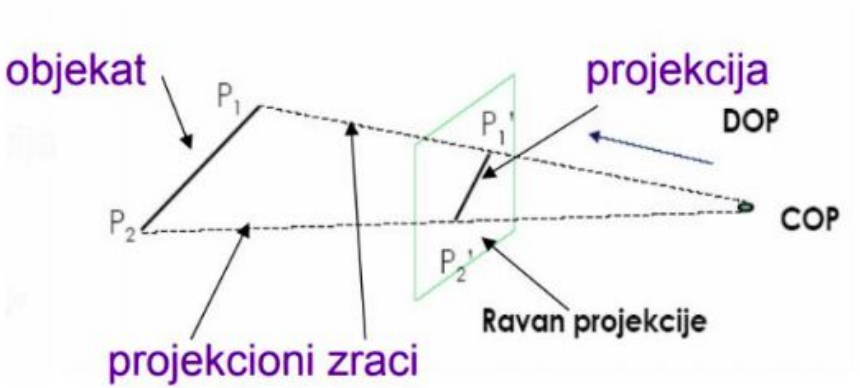
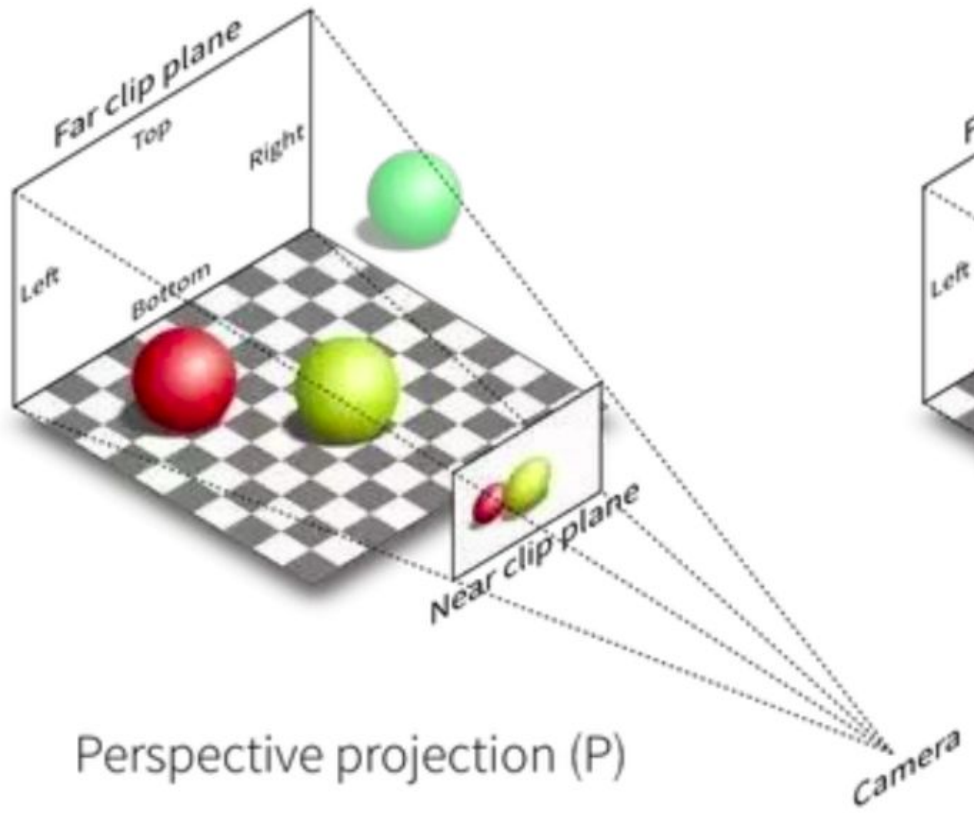
Računarska grafika
OpenGL 3

