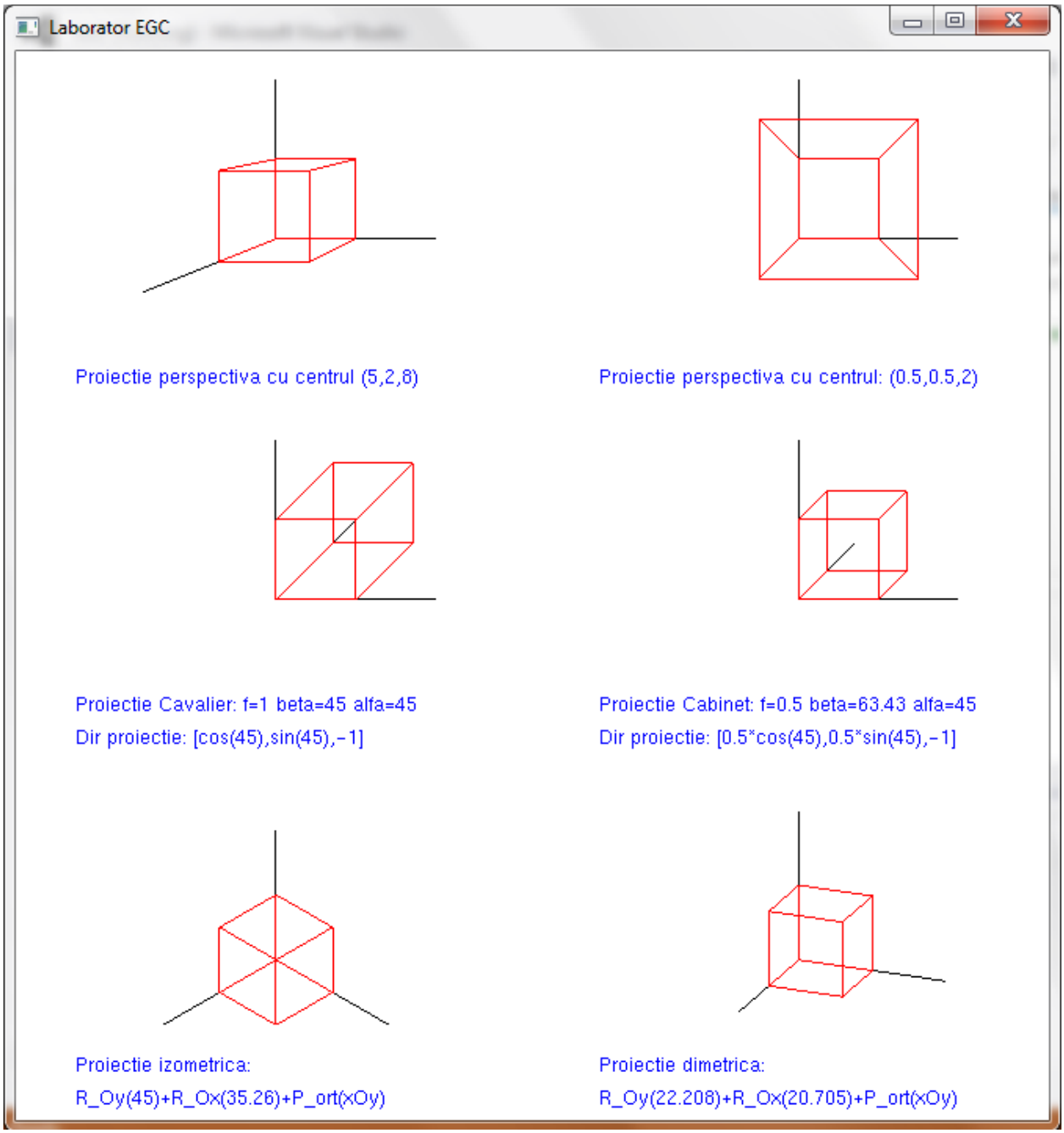
Program care afişează în fereastra aplicației 6 proiecții ale unui cub

In spațiul 3D cubul are laturile egale cu 1, fețele paralele cu planele principale ale sistemului de cordonate si colțul de (xmin, ymin, zmin) in originea sistemului de coordonate.



