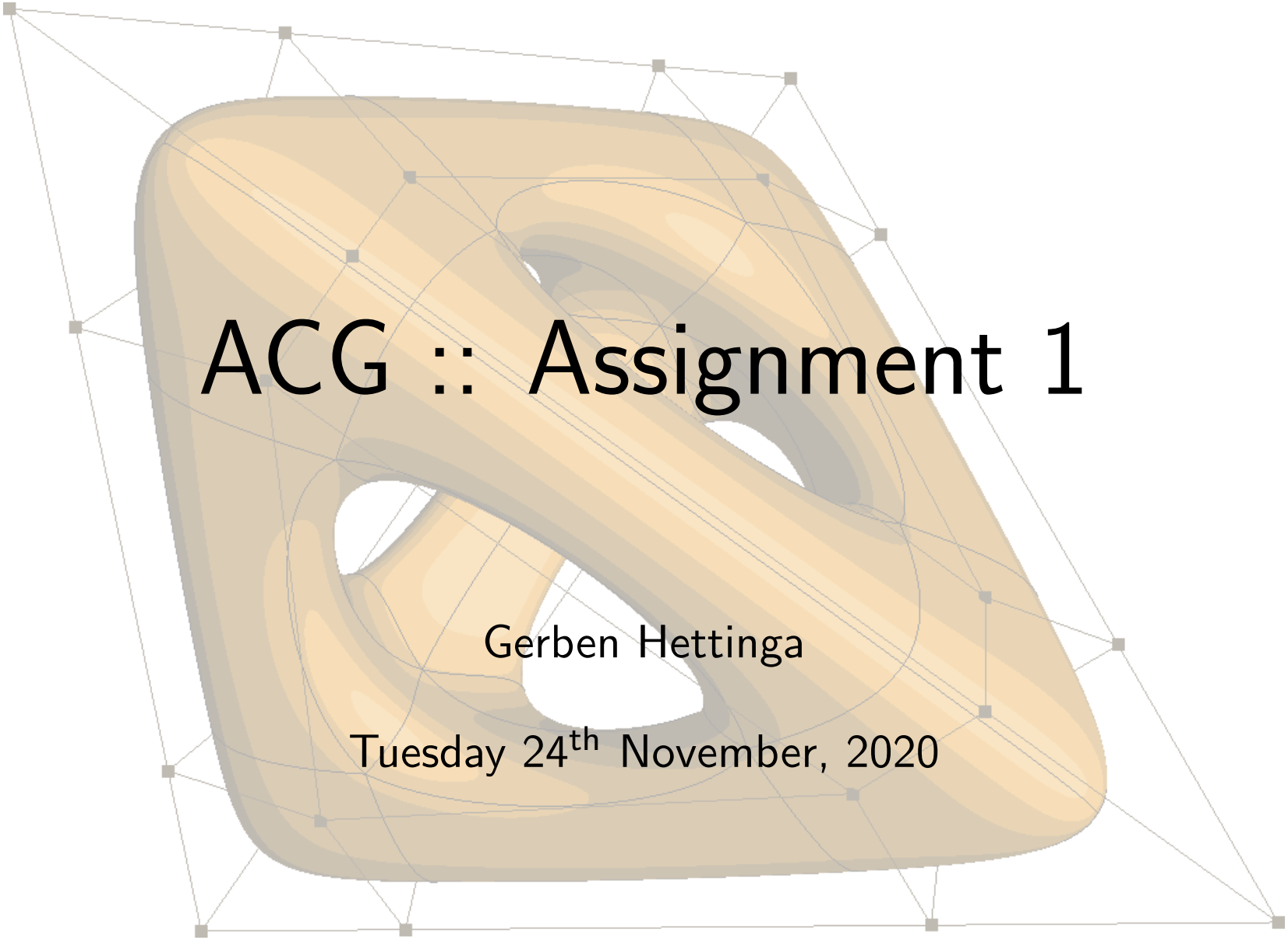


ACG :: Assignment 1

Gerben Hettinga

Tuesday 24th November, 2020

Assignment

MA General curve subdivision algorithm accepting binary subdivision masks (e.g. [1 4 6 4 1] $\rightarrow$ [1 6 1] and [4 4])

Assignment

MA General curve subdivision algorithm accepting binary subdivision masks (e.g. [1 4 6 4 1] $\rightarrow$ [1 6 1] and [4 4])

- Interpolation of the first and last control point is optional.

