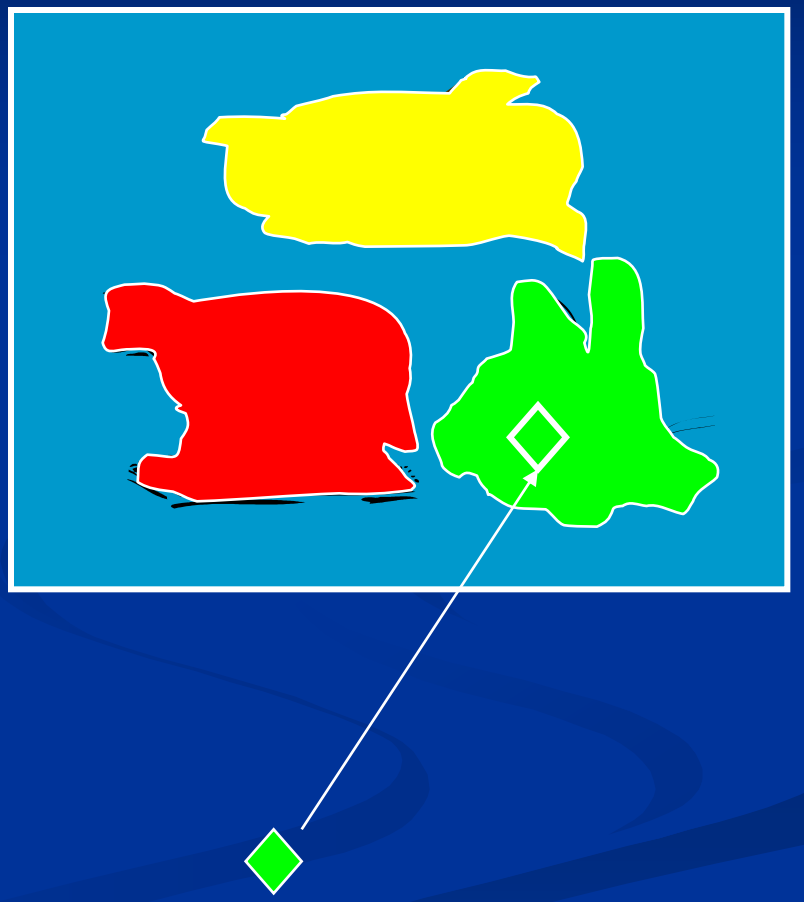
Object Selection

- Picking an object on the screen based on mouse click.
- Problem :
  - How to relate **2D** Mouse position to **3D** objects ?



Object Selection

- First Idea : Color coding
  - Disable Lighting
  - Draw objects in different colors
  - Check pixel color to identify the object
  - Apply “glReadPixels” for each mouse click to obtain pixel values.

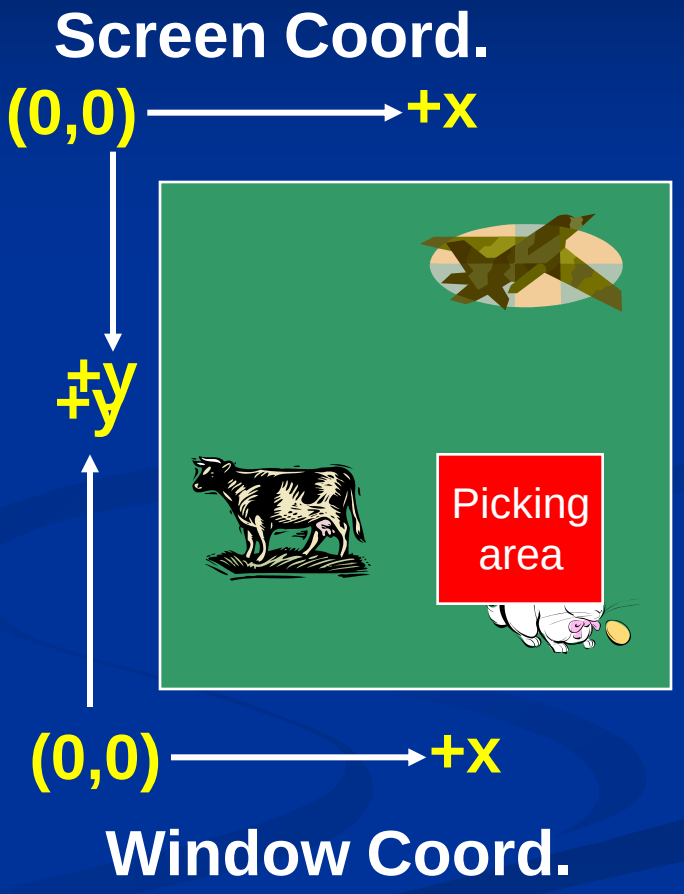


Object Selection

- OpenGL Picking : Similar idea
  - Define a screen area for picking
  - Set each object with a "Name" (ID)
  - Draw Objects in "Select Mode".
  - OpenGL identify "Name" of objects inside the picking region.

