Московский авиационный институт
(национальный исследовательский университет)

Институт №8 «Информационные технологии и прикладная математика»

Кафедра 806 «Вычислительная математика и программирование»

**Лабораторная работа №2 по курсу «Компьютерная графика»**

Студент: М. А. Инютин
Преподаватель: А. В. Морозов
Группа: М8О-307Б-19
Дата:
Оценка:
Подпись:

**Москва, 2021**

# Каркасная визуализация выпуклого многогранника. Удаление невидимых линий

**Задача:** Разработать формат представления многогранника и процедуру его каркасной отрисовки в ортографической и изометрической проекциях. Обеспечить удаление невидимых линий и возможность пространственных поворотов и масштабирования многогранника. Обеспечить автоматическое центрирование и изменение размеров изображения при изменении размеров окна.

**Вариант задания:** Восьмигранная прямая правильная призма.

# 1 Описание

Для параметризации правильной призмы нужно параметризовать правильный многоугольник, лежащий в основаниях призмы. Точки правильного многоугольника расположены на окружности, поэтому можно составить уравнение окружности и вычислять координаты вершин многоугольника.

Вся окружность составляет угол $2 \cdot \pi$ радиан, тогда для правильного $n$-угольника угол поворота между соседними вершинами $\Delta\phi = \frac{2 \cdot \pi}{n}$, причём $\phi_0 = 0$.

После генерации многоугольника следует переместить его параллельно оси $Oz$ и соединить точки в правильном порядке, чтобы задать полигоны. Они должны быть заданы единообразно, чтобы вектора нормали смотрели в одну и ту же сторону относительно плоскости.

# 2 Исходный код

Файл *Figure.cs* содержит код для генерации фигуры.

```
1  public Prism(int n, double r, double h)
2  {
3      dPhi = Misc.MAX_DEG / (double)n;
4      polygonsCount = 4 * n;
5      baseVerticiesCount = n;
6      low = new List<Vector4D>(baseVerticiesCount);
7      high = new List<Vector4D>(baseVerticiesCount);
8      _polygons = new List<Polygon>(polygonsCount);
9      GenVertices(r, h);
10     GenFigure();
11     GenVertexPolygons();
12 }
13
14 private void GenVertices(double r, double h)
15 {
16     _vertices = new List<VertexPolygonsPair>(2 * baseVerticiesCount + 2);
17     centerLow = new Vector4D(0, 0, 0, 1);
18     centerHigh = new Vector4D(0, 0, h, 1);
19     _vertices.Add(new VertexPolygonsPair(centerLow));
20     _vertices.Add(new VertexPolygonsPair(centerHigh));
21     double phi = 0;
22     for (int i = 0; i < baseVerticiesCount; ++i)
23     {
24         Vector4D vertex = new Vector4D(r * Math.Cos(Misc.ToRadians(phi)), r * Math.Sin(
               Misc.ToRadians(phi)), 0, 0);
25         low.Add(vertex + centerLow);
26         high.Add(vertex + centerHigh);
27         phi = phi + dPhi;
28     }
29     for (int i = 0; i < baseVerticiesCount; ++i)
30     {
31         _vertices.Add(new VertexPolygonsPair(low[i]));
32         _vertices.Add(new VertexPolygonsPair(high[i]));
33     }
34 }
35
36 private void GenFigure()
37 {
38     for (int i = 0; i < baseVerticiesCount; ++i)
39     {
40         Vector4D a = low[i];
41         Vector4D b = high[i];
42         Vector4D c = high[(i + 1) % baseVerticiesCount];
43         Vector4D d = low[(i + 1) % baseVerticiesCount];
44         _polygons.Add(new Polygon(d, c, a));
45         _polygons.Add(new Polygon(b, a, c));
```

```
46          _polygons.Add(new Polygon(centerLow, d, a));
47          _polygons.Add(new Polygon(centerHigh, b, c));
48      }
49  }
```

*Geometry.cs* содержит классы векторов и матрицы, методы для работы с ними.

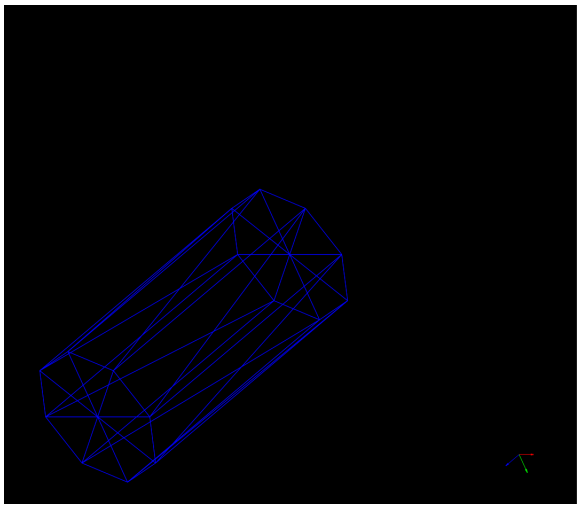
Файл *MainWindow.cs* содержит интерфейс и методы отрисовки фигуры.