Assignment

MA General curve subdivision algorithm accepting binary subdivision masks (e.g. [1 4 6 4 1] $\rightarrow$ [1 6 1] and [4 4])

- Interpolation of the first and last control point is optional.

MA Visualise (discrete) curvature

Assignment

MA General curve subdivision algorithm accepting binary subdivision masks (e.g. [1 4 6 4 1] $\rightarrow$ [1 6 1] and [4 4])
- Interpolation of the first and last control point is optional.

MA Visualise (discrete) curvature
- Using e.g. curvature combs, (centres of) osculating circles, colour. For (more) inspiration, look at CAD software!

Assignment

MA General curve subdivision algorithm accepting binary subdivision masks (e.g. [1 4 6 4 1] $\rightarrow$ [1 6 1] and [4 4])
- Interpolation of the first and last control point is optional.

MA Visualise (discrete) curvature
- Using e.g. curvature combs, (centres of) osculating circles, colour. For (more) inspiration, look at CAD software!

AF Visualise the influence of a control point (...)
- Clearly indicate the boundaries of the influenced part.

Assignment

MA General curve subdivision algorithm accepting binary subdivision masks (e.g. [1 4 6 4 1] $\rightarrow$ [1 6 1] and [4 4])
- Interpolation of the first and last control point is optional.

MA Visualise (discrete) curvature
- Using e.g. curvature combs, (centres of) osculating circles, colour. For (more) inspiration, look at CAD software!

AF Visualise the influence of a control point (...)
- Clearly indicate the boundaries of the influenced part.

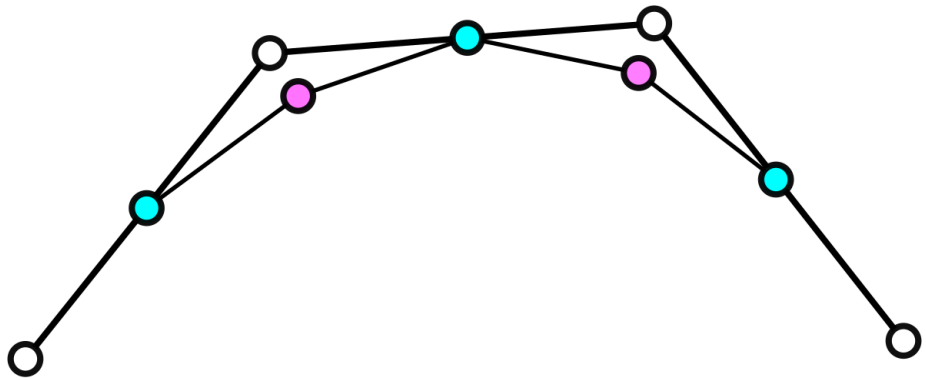
B Multiple ways to visualise curvature information

Curve Subdivision With Subdivision Masks

Consider an initial control polygon with n points P_i^0 $i = 0, \ldots n-1$
we can subdivide this curve with the stencils odd $[1, 6, 1]$ and even $[4, 4]$

$$P_{2i+1}^{k+1} = \frac{P_{i-1}^k \cdot 1 + P_i^k \cdot 6 + P_{i+1}^k \cdot 1}{8}$$

$$P_{2i}^{k+1} = \frac{P_{i-1} \cdot 4 + P_i \cdot 4}{8}$$



Binary Subdivision Masks: Derivation

Simplest possible mask: $[1, 1] \rightarrow [1]$ & $[1]$ (Identity)
Convolve mask to obtain higher order masks