```
// Cele 6 proiectii se creaza in 6 contexte vizuale
Visual2D v2d1,v2d2,v2d3,v2d4,v2d5,v2d6;

Object3D cube;
PolyLine3D axa_ox, axa_oy, axa_oz;
Text text1, text2, text3, text4, text5, text6, text32, text42, text52, text62;
float n=1;  // latura cubului

//functia care permite adaugarea de obiecte
void DrawingWindow::init()
{
    //adaugare contexte vizuale

    v2d1.init(-2.5,-2,2.5,2.5,0,0,DrawingWindow::width/2,DrawingWindow::height/3);
    v2d2.init(-2.5,-
2,2.5,2.5,DrawingWindow::width/2,0,DrawingWindow::width,DrawingWindow::height/3);
v2d3.init(-2.5,-2,2.5,2.5,0,DrawingWindow::height/3,DrawingWindow::width/2, 2 *
                                                                    DrawingWindow::height/3);
    v2d4.init(-2.5,-
2,2.5,2.5,DrawingWindow::width/2,DrawingWindow::height/3,DrawingWindow::width, 2*
                                                                     DrawingWindow::height/3);
    v2d5.init(-2.5,-2,2.5,2.5,0,2 *
DrawingWindow::height/3,DrawingWindow::width/2,DrawingWindow::height);
    v2d6.init(-2.5,-2,2.5,2.5,DrawingWindow::width/2,2 *
DrawingWindow::height/3,DrawingWindow::width,DrawingWindow::height);

    v2d1.tipTran(true);
    v2d2.tipTran(true);
    v2d3.tipTran(true);
    v2d4.tipTran(true);
    v2d5.tipTran(true);
    v2d6.tipTran(true);

//creare cub
    vector <Point3D> vertices;
    vector <Face> faces;

  //varfurile de jos
    vertices.push_back( Point3D(0,0,0));
   vertices.push_back( Point3D(n,0,0));
```

```
   vertices.push_back( Point3D(n,0,n));
   vertices.push_back( Point3D(0,0,n));
//varfurile de sus
   vertices.push_back( Point3D(0,n,0));
   vertices.push_back( Point3D(n,n,0));
   vertices.push_back( Point3D(n,n,n));
   vertices.push_back( Point3D(0,n,n));

//cele 6 fete
   vector <int> contour;
  //fata jos
        contour.clear();
        contour.push_back(0);
        contour.push_back(1);
        contour.push_back(2);
        contour.push_back(3);
        faces.push_back( Face(contour));
//fata sus
        contour.clear();
        contour.push_back(4);
        contour.push_back(5);
        contour.push_back(6);
        contour.push_back(7);
        faces.push_back( Face(contour));
//fata fata
        contour.clear();
        contour.push_back(0);
        contour.push_back(1);
        contour.push_back(5);
        contour.push_back(4);
        faces.push_back( Face(contour));
//fata dreapta
        contour.clear();
        contour.push_back(1);
        contour.push_back(2);
        contour.push_back(6);
        contour.push_back(5);
        faces.push_back( Face(contour));
//fata spate
        contour.clear();
```

```
        contour.push_back(2);
        contour.push_back(3);
        contour.push_back(7);
        contour.push_back(6);
        faces.push_back( Face(contour));
//fata stanga
        contour.clear();
        contour.push_back(3);
        contour.push_back(0);
        contour.push_back(4);
        contour.push_back(7);
        faces.push_back( Face(contour));

 cube.init(vertices,faces,Color(1,0,0),false);

// adauga cubul in cele 6 contexte vizuale
        addObject3D_to_Visual2D(cube,v2d1);
        addObject3D_to_Visual2D(cube,v2d2);
        addObject3D_to_Visual2D(cube,v2d3);
        addObject3D_to_Visual2D(cube,v2d4);
        addObject3D_to_Visual2D(cube,v2d5);