Projekcije

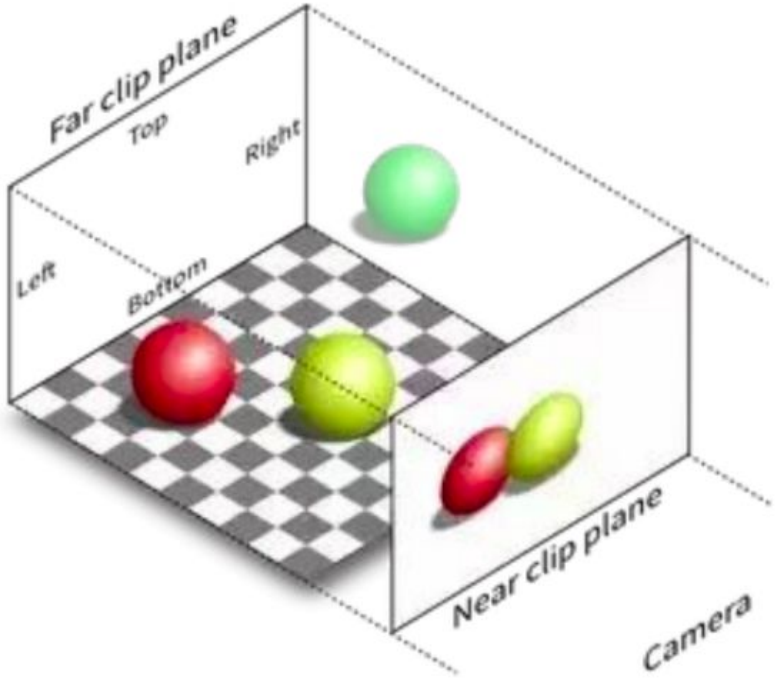
- Prikaz 3D objekta za 2D površini
- Osnovni parametri projekcije su
  - projekcioni zraci (projektori)
  - projekciona ravan
  - centar projekcije (COP)
  - pravac projekcije (DOP)



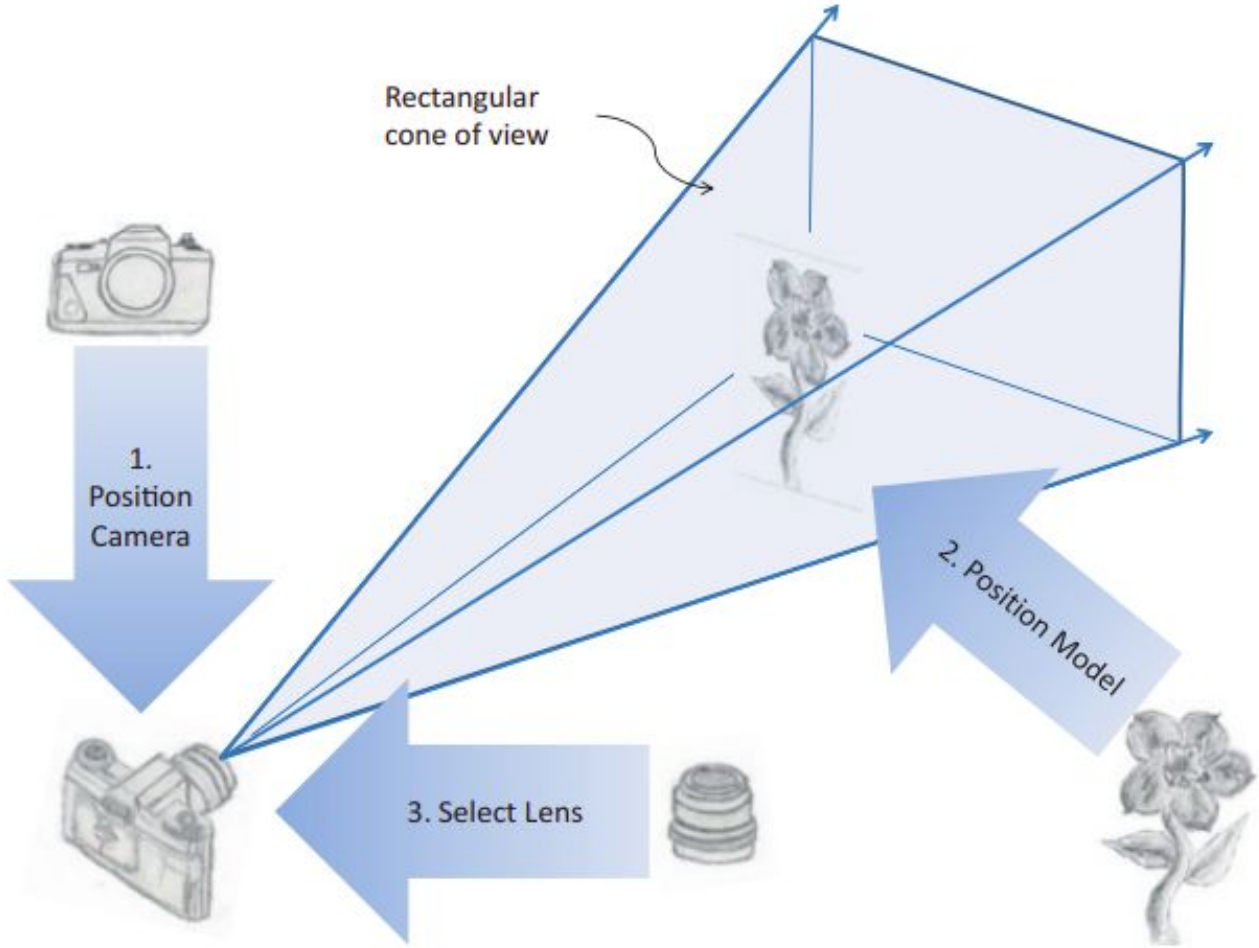
- Projekcija 3D objekta je definisana projekcionim zracima koji se emituju iz centra projekcije kroz svaku tačku objekta. Projekcija 3D objekta se nalazi na presjeku projektora i projekcione ravni.

