Fall 2018

CSCI 420: **Computer Graphics**

Exercise 2. Simulate a Roller Coaster

Haiwei Chen

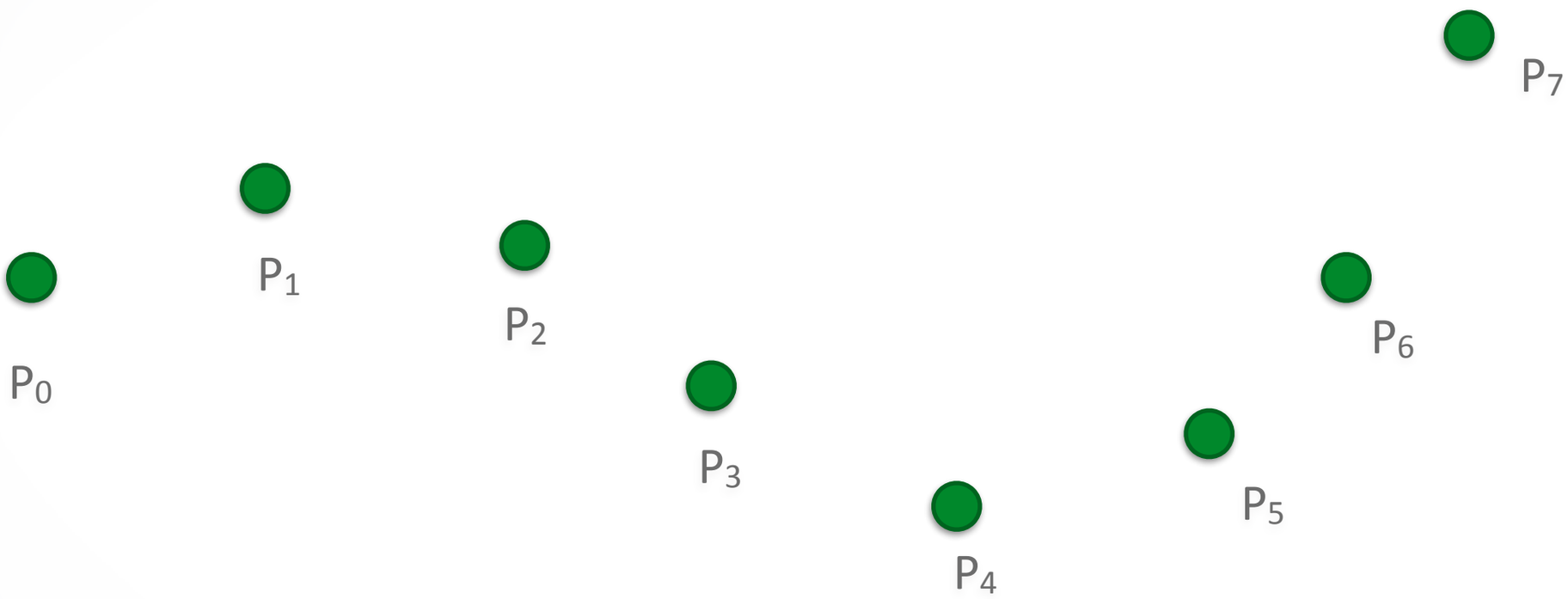
http://cs420.hao-li.com

How it will look like…

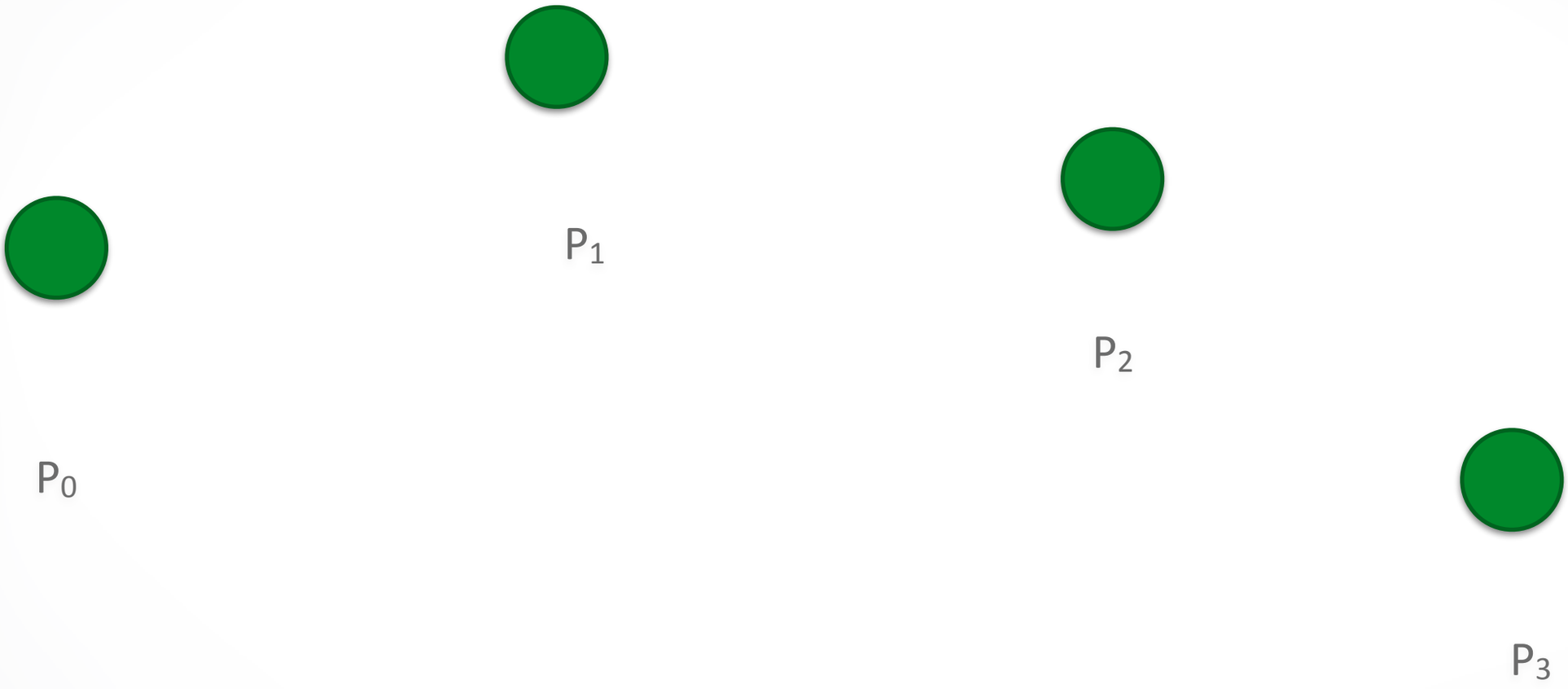
Level 1: Roller Coaster

- Splines
- Texture maps and their use in OpenGL
- Camera manipulations--the use of transformations to create realistic first-person movement

