作业三：曲线与网格

代码目录结构如下：

```
code/
├── main.py                          # 主程序入口
├── bezier_circle.py                 # Task 1: Bezier曲线绘制圆（待完成）
├── halfedge_mesh.py                 # Task 2.1: 半边数据结构
├── loop_subdivision.py              # Task 2.2: Loop细分算法（待完成）
│
├── bunny.obj                        # 兔子模型
│
└── requirements.txt                 # 依赖库
```

主要考核：Bezier 曲线、网格表示、半边结构、网格细分等。具体任务要求如下：

1. 利用分段 Bezier 曲线表示一个圆，简要描述原理，并用 de Casteljau 算法绘制封闭曲线及控制点。(5 分)

   · 修改 bezier_circle.py 里的第 68 行的_create_segments 函数，计算每段 Bezier 曲线的控制点；

```python
def _create_segments(self):
    """
    TODO: 创建分段Bezier曲线来逼近圆

    提示:
    1. 将圆分成num_segments段
    2. 每段使用三次Bezier曲线（4个控制点）
    3. 对于圆，可以使用以下控制点布局:
       - P0: 段起点（在圆上）
       - P1: 起点处的切线方向控制点
       - P2: 终点处的切线方向控制点
       - P3: 段终点（在圆上）
    4. 关键参数: 对于n段圆，每段圆弧对应角度为2π/n
    """


    #for i in range(self.num_segments):
    #    # 创建Bezier曲线段
    #    control_points = np.array([p0, p1, p2, p3])
    #    self.segments.append(BezierCurve(control_points))

    raise NotImplementedError("请计算分段Bezier曲线的控制点列表")
```

   · 修改 bezier_circle.py 里的第 18 行的 de_casteljau 函数，计算 Bezier 曲线上采样点的位置。

```
def de_casteljau(self, t):
    """
    TODO: de Casteljau算法计算曲线上的点

    Args:
        t: 参数值，范围[0, 1]

    Returns:
        numpy array of shape (2,): 曲线上t对应的点坐标

    提示:
    1. 使用递归或迭代的方式实现de Casteljau算法
    2. 每次迭代，相邻控制点线性插值
    3. 最终得到一个点即为曲线上的点
    """
    raise NotImplementedError("请实现de_casteljau算法")
```

2. 将三角网格应用 Loop 细分算法（2 轮 Loop 细分）。(5 分)

· 修改 loop_subdivision.py 里的第 5 行的 compute_old_vertex_position 函数，实现 Loop 细分中旧点的位置更新；

```
def compute_old_vertex_position(old_mesh, old_vertex):
    """
    TODO: 计算旧顶点的新位置（Loop细分公式）

    Args:
        old_mesh: 半边结构的三角网格
        old_vertex: Vertex对象

    Returns:
        numpy array: 新位置坐标

    提示:
        1. 使用 TriangleMesh类的get_vertex_one_ring_neighbors(old_vertex) 获取邻居顶点列表
        2. 计算邻居数量 n = len(neighbors)
        3. 根据n的值计算权重u
        4. 应用Loop计算公式
    """
    raise NotImplementedError("请实现Loop细分中旧顶点的位置更新")
```

· 修改 loop_subdivision.py 里的第 27 行的 compute_edge_midpoint_position 函数，实现 Loop 细分中新插入点（边的中点）的位置更新。

```python
def compute_edge_midpoint_position(halfedge):
    """
    TODO: 计算边中点（新顶点）的位置（Loop细分公式）

    Args:
        halfedge: HalfEdge对象

    Returns:
        numpy array: 中点位置坐标

    提示:
        1. 获取边的起点和终点
        2. 获取边对应的两个对立点
        3. 应用Loop计算公式
    """
    raise NotImplementedError("请实现Loop细分中新顶点的位置更新")
```

作业提交要求：

i) 代码 + 报告

j) 报告里需简要说明每个考核点的实现过程，并给出最终的结果图。

k) 会根据代码实现细节和结果图进行给分。

l) 文件命名：学号+姓名+作业三.zip 上传地址为

https://bhpan.buaa.edu.cn/link/AAC1E75F220E684D27A4025913623F472B

文件夹名：作业三
有效期限：2026-01-05 23:59
提取码：jgh5