        addObject3D_to_Visual2D(cube,v2d6);

//Creaza axele sistemului de coordonate 3D
        vector <Point3D> vertices_Ox;
        vector <Point3D> vertices_Oy;
        vector <Point3D> vertices_Oz;

        vertices_Ox.push_back( Point3D(0,0,0));
        vertices_Ox.push_back( Point3D(2,0,0));

        vertices_Oy.push_back( Point3D(0,0,0));
        vertices_Oy.push_back( Point3D(0,2,0));

        vertices_Oz.push_back( Point3D(0,0,0));
        vertices_Oz.push_back( Point3D(0,0,2));

        axa_ox.init(vertices_Ox,Color(0,0,0));
        axa_oy.init(vertices_Oy,Color(0,0,0));
        axa_oz.init(vertices_Oz,Color(0,0,0));
```

```
// adauga axele in cele 6 contexte vizuale
        addObject3D_to_Visual2D(axa_ox,v2d1);
        addObject3D_to_Visual2D(axa_oy,v2d1);
        addObject3D_to_Visual2D(axa_oz,v2d1);

        addObject3D_to_Visual2D(axa_ox,v2d2);
        addObject3D_to_Visual2D(axa_oy,v2d2);
        addObject3D_to_Visual2D(axa_oz,v2d2);

        addObject3D_to_Visual2D(axa_ox,v2d3);
        addObject3D_to_Visual2D(axa_oy,v2d3);
        addObject3D_to_Visual2D(axa_oz,v2d3);

        addObject3D_to_Visual2D(axa_ox,v2d4);
        addObject3D_to_Visual2D(axa_oy,v2d4);
        addObject3D_to_Visual2D(axa_oz,v2d4);

        addObject3D_to_Visual2D(axa_ox,v2d5);
        addObject3D_to_Visual2D(axa_oy,v2d5);
        addObject3D_to_Visual2D(axa_oz,v2d5);

        addObject3D_to_Visual2D(axa_ox,v2d6);
        addObject3D_to_Visual2D(axa_oy,v2d6);
        addObject3D_to_Visual2D(axa_oz,v2d6);

//text
        text1.init("Proiectie perspectiva cu centrul (5,2,8)",Point2D(-2.5,-
1.8),Color(0,0,1),BITMAP_HELVETICA_12);
        text2.init("Proiectie perspectiva standard: (0,0,-d)",Point2D(-2.5,-
1.8),Color(0,0,1),BITMAP_HELVETICA_12);
        text3.init("Proiectie Cavalier: f=1 beta=45 alfa=45",Point2D(-2.5,-
1.4),Color(0,0,1),BITMAP_HELVETICA_12);
        text32.init("Dir proiectie: [-cos(45),-sin(45),-1]",Point2D(-2.5,-
1.8),Color(0,0,1),BITMAP_HELVETICA_12);
        text4.init("Proiectie Cabinet: f=0.5 beta=63.43 alfa=45",Point2D(-2.5,-
1.4),Color(0,0,1),BITMAP_HELVETICA_12);
        text42.init("Dir proiectie: [-0.5*cos(45),-0.5*sin(45),-1]",Point2D(-2.5,-
1.8),Color(0,0,1),BITMAP_HELVETICA_12);
```

```
        text5.init("Proiectie izometrica:",Point2D(-2.5,-
1.4),Color(0,0,1),BITMAP_HELVETICA_12);
        text52.init("R_Oy(45)+R_Ox(35.26)+P_ort(xOy)",Point2D(-2.5,-
1.8),Color(0,0,1),BITMAP_HELVETICA_12);
        text6.init("Proiectie dimetrica:",Point2D(-2.5,-
1.4),Color(0,0,1),BITMAP_HELVETICA_12);
        text62.init("R_Oy(22.208)+R_Ox(20.705)+P_ort(xOy)",Point2D(-2.5,-
1.8),Color(0,0,1),BITMAP_HELVETICA_12);

        addText_to_Visual2D(text1,v2d1);
        addText_to_Visual2D(text2,v2d2);
        addText_to_Visual2D(text3,v2d3);
        addText_to_Visual2D(text32,v2d3);
        addText_to_Visual2D(text4,v2d4);
        addText_to_Visual2D(text42,v2d4);
        addText_to_Visual2D(text5,v2d5);
        addText_to_Visual2D(text52,v2d5);
        addText_to_Visual2D(text6,v2d6);
        addText_to_Visual2D(text62,v2d6);


        int i;
        Transform3D::loadIdentityModelMatrix();

//primul cub - proiectie perspectiva cu centru oarecare de proiectie
        Transform3D::perspectiveProjectionMatrix(5,2,8);