Level 1: Spline

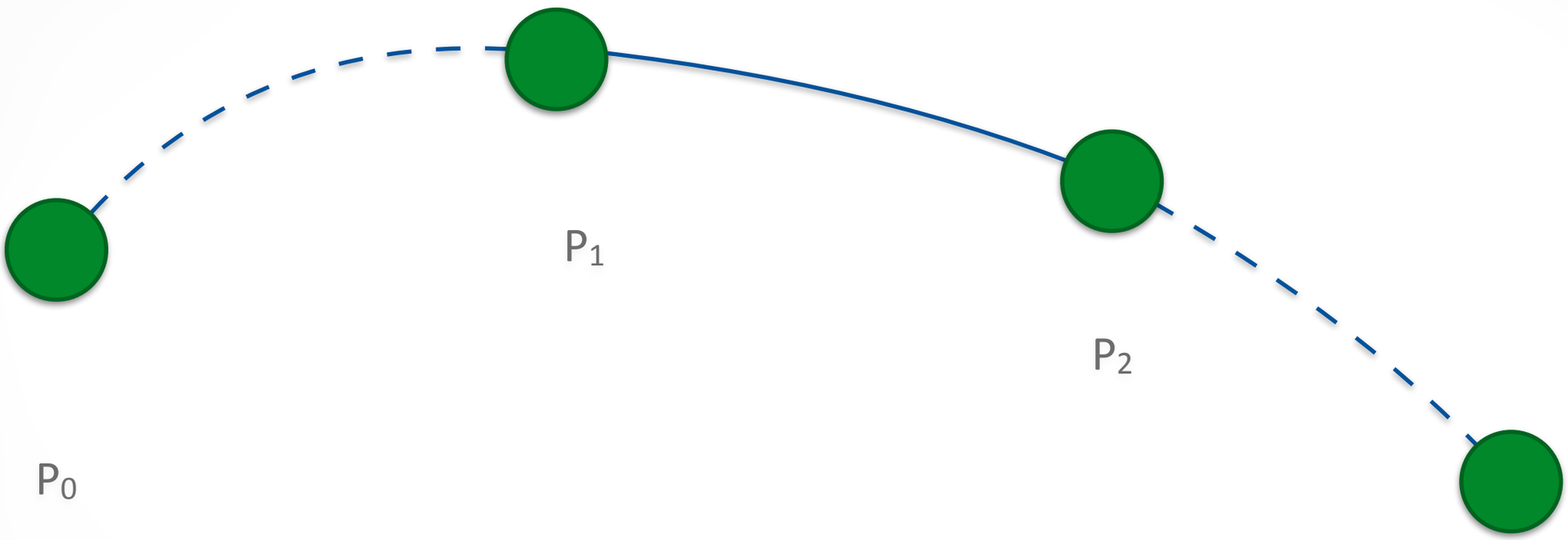


Level 1: Spline



Catmull-Rom Spline

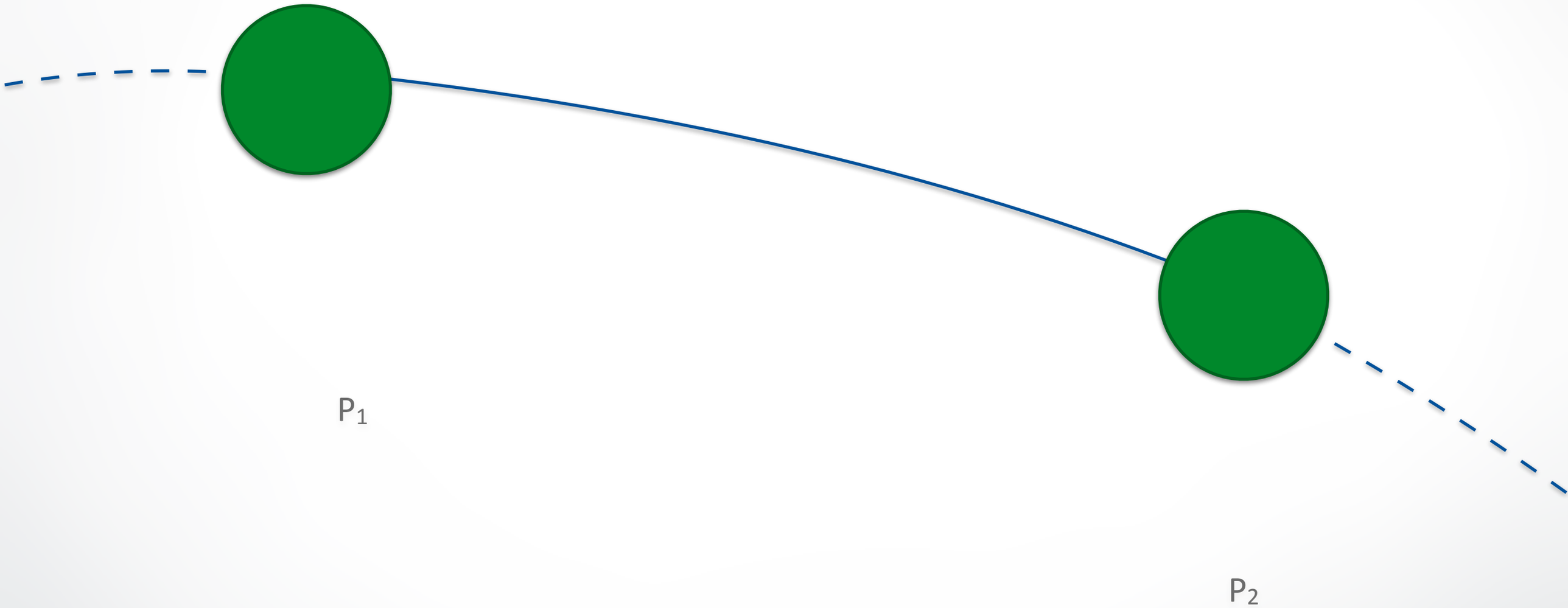
Level 1: Spline



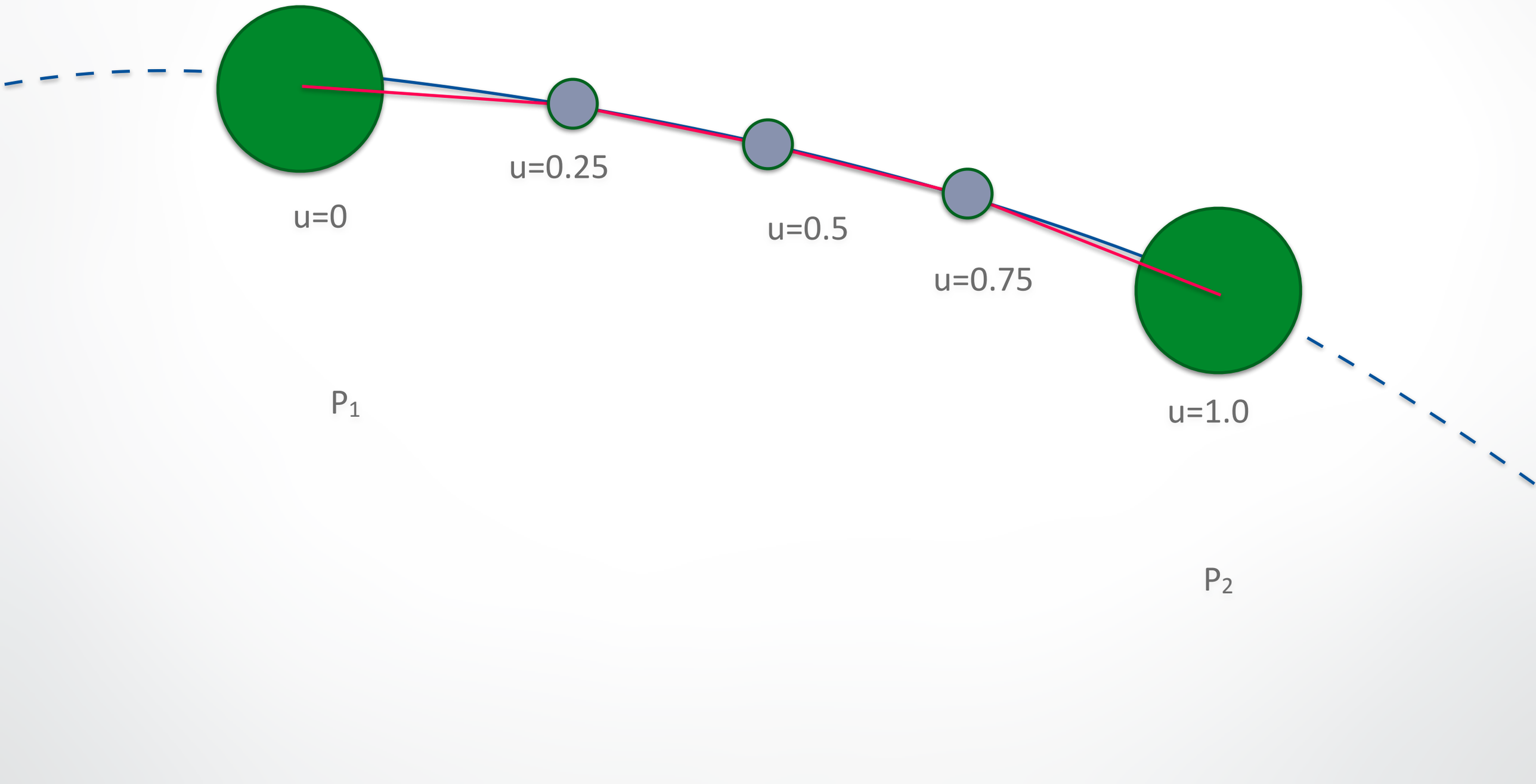
$$\begin{bmatrix} x & y & z \end{bmatrix} = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} -s & 2-s & s-2 & s \\ 2s & s-3 & 3-2s & -s \\ -s & 0 & s & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \end{bmatrix}$$

