CS 3451 Project 4: XOR of Triangles
Teams of 1 or 2 students. Due Thursday Sept 18.
A perfect implementation, along with a convincing video and an impressive paper will not only give you 10 points out of 10, but will also allow you to replace your lowest project grade by a 10.

Required functionality of the code:

1) Start with code from previous projects and provide support that lets the user click& drag 6 points: A, B, C, D, E, F. Draw the edges of ABC in green and the edges of DEF in blue. Write the letters next to or on the vertices. When a triangle is counter-clockwise, fill it in light gray.
2) At each frame, compute all points of intersections between an edge of ABC and an edge of DEF, store their location in an array of points (or in two parallel arrays x[] and y[]), show them as black dots. Write the number of components on the canvas. These point and writing should be updated in realtime as the used edits the vertices.
3) At each frame, compute a GEL representation (discussed in class) of the polyloops that form the boundary of the connected components of the interior of the XOR (symmetric difference), [ABC] ⊗[DEF] of the two triangles. Make sure that each loop is oriented so that the interior of the region it bounds is on its right.
4) At each frame, compute the area of each connected component and fill the component with a matching color (using a warm-to-cold color ramp). Make sure that the color ramp is normalized so that it is easy to use color to identify the components with largest and smallest area when these are sufficiently different.

Deliverables (one version for each team) :
A zip file on T-square with the following content

A) Short video showing the 4 functionalities in realtime as the user manipulates the vertices over a wide range of configurations (including cases where one triangle is inside the other and vice versa, and cases where the triangles intersect but no vertex of one lies inside the other, and including cases for each possible number of connected components. The screen should show the photos of the team members and their names.
B) The source code of the complete outside folder of your sketch (not just the files in it), but not containing any pictures that you used to make the videos.
C) A report (PDF file) of about 2 pages with:
    a. Header specifying “CS34515 Fall 2014: Project 4”, First and LAST names of the team members written below their photos, descriptive title of the project.
    b. Clear problem statement presented twice:
        i. Using a complete and unambiguous math formulation. For example: “Given two triangles, identify the components of their … and fill each with a color…”
        ii. Using easy to understand plain English explaining what these terms mean.
    c. Motivation: Explain why being able to compute the XOR between shape is important, why it is important to be able to separate a shape into its

components, and why one may want to measure the area of each component. Also explain briefly what concepts and techniques this project has helped you understand and learn.

d. Overview: Explain in plain English, at a high level, the structure of your approach. For example, you may say: "Our solution involves x steps: (1) Compute … and store them in a …, (2) Construct … by …, (3) For each … measure…
e. Technical details: Provide succinct math formulations (using the notation defined in class) or algorithms (using pseudocode) for all non-trivial sub-problems identified in the Overview. For example
    i. To compute the intersection points, for each edge [PQ] of ABC and each edge [RS] of DEF, we test whether they intersect using … and if so, compute their intersection point using … These intersection points as well as the 6 original vertices are stored in an array G
    ii. To identify and orient the edges of the result, we…
    iii. To build the V and N tables of the GEL representation, and use the following algorithm: …
    iv. To identify the connected components of the XOR and the bounding polyloops associated with each component, we…
    v. To measure the area of each component, we …
    vi. To assign a color to each component, we use the following color ramp … which we normalize by identifying the smallest s and largest m areas.
f. Discussion of limitations:
    i. Say that you assume *General Position* and explain what it means.
    ii. Discuss any other problems or limitations of your implementation
g. Justification: Argue why the approach you have chosen should always work (assuming General Position). For example, discuss properties of the results (where its edges and vertices) come from and argue that your approach will find all of these.

GEL representation:

1) G: represents the vertices as an array G[] of points or as coordinate arrays X[], Y[]. Given index v, the geometric location (point) of that vertex will be denoted v.g
2) E: represents the oriented edges as two arrays V[] and N[]. V[e] is denoted e.v and represents the integer index to the starting vertex of edge e. N[e], denoted e.n represents the next edge along the polyloop.
3) L: represents the loops for each component by storing, for each component: the number of loops and, for each loop: the index of one of its edges.

For example, the points where the ends of edge with index e are located can be referenced using our notation as e.v.g and e.n.v.g.

Primitive tools:

Your implementation may need to use the following primitive tools:

- Test whether two edges intersect
- Compute their intersection point
- Test whether a point lies inside a triangle

- Test whether a triangle is oriented clockwise
- Compute the signed area of a triangle

Make sure that you understand what these tools take as input, what they compute, and what they produce as output. Make sure that you know the mathematical formulations of these results and include your own presentation of these formulations in your write up. Doing so will help you memorize it. Explaining it in writing will help you understand it. Implement them yourself or, if you can't and find existing code that is publicly available or posted on the class Dropbox, use that code, but clearly acknowledge where it comes from and provide a link or reference (both in the comments of your code and in your write up). DO NOT FORGET TO DO THIS!

There are some theoretical aspects that you need to consider for this project:

- What is the XOR of two sets
- What are the interior, boundary, exterior, and closure of two sets
- What is the relation between the boundary of the XOR and the boundaries of the two triangles
- What are the connected components of a set
- Why do we identify the connected components of the interior of the XOR
- How to compute the proper orientation of the edges of the boundary of a polygonal region
- How to define the bounding loops of a connected component
- What are the non-manifold vertices of a loop
- How is e.n defined when the edge arrives at a non-manifold vertex
- Why will repeating e=e.n always trace a loop
- What are the maximum numbers of components, of loops, of loops in a component
- How to test whether two loops are bounding the same components

Make sure that you know and understand the answers to these questions.

There are some interesting algorithmic challenges:

- What is the simplest (most elegant, easiest to code) approach for computing the loops
- What information will you collect during the phase that computes edge-edge intersections to help you implement that approach
- How will you test whether you have components with more than one loop

Think a lot about them and brainstorm with your partner before you start implementing anything. This part of the assignment is where you can show your abstract thinking, creativity, and problem solving skills. Enjoy!