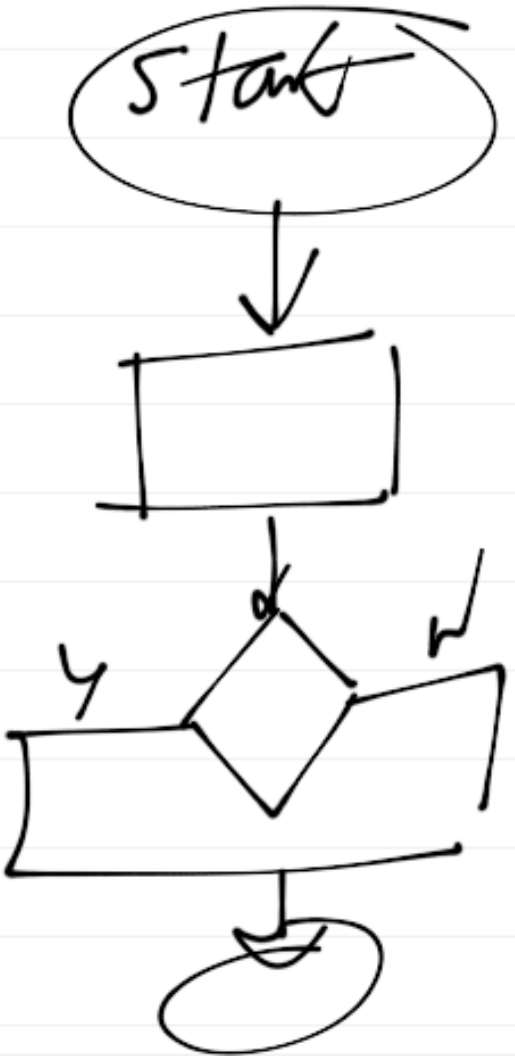
Procedural / functional programming:

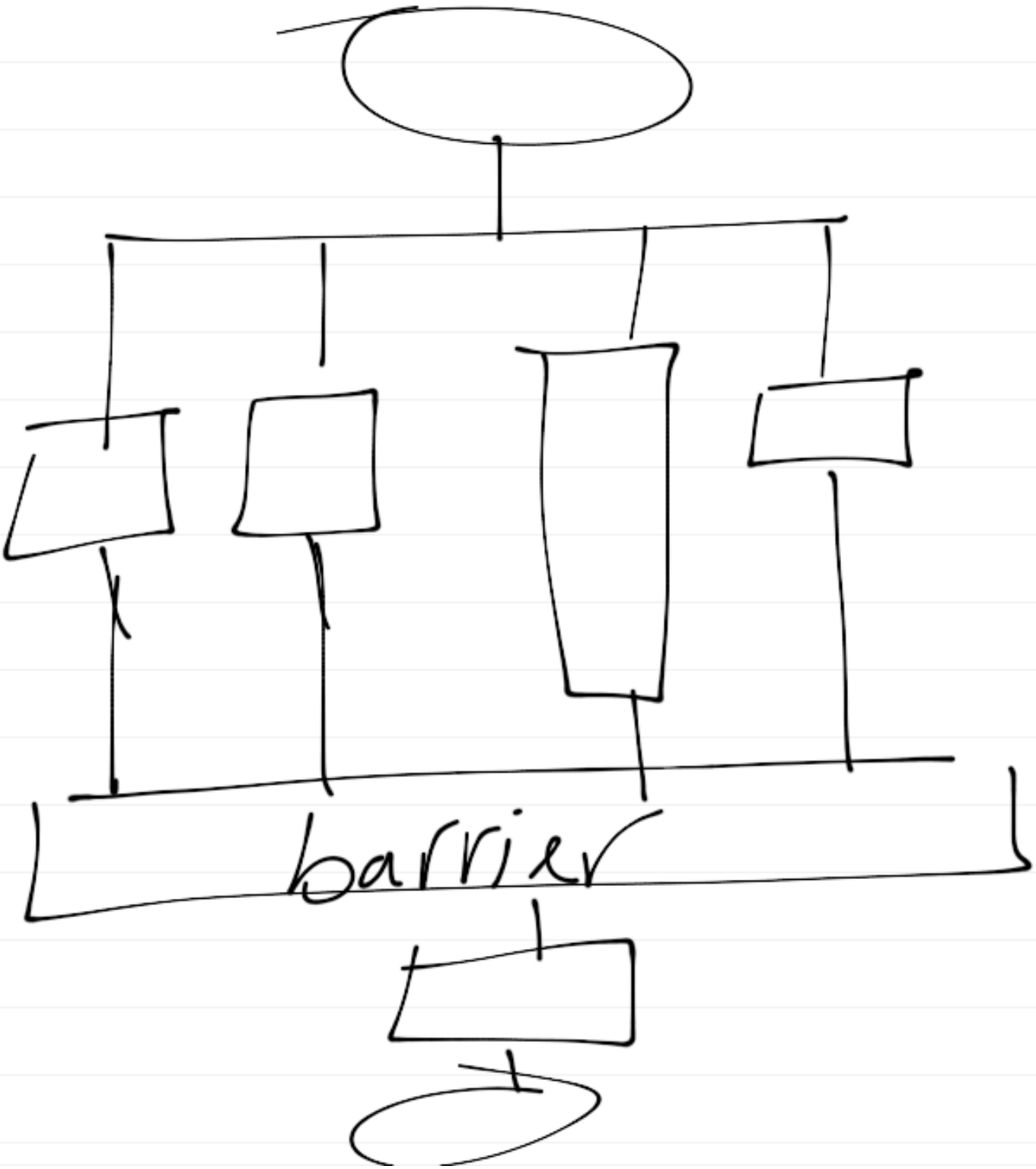
- "regular" flow of control

- main():

