Exam 2016 ICG

Cyril Wendl

June 16, 2017

Contents

1 Parametric Curves

1.1 De Casteljau's algorithm

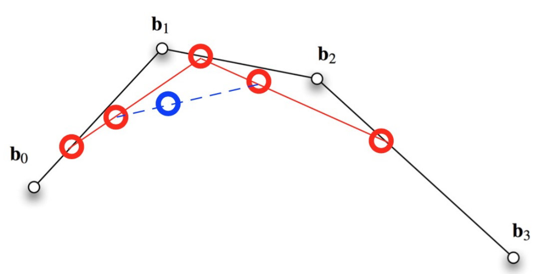
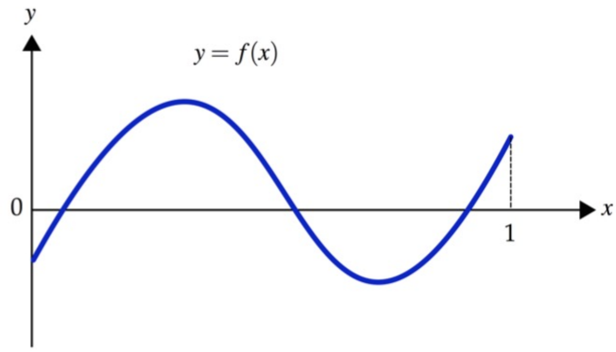


Figure 1.1: De Casteljau's algorithm

blue: point at t=.25 and tangent

1.2 Bézier curves - properties

(2 points) *Suppose the graph of a function $y = f(x)$ on interval [0, 1] can be represented by a Bézier curve $b(t)$. If there are 3 intersection points between $b(t)$ and the x-axis, what is the minimum degree of $b(t)$? Which property of Bézier curves can you use to explain your result?*



Bézier curve of degree n defined by n + 1 control points $P_0, P_1, P_2, ..., P_n$
→ Minimum degree = 3, since we have 4 control points
→ The variation diminishing property of Bézier curves is that they are smoother than the polygon formed by their control points
(2 points) *From the graph above, can you determine the maximum degree of b(t)? Explain your answer.*

No, there could be infinitely many points along the line above and line would not change its shape if more points are added that lie exactly on the line. Therefore the maximum degree can be infinite.

(3 points) *List three more properties of Bézier curves and provide a short explananation Bezier and Bernstein polynomials tion of each one.*

1. Affine invariance: it is similar to compute an affine transformation of all the points of the curve or only the control points.
2. Invariance under affine transformation: for affine transformations such as scaling or rotation, Bezier curves keep their properties
3. Convex hulls
4. Endpoint interpolation: the first and the last points of a Bezier curve defined by points P0,P1,... PN are P0 and PN
5. Symmetry: The same Bezier curve shape is obtained if the control points are specified in the opposite order.
6. Linear precisions: if the control points are uniformly distributed on a straight line joining two points p and q, the Bézier Curve is a straight line linearly parametrized.

- https://people.eecs.ku.edu/~miller/Courses/IntroToCurvesAndSurfaces/BezierCurveProperties.html

1.3 Bezier and Bernstein polynomials

Every polynomial curve $c(t) \in \mathbb{R}^2$ of degree n with parameter $t \in [0,1]$ can be expressed by a Bézier curve $b(t) = \sum_{i=0}^{n} B_i^n(t) b_i$ with corresponding Bézier control points $b_0, ..., b_n$ and Bernstein polynomials

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}, \binom{n}{i} = \frac{n!}{i!(n-i!)}$$

(1 point) *Write down the formula of a cubic Bézier curve with control points (b_0, b_1, b_2, b_3) using Bernstein polynomials.*

$$
\begin{aligned}
B_{0,3} &= \frac{3!}{0!(3-0)!}t^0(1-t)^3 = (1-t)^3 \\
B_{1,3} &= \frac{3!}{1!(3-1)!}t^1(1-t)^2 = 3t(1-t)^2 \\
B_{2,3} &= \frac{3!}{2!(3-2)!}t^2(1-t)^1 = 3t^2(1-t) \\
B_{3,3} &= \frac{3!}{3!(3-3)!}t^3(1-t)^0 = t^3 \\
\Rightarrow b(t) &= (1-t)^3 b_0 + 3t(1-t)^2 b_1 + 3t^2(1-t)b_2 + t^3 b_3
\end{aligned}
$$

