Programming assignment no. 2

(15pts)

Topic: Triangle Bézier patch

Deadline: 17 March 2020 – 12 April 2020

Goal: The goal of the second assignment is to:

1. draw three types of surfaces using the triangle Bézier patch of the degree n,
2. draw isoparametric curves of the patch for given parameters s, t and u.

Instructions:

3pts Implement procedures which draw the listed types of surfaces in the reserved space in the given template:

Surface interpolating the given set of points – let $\mathcal{I}$ be the set containing $\frac{(n+2)(n+1)}{2}$ points, which are distributed along a planar grid. Then the triangle Bézier patch of the degree n is constructed, so the set $\mathcal{I}$ is **interpolated**.

After modifying the bidegree or poisition of an interpolated point, the new patch is generated automatically, so it interpolates $\mathcal{I}$.

Surface approximating the spherical triangle and the conical sector – you may consider the same sectors as in the previous assignment (the cylindrical sector is not implemented in this assignment). The surface is constructed using the triangle Bézier patch of the **fixed** degree.

3pts In all cases, the tensor-product Bézier patch is rendered as a triangle mesh having $\frac{(k+2)(k+1)}{2}$ points, while each point is computed using **de Casteljau algorithm**. Also, the normal for each sampled point is compute. The **control net** of the patch needs to be drawn, i.e. the position of the control vertices needs to be computed.

For correct depiction of the sampled surface and coordinate net, the suitable sequence of vertex indices needs to be constructed, so each triplet determines a triangle. The guideline is included directly in the code.

The default surface is a triangle with vertices

$$(-1,-1,0)^\top,\ (1,-1,0)^\top,\ (0,1,0)^\top.$$

2pts Also it is possible to depict the **isoparametric curves** of the patch for the given parameters s, t and u. Each of the curves is represented as a polyline having $k+1$ points. Again, apart from the curves, also their control polygons need to be drawn, too.

The values s, t and u are loaded from the triangle drawn on the canvas named `Domain`, which depicts the domain of the patch. The coordinates of the domain with respect to the canvas (in pixels) are

$$(0,200)^\top,\ (200,200)^\top,\ (100,0)^\top.$$

marcel.makovnik@fmph.uniba.sk

Also, draw the line segments in the domain, which are mapped to the isoparametric curves for the parameters s, t and u. Also, ensure that the input is correct, i.e. the point, where the curves intersect is inside the domain triangle.

After modification of any parameter in the user interface, the surface and the domain is redrawn automatically.

Notes on the code: the comments `TODO` may be found in the following lines:

`MainWindow.xaml.cs - 104, 115, 144, 190`, also see the comment in the line `74`,

`Patch.cs - 135, 148, 161, 175, 186, 197, 206, 216`, also see the comment in the line `120`.

Output:

5pts Apart from the code implementing the patches (this part needs to be easily identifiable and created by the author, i.e. created explicitly for the purposes of this assignment), there is a discussion regarding the submitted solution. Details of the discussion may be found at `https://mkvnk.sk/GM1/#grading`.

Usage or integration of solely external libraries is strictly forbidden!

Apart from these, the code needs to be **richly** commented and **neatly** formatted. Poor commenting and ugly formatting might be penalized by loss up to *3 pts.*

Sample application and template is available on the webpage, together with this .pdf file.

The sample application lacks the control nets except for the case of the interpolating surface. However, your solution is supposed to render control nets for all four types of surfaces.