COS 426, Spring 2013

Midterm 1

Name:

NetID:

Honor Code pledge:

Signature:

The exam consists of 5 questions. Do all of your work on these pages (use the back for scratch space), giving the answer in the space provided. This is a closed-book exam, but you may use one page (both sides) of notes during the exam. **Put your NetID on every page (1 point), and write out and sign the Honor Code pledge before turning in the test:**

"I pledge my honor that I have not violated the Honor Code during this examination."

Question	Score
1	/ 20
2	/ 16
3	/ 10
4	/ 4
5	/ 9
NetID on every page	/ 1
Total	/ 60

NetID:

1. **Image processing** (20 points)

Describe the result of applying the following 3x3 filters to an image. Write your answer next to each filter.

a) (2 points)

0	-1	0
0	2	0
0	-1	0

detects horizontal edges (y gradient)

b) (2 points)

0	0	0
1	0	0
0	0	0

shifts image right one pixel

c) (2 points)

0	0	0
1	1	1
0	0	0

x 1/3 blurs in the x-direction

d) (2 points)

-2	-2	-2
-2	17	-2
-2	-2	-2

sharpens

e) (2 points)

1	1	1
1	1	1
1	1	1

x 1/9 blurs

f) (2 points)

1	1	1
1	1	1
1	1	1

x 1/12 blurs and darkens

NetID:

g) (2 points) Describe the result of applying the filter in (b) n times.
shifts image right n pixels

h) (2 points) Describe the image obtained in the limit as the filter in (f) is applied repeatedly.
black image

i) (4 points) Given the small 8-bit grayscale image shown below:

3	141	59	26
53	58	97	93
238	46	26	43
38	32	79	50

Do ordered dithering to convert the image to a 1-bit image. Start from the top left. The ordered dithering matrix is:

$$\begin{pmatrix} 3 & 1 \\ 0 & 2 \end{pmatrix}$$

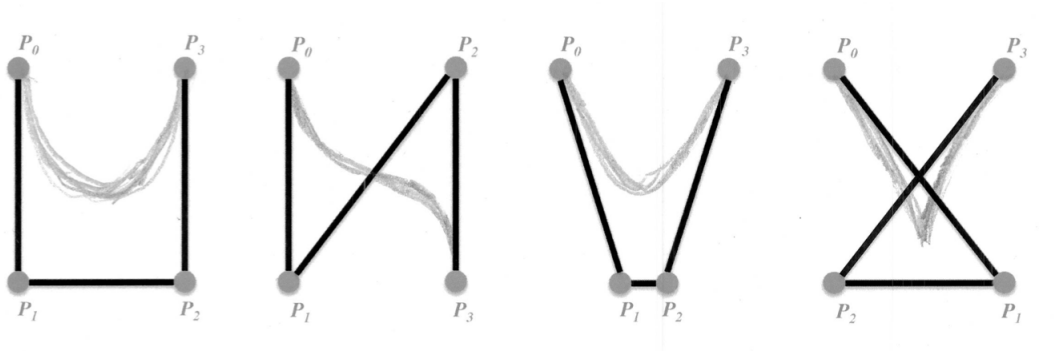
Fill in the output using 0s and 1s:
multiple answers, depending on algorithm details; one possibility:

0	1	0	0
1	0	1	0
1	0	0	0
0	0	1	0

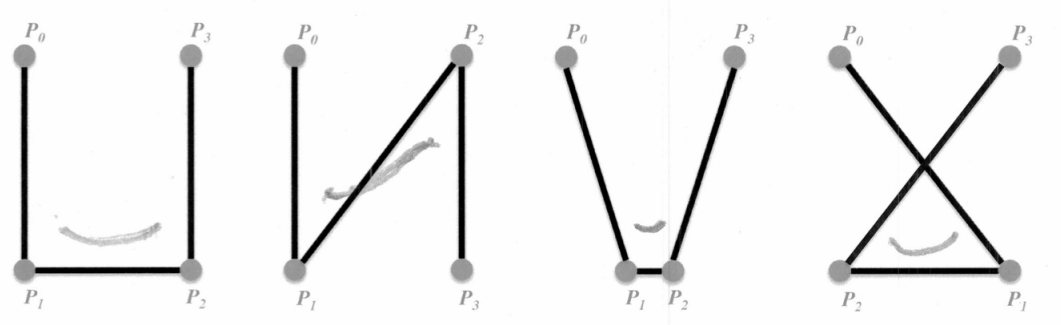
NetID:

2. **Polynomial curves** (16 points)

a) (4 points) Sketch the cubic Bezier curve for each of the four control polygons below with point P0 through P3. Your sketches can be approximate of course, but be sure to interpolate or approximate the control points as appropriate, and get the tangents approximately right.

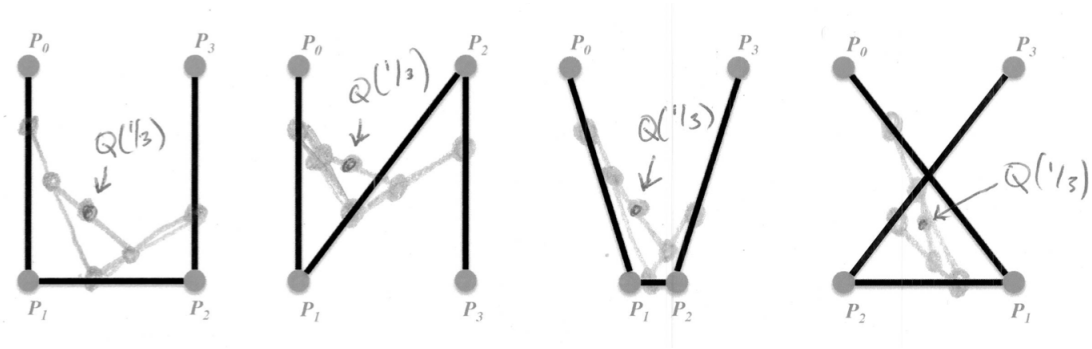


b) (2 points) Suppose P0 through P3 shown below are in the middle of a longer sequence of control points for a cubic B-spline. Sketch just the portion of the spline corresponding to these four control points (the part that is not affected by any control points other than P0 through P3). For this part just try to get the rough shape and location; it is hard to know exactly where the curve goes without some extra machinery.