basis control matrix

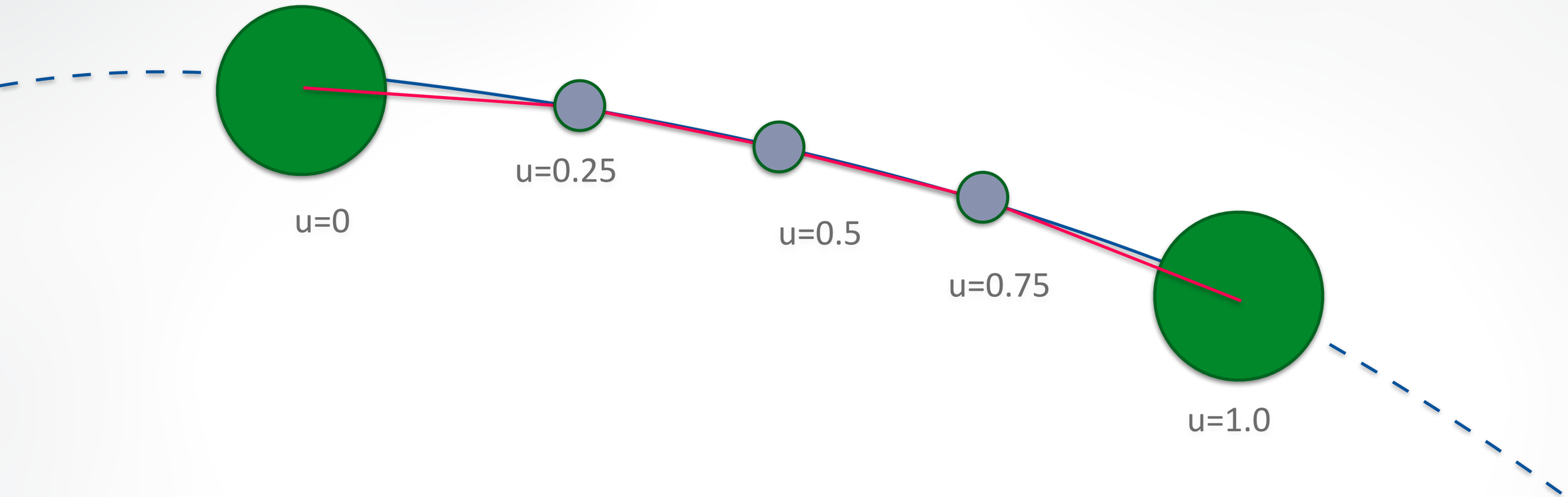
Level 1: Spline



Level 1: Spline

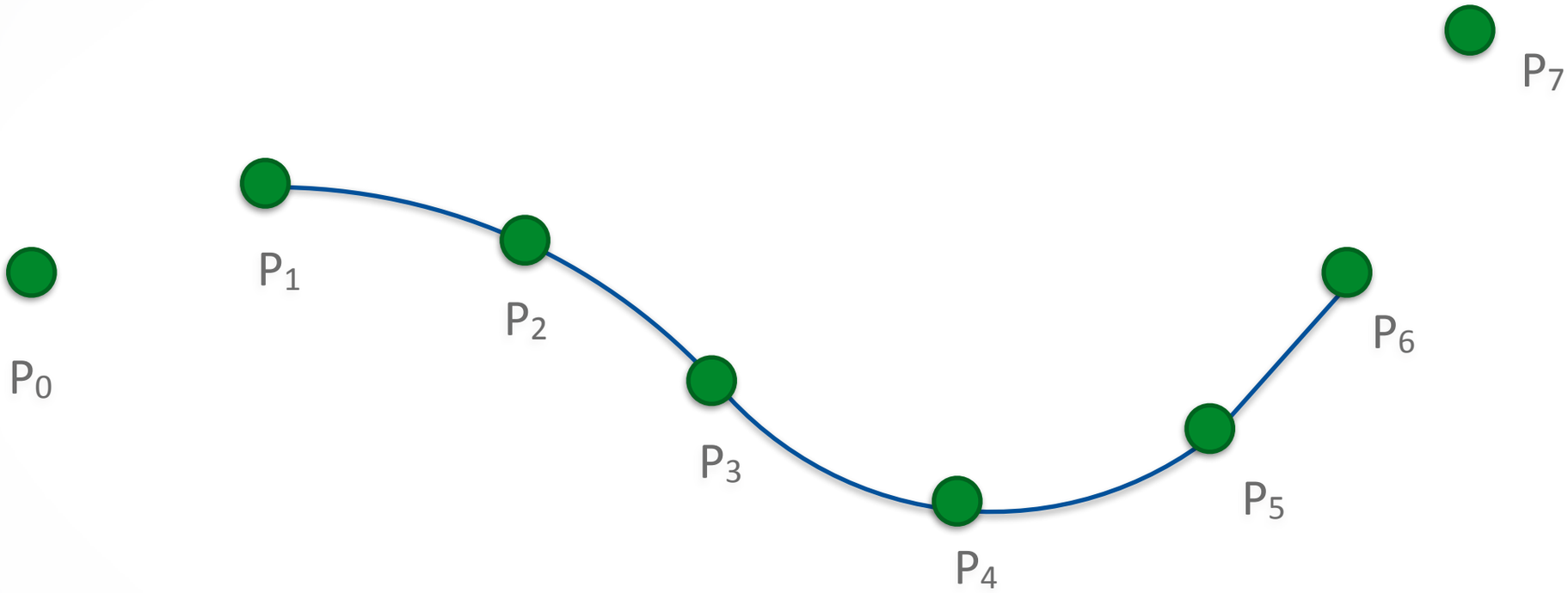


Level 1: Spline

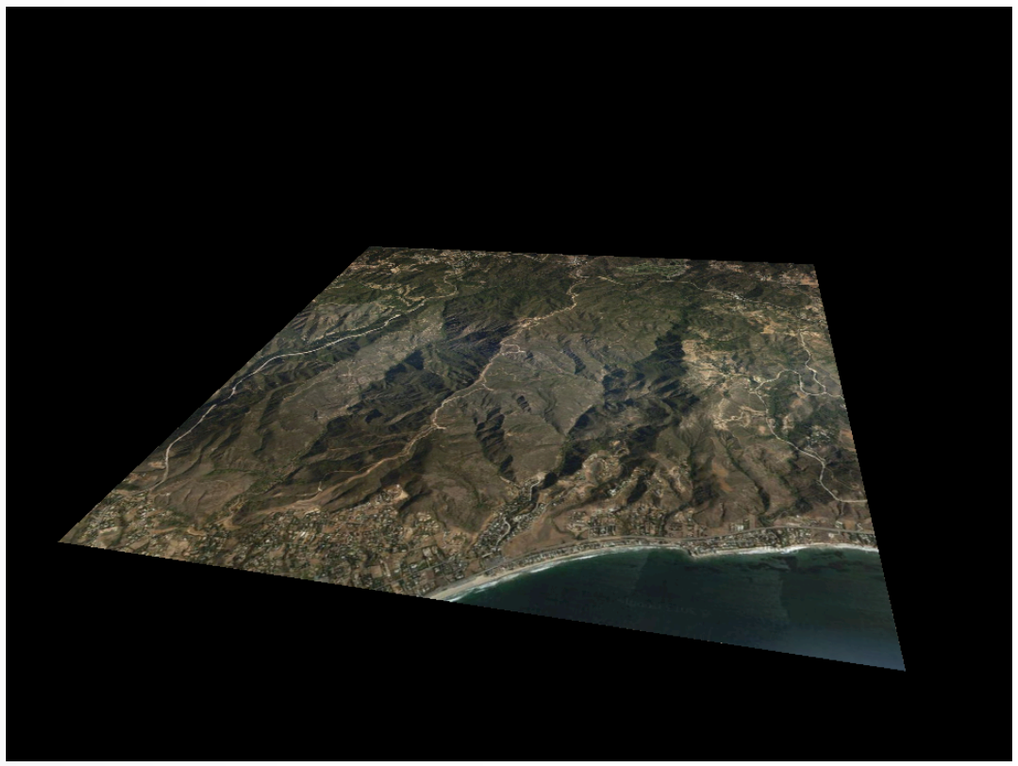


- u = 0.0, 0.001, 0.002, ..., 0.999, 1.0.
- connect consecutive points with lines

Level 1: Spline



Level 2: Ground



Plane

Height Field

Level 2: Ground

Level 2: Ground

```
GLuint texture[1];

void texload(int i,char *filename)
{
    Pic* img;
    img = jpeg_read(filename, NULL);
    glBindTexture(GL_TEXTURE_2D, texture[i]);
    glTexImage2D(GL_TEXTURE_2D,
                 0,
                 GL_RGB,
                 img->nx,
                 img->ny,
                 0,
                 GL_RGB,
                 GL_UNSIGNED_BYTE,
                 &img->pix[0]);
    pic_free(img);
}

void myinit()
{
    glGenTextures(1, texture);
    texload(0,"fileName.jpg");
}
```

