Do not open the test until told to begin!

CMSC 435/634: Computer Graphics

Final Exam

Due 3:00 PM May 16, 2021

Instructions: You have 48 hours to complete this exam, including time to submit at the end. You can do it on paper to scan or photograph, complete it electronically directly in the pdf on a tablet, or as clearly numbered answers in a text, latex, or word document. Be sure to **add**, **commit**, and ****push**** your answers to your git repository by 3:00 PM on May 16th.

This exam is open book and notes: you may use any notes you have taken, the class texts, any of the class lecture slides, or any of the recorded lectures. You can use a calculator to assist you in coming up with numeric answers where one makes sense, and a computer for numeric computation or drawing. You cannot post new public messages on webex, discord, or other forum, do searches of the wider web, or consult with anyone besides the TA and instructor about the exam. If anything is not clear, post a private message to the instructor and/or TA on webex or by email. If any bugs come up in the exam, I will post corrections or clarifications on webex to the class as a whole.

Good luck

Name: Nick Conway (JJ02259 / conwan1@umbc.edu)

Do not open the test until told to begin!

1 Odds and ends **(20 points total)**

1.1 If assignments 3–5 were the first steps toward a future video game, what would it be called? **(5 points)**

1.2 What line(s) of this shader are likely to cause reduced SIMD utilization? **(5 points)**

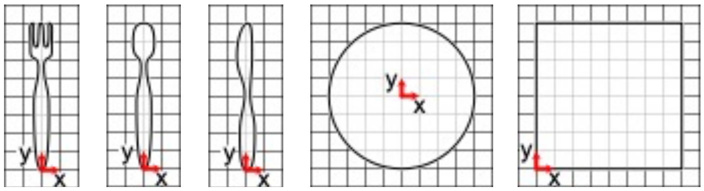
```
1 for(x = floor(uv.x - 1.5); x <= floor(uv.x + 2.5); ++x) {
2     for(y = floor(uv.y - 1.5); y <= floor(uv.y + 2.5); ++ y) {
3         vec2 pt = vec2(x,y) + hash(ivec3(x,y,0)).xy;
4         float dist = length(uv - pt);
5         if (dist < vnoise) vnoise = dist;
6     }
7 }
```

1.3 Name one graphics thing Ken Perlin is known for. **(5 points)**

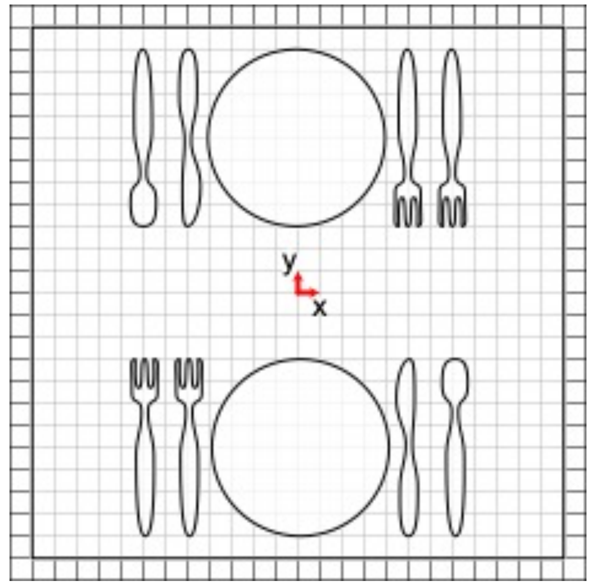
1.4 Write pseudocode for bilinear sampling at texture coordinates `(u,v)` where `0 ≤ u < 1` and `0 ≤ v < 1`, given a 1D linear interpolation function, `lerp(a, b, t)`. The texture is stored in a 2D array as `u8vec3 image[j][i]`, for integer `0 ≤ i < w` and `0 ≤ j < h`. **(5 points)**

2 Transformations **(20 points total)**

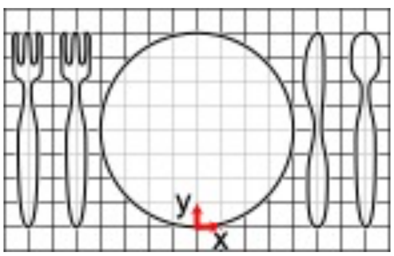
For this problem, you will be modeling a table with three equally spaced place settings around a round table. You have already written the drawing functions (a) `fork()`, (b) `spoon()`, (c) `knife()`, (d) `plate()`, and (e) `table()`. Each of these draws the respective object with the origin, orientation and scale show below.



(a) (b) (c) (d) (e)



(g)



(f)

You also have functions to operate on a transformation matrix stack: `PushTransform()` pushes a copy of the top matrix on the stack, `PopTransform()` removes the top matrix from the stack. `Scale(x,y)`, `Translate(x,y)`, and `Rotate(θ)` all multiply the top matrix on the stack by a scaling, translation, or rotation matrix. When you call one of the drawing functions, it uses the top matrix on the matrix stack as its `WorldFromObject` transformation.