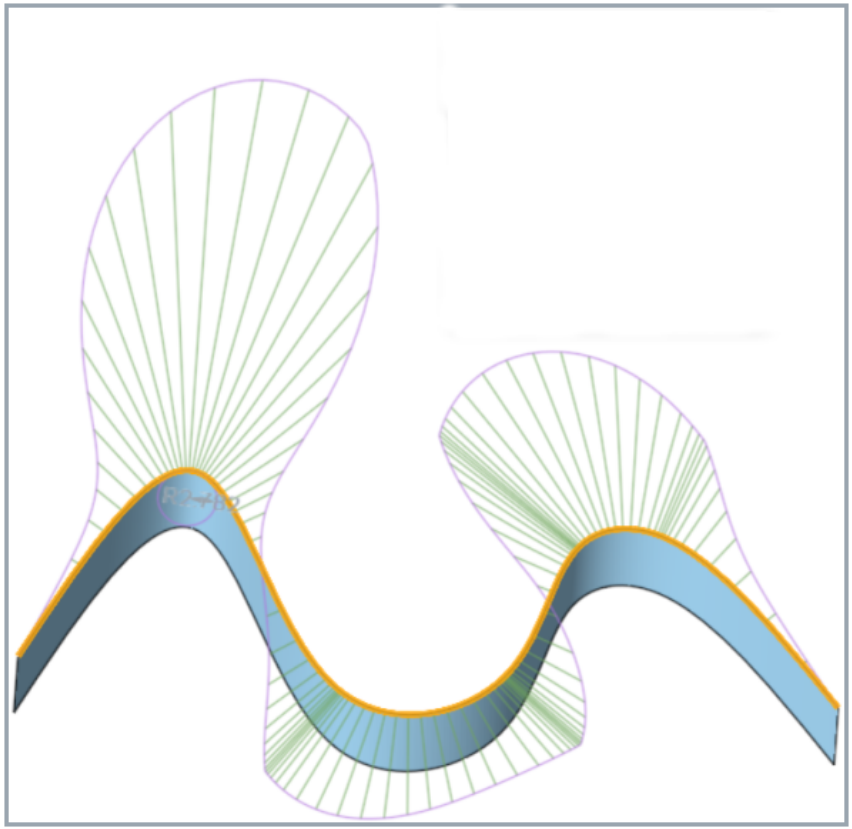
$$[1, 1] \star [1, 1] = ?$$

	1	1		
1	1			1
	1	1		2
		1	1	1

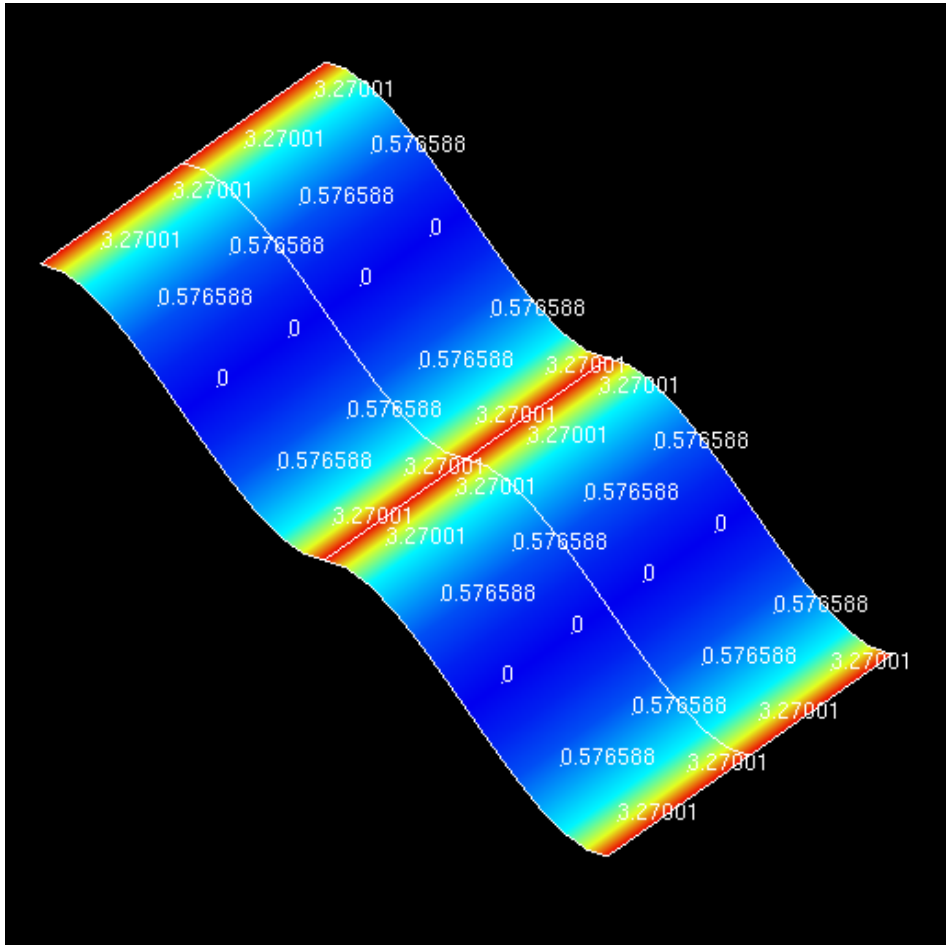
$$[1, 2, 1] \star [1, 1] = [1, 3, 3, 1]$$

