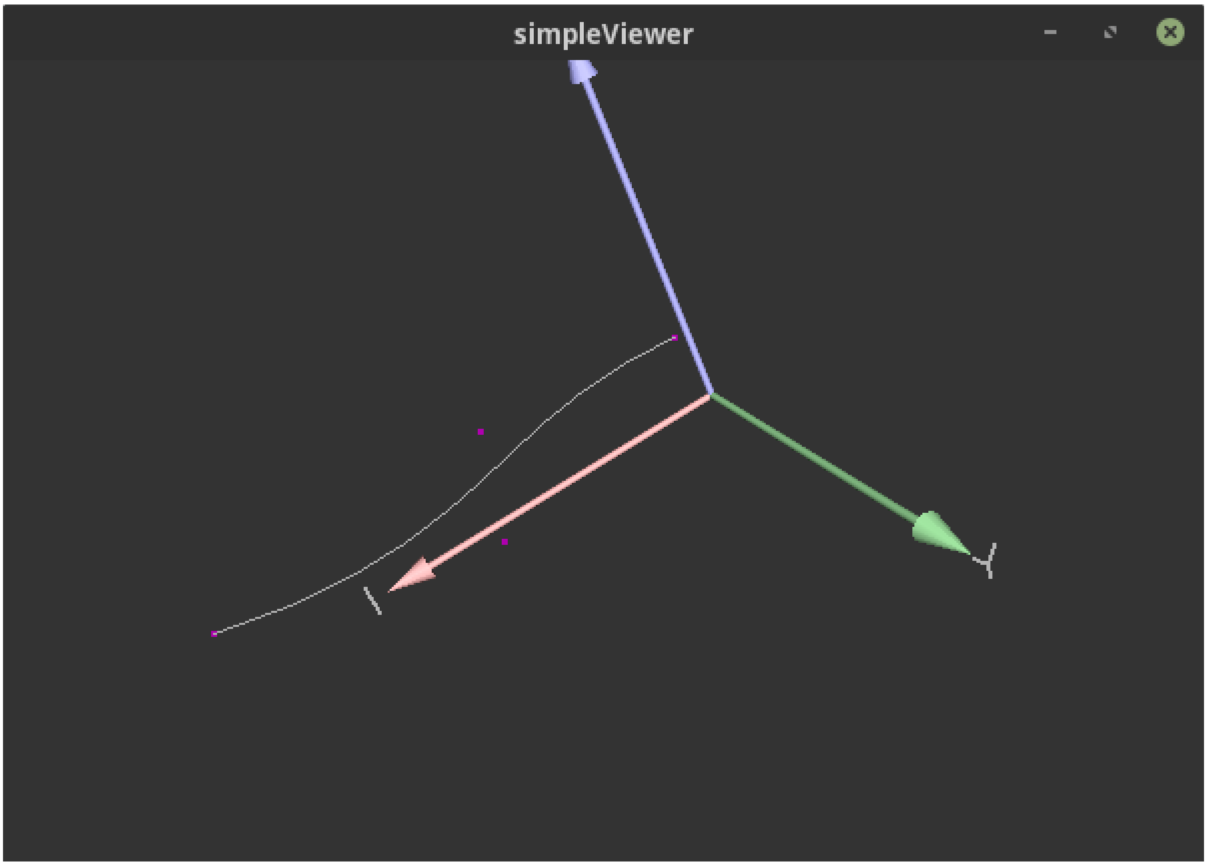
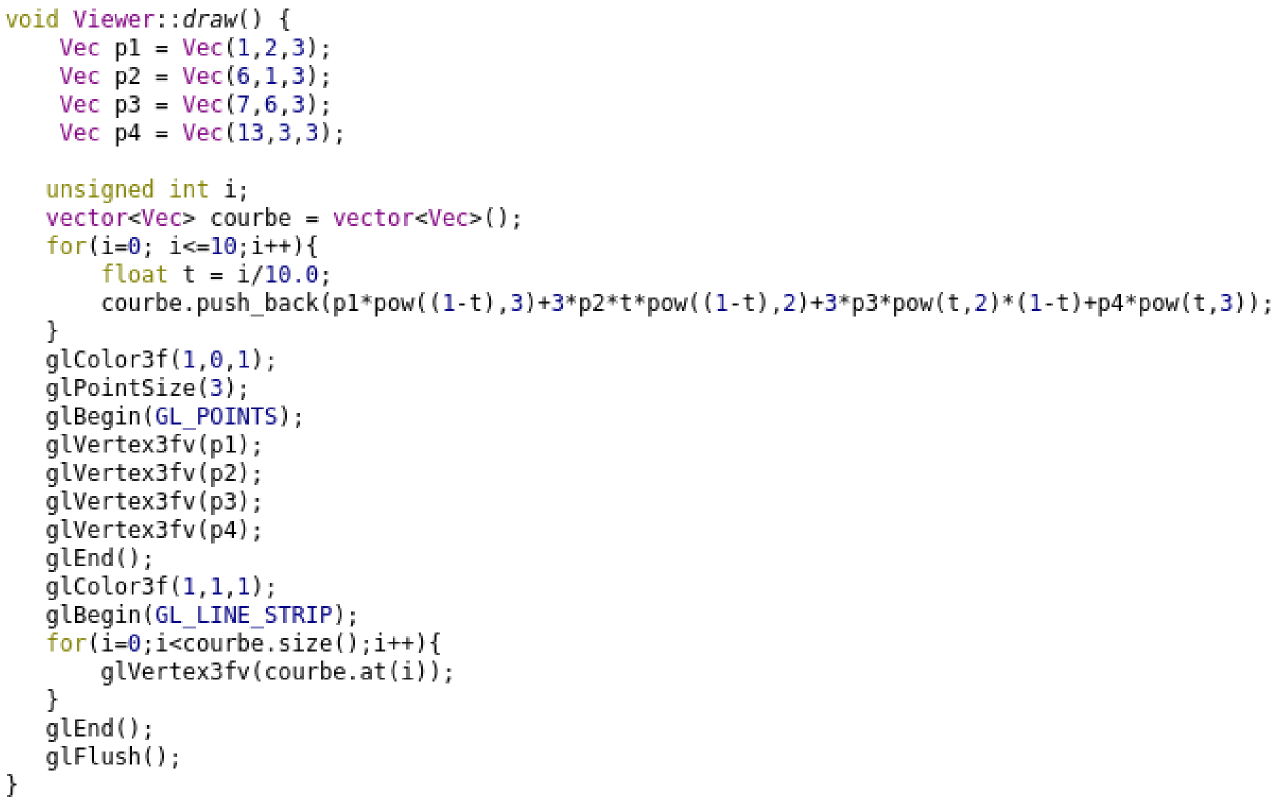
On utilise l’équation de la courbe de Bézier cubique.

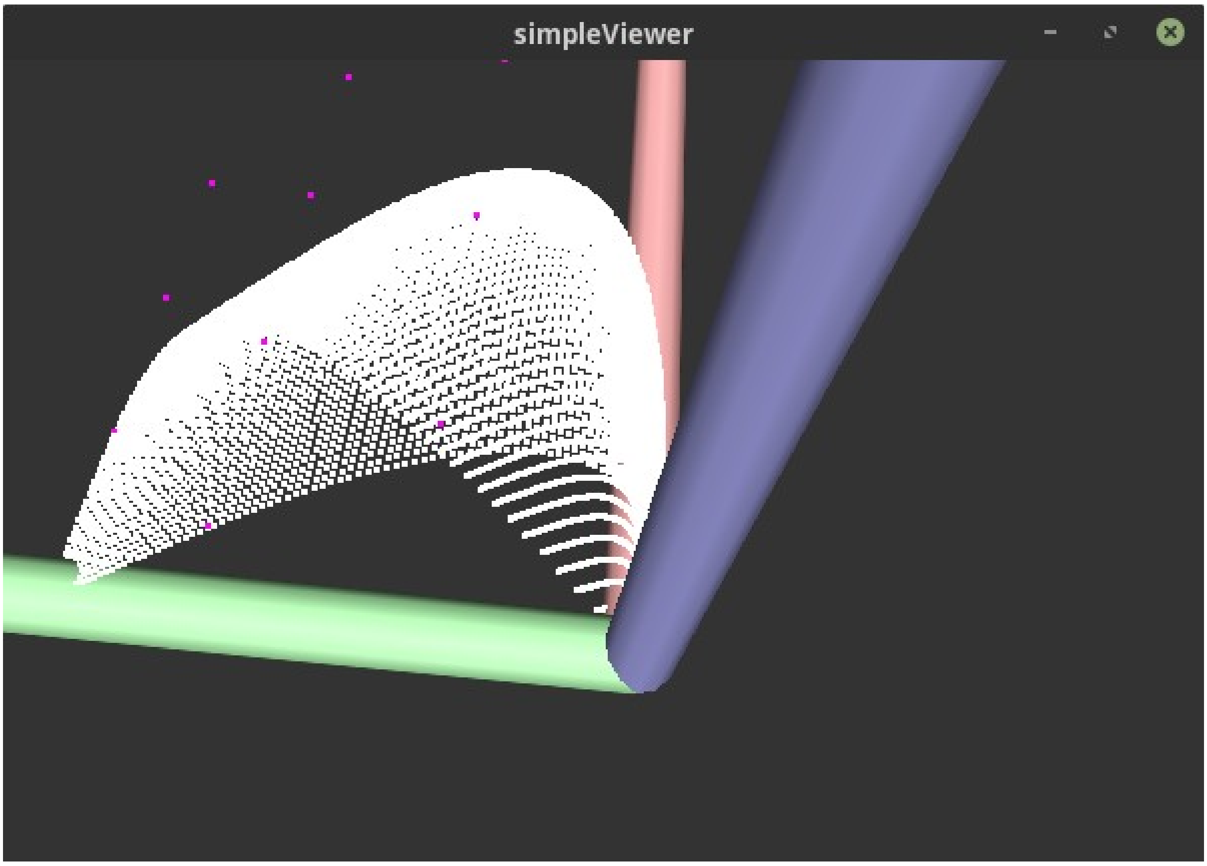


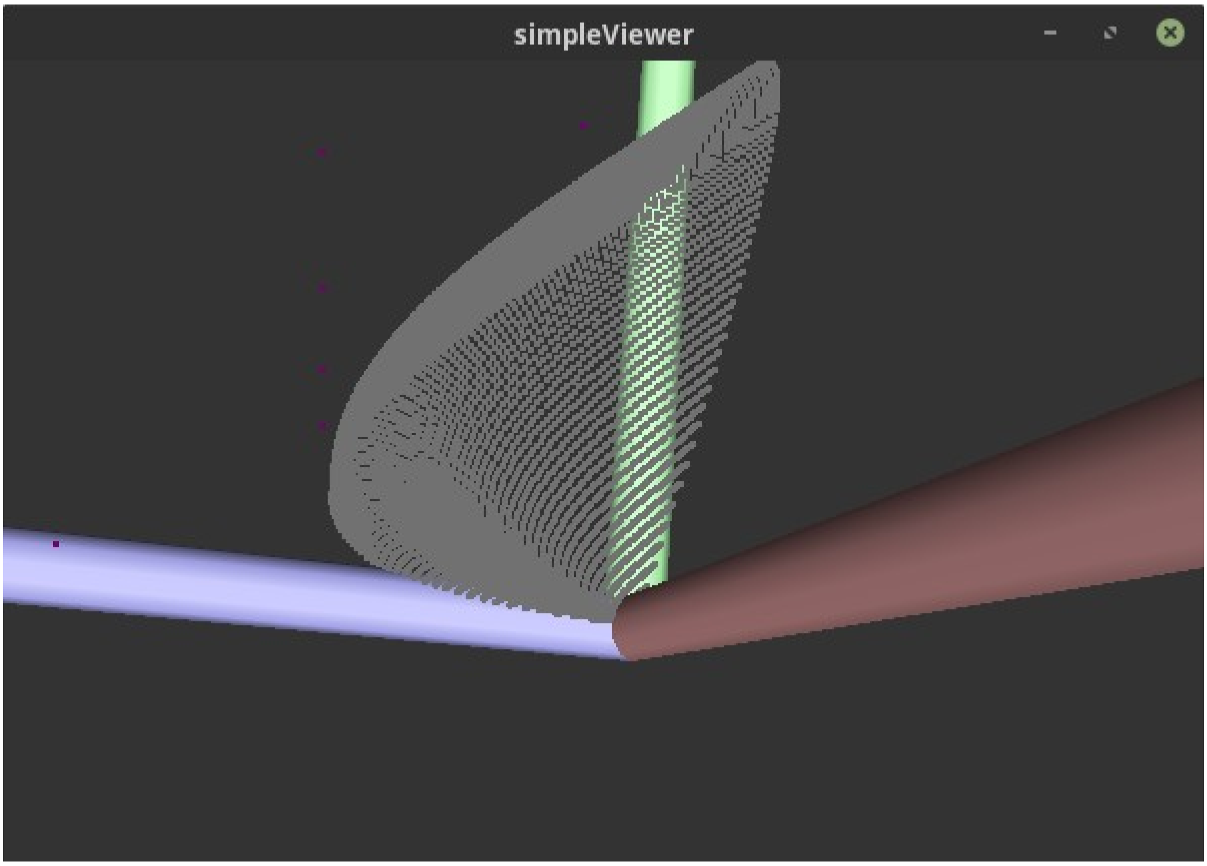


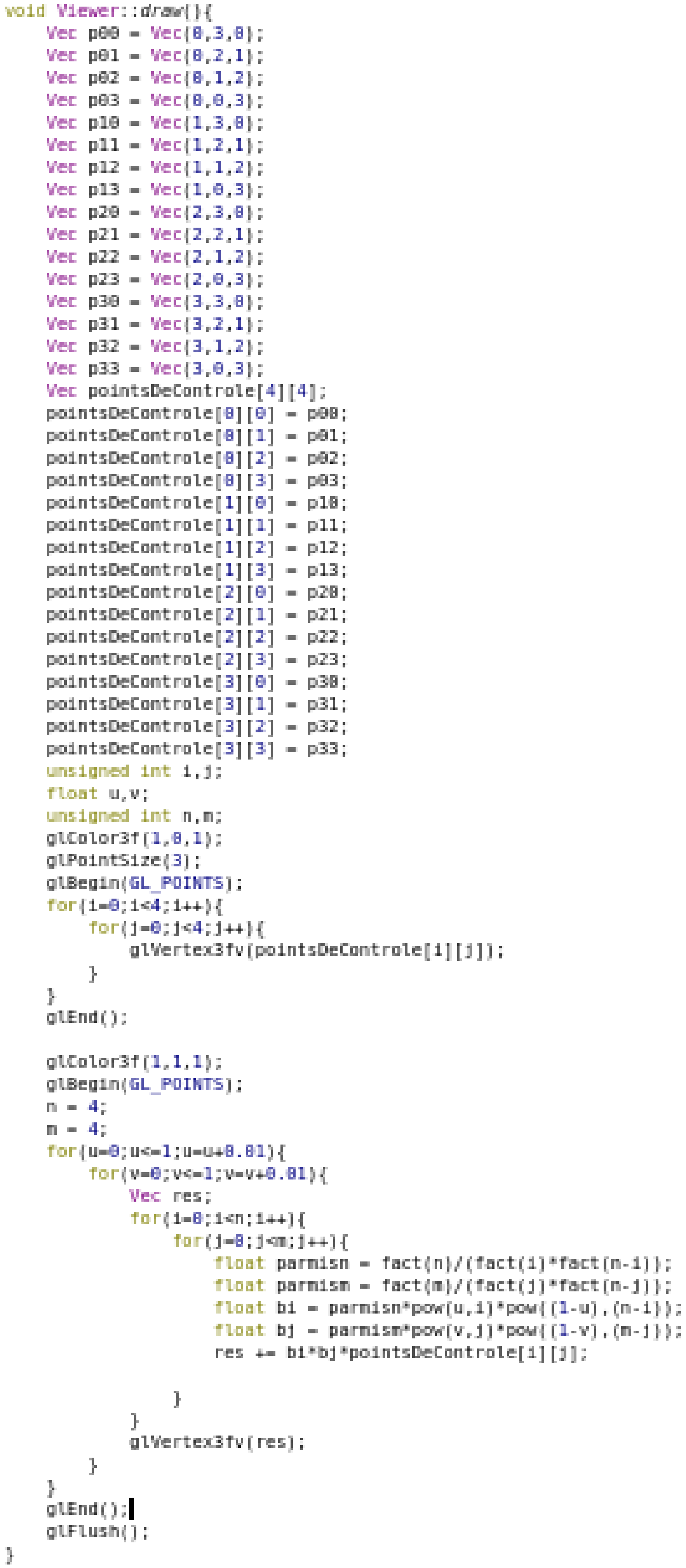
```
void Viewer::draw() {
    Vec p1 = Vec(1,2,3);
    Vec p2 = Vec(6,1,3);
    Vec p3 = Vec(7,6,3);
    Vec p4 = Vec(13,3,3);

   unsigned int i;
   vector<Vec> courbe = vector<Vec>();
   for(i=0; i<=10;i++){
       float t = i/10.0;
       courbe.push_back(p1*pow((1-t),3)+3*p2*t*pow((1-t),2)+3*p3*pow(t,2)*(1-t)+p4*pow(t,3));