NetID:

c) (2 points) Suppose P0 through P3 are the control points for a cubic curve generated by De Casteljau's algorithm. Moreover, they are ordered so that the curve Q(t) starts with Q(0) at the P0 end of the polygon, and ends with Q(1) at the other end (P3). Draw De Casteljau's construction for finding Q(1/3).



d) (2 points) Briefly, what is the difference between parametric and geometric continuity (one or two sentences)?
parametric - continuity of underlying parametrization
geometric - visual/intrinsic continuity

e) (2 points) What kind of continuity is there in the middle of curves like those in part (b)? Explain briefly and be as precise as possible.

C^∞ - polynomial is infinitely differentiable

f) (2 points) What kind of continuity is there for curves like those in part (b) at the ends of the portion you drew ("joints" in the spline)? Explain briefly and be as precise as possible.

B-splines ensure C^2 continuity at joints

g) (2 points) What kind of continuity is there for curves like those in part (c) at the location Q(1/3)? Explain briefly and be as precise as possible.

C^∞ - same answer as (e) because it's the same curve

NetID:

3. **Shape representation** (10 points)

Suppose you have a class `Shape` that represents a watertight, orientable 3D shape. `Shape` has the following method:

```
// returns true if point is inside the shape, false otherwise
bool IsInside(R3Point const & point)
```

a) (3 points) Suppose you also have classes `Voxel` and `VoxelGrid`, representing a voxel and voxel grid, respectively. `Voxel` has the following methods:

```
// returns the point at the center of the voxel
R3Point Center()
```

`VoxelGrid` has the following methods:

```
// returns the number of voxels in the grid
unsigned int NVoxels()

// returns the kth voxel in the grid
Voxel * Voxel(unsigned int k)

// set occupancy for the kth voxel
// true for occupied, false for unoccupied
void SetOccupancy(unsigned int k, bool isOccupied)
```

Fill in the following method of the `Shape` class:

```
// rasterize the shape to the provided voxel grid
void Rasterize(VoxelGrid * grid)
{
   // fill in your answer here
   for (unsigned int k = 0; k < grid->NVoxels(); k++)
   {
      grid->SetOccupancy(k, IsInside(grid->Voxel(k)->Center()));
   }
}
```

NetID:

b) (2 points) For this question, suppose the `Shape` class represents the shape using an implicit surface and has a method `double f(R3Point const & point)` that returns the value of the function defining the implicit surface at `point`. Write `IsInside` for this version of the class:

```
bool IsInside(R3Point const & point)
{
   // fill in your answer here
   return (f(point) < 0.0);
}
```

For parts (c) and (d), suppose the `Shape` class represents the shape using a polygonal mesh, as in the starter code for assignment 2. In particular, `Shape` has methods `unsigned int NFaces()` and `R3MeshFace * Face(unsigned int k)` that return the number of faces and the kth face, respectively.

Suppose `ConvexShape` is a sublass of `Shape` that is restricted to representing convex shapes. Furthermore, suppose `R3MeshFace` is augmented with the following method:

```
// returns the vector, from the plane containing the face
// (i.e.  face->plane) to the specified point, which is
// perpendicular to said plane
R3Vector fromPlane(R3Point const & point)
```

c) (3 points) Using sentences or pseudocode, describe an implementation of `IsInside` for `ConvexShape`.

check if the point is on the inner side of each face's plane

NetID:

d) (2 points) Convert your answer to part (c) into code. Recall that the `R3Plane` class has a method `R3Vector Normal()` that returns the normal to the plane.

```
bool IsInside(R3Point const & point)
{
   // fill in your answer here
   for (unsigned int k = 0; k < NFaces(); k++)
   {
      R3MeshFace * f = Face(k);
      if (f->fromPlane(point).Dot(f->plane.Normal()) > 0.0)
      {
         return false;
      }
   }
   return true;
}
```

NetID:

4. **Shape representation continued** (4 points)

Suppose you have a set of n 3D shapes represented as implicit surfaces by the functions $f_1(x, y, z), \ldots, f_n(x, y, z)$.

a) Write down the function that defines the shape that is the union of these n shapes.

$\min_i f_i$

b) Write down the function that defines the shape that is the XOR of the shapes defined by f_1 and f_2.

$f_1 f_2$

NetID:

5. **Transformations** (9 points)

a) (2 points) Describe in a sentence what the following 2D transformation matrix does. Your answer should demonstrate understanding; do not give an answer like "maps (x, y) to $(f(x, y), g(x, y))$."

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$ reflects across y = x

b) (3 points) Express the transformation in part (a) as a product of translation, scale, rotation, and/or shear matrices.

multiple answers; one possibility (ccw rotation by $\pi/2$ followed by scale of x by -1 (reflection across y-axis)):

$$\begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

c) (4 points) Consider the permutations of of $T_1T_2T_3T_4$. There are 4! = 24 such permutations. Write down all the unique points to which these permutations map the point (1, 0).

$$T_1 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} T_2 = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} T_3 = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} T_4 = \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(1,0), (0,1), (0,-1), (2,1), (2,-1)