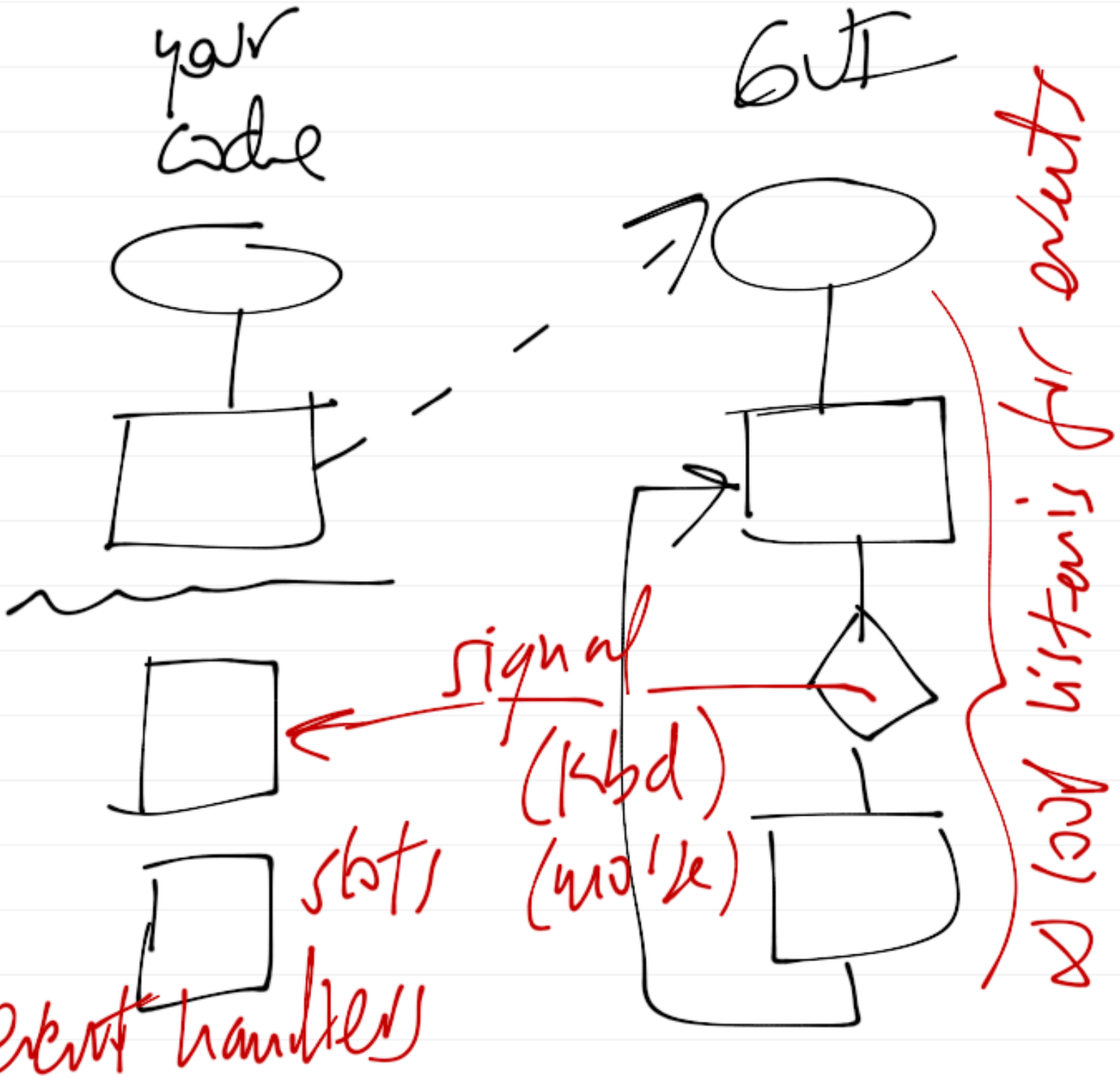


Concurrent programming (OpenMP)

multi-threads



event-driven programming (GUI)

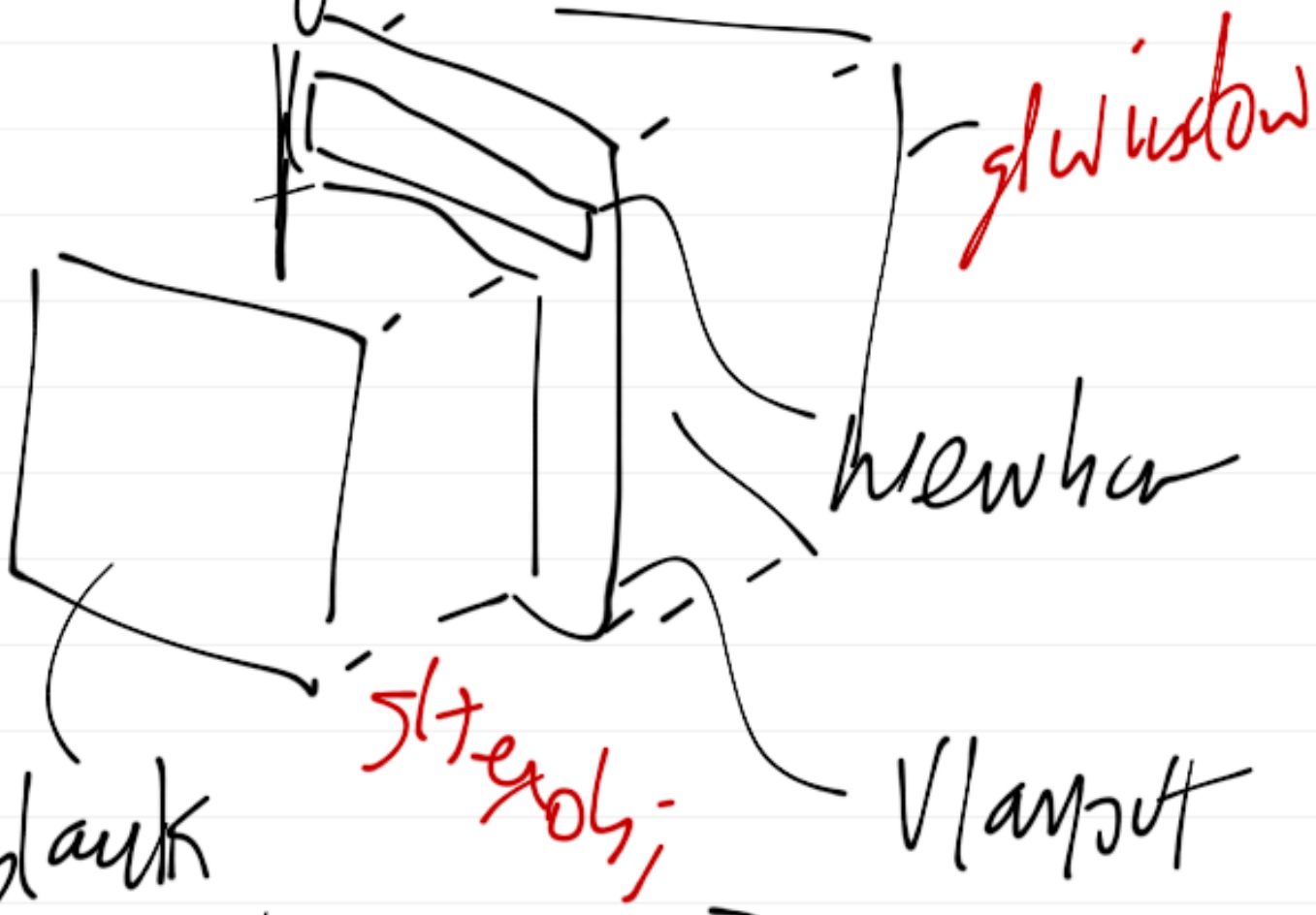


Qt: ——— all in C++

- signals:
  issued when events occur (mouse moves, button pressed, etc.)