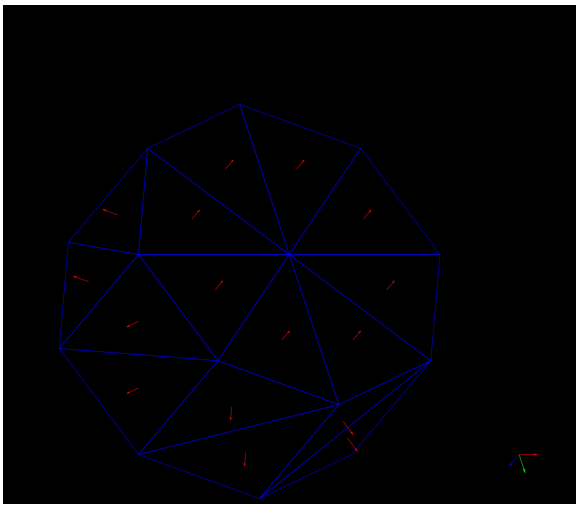
```
50  private void DrawFigure(Context cr)
51  {
52      for (int i = 0; i < prism._polygons.Count; ++i)
53      {
54          if (!ignoreInvisible || prism._polygons[i].Visible()) {
55              if (fillPolygons)
56              {
57                  FillPolygon(cr, i, polygonColors[i]);
58                  if (drawFrame)
59                  {
60                      DrawPolygon(cr, i, DEFAULT_LINE_COLOR_FILL);
61                  }
62              }
63              else if (drawFrame)
64              {
65                  DrawPolygon(cr, i, DEFAULT_LINE_COLOR);
66              }
67          }
68      }
69  }
70
71  private void DrawAxises(Context cr, Vector2D shift)
72  {
73      Matrix4D matr = new Matrix4D();
74      if (_radioButtonIsometric.Active)
75      {
76          matr = matr * Matrix4D.RotX(ISOMETRIC_X) * Matrix4D.RotY(ISOMETRIC_Y) *
                Matrix4D.RotZ(ISOMETRIC_Z);
77      }
78      else
79      {
80          matr = matr * Matrix4D.RotX(alpha) * Matrix4D.RotY(beta) * Matrix4D.RotZ(gamma)
                ;
81      }
82      Vector4D start = new Vector4D();
83      Vector4D ox = new Vector4D(1, 0, 0, 0) * matr * AXIS_SIZE;
84      Vector4D oy = new Vector4D(0, 1, 0, 0) * matr * AXIS_SIZE;
85      Vector4D oz = new Vector4D(0, 0, 1, 0) * matr * AXIS_SIZE;
86      Cairo.Color oxColor = new Cairo.Color(1, 0, 0);
87      Cairo.Color oyColor = new Cairo.Color(0, 1, 0);
88      Cairo.Color ozColor = new Cairo.Color(0, 0, 1);
89      DrawVector(cr, shift + start.Proj(), shift + ox.Proj(), oxColor);
```

```
90      DrawVector(cr, shift + start.Proj(), shift + oy.Proj(), oyColor);
91      DrawVector(cr, shift + start.Proj(), shift + oz.Proj(), ozColor);
92  }
93
94  private void DrawNormalVectors(Context cr, Prism prism)
95  {
96      for (int i = 0; i < prism._polygons.Count; ++i)
97      {
98          if (!ignoreInvisible || prism._polygons[i].Visible()) {
99              cr.SetSourceColor(DEFAULT_NORMAL_COLOR);
100             DrawNormal(cr, prism._polygons[i]);
101         }
102     }
103 }
104
105 private void FillPolygon(Context cr, int id, Cairo.Color col)
106 {
107     Polygon poly = prism._polygons[id];
108     Vector4D vertex = poly[0];
109     cr.NewPath();
110     cr.LineWidth = 1;
111     cr.SetSourceRGB(0, 0, 0);
112     cr.MoveTo(vertex.X + windowCenter.X, vertex.Y + windowCenter.Y);
113     for (int i = 1; i < poly.Count; ++i)
114     {
115         vertex = poly[i];
116         cr.LineTo(vertex.X + windowCenter.X, vertex.Y + windowCenter.Y);
117     }
118     cr.ClosePath();
119     cr.SetSourceColor(col);
120     cr.Fill();
121 }
122
123 private void DrawPolygon(Context cr, int id, Cairo.Color col)
124 {
125     Polygon poly = prism._polygons[id];
126     for (int i = 0; i < poly.Count; ++i)
127     {
128         Vector4D a = poly[i];
129         Vector4D b = poly[(i + 1) % poly.Count];
130         DrawLine(cr, windowCenter + a.Proj(), windowCenter + b.Proj(), col);
131     }
132 }
```

# 3 Демонстрация работы программы

Каркасная визуализация призмы. Предусмотрена возможность удаления невидимых линий и отрисовки нормалей к поверхности.





Закраска полигонов. Предусмотрена возможность закраски полигонов в случайные цвета.