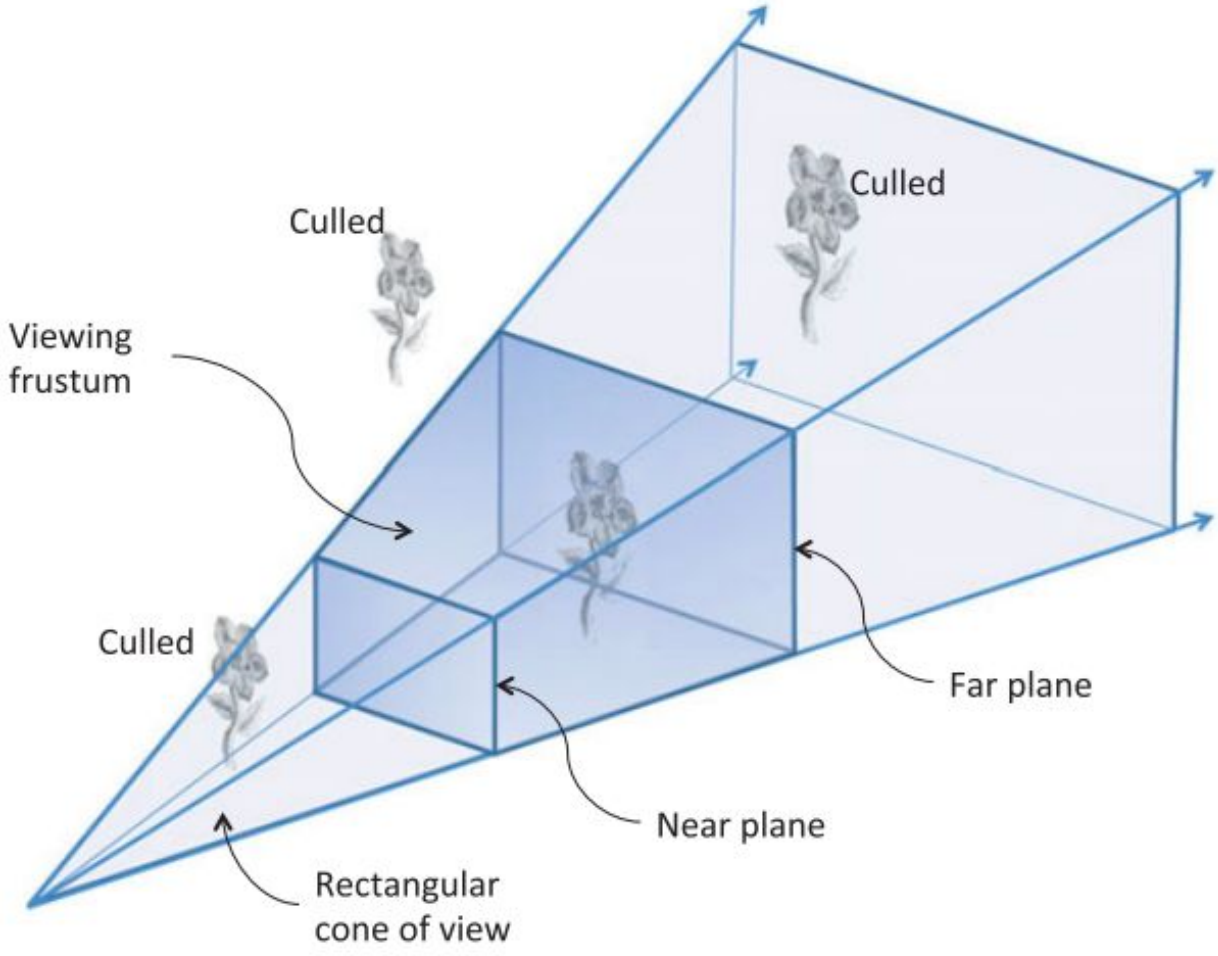


Perspective projection (P)