(5 points) *Prove that the derivative of a cubic Bézier curve with control points (b_0, b_1, b_2, b_3) is also a Bézier curve. What is the polynomial order of the derivative?*

$$
\begin{aligned}
\frac{d}{dt}b^n(t) &= \sum_{i=0}^{n} b_i \frac{d}{dt}B_i^n(t) \\
\frac{d}{dt}B_i^n(t) &= n(B_{i-1}^{n-1}(t) - B_i^{n-1}(t)) \\
\frac{d}{dt}b^n(t) &= n\sum_{i=0}^{n-1}(b_{i+1}-b_i)B_i^{n-1}(t) \\
\frac{d}{dt}b^n(3) &= 3\sum_{i=0}^{2}(b_{i+1}-b_i)B_i^2(t) \\
&= 3(\Delta b_0(1-t)^2 + 2\Delta b_1 t(1-t)) + \Delta b_2 t^2) \\
&\Rightarrow \text{second order}
\end{aligned}
$$

(3 points) *Prove the partition of unit property $\sum_{i=0}^{n} B_i^n(t) = 1$ for $n = 2$.*

$$
\begin{aligned}
\sum_{i=0}^{2} B_i^2(t) &= t^2 + 2t(1-t) + (1-t)^2 \\
&= t^2 + 2t - 2t^2 + 1 - 2t + t^2 \\
&= 1
\end{aligned}
$$

2 Projections and Transformations

2.1 Rotation matrices

(2 points) *2D rotation matrices are defined as $R = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}$. Prove that any 2D rotation matrix is orthonormal.*

Definition of orthonormal:

$$
\begin{aligned}
R^T = R^{-1} &\Rightarrow \det(R) = 1 \\
\det(R) &= \underbrace{\cos(\alpha)^2 + \sin(\alpha)^2}_{=1}
\end{aligned}
$$

(1 points) *Are there other orthonormal matrices that are not rotations of the form above? If yes, give an example.*

Reflection matrix

$$
R = \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ \sin(\alpha) & -\cos(\alpha) \end{bmatrix}
$$

(2 points) *Prove that the inverse of a rotation matrix is its transpose.*

$$
\begin{aligned}
\because \det(R) &= 1 \\
R^{-1} &= \frac{1}{1}\begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{bmatrix} \\
&= R^T
\end{aligned}
$$

(2 points) *Two 2D vectors v and n are orthogonal. Show that their orthogonality is preserved under a rotation R. (Hint: express their relationship in terms of a dot product).*

$$
\begin{aligned}
A &= \begin{bmatrix} A_y \\ A_z \end{bmatrix}, B = \begin{bmatrix} B_y \\ B_z \end{bmatrix} \\
\overline{A} &= \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} A_y \\ A_z \end{bmatrix} = \begin{bmatrix} \cos(\alpha)A_y - \sin(\alpha)A_z \\ \sin(\alpha)A_y + \cos(\alpha)A_z \end{bmatrix}, \\
\overline{B} &= \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} B_y \\ B_z \end{bmatrix} = \begin{bmatrix} \cos(\alpha)B_y - \sin(\alpha)B_z \\ \sin(\alpha)B_y + \cos(\alpha)B_z \end{bmatrix}, \\
\overline{A} \cdot \overline{B} &= A_y B_y \cos(\alpha)^2 - \cancel{A_y B_z(\cos(\alpha)\sin(\alpha))} - \cancel{A_z B_y(\sin(\alpha)\cos(\alpha))} + A_z B_z(\sin(\alpha)^2)) \\
&+ A_y B_y \sin(\alpha)^2 + \cancel{A_y B_z(\cos(\alpha)\sin(\alpha))} + \cancel{A_z B_y(\sin(\alpha)\cos(\alpha))} + A_z B_z(\sin(\alpha)^2)) \\
&= A_y B_y(\underbrace{\cos(\alpha)^2 + sin(\alpha)^2}_{=1}) + A_z B_z(\underbrace{\cos(\alpha)^2 + sin(\alpha)^2}_{=1}) \\
&= A_y B_y + A_z B_z
\end{aligned}
$$

Or, a bit shorter:

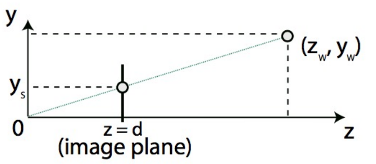
$$n^T v = 0 \Rightarrow (Rn)^T(Rv) = n^T R^T R v = n^T v = 0$$

(2 points) *An object is rendered on the screen by multiplying its vertices by the model, view, and projection matrices: $\tilde{v} = PVMv$. If you are given $\tilde{v}$, how do you compute its world coordinates?*

$$v_{world} = Mv = (PV)^{-1}\tilde{v}$$

2.2 Perspective Projection

(2 points) *You are given a 2D set up as shown below. The camera is at the origin looking down z. The image plane is at $z = d$. Derive the screen space position y_s of the point (y_w, z_w).*