        for (i = 0; i < v2d1.visual2D_objects3D.size(); i++)
        {
                Transform3D::applyTransform(v2d1.visual2D_objects3D[i]);
        }

//al doilea cub - proiectie perspectiva cu centru oarecare de proiectie
        Transform3D::perspectiveProjectionMatrix(0.5,0.5,2);
        for (i = 0; i < v2d2.visual2D_objects3D.size(); i++)
        {
                Transform3D::applyTransform(v2d2.visual2D_objects3D[i]);
        }

//al treilea cub - proiectie Cavalier: f = 1, alfa = 45 grade => a = cos(PI/4),
// b = sin(PI/4), c = - 1
```

```
		Transform3D::parallelProjectionMatrix(cos(PI/4), sin(PI/4), -1);
		// sau Transform3D::obliqueProjectionMatrix (1, 45);
		for (i = 0; i < v2d3.visual2D_objects3D.size(); i++)
		{
			Transform3D::applyTransform(v2d3.visual2D_objects3D[i]);
		}

// al patrulea cub - proiectie Cabinet: f = 0.5, alfa = 45 grade => a = 0.5 * cos(PI/4),
// b = 0.5 * sin(PI/4), c = - 1
		Transform3D::parallelProjectionMatrix(0.5 * cos(PI/4), 0.5 * sin(PI/4),-1);
		// sau Transform3D::obliqueProjectionMatrix (0.5, 45);
		for (i = 0; i < v2d4.visual2D_objects3D.size(); i++)
		{
			Transform3D::applyTransform(v2d4.visual2D_objects3D[i]);
		}
//al cincilea cub - proiectie izometrica
		//rotatie fata de Oy (cu unghiul 45 grade),
		// urmata de rotatie fata de Ox (cu unghiul 35.26 grade)
		// urmata de proiectie ortografica in planul xOy
		Transform3D::loadIdentityModelMatrix();
		Transform3D::rotateMatrixOy(45.0f * PI/180.0f);
		Transform3D::rotateMatrixOx(35.26f * PI/180.0f);
		Transform3D::parallelProjectionMatrix(0,0,1);
		// sau Transform3D::axonometricProjectionMatrix(35.26, 45);
		for (i = 0; i < v2d5.visual2D_objects3D.size(); i++)
		{
			Transform3D::applyTransform(v2d5.visual2D_objects3D[i]);
		}

//al saselea cub - proiectie dimetrica
		//rotatie fata de Oy (cu unghiul 22.208 grade),
		// urmata de rotatie fata de Ox (cu unghiul 20.705 grade)
		// urmata de proiectie ortografica in planul xOy
		Transform3D::loadIdentityModelMatrix();
		Transform3D::rotateMatrixOy(22.208f * PI/180.0f);
		Transform3D::rotateMatrixOx(20.705f * PI/180.0f);
		Transform3D::parallelProjectionMatrix(0,0,1);
		// sau Transform3D::axonometricProjectionMatrix(20.705, 22.208);

		for (i = 0; i < v2d6.visual2D_objects3D.size(); i++)
```

```
        {
                Transform3D::applyTransform(v2d6.visual2D_objects3D[i]);
        }
// adauga cele 6 contexte vizuale in fereastra curenta
        addVisual2D(v2d1);
        addVisual2D(v2d2);
        addVisual2D(v2d3);
        addVisual2D(v2d4);
        addVisual2D(v2d5);
        addVisual2D(v2d6);

}

//functia care se apeleaza la redimensionarea ferestrei
void DrawingWindow::onReshape(int width,int height)
{
        v2d1.poarta(0,0,width/2,height/3);
        v2d2.poarta(width/2,0,width,height/3);

        v2d3.poarta(0,height/3,width/2,2 * height/3);
        v2d4.poarta(width/2,height/3,width,2 * height/3);

        v2d5.poarta(0,2 * height/3,width/2,height);
        v2d6.poarta(width/2,2 * height/3,width,height);
}

int main(int argc, char** argv)
{
   //creare fereastra
        DrawingWindow dw(argc, argv, 600, 600, 200, 50, "Laborator EGC");
        //se apeleaza functia init() - in care s-au adaugat obiecte
        dw.init();
  //se intra in bucla principala de desenare
        dw.run();
        return 0;

}
```