Orthographic projection (O)


Rectangular
cone of view
1.
Position
Camera
2. Position Model
3. Select Lens


Culled
Culled
Viewing
frustum
Culled
Far plane
Near plane
Rectangular
cone of view

gluPerspective

- Definiše matricu perspektivne projekcije:

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

- Parametar `fovy` definiše ugao pogleda (u stepenima) u smjeru y-ose
- `aspect` definiše rezoluciju (odnos širine i visine)
- `zNear` i `zFar` definišu udaljenost kamere od bliže i dalje ravni (uvijek pozitivne vrijednosti).

gluLookAt

- Definiše transformaciju pogleda:

```
void gluLookAt(GLdouble eyeX, GLdouble eyeY, GLdouble eyeZ,
GLdouble centerX, GLdouble centerY, GLdouble centerZ,
GLdouble upX, GLdouble upY, GLdouble upZ);
```

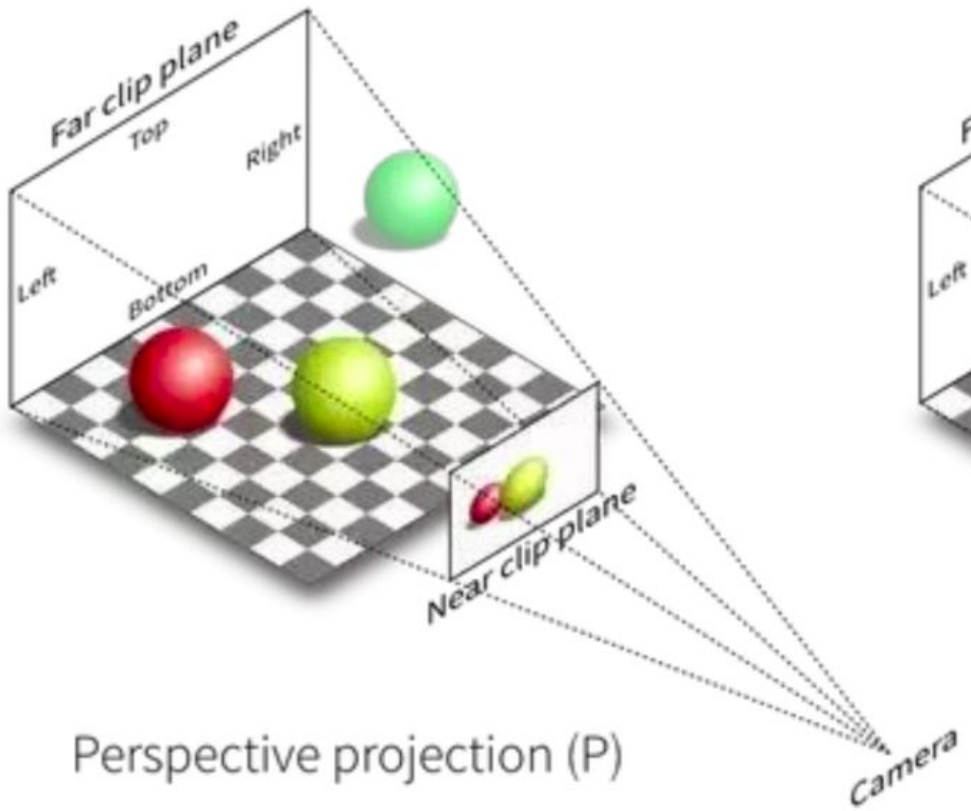
- `eyeX`, `eyeY`, `eyeZ` definišu poziciju oka (kamere)
- `centerX`, `centerY`, `centerZ` definišu poziciju referentne tačke (gdje kamera gleda)
- `upX`, `upY`, `upZ` su komponente jediničnog vektora koji određuje orijentaciju kamere.