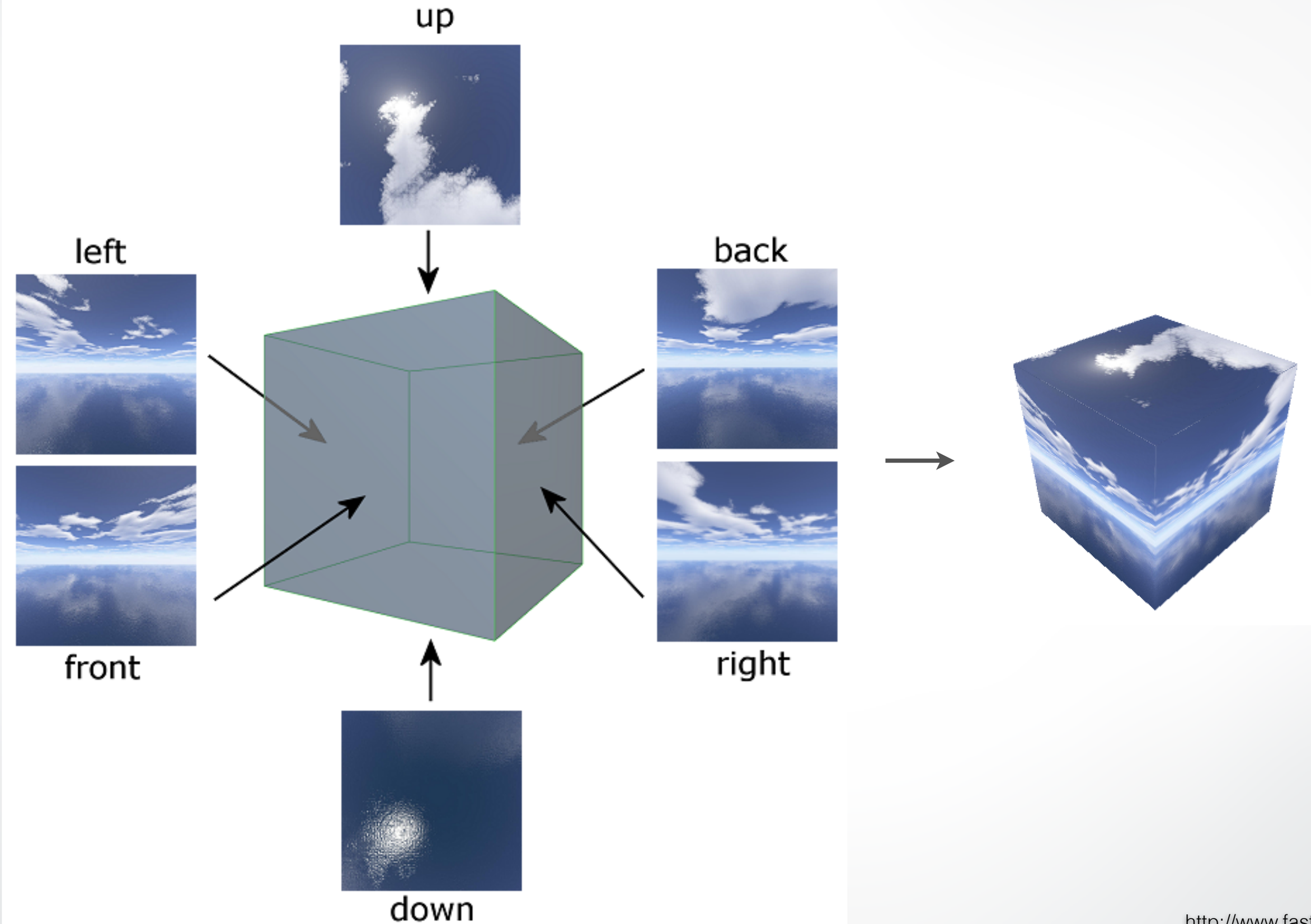
Level 2: Ground

```
void display()
{
    ……

    glBindTexture(GL_TEXTURE_2D, texture[0]);
    glTexEnvf(GL_TEXTURE_ENV,GL_TEXTURE_ENV_MODE,GL_MODULATE);
    glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_LINEAR);
    glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);

    glBegin(GL_POLYGON);
    glTexCoord2f(1.0, 0.0);
    glVertex3f(1.0, 1.0, 1.0);
    glTexCoord2f(0.0, 0.0);
    glVertex3f(-1.0, 1.0, 1.0);
    glTexCoord2f(0.0, 1.0);
    glVertex3f(-1.0, 1.0, -1.0);
    glTexCoord2f(1.0, 1.0);
    glVertex3f(1.0, 1.0, -1.0);
    glEnd();
    glDisable(GL_TEXTURE_2D);

    ……
}
```