Object Selection

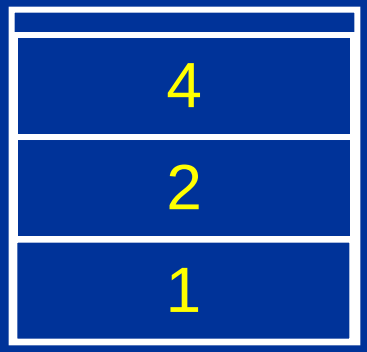
- Picking Matrix
  - Define a screen drawing area for picking
  - Set projection matrix with smaller screen area.
  - gluPickMatrix(**x,y,w,h,viewport**)
    - x,y : **window** coordinates
    - w,h : picking region
    - Viewport : screen size



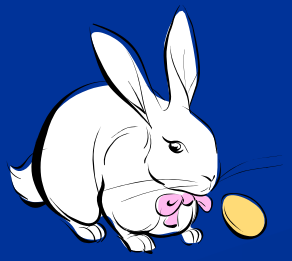
Object Selection

- Name Stack
  - OpenGL maintains a stack of "Names"
  - Each name is a number/ID
  - When an object is drawn, all "Names" in current stack are used as IDs.

Name Stack



Name : {1}



Name : {1,2,4}

Object Selection

- Hit Records
  - Need to set up a **"GLuint"** buffer to have results returned to you

```
glSelectBuffer( size, buffer )
```

  - Depth : 0~2^32
  - Multiple records might intersect the picking region.
  - Choose the record with smallest depth. (Closest to you)

#Names
Min Depth
Max Depth
Name1, Name2, …

#Name = 1
10
300
"10"

Records1

#Names = 2
30
60
" 1" " 2"

Records2

Object Selection

- OpenGL Picking example

Init Selection :

```
glSelectBuffer(BUFSIZE,selectBuf);
glRenderMode(GL_SELECT);
glInitNames();
```

Define Pick Region :

```
glGetIntegerv(GL_VIEWPORT,vp);
glMatrixMode(GL_PROJECTION);
glPushMatrix();  // backup projection
glLoadIdentity();
gluPickMatrix(mouseX,vp[3]-mouseY,5,5,vp);
gluPerspective(…)
```

Drawing :

```
glPushName(1)
drawObj1()
glPopName()

glPushName(2)
drawObj2()
glPopName()
```

After Picking :

```
glMatrixMode(GL_PROJECTION);
glPopMatrix(); // restore projection

Hits = glRenderMode(GL_RENDER);

// process Hit records from selectBuf
```

Object Manipulation

- Once we select the object, we can animate that specific object.
- Object translational animation
  - Move object along a pre-defined direction.
  - Update its position periodically and redraw.
  - Change velocity based on UI.

Object Manipulation

- Animation Example

Init :
```
m_T = Vec3(0,0,0);
m_V = Vec3(0,0,1);
m_Speed = 1.0;
```

Timer :
```
m_T += m_V*m_Speed
```

Change Speed :
```
m_Speed ++ (or --)
```

Rendering :
```
glPushMatrix();
glTranslate(m_T);
glBegin();
....
glEnd();
glPopMatrix();
```

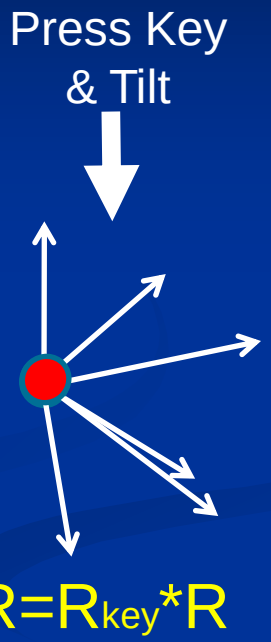
Object Manipulation

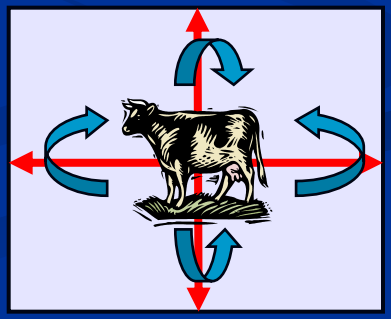
- Move object along a fixed direction is not enough.
- Rotate the object to change its moving direction.

- Problem :
  - What kind of UI to use ?
    - Keyboard ? Mouse ?
  - How to rotate its moving direction ?

Object Manipulation

- Choice of UI ?
  - Key requirements : Must be able to orient object to any directions.
  - Rotation about only one fixed axis won't get full credit.
- Keyboard
  - Change the angle/axis of rotation based on key-press.
  - Analogy to flight simulator → 3rd Person view control.
  - Keep track a total accumulated rotations.
- Mouse
  - Make use of mouse movement to define a rotation.
  - Sounds familiar ? → Euler's Angle, Arcball, etc.





Object Manipulation

- How to re-orient object ?
- Maintain a model rotation matrix.
  - Change its values based on user input.
  - Object also needs to be rotated accordingly.
  - Apply the rotation for both
    - Velocity vector
    - Object model transformation

Object Manipulation

- Re-orientation Example

Init :
m_Rot = Identity
m_InitV = m_V = Vec3(0,0,1)

UI Input :
Determine fAngle,vAxis
Mat4 newR = (fAngle,vAxis);
m_Rot = newR*m_Rot;

Update Orientation
m_V = m_Rot*m_InitV;
m_T += m_V*m_Speed

Rendering :
glPushMatrix();
glTranslate(m_T);
glMultMatrix(m_Rot);
glBegin();
….
glEnd();
glPopMatrix();

Q&A