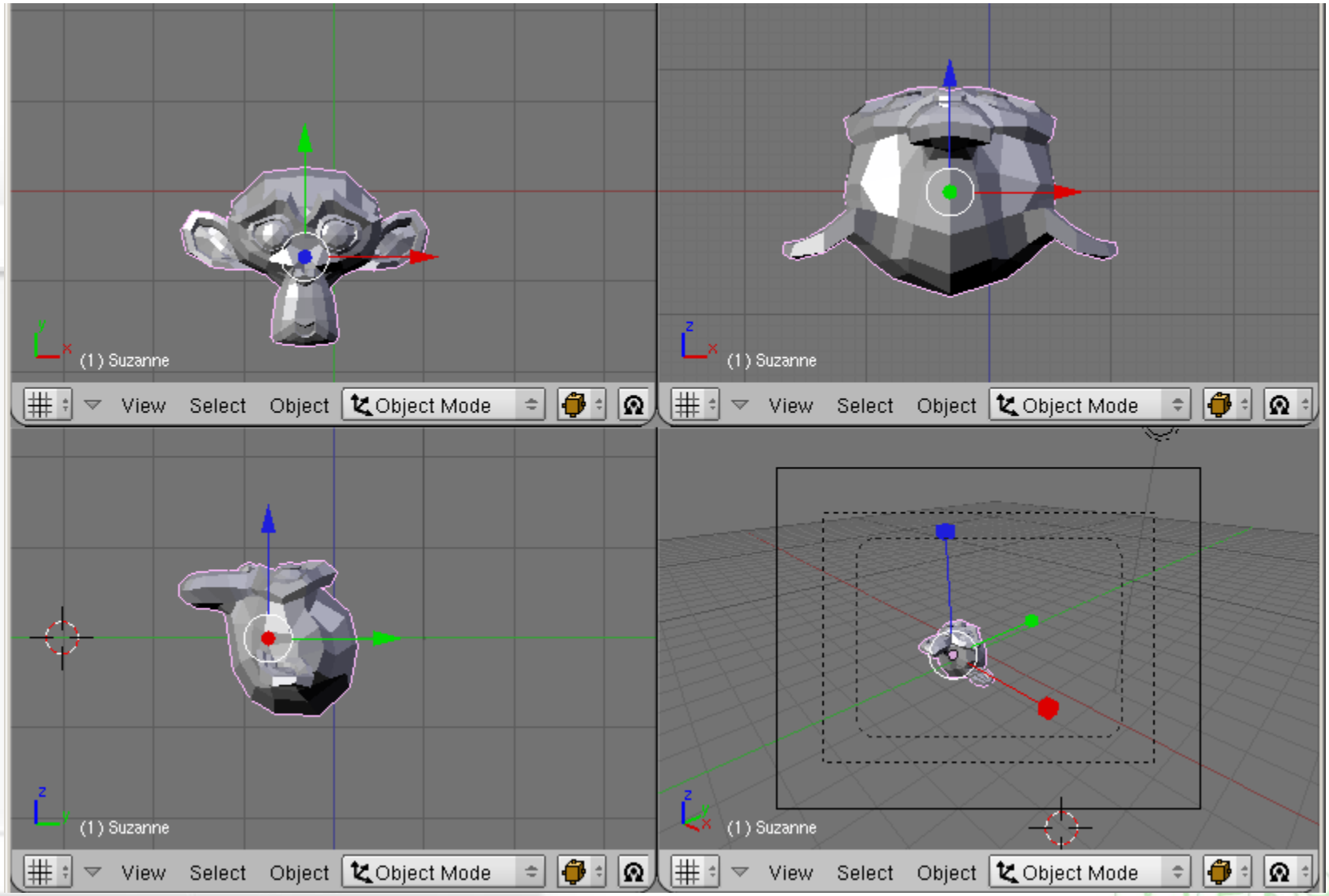
3D Game Programming camera

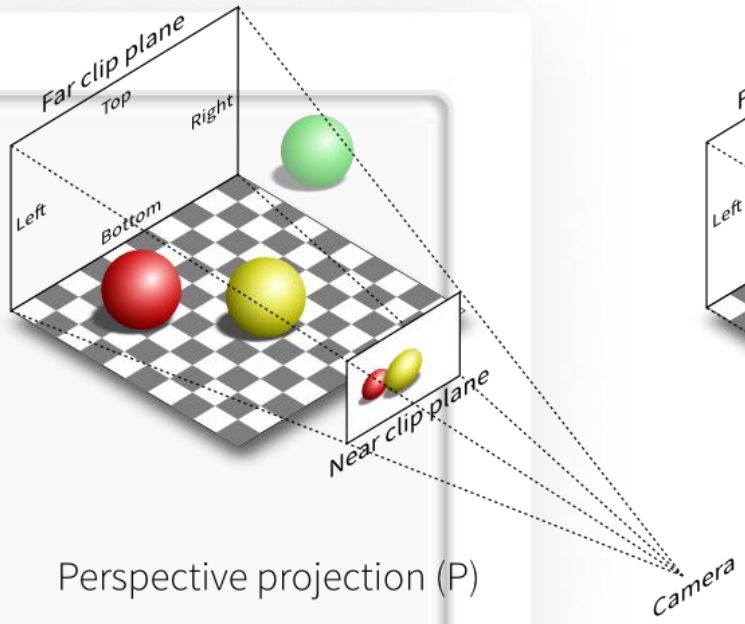
Ming-Te Chi
Department of Computer Science,
National Chengchi University