- slots:
  public member fcns that process events (event handlers or callbacks)

- the img-texture code:



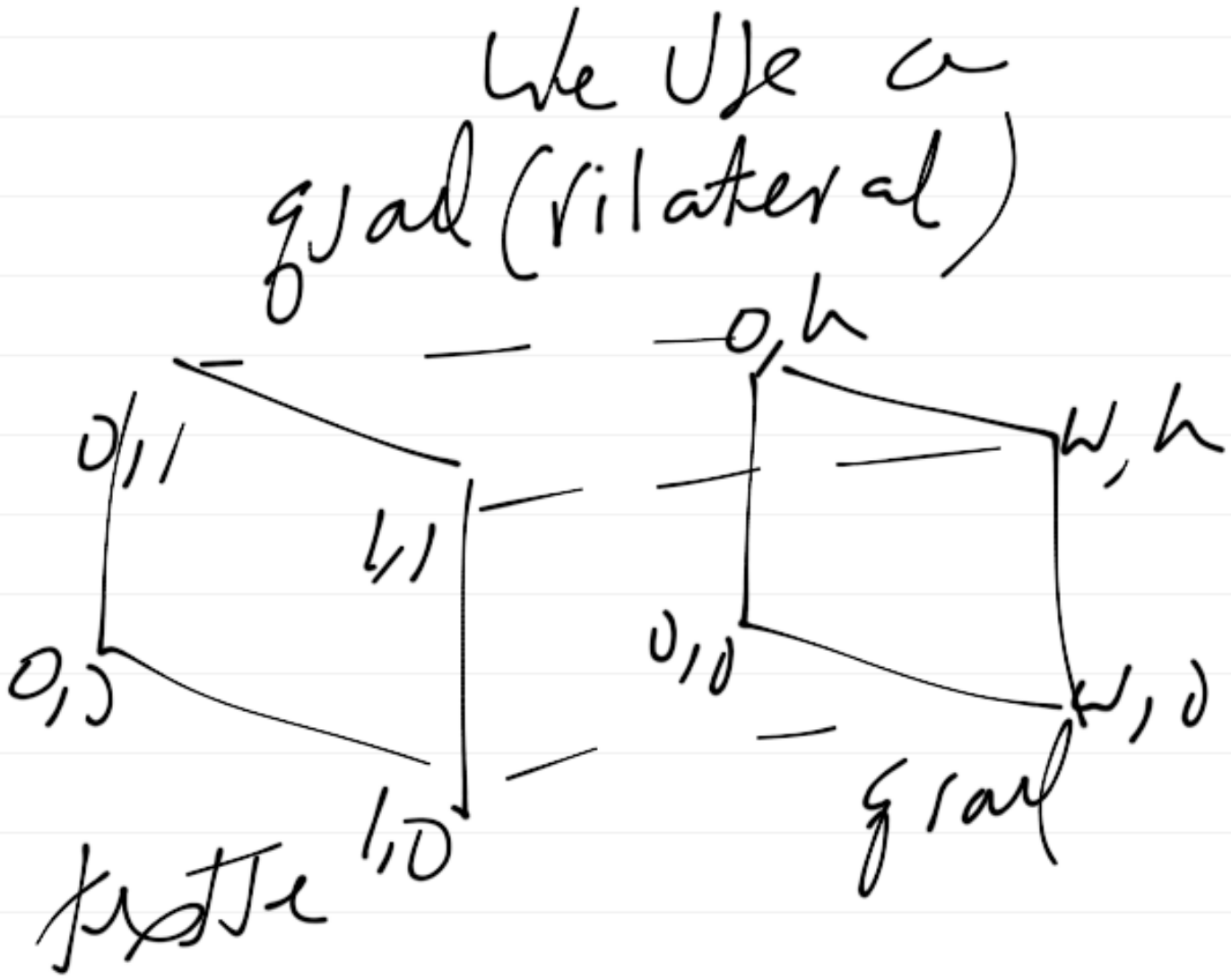
- OpenGL:
  graphics library

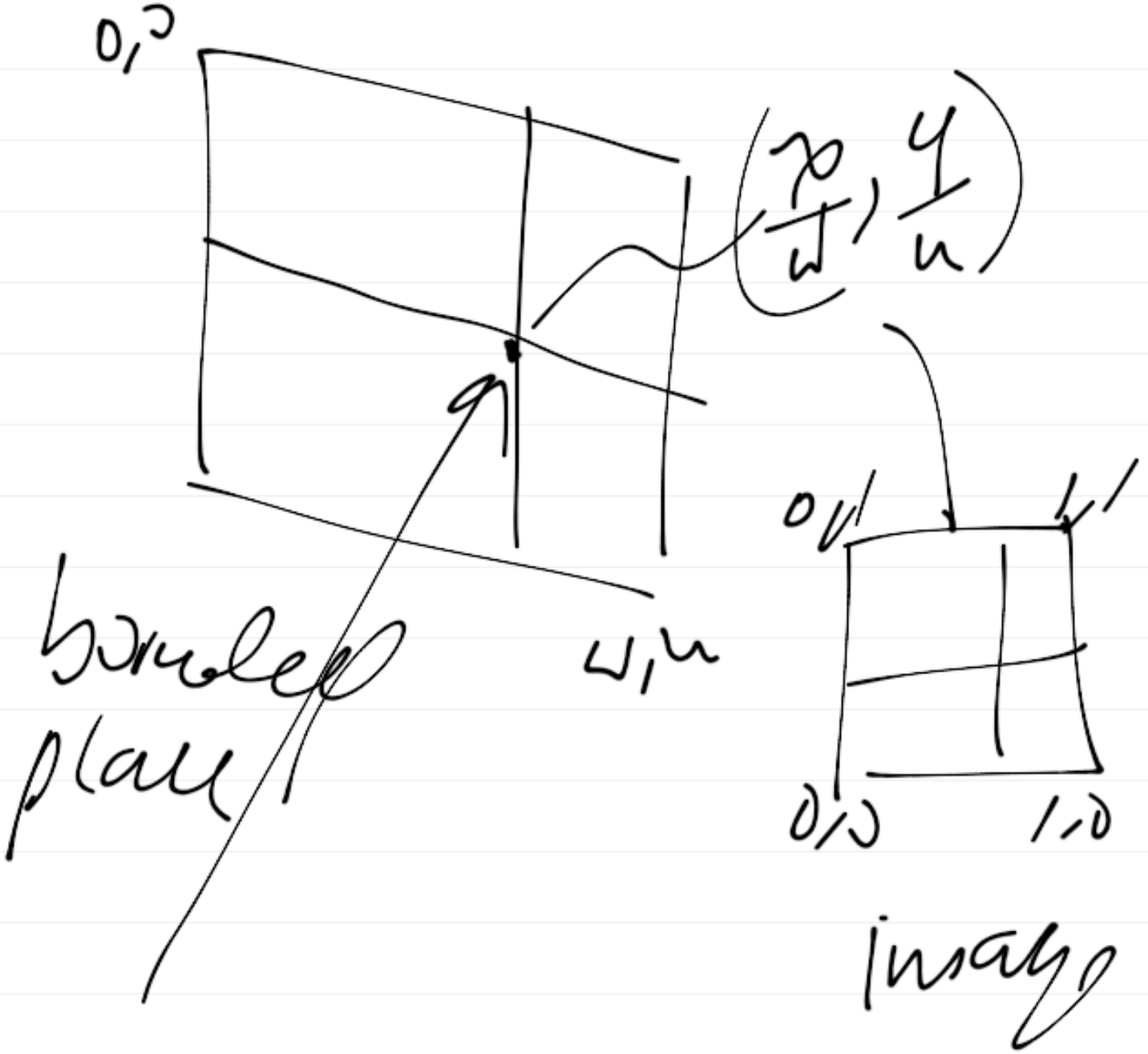
- think of GL as a
  state machine

e.g. glColor4f(1,0,0,1)
                R G B alpha (transp.)

sets color to red,
fully opaque

- img-texture:
  - maps texture (image pixels) to geometry




0,0
(x/w, y/w)
q
w,w
bounded plane
0,1
1,1
0,0
1,0
image

- Qt photon visualizer:
  - glteapot needs to maintain (private data member) a std::vector<photon_t*> photons;
- glteapot needs to read in 'photons.pts'