   }
   glColor3f(1,0,1);
   glPointSize(3);
   glBegin(GL_POINTS);
   glVertex3fv(p1);
   glVertex3fv(p2);
   glVertex3fv(p3);
   glVertex3fv(p4);
   glEnd();
   glColor3f(1,1,1);
   glBegin(GL_LINE_STRIP);
   for(i=0;i<courbe.size();i++){
       glVertex3fv(courbe.at(i));
   }
   glEnd();
   glFlush();
}
```

On prend dessine une surface de Bézier :







```
void Viewer::draw(){
    Vec p00 = Vec(0,3,0);
    Vec p01 = Vec(0,2,1);
    Vec p02 = Vec(0,1,2);
    Vec p03 = Vec(0,0,3);
    Vec p10 = Vec(1,3,0);
    Vec p11 = Vec(1,2,1);
    Vec p12 = Vec(1,1,2);
    Vec p13 = Vec(1,0,3);
    Vec p20 = Vec(2,3,0);
    Vec p21 = Vec(2,2,1);
    Vec p22 = Vec(2,1,2);
    Vec p23 = Vec(2,0,3);
    Vec p30 = Vec(3,3,0);
    Vec p31 = Vec(3,2,1);
    Vec p32 = Vec(3,1,2);
    Vec p33 = Vec(3,0,3);
    Vec pointsDeControle[4][4];
    pointsDeControle[0][0] = p00;
    pointsDeControle[0][1] = p01;
    pointsDeControle[0][2] = p02;
    pointsDeControle[0][3] = p03;
    pointsDeControle[1][0] = p10;
    pointsDeControle[1][1] = p11;
    pointsDeControle[1][2] = p12;
    pointsDeControle[1][3] = p13;
    pointsDeControle[2][0] = p20;