Because of the triangle similarity we have:

$$
\begin{aligned}
\frac{y_s}{d} &= \frac{y_w}{z_w} \\
\Rightarrow y_s &= d\frac{y_w}{z_w}
\end{aligned}
$$

(2 points) *Generalize the previous question to the 3D setup. That is, derive the expression of the image plane coordinates (x_s, y_s) for the world coordinate point (x_w, y_w, z_w).*

$$x_s = d\frac{x_w}{z_w}, y_s = d\frac{y_w}{z_w}$$

(2 points) *Is this transformation affine? (briefly justify your answer)*

No, you cannot perform a division between elements of the same vector (e.g. x_w/z_w) with an affine transformation.

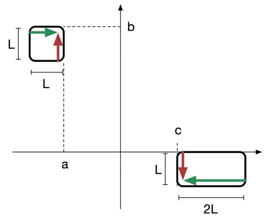
(2 points) *Explain how homogeneous coordinates can be used to represent the perspective transformation above in matrix form?*

$$\begin{bmatrix} d & & \\ & d & \\ & & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = \begin{bmatrix} dx_w \\ dy_w \\ z_w \end{bmatrix} \overset{hom.div.}{=} \begin{bmatrix} dx_w/z_w \\ dy_w/z_w \end{bmatrix}$$

2.3 Concatenating Affine Transformations

Derive the 3×3 homogeneous matrix that achieves the transformation in the figure.

a L Translate (a, b), scale (2, 1,1), then translate to (c,L). 241 0 c 35 24 2 0 035 241 0 a35 24t od o 1 0 a 35



$$R = \begin{bmatrix} cos(\pi) & -sin(\pi) & 0 \\ sin(\pi) & cos(\pi) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} S = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} T = \begin{bmatrix} 1 & 0 & (-a-L) \\ 0 & 1 & (b+c) \\ 0 & 0 & 1 \end{bmatrix}$$

$$M = RST = \begin{bmatrix} -1 & 0 & (-a+L) \\ 0 & -2 & -2(b+c) \\ 0 & 0 & 1 \end{bmatrix}$$

(2 points) *Consider the following affine transformation (note that the 3^{rd} dimension is not modified):* $M = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$*. If a triangle is transformed with this matrix, how will its area change?*

Use only the upper-left 2x2 part of M. The triangle area will not change. The area is preserved if the transformation matrix has a determinant of 1.

(1 points) *How can you in general check if an affine transformation matrix pre- serves the area?*

The area is preserved if the transformation matrix has a determinant of 1.

2.4 Rasterizing Planar Reflections

3 Textures, Rasterization and Visibility

3.1 Rasterization

In two sentences, quickly describe the painter's algorithm

Sort polygons from back to front, and paint in that order. that leads to overwriting some of the already filled parts.

Complete the pseudo code to implement the z-buffer algorithm.

```
// zbuffer[.,.]: R/Waccess
// framebuffer [ . , . ] : Read access
for (each polygon P) :
  for (each pixel (x,y,z) in P) :

  if (z < zbuffer(x,y)){
    framebuffer[x,y]=rgb; // write pixel's color to frame buffer
    zbuffer(x,y)=pixel(z);// update depth buffer
  }
```

04-Lighting slide 4

The z-buffer algorithm has two main advantages over the painter's algorithm? Describe and explain them.

1. It processes one object at a time, doesn't require sorting of objects and avoids nested splitting
2. It is very easy to implement and can be put in the hardware

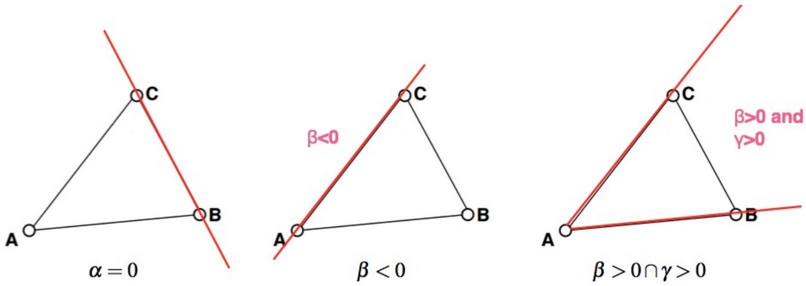
Sorting from back to front is not always possible without splitting polygons, and sorting is $O(n \log n)$. Z-buffer is constant time and never ambiguous.