- glutMainLoop:
  - in paintGL():
    - loop through photons
      std::vector <

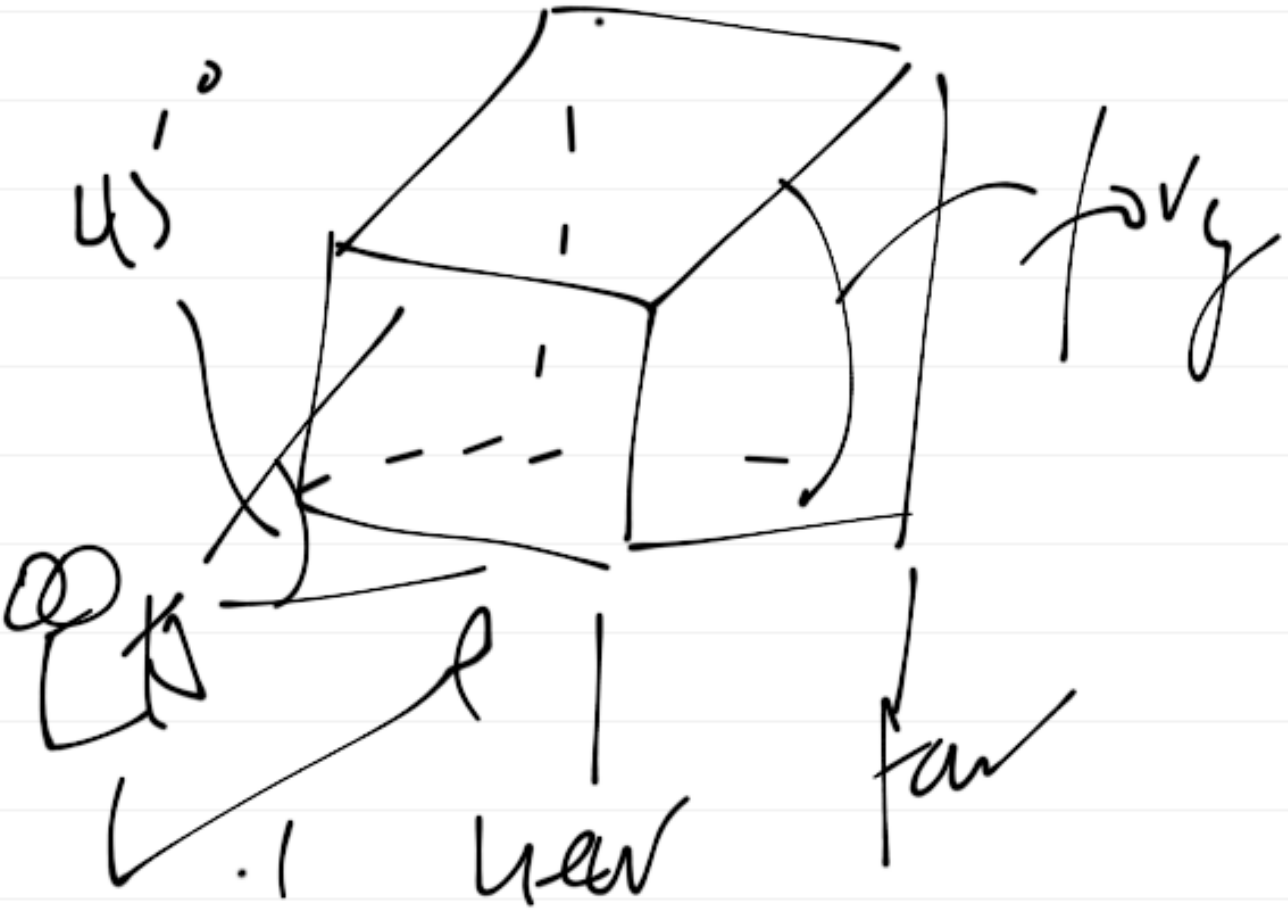
```
glBegin(GL_POINTS);
for(i=0; i < photons.size();
                i++)
    glVertex3f(
     (*photons[i])[0],
     (*photons[i])[1],
     (*photons[i])[2]);
glEnd();
```

- ahead of paint rendering,
set up camera view:

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(posx, posy, posz,
          refx, refy, refz,
          0, 1, 0);
```

(all in PaintGL)

```
if resize GL (int w, int h):
    glViewport(0, 0, w, h)
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluPerspective(fov,
                   aspect, (h/w)
                   .1 ← near,
                   100 ← far);
```



View frustum