    pointsDeControle[2][1] = p21;
    pointsDeControle[2][2] = p22;
    pointsDeControle[2][3] = p23;
    pointsDeControle[3][0] = p30;
    pointsDeControle[3][1] = p31;
    pointsDeControle[3][2] = p32;
    pointsDeControle[3][3] = p33;
    unsigned int i,j;
    float u,v;
    unsigned int n,m;
    glColor3f(1,0,1);
    glPointSize(3);
    glBegin(GL_POINTS);
    for(i=0;i<4;i++){
        for(j=0;j<4;j++){
            glVertex3fv(pointsDeControle[i][j]);
        }
    }
    glEnd();

    glColor3f(1,1,1);
    glBegin(GL_POINTS);
    n = 4;
    m = 4;
    for(u=0;u<=1;u=u+0.01){
        for(v=0;v<=1;v=v+0.01){
            Vec res;
            for(i=0;i<n;i++){
                for(j=0;j<m;j++){
                    float parmisn = fact(n)/(fact(i)*fact(n-i));
                    float parmism = fact(m)/(fact(j)*fact(n-j));
                    float bi = parmisn*pow(u,i)*pow((1-u),(n-i));
                    float bj = parmism*pow(v,j)*pow((1-v),(m-j));
                    res += bi*bj*pointsDeControle[i][j];

                }
            }
            glVertex3fv(res);
        }
    }
    glEnd();
    glFlush();
}
```