http://www.cs.cmu.edu/afs/cs/project/anim/ph/463.96/pub/www/notes/zbuf.2.pdf

(2 points) *Barycentric coordinates can be used for simple 2D rasterization of a tri- angle. Complete the following pseudo code for rasterizing one triangle.*

```
// You can use
// set pixel(x,y) to fill a pixel
// (a,b,c)=bary_coords(x,y) to compute barycentric coordinates within the triangle
 for (all x):
  for (all y):
  (a,b,c)=bary_coords(x,y)
    if (a in [0,1] and b in [0,1] and c in [0,1]
      set_pixel (x,y)
```

(3 points) *We have a triangle with vertices A,B,C. A point on the triangle's plane can be represented with barycentric coordinates as $P = \alpha A + \beta B + \gamma C$. Sketch the three following regions:*



3.2 Texturing

(3 points) *Your program renders a quad whose texture coordinates "`in vec2 uv`" are $\in [0, 1]$. Write a fragment shader (GLSL code) that generates a 8×8 checkerboard texture in blue and red.*

(2 points) *What are light maps and what is the underlying assumption? When and why are they useful?*

Storing the expensive lighting calculation in textures in a preprocess. Lights are static and don't move. because texturing is fast and much cheaper and dynamic lighting. you can even precompute global illumination.

(1 points) *Which of the following three spheres was rendered with displacement mapping and which one with bump mapping?*

	bump	displacement

(1points) *Is the bump mapped or the displacement mapped sphere more expensive to render? Why?*

Displacement mapping is more expensive since it requires an adaptive tesselation of the surface to get enough micro-polygons to match the size of a screen pixel. https://en.wikipedia.org/wiki/Displacement_mapping

(1 points) *Using the following figure, show and explain that with perspective projection, barycentric interpolation in screen-space and worldspace differ.*

- In screen space: affine math: scaling, tearing and rotating but not perspective (has to be in object space)

(1 point) *What type of interpolation does OpenGL employ y default?*

(1 point) *Which of the two interpolation types is slower to compute for vertex attributes?*

(3 points) *Why is the correct answer to the previous question not true for interpolating the depth-related information used in the depth test?*

Because attributes of the form I/z can be linearly interpolated. the depth buffer stores stuff in form const + 1/z. the const doesn't matter.

3.3 Mip Mapping

(2 point) *Which problem does mip mapping address? Why does this problem occur?*

Aliasing due to insufficient sampling frequency of the texture. A more remote texture that has features below the scale of a pixel can show strange artifacts due to the fact that they are „sampled“ incorrectly (avaraged over a pixel).

They can also be used to improve the Level of Detail (LoD) of closer objects and speed up rendering by not rendering remote objects in full resolution

(2 points) *How do you compute the mip map texture T_{n+1} from the finer level T_n?*

$$T_{n+1}(x, y) = mean(T_n(x, y) + T_n(x + 1, y) + T_n(x, y + 1) + T_n(x + 1, y + 1))$$

(3 points) *Show that the increase of storage space required of all mip-map levels is at most one-third of the original texture.*

Every level uses a fourth of the pixels. E.g.,: $l(1) = 64^2, l(2) = 32^2 \ldots$, therefore the storage requirement of each level is a fourth of its preceding one storage requirement for mipmapping:

$$\sum_{i=1}^{k\to\infty} \frac{1}{4^i} = \frac{1}{4^1} + \frac{1}{4^2} + ... + \frac{1}{4^\infty}$$

$$\begin{aligned} s &= 1/4 + 1/16 + 1/64 \ldots \\ 4 * s &= 1 + 1/4 \ldots \\ 4s &= 1 + s \\ 3s &= 1 \\ s &= 1/3 \end{aligned}$$

Or simply from the geometric series: a=1/4, r=1/4, then

$$\begin{aligned} a &= 1/4, r = 1/4 \\ s &= \frac{a}{1 - r} \\ s &= \frac{1/4}{1 - 1/4} = \frac{1/4}{3/4} = \frac{1}{3} \end{aligned}$$

- https://en.wikipedia.org/wiki/Geometric_series
- https://www.reddit.com/r/math/comments/931jp/a_visualization_of_why_14_116_164_1256_13/

4 Coding, Lighting, Shading and Rendering Pipeline

4.1 Rendering Pipeline

4.2 OpenGL syntax

correct order: 7, 2, 8, 5, 6, 4, 1, 3
see `ex. 2, triangle.h`

4.3 Phong Lighting

4.4 Phong Shading

(2 points) *Explain the difference between Gouraud and Phong shading. Which one is computationally more expensive. Why?*