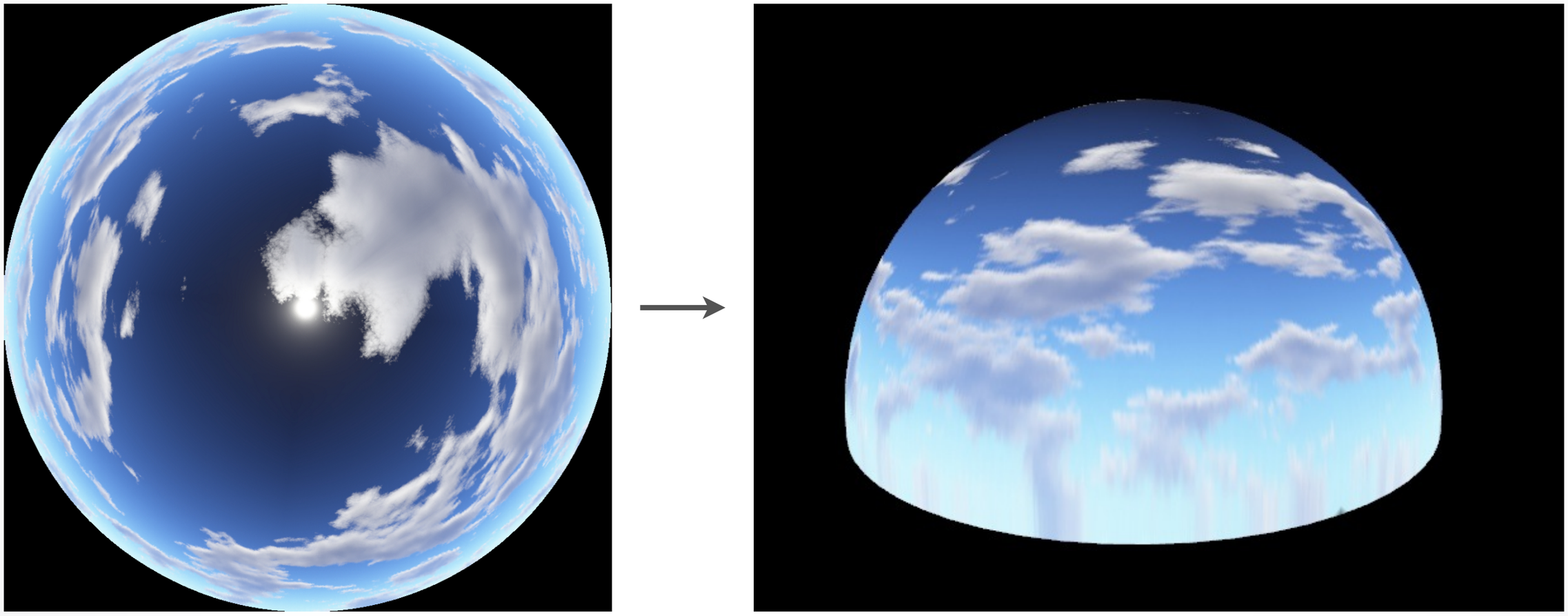
Level 3: Sky

Sky cube texture

Level 3: Sky

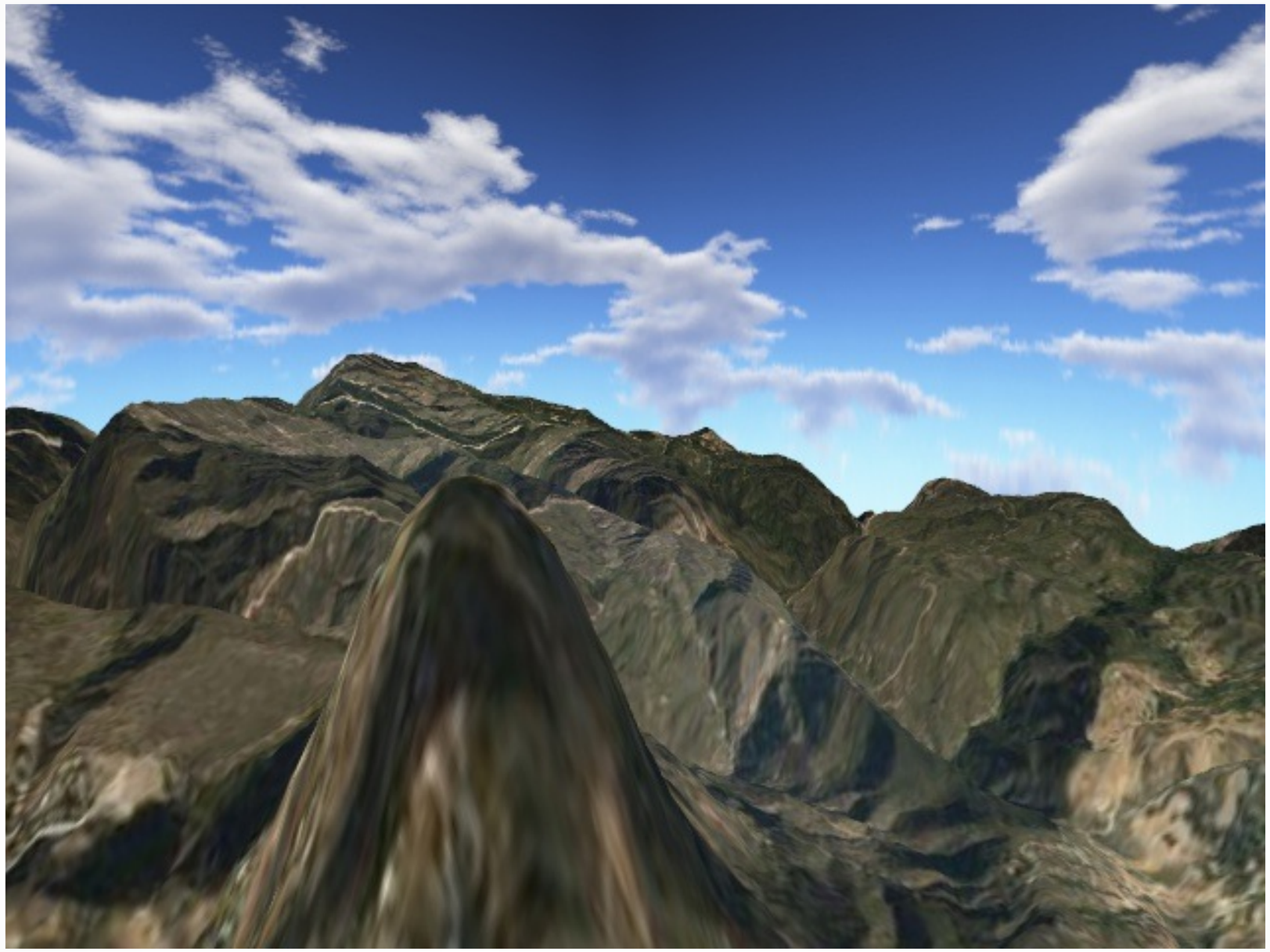


Level 3: Sky



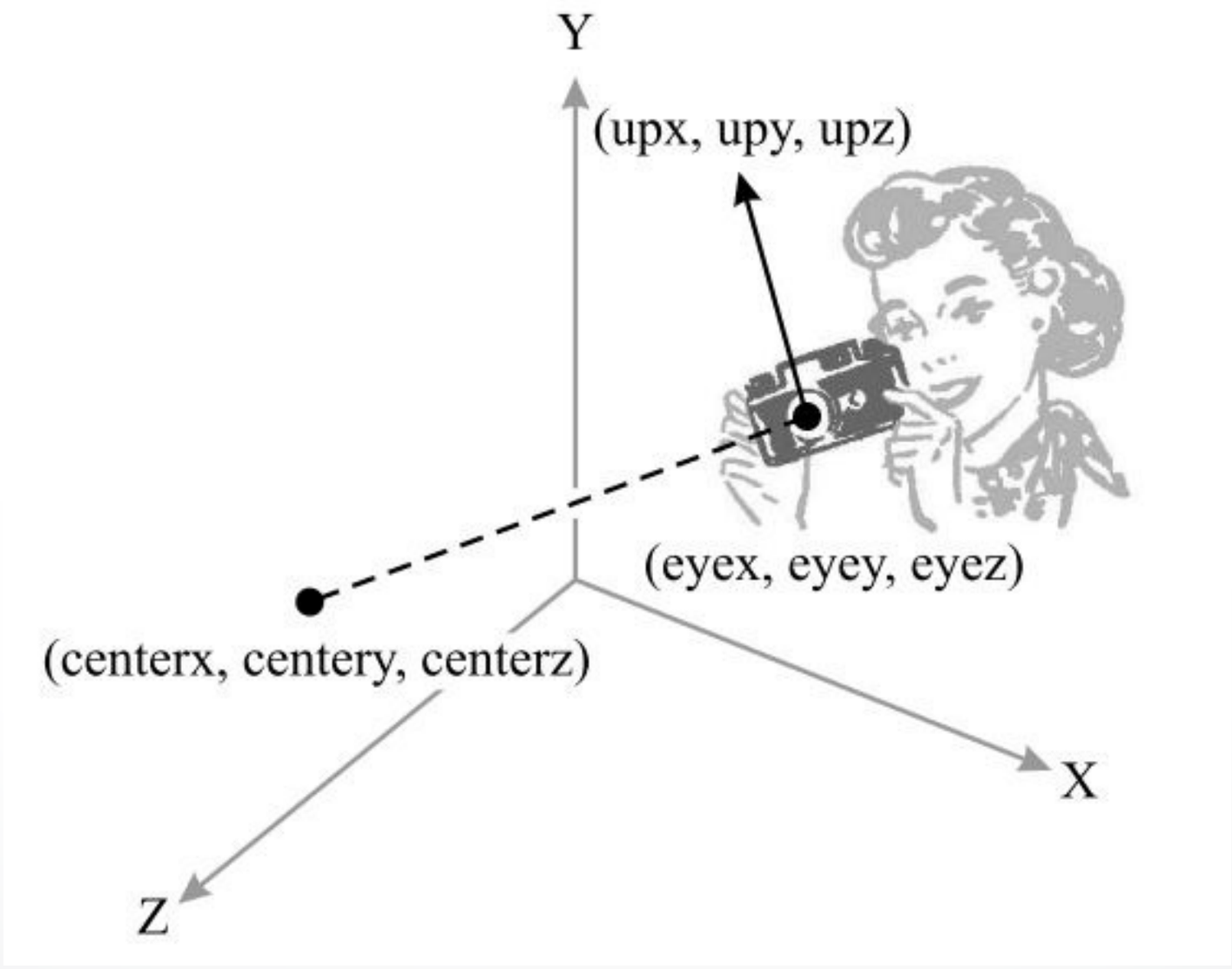
Sky dome texture

Level 3: Sky



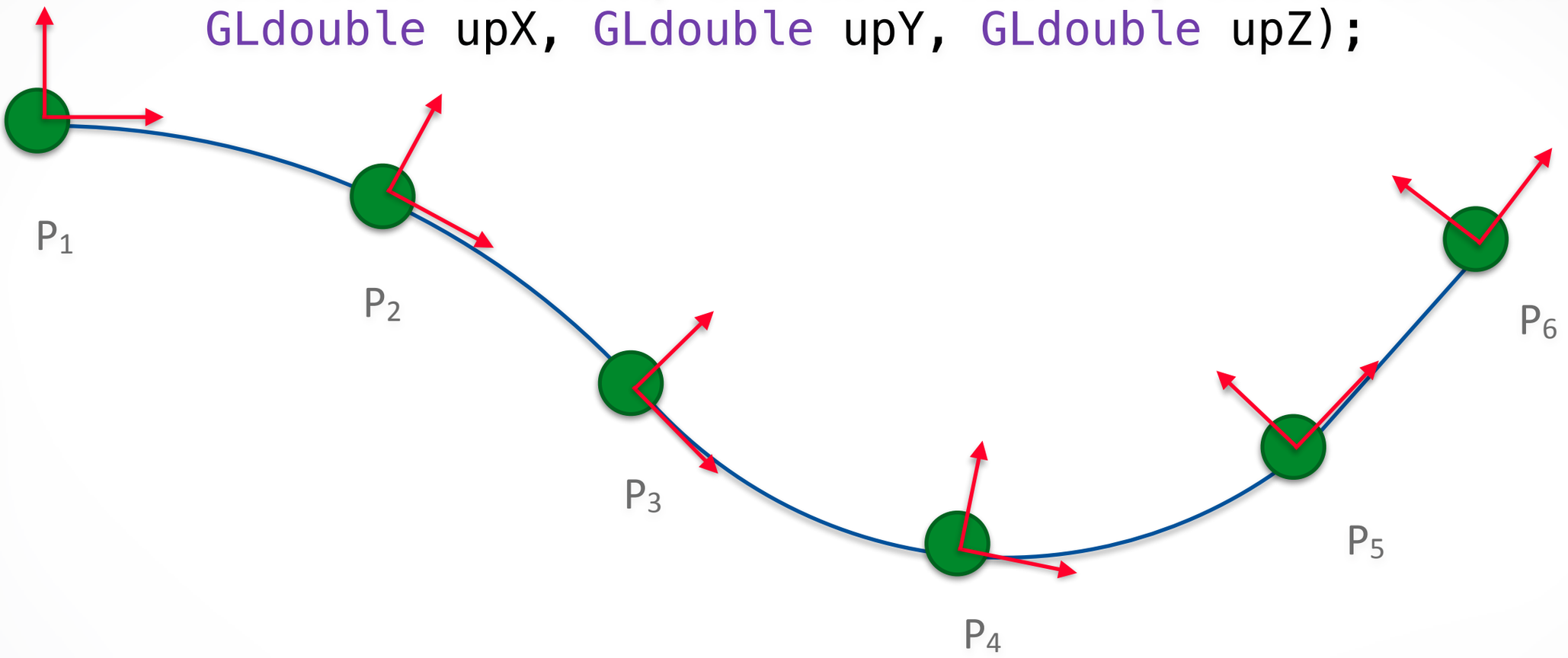
Level 4: The Ride

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



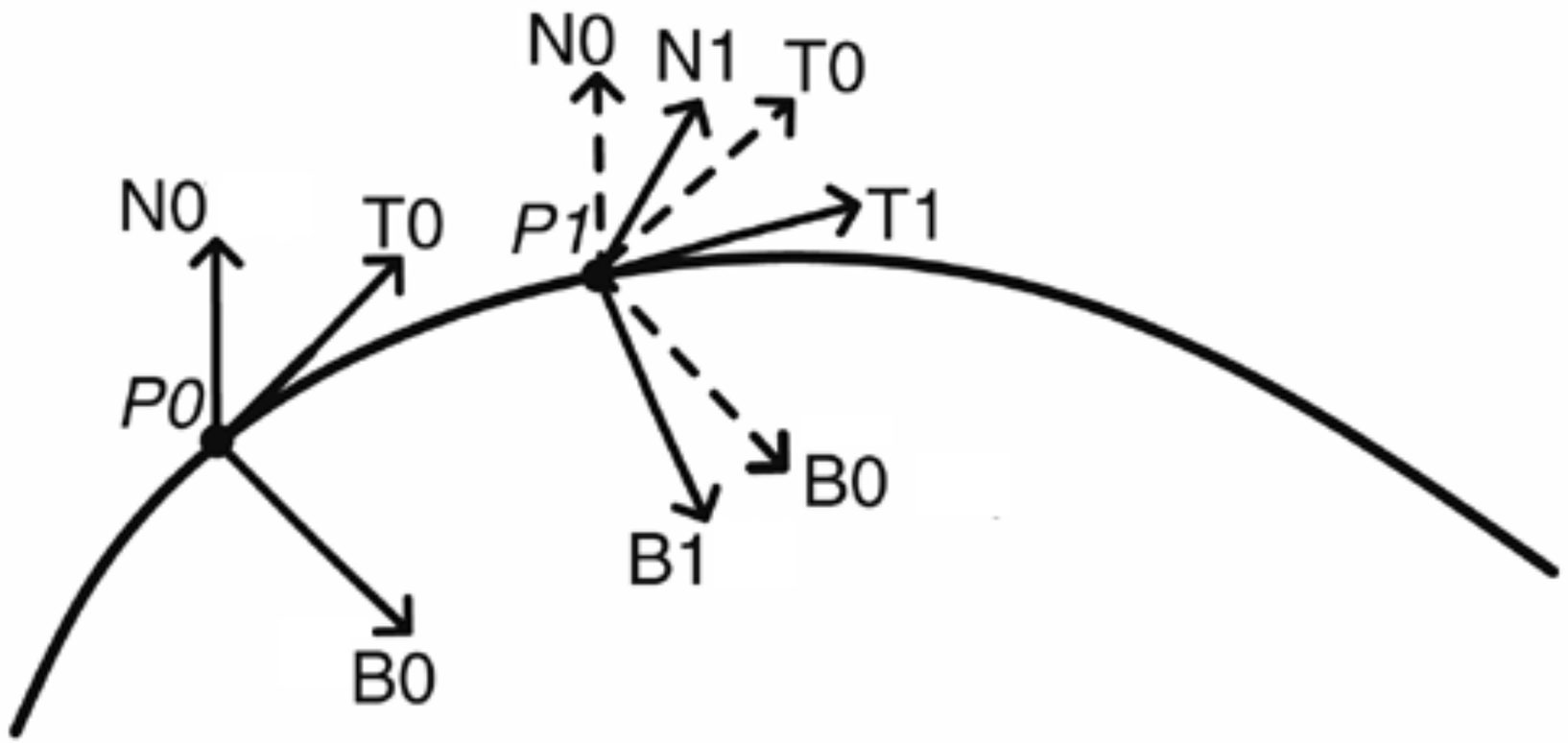
Level 4: The Ride

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



- eye: p(u)
- tangent (orientation): t(u) =unit(p'(u)) = unit([3u^2 2u 1 0] M C).