2.1 Write the code for a `placeSetting()` function that produces a single place setting with origin, orientation and scale shown in (f). **(5 points)**

2.2 Now write code to lay out the entire table as shown in (g), using any of the functions mentioned above, as well as the new `placeSetting()` function **(5 points)**

2.3 Draw the final layout as a transformation tree, with fork, knife, spoon, plate and table labeled on the nodes where they would be drawn. You can draw by hand, with your favorite drawing program, or using the drawing tools in Word, PowerPoint, or Google Docs. **(10 points)**

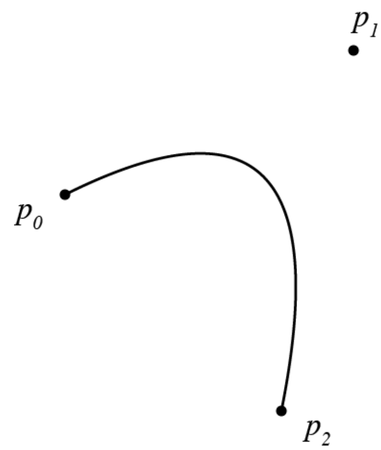
3 Curves **(40 points total)**

A quadratic spline has the equation for $t \in [0, 1]$

$$p(t) = a\ t^2 + b\ t + c$$

With velocity and acceleration

$$p'(t) = 2\ a\ t + b$$
$$p''(t) = 2\ a$$



A *quadratic Bézier* curve obeys the following constraints:

1. Passes through p_0 at $t=0$
2. Velocity $2(p_2 - p_1)$ at $t=1$
3. Passes through p_2 at $t=1$

3.1 Write the constraint equations for this curve. **(10 points)**

3.2 Write the equations in matrix form in terms of the vector of control points and vector of coefficients. You should have a number for every box in the matrices. **(10 points)**

$$\begin{bmatrix} \Box & \Box & \Box \\ \Box & \Box & \Box \\ \Box & \Box & \Box \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} \Box & \Box & \Box \\ \Box & \Box & \Box \\ \Box & \Box & \Box \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

3.3 Solve for the *basis functions* for this curve in matrix form. **(10 points)**

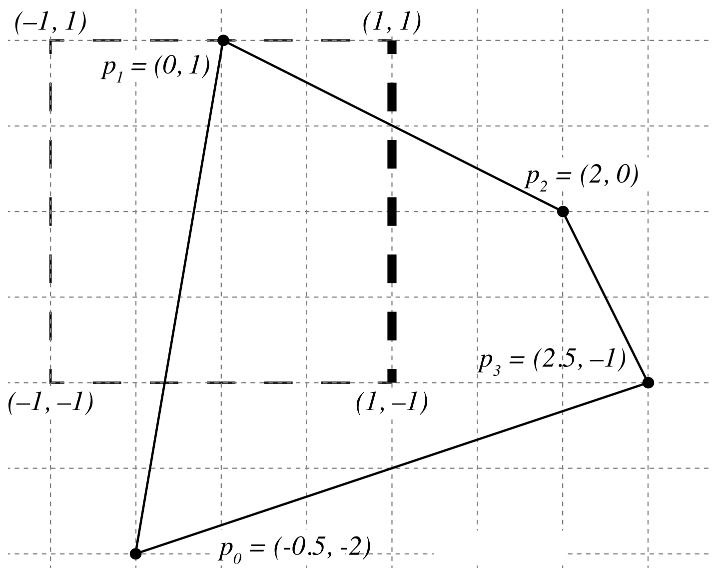
For this question, and the next, you can use Wolfram Alpha for matrix multiplication and inverse:

- https://www.wolframalpha.com/input/?i=matrix+multiplication
- https://www.wolframalpha.com/input/?i=matrix+inverse

3.4 I used *Adobe Illustrator* for the drawing Bézier curve drawing above. Illustrator has a cubic Bézier tool, but no *quadratic* Bézier tool. Any quadratic Bézier curve is also a cubic curve with a 0 for the t^3 coefficient. Consequently, you can solve the quadratic Bézier equations for the quadratic a, b, and c, then use the cubic Bézier equations to solve for the cubic control point positions that would create the algebraically identical curve (this is called *degree elevation*). Solve for the matrix equation for those cubic Bézier control point positions in terms of p_0, p_1, and p_2. **(10 points)**

4 Clipping **(20 points total)**

Consider clipping the polygon shown below, with vertices p_0, p_1, p_2, and p_3, against the right clipping plane at $x = 1$ (the dark dashed line)



4.1 Draw the result of clipping the polygon against the $x = 1$ clipping edge, labelling any new points. **(5 points)**

4.2 Solve for the location(s) of any new vertices. The clip intersections are intentionally placed at integer locations so you can easily tell if your results are correct. For this question, you **must** show your math, you will not get credit if you just say where they are by looking at the picture. **(10 points)**

4.3 What is the order of points output by the Sutherland-Hodgeman clipping algorithm when clipping polygon p_0, p_1, p_2, p_3, against the edge at $x = 1$? **(5 points)**