4.5 Phong Lighting Shader

(6 points) *Implement the fragment shader of a GLSL program that uses the Phong Lighting Model to render a mesh. You are not allowed to define any new uniform or in variables.* ***Hint****: Remember to consider back-facing geometry.*

```
in vec3 normal; // normal in camera space
in vec3 pos; // vertex position in camera space
out vec3 color; // write the final color to this output variable
uniform vec3 light_pos; // light position in eye coordinates
uniform vec3 Ia, Id, Is, ka, kd, ks;
uniform float n;
void main() {
  color = vec3(0.0) ;
```

4.6 Shadow Maps

(1 point) *What kind of projection do you use to generate the shadow map for a point light source?*

Perspective projection

(2 points) *How should you set the extents of the frustum when you render from the light's point of view to generate the shadow map to achieve best possible quality (least amount of jagged shadow outlines such as the ones shown in the image)?*

Tightly around the view frustum and so that points outside the view frustum that can cast shadows into the view frustum

(3 points) *For large outdoor scenes, explain what you can do to avoid shadow mapping artifacts similar to the ones in the image above. Assume that your GPU does not support textures larger than 2048 × 2048.*

Cascade shadow maps

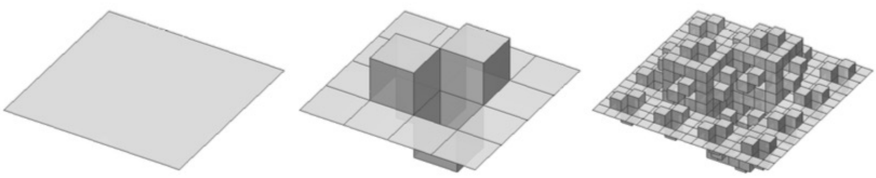
5 Procedural Modeling and Fractals

5.1 Measuring the dimensions of fractals

(2 points) *What is the Hausdorff dimension for fractals, how is it defined?*

$$
\begin{aligned}
N &= r^D \\
D &= \log(N)/\log(r) \\
D &= \text{fractal dimension} r \qquad = \text{fractal increment} \\
N &= \#\ \text{self-similar objects at smaller scale}
\end{aligned}
$$

(2 points) *What is the Hausdorff dimension of the 3D quadratic Koch surface (type 2)?*



$$r = 4, N = (12 + 10 * 2) = 32$$
$$\log(32)/\log(4) = 2.5$$

(1 point) *What is the Hausdorff dimension of the following fractal: take a unit cube, split it into 27 cubes, keep nine cubes, namely the bottom nine cubes.*

$$\log(9)/\log(3) = 2$$

(2 points) Give formulas for the volume and surface of the above fractal as a function of the recursion level k. What do they converge to for $\lim k \to \infty$

Volume:

$$v_1 = \left(\frac{1}{3}\right)^{1*3} * 9^1, v_2 = \left(\frac{1}{3}\right)^{2*3} * 9^2 ...$$
$$v_i = \left(\frac{1}{3}\right)^{i*3} * 9^i = \left(\frac{1}{3}\right)^{i}$$

Converges to 0. **Surface:**

$$s_1 = \left(\frac{1}{3}\right)^{1*2} * 6 * 9^1, s_2 = \left(\frac{1}{3}\right)^{2*2} * 6 * 9^2 ...$$
$$s_i = \left(\frac{1}{3}\right)^{i*2} * 6 * 9^i = \left(\frac{1}{3}\right)^{i} = 6$$

5.2 Noise Functions

(1 points) *When calculating Perlin Noise, the domain is divided into a grid. What are the noise values on the corners of the grid cells?*

0

(1 point) *What is the advantage of Perlin Noise over value noise?*

Interpolation over tangents and not points, in value noise there is a higher chance that several values in a row only differ a little.

https://computergraphics.stackexchange.com/questions/3608/benefit-of-perlin-noise-over-value-noise/3609

(2 point) *What is Fractal Brownian Motion? Give a short explanation.*

- Spectral synthesis of noise function
- Progressively smaller frequency
- Progressively smaller amplitude
- Each term in the summation is called an octave

(1 point) *Write down the pseudocode for fBM*

```
float fbm (in vec2 st) {
  // Initial values
  float value = 0.0;
  float amplitude = .5;
  float frequency = 0.;
  //
  // Loop of octaves
  for (int i = 0; i < OCTAVES; i++) {
    value += amplitude * noise(st);
    st *= 2.;
    amplitude *= .5;
  }
  return value;
}
```