- center = eye + a*tangent
- How to compute up vector?

Level 4: The Ride

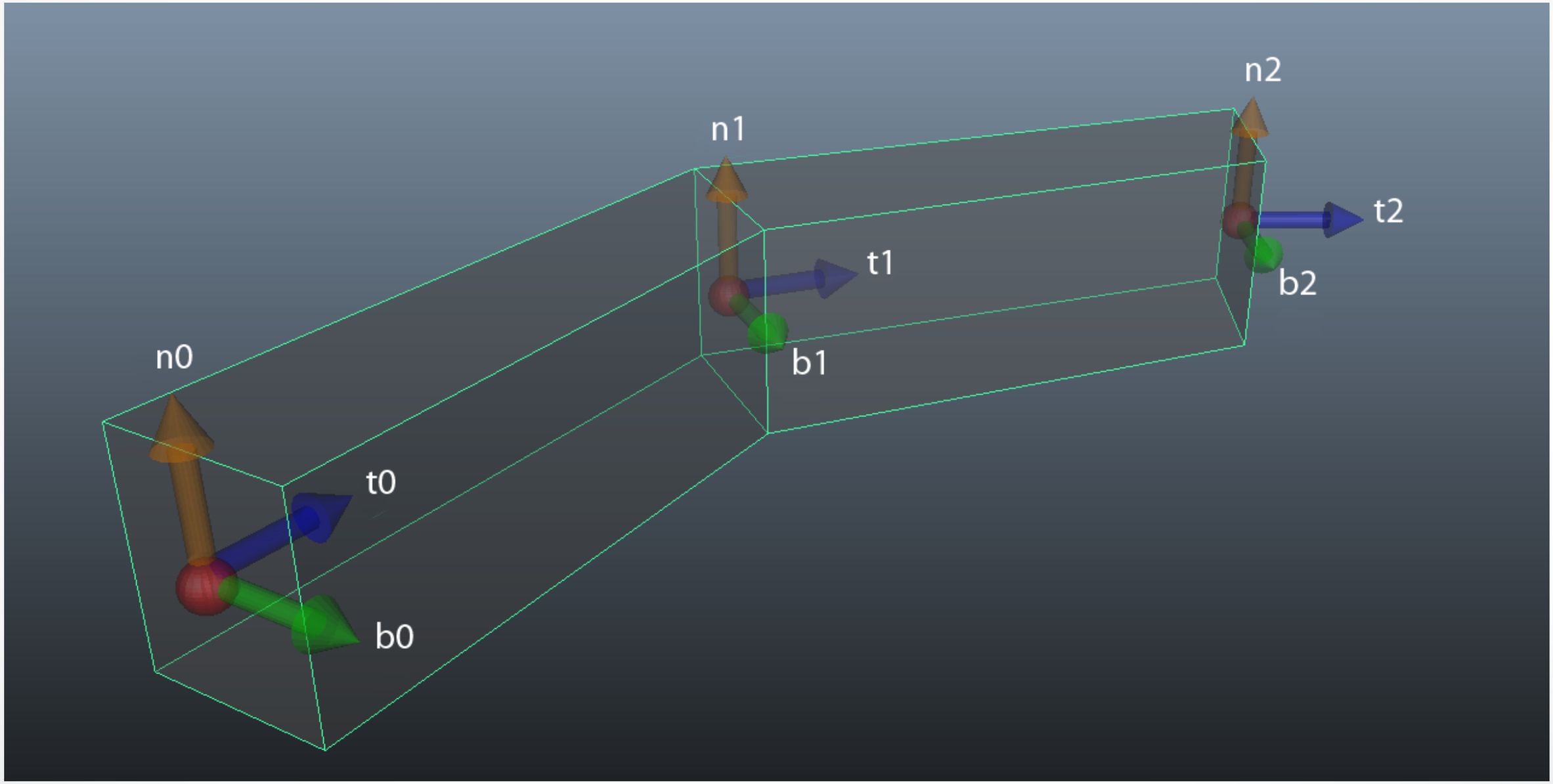


Computing a reference frame from the previous frame

Level 4: The Ride

- Establish a local coordinate system for
  each point on the curve
    T, N, B
- Initialization: T0, N0, B0
  u = 0 => T0
  Pick an arbitrary unit(V). N0 = unit(T0 x V) and
  B0 = unit(T0 x N0). This guarantees T0,
  N0, B0 orthonormal to each other.
- Next view: T1, N1, B1
  Move u ahead, compute T1 based on new u
  N1 = unit(B0 x T1) and B1 = unit(T1 x N1)
- T2, N2, B2, ...
- Update camera “up” vector to be N or B
- Guarantee camera orientation changes continuously

Level 5: Rail Cross-section



Submission

- Deadline: **Oct 22, 2018 11:59 pm**
- **Start this assignment as soon as you can**
- Upload a .zip compressed file named “Exercise2-YourName.zip” to blackboard
- Include your code with comments
- Include a readme file
- Include JPEG frames or a video

Enjoy it!