PROJECTION

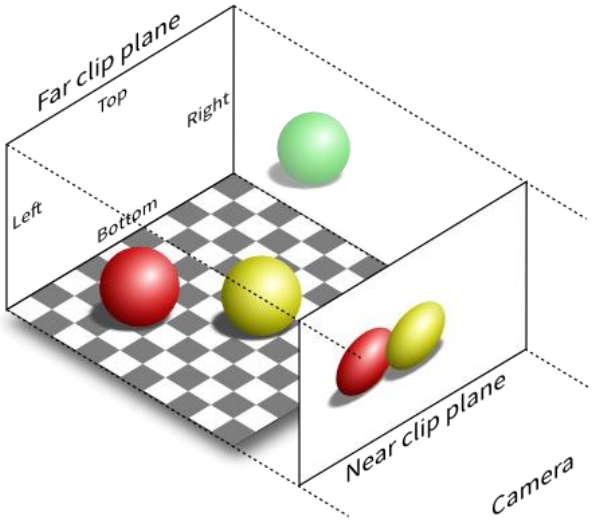
Multi-view



Projection Transformation



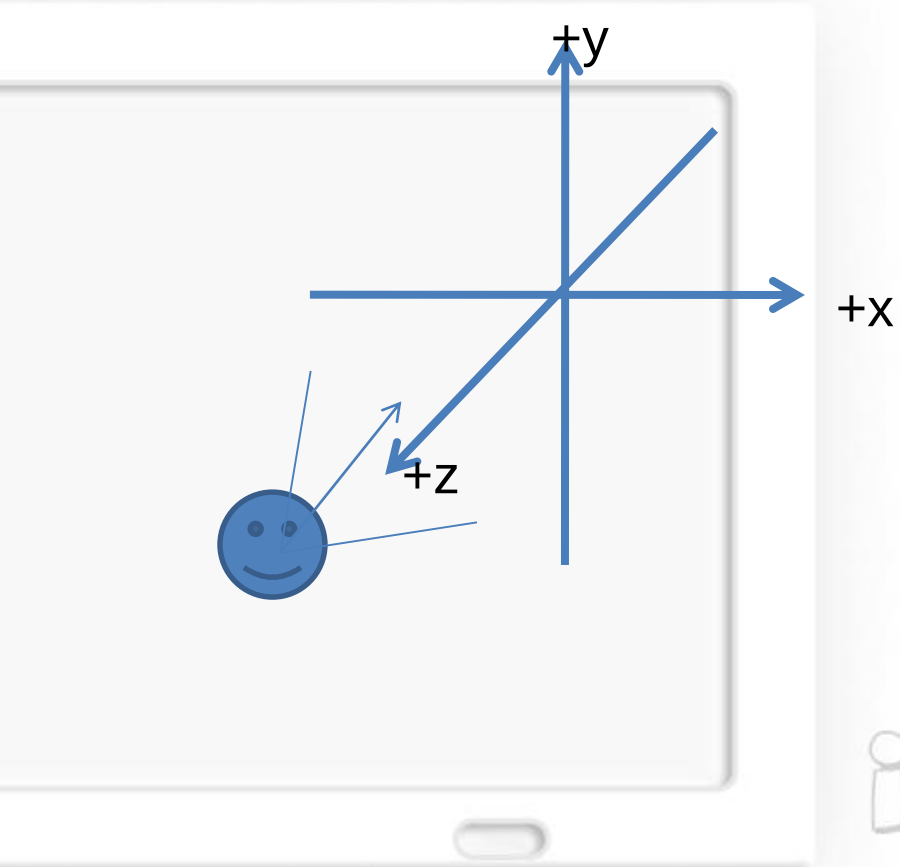
Perspective projection (P)



Orthographic projection (O)

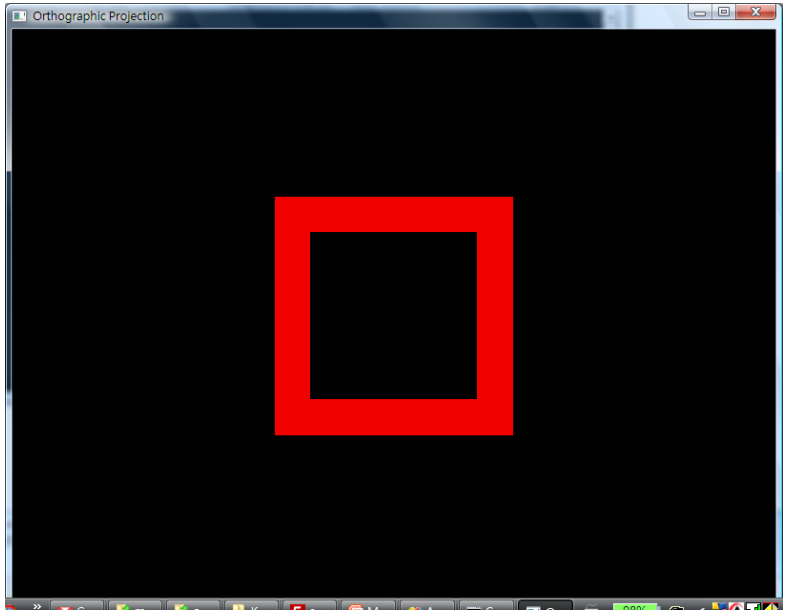
https://stackoverflow.com/questions/36573283/from-perspective-picture-to-orthographic-picture