$$[1, 3, 3, 1] \star [1, 1] = ?$$

Visualising Curvature



Curvature comb



Color map

How to compute curvature from a polyline? Consult the lecture slides: Spline and Subdivision Curves!

Geometry shaders

- Requires an input primitive (e.g. lines_adjacency) and output primitive (e.g. line_strip), including a value for max_vertices

Geometry shaders

- Requires an input primitive (e.g. lines_adjacency) and output primitive (e.g. line_strip), including a value for max_vertices
- Use the functions EmitVertex() and EndPrimitive()

Geometry shaders

- Requires an input primitive (e.g. lines_adjacency) and output primitive (e.g. line_strip), including a value for max_vertices
- Use the functions EmitVertex() and EndPrimitive()
- Built-in inputs gl_PrimitiveIDIn and gl_InvocationID might be useful
- After Vertex Shader & before Fragment Shader!

Geometry shaders

- Requires an input primitive (e.g. lines_adjacency) and output primitive (e.g. line_strip), including a value for max_vertices
- Use the functions EmitVertex() and EndPrimitive()
- Built-in inputs gl_PrimitiveIDIn and gl_InvocationID might be useful
- After Vertex Shader & before Fragment Shader!

Remember — data can be passed to the next shader based on name matching or location matching.

Geometry shaders

- Requires an input primitive (e.g. lines_adjacency) and output primitive (e.g. line_strip), including a value for max_vertices
- Use the functions EmitVertex() and EndPrimitive()
- Built-in inputs gl_PrimitiveIDIn and gl_InvocationID might be useful
- After Vertex Shader & before Fragment Shader!

Remember — data can be passed to the next shader based on name matching or location matching.

Some good sources (details and examples):
https://www.opengl.org/wiki/Geometry_Shader, http://www.lighthouse3d.com/tutorials/glsl-tutorial/geometry-shader/, http://learnopengl.com/#!Advanced-OpenGL/Geometry-Shader

Geometry shaders

- Input: polyline (lines_adjacency))
- Compute: (Discrete) Curvature
- Output:
  - Osculating circle(s): generate circle geometry
  - Curvature comb: generate teeth along normal direction scaled w.r.t. curvature
  - Colour: draw geometry colour mapped
  - Combinations

What and how to submit

- Read the relevant sections on Nestor (Lab Sessions)

What and how to submit

- Read the relevant sections on Nestor (Lab Sessions)
- Do not include files that are generated during compilation! For instance, these are the contents of the `.gitignore` file for this week's skeleton code —

```
*.pro.user
*.o
SubdivCurves
ui_mainwindow.h
moc_mainview.cpp
moc_mainwindow.cpp
qrc_resources.cpp
Makefile
.qmake.stash
```

Miscellaneous OpenGL/GLSL

- Separate buffers or interleaved ones? In the latter case, use the last two arguments of glVertexAttribPointer to set the stride and offset. See e.g. http://goharsha.com/lwjgl-tutorial-series/interleaving-buffer-objects/

Miscellaneous OpenGL/GLSL

- Separate buffers or interleaved ones? In the latter case, use the last two arguments of glVertexAttribPointer to set the stride and offset. See e.g. http://goharsha.com/lwjgl-tutorial-series/interleaving-buffer-objects/
- glBufferSubData can be used to update only part of a buffer, see e.g. https://www.opengl.org/wiki/GLAPI/glBufferSubData