CSC 240 Computer Graphics
Day 15: Hierarchical Models

Nick Howe
Smith College
Midterm Exam

*If you score better on the final exam than the midterm, the final grade will replace the midterm grade.*
Problem 2:

• Algorithm A is an 8-connected recursive fill with a variable call order. It is otherwise very similar to the 8-connected fill we studied.

• Algorithm B will not fill shapes with variable side boundaries. It makes fewer recursive calls than our standard sweep algorithm.
Midterm Exam

Problem 3:

- Check that your matrix transforms the original corners to the desired new corners. You can compute it by hand or by coding it.

- Example (part d):

\[
\begin{bmatrix}
-1 & 0 & 4 \\
1 & 1 & -4 \\
0 & 0 & 1 \\
\end{bmatrix}
\begin{bmatrix}
0 & 4 & 4 & 0 \\
0 & 0 & 4 & 4 \\
1 & 1 & 1 & 1 \\
\end{bmatrix}
=\begin{bmatrix}
4 & 0 & 0 & 4 \\
-4 & 0 & 4 & 0 \\
1 & 1 & 1 & 1 \\
\end{bmatrix}
\]
Problem 4:
- *Trickiest part is order of transformations (translate first, then rotate)*
- See [demo](#).

Problem 5:
- *Clip all lines on a side, so you keep a polygon after each step*
Problem 6:

- Use the direct formula for a cubic spline:

\[ \mathbf{p} = (1 - t)^3 \mathbf{p}_0 + 3t(1 - t)^2 \mathbf{p}_1 + 3t^2(1 - t)\mathbf{p}_2 + t^3 \mathbf{p}_3 \]

(works for any t)
Your Questions

Q. I didn't quite understand the relatedness principle, can you explain it further?

A. It’s not so much a principle as a guideline. If you see a set of things in your scene as related in some way, it can make sense to group them together for that reason alone. On the other hand, the practical consideration of what moves together may be a more useful motivator.
Q. Why is B not neither?

A. An object can only attach to one parent. When you add something twice, the second call supersedes the first.

Q. Why can't we add the same object to two different objects? What if the object's motion is dependent on two different objects?

A. Each object may have only one parent. The reason is that the object’s final transformation matrix includes the composition of all its ancestors in the tree. If it belonged to multiple parents, the correct transformation would not be well defined.
Your Questions

Q. When we "R.add(Q); S.add(Q)", how does the computer know that we remove Q at R and add it to S?

A. Most likely, there is a variable pointing to the parent of an object, and assigning a new one overwrites the previous value.

Q. So for matching B with picture Y, is it true that whenever we have multiple lines adding the same child to several different parents, only the last parent specified will ultimately contain the child?

A. Yes.
Q. Is the object hierarchy like a tree?
A. Yes! It is a tree.

Q. What if we apply T_left first and then R_left? Do we still need a Lhip?
A. The application order of the transformations is fixed by THREE.js, so we don’t have the option of switching them.
Q. I don't quite understand the animate part in the demo when you used the body.rotation variable within the body.position variable? Also how do the 3D objects know where to go? I didn't really see coordinates for them so how is it that the rotation point changes?

Q. We didn't set the coordinates for lhip and rhip, yet their center is the bottom of the big rectangle, why?

Q. Can you explain more on lhip and rhip? I don't get the reason why the rotation center change. Thank you!

Q. In the demo code, how does the lhip and rhip make the rotation center change?

Q. Could you explain more about why adding the left or right hip would change the points that the legs are rotating about?

A. The position of left and right is interpreted in the frame of the parent. If we rotate lhip and rhip, then the leg position moves accordingly.
Q. Conceptually, are we rendering the scene as a whole or are we rendering each object separately? (I got a little confused with the vanes of the windmill in the reading. Are we rotating one vane and then adding the "rotating vane" as a child to the higher hierarchy level, or are we building our hierarchy structure first and then rendering everything all together?)

A. I would