Eye coordinates

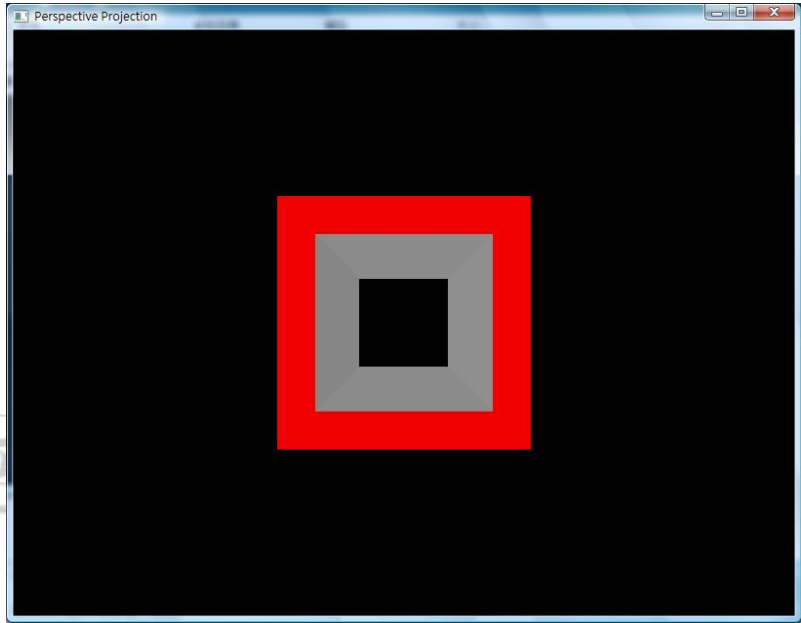


Projections

- Orthographic Projections

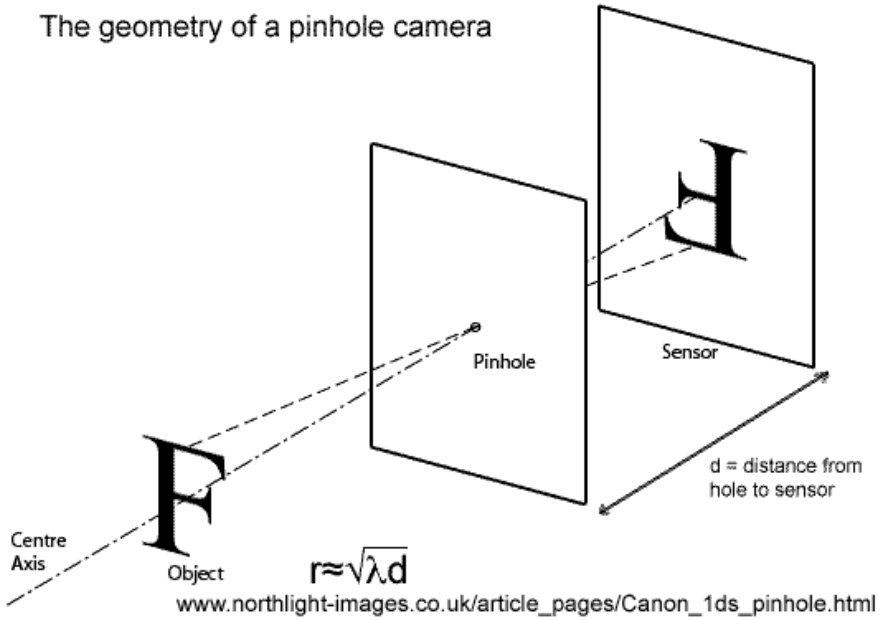


- Perspective Projections



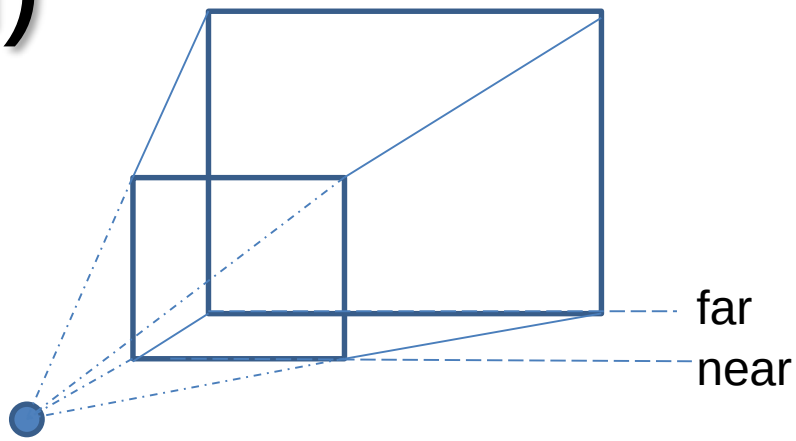
Pinhole camera

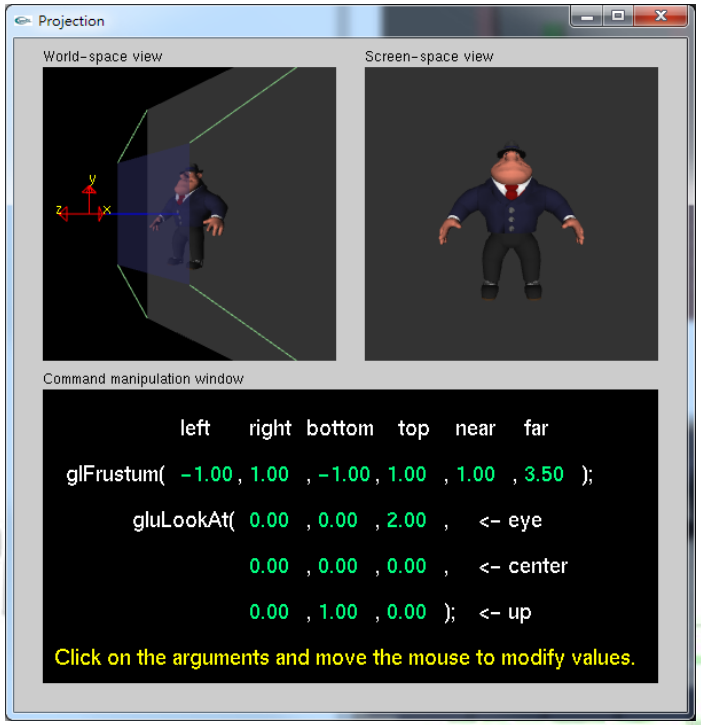
A pinhole camera is a simple camera without a lens and with a single small aperture – effectively a light-proof box with a small hole in one side.



Frustum(opengl)

```
void glFrustum(
    GLdouble  left,
    GLdouble  right,
    GLdouble  bottom,
    GLdouble  top,
    GLdouble  nearVal,
    GLdouble  farVal
);
```



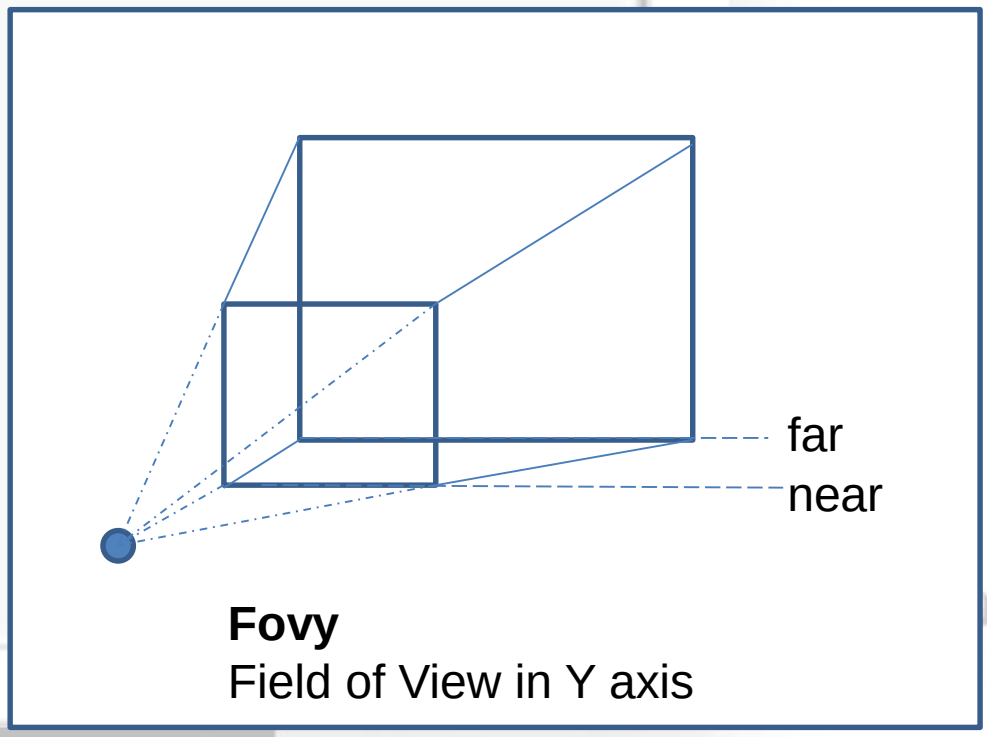


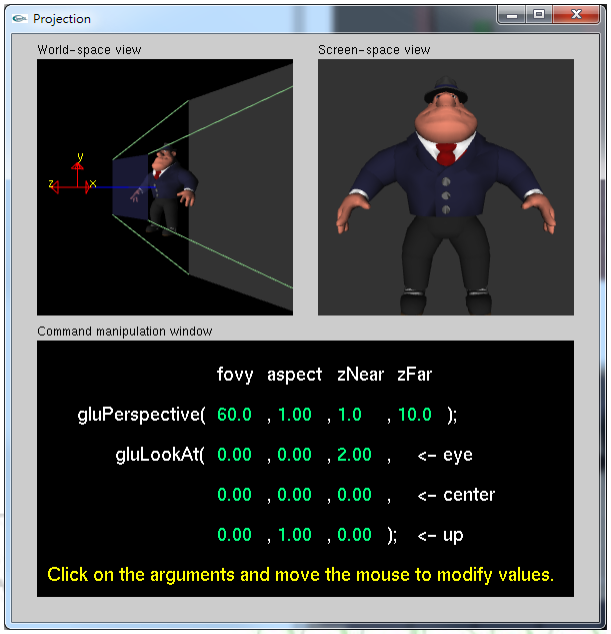
Perspective Projections(opengl)

// Produce the perspective projection

Perspective(45.0f, fAspect, 1.0, 400.0);

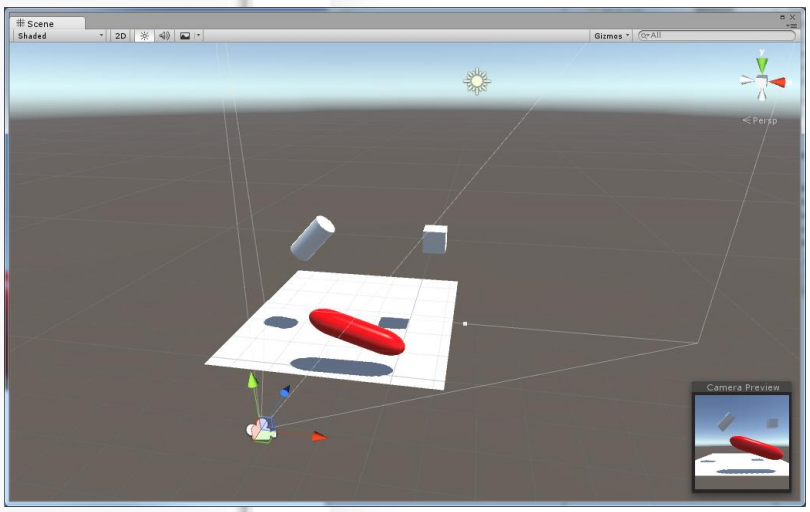
```
void gluPerspective(
GLdouble  fovy,
GLdouble  aspect,
GLdouble  zNear,
GLdouble  zFar);
```





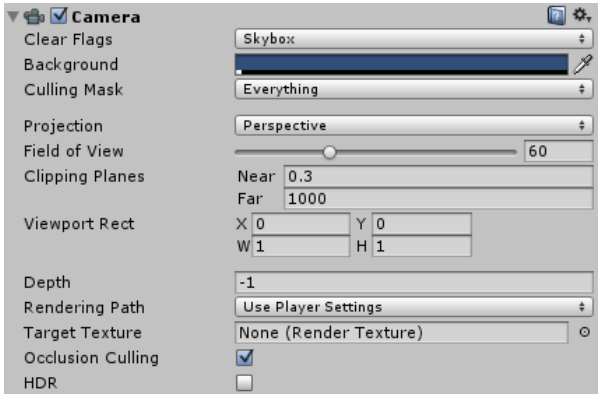
攝影機使用與設定

- Main Camera: 遊戲預設視角
  - 點擊場景中的攝影機，可顯示預覽畫面

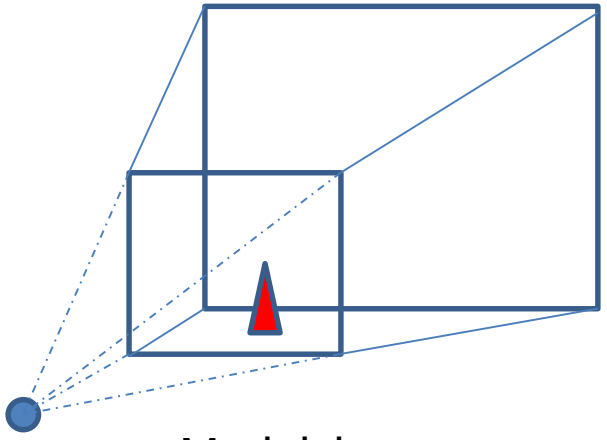


攝影機使用與設定

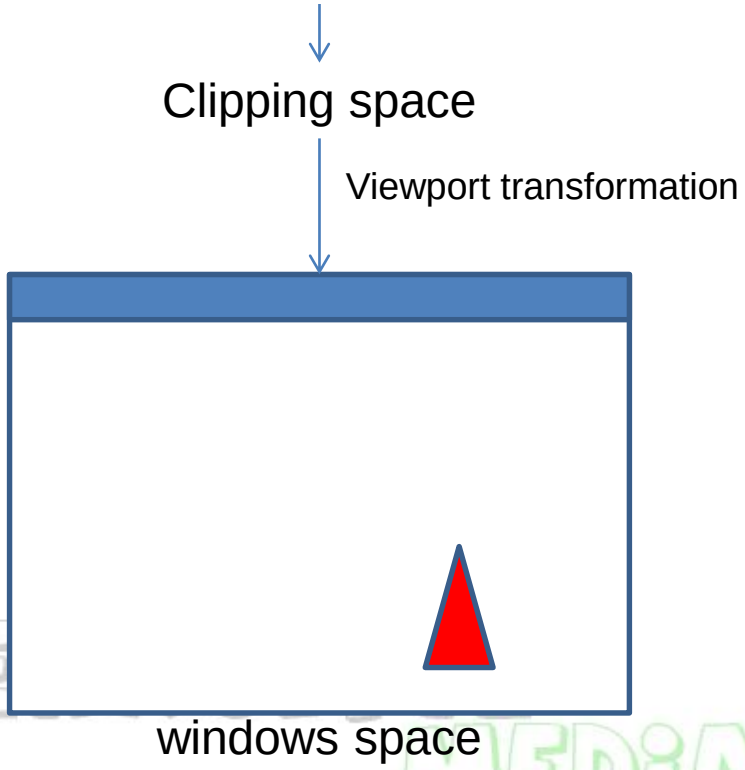
- 攝影機屬性
  - Background: 背景顏色
  - Projection:投影方式
    - Perspective: 透視投影
    - Orthographic: 正交投影
  - Field of View(FOV)
    - 可見視角範圍
  - Clipping Planes
    - 最近(Near)與最遠(Far)能看到的範圍



Viewport



Modelview space



GRAPHICAL PERSPECTIVE

First-Person perspective

- A graphical perspective rendered from the viewpoint of the player character

doom

Mirror's Edge

Second-Person Perspective

- In second person point of view, the action is shot from the perspective of a character that is not the protagonist.
- Let players feel as if they are actually the character they are controlling

Mario 64

Third-person shooter

- The player character is seen at a distance from a number of different possible perspective angles.
- the player character is visible on-screen, and the gameplay consists primarily of shooting.

Gear of war 3

Top-down perspective

- From God view

simcity

Star craft 2

CAMERAS AND ACTORS

CameraFollow

```
public GameObject player;
private Vector3 offset;

void start() {
    offset = transform.position –player.transform.position;
}
void LateUpdate() {
    transform.position = player.transform.position +
offset;
}
```

Vector3.Lerp

Linearly interpolates

public static Vector3 **Lerp**(Vector3 **a**, Vector3 **b**, float **t**);

$$out = (1 - t) * a + t * b$$

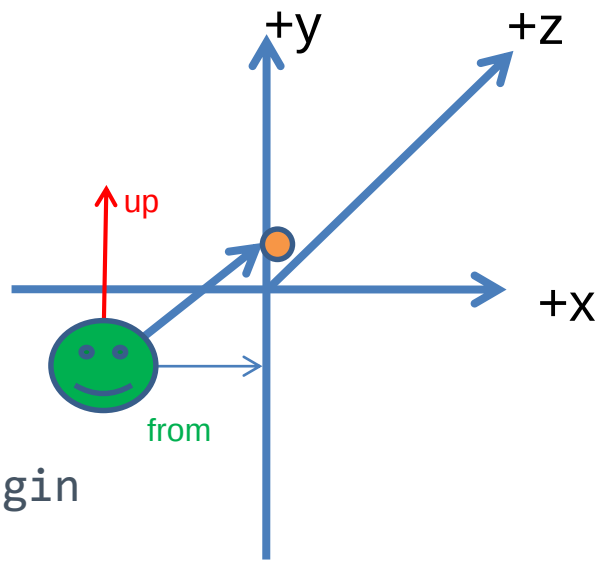
CameraFollow2

```
public GameObject player;
public  float smooth;
private Vector3 offset;
void start() {
    offset = transform.position –player.transform.position;
}
void LateUpdate() {
    Vector3 targetPos = player.transform.position +
offset;
    transform.position =  Vector3.Lerp
(transform.position, targetPos, smooth*Time.deltaTime);
}
```