glFrustum

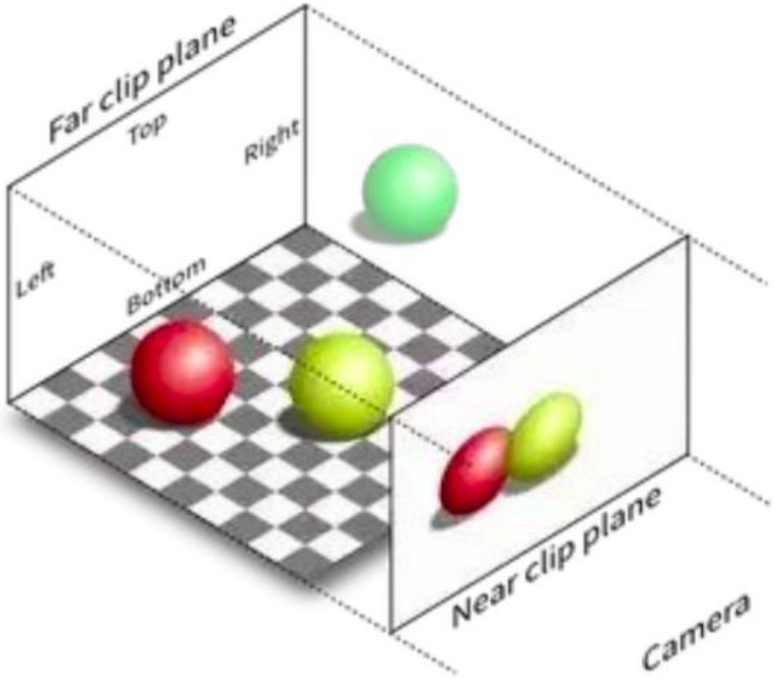
```
void glFrustum(GLdouble left, GLdouble right, GLdouble
bottom, GLdouble top, GLdouble nearVal, GLdouble farVal);
```

- `left`, `right`, `bottom`, `top` definišu površinu koju posmatramo
- `nearVal`, `farVal` definišu udaljenost kamere od bliže i dalje ravni

Primjer 1



Perspective projection (P)



Orthographic projection (O)

glOrtho

- Definiše matricu ortogonalne projekcije:

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

- `left`, `right`, `bottom`, `top` definišu površinu koju posmatramo
- `nearVal`, `farVal` definišu udaljenost kamere od bliže i dalje ravni (negativne vrijednosti ukoliko je ravan iza kamere)

Primjer 2

3D transformacije

- Skaliranje:

```
void glScalef(GLfloat x, GLfloat y, GLfloat z);
void glScaled(GLdouble x, GLdouble y, GLdouble z);
```

- Rotacija:

```
void glRotatef(GLfloat angle, GLfloat x, GLfloat y, GLfloat z);
void glRotated(GLdouble angle, GLdouble x, GLdouble y, GLdouble
z);
```

- Translacija:

```
void glTranslatef(GLfloat x, GLfloat y, GLfloat z);
void glTranslated(GLdouble x, GLdouble y, GLdouble z);
```

Primjer 3

Double buffering

- Postoje dva *buffer*-a za crtanje. Dok je jedan prikazan, drugi se priprema za prikaz, nakon čega se vrši *swap*.
- Operacija zamjene *buffer*-a je skoro trenutačna, što omogućava animacije (kao smjenu statičnih slika koje se prikazuju)

```
void glutSwapBuffers();
```

- Obično posljednja funkcija u *display callback*-u.
- Da bismo koristili *double buffering* potrebno je dodati konstantu `GLUT_DOUBLE` u *bitmask*-u koju dajemo kao argument pri pozivu `glutInitDisplayMode`.

Animacije

- GLUT sistem je *event-driven*, što omogućava tretiranje *input*-a od korisnika.
- Ipak, možemo napraviti da GLUT nešto radi i kad ne dobija naredbe iz vanjskog svijeta.

```
void glutIdleFunc(void (*func)(void));
```

- Omogućava zadavanje funkcije koja će se kontinualno pokretati (kada nema događaja za tretirati). Na ovaj način dobijamo dinamičnu scenu, bez zahtijevanja *input*-a od korisnika.
- Kod u funkciji koja se postavi kao *idle callback* treba minimizirati - izbjegavati složene proračune i operacije.
- Prosljeđivanje NULL vrijednosti isključuje pokretanje *callback*-a.

glutPostRedisplay *(pseudo)*

```
int do_display = 0;

void glutPostRedisplay(void) {
    do_display = 1;
}

void glutMainLoop(void) {
    for(;;) {
        event ev = get_OS_event();
        switch(ev.type) {
        case keyboard_event: call_keyboard_function(ev); return;
        case mouse_event:    call_mouse_function(ev); return;
        /* ... */
        }
        if( do_display ) {
            call_display_function();
            do_display = 0;
        } else {
            call_idle_function();
        }
    }
}
```

Primjer 4

Hvala na pažnji!