Lookat

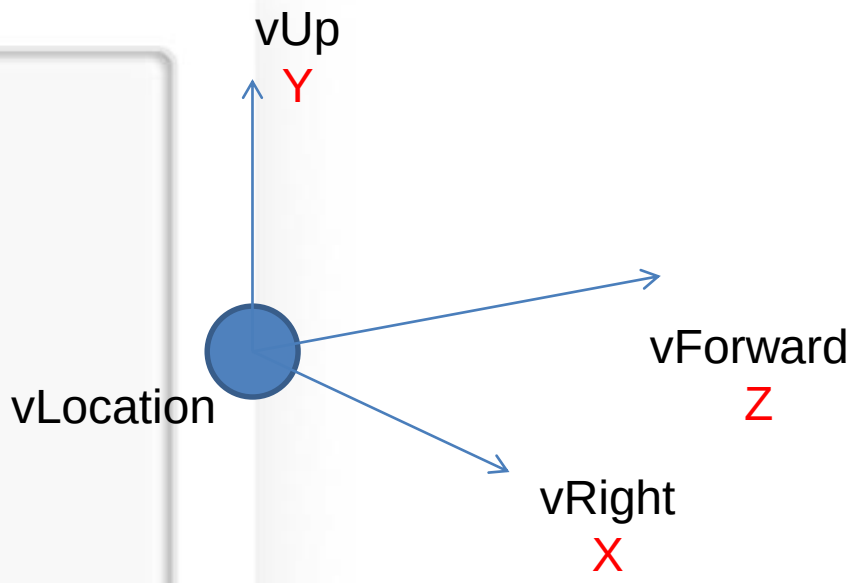
public void **LookAt**(Vector3 **worldPosition**);

public void **LookAt**(Vector3 **worldPosition**, Vector3 **worldUp** = Vector3.up);




```
void Update()
{
        // Point the object at the world origin
        transform.LookAt(Vector3.zero);
}
```

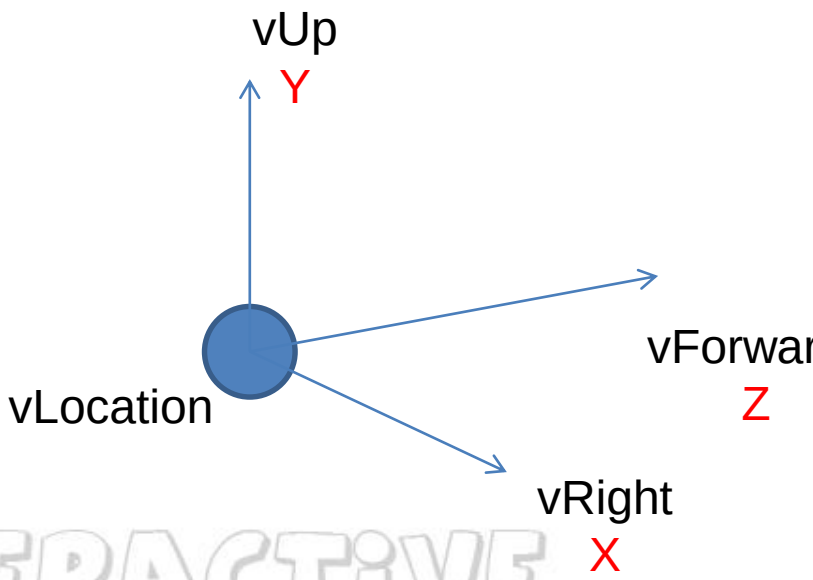
An Actor Frame



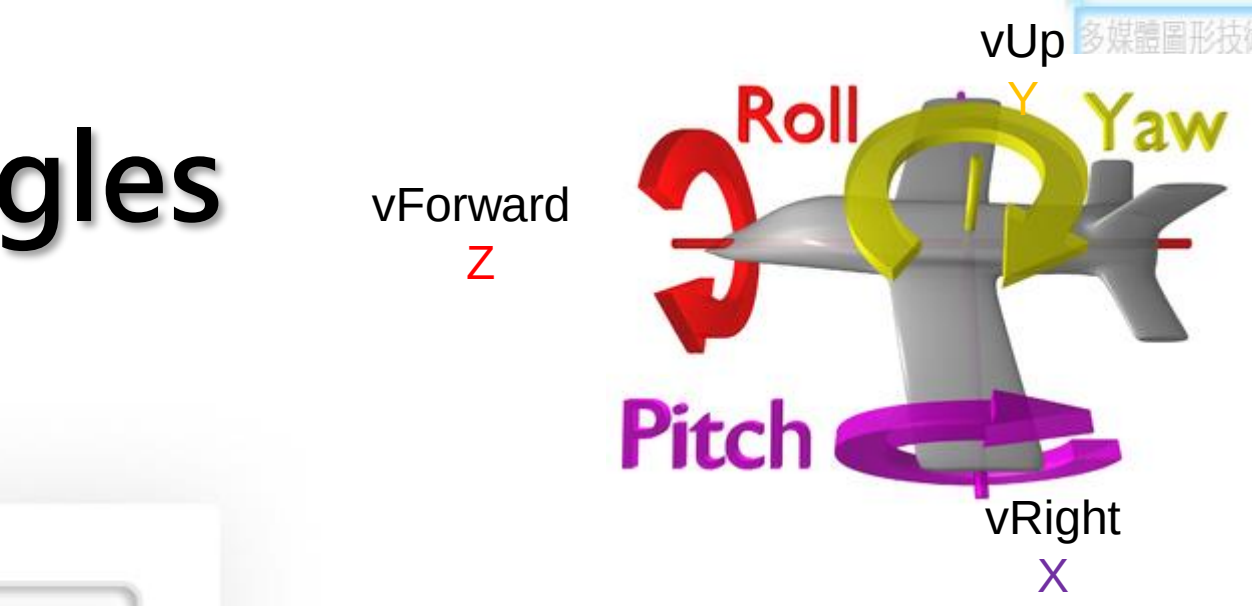
Move object

Translate

- public void **Translate**(Vector3 **translation**, Space **relativeTo** = Space.Self);
- public void **Translate**(float **x**, float **y**, float **z**, Space **relativeTo** = Space.Self);

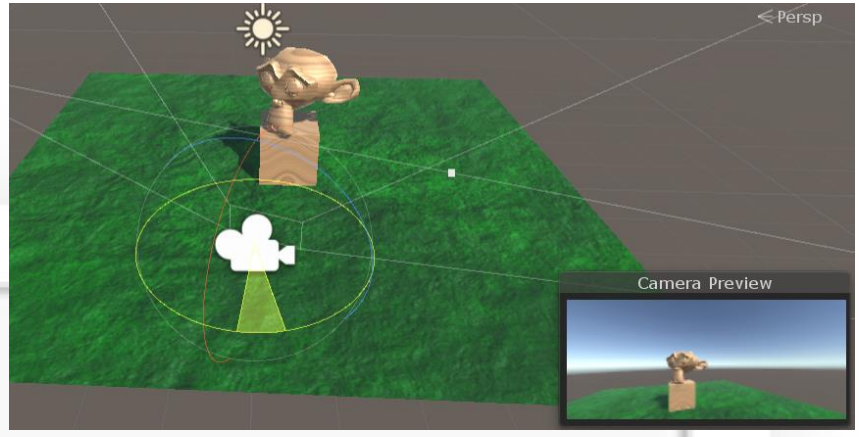


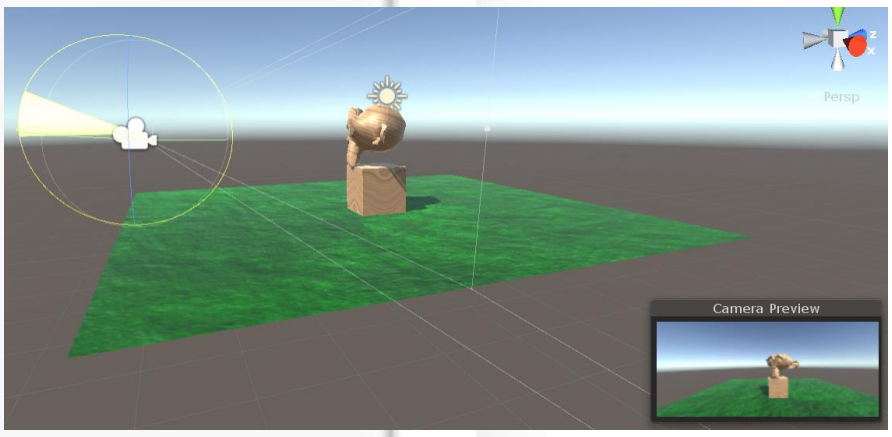
Euler Angles



- public void **Rotate**(Vector3 **eulers**, Space **relativeTo** = Space.Self);
- public void **Rotate**(float **xAngle**, float **yAngle**, float **zAngle**, Space **relativeTo** = Space.Self);

Camera movement



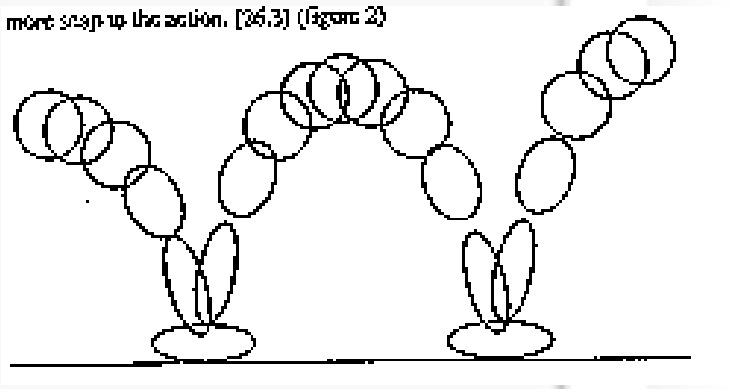


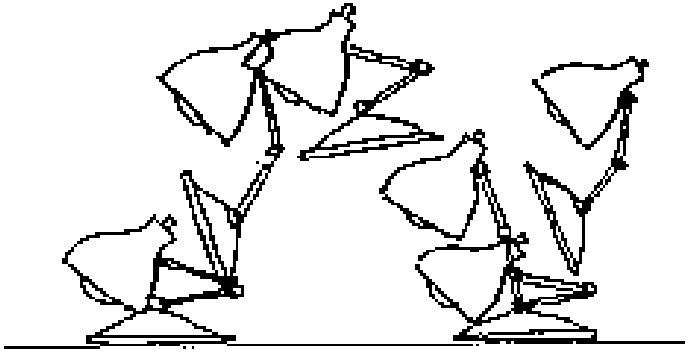
Pan
move your camera horizontally

Tilt
move your camera vertically

Dolly
move the entire camera forwards and backwards

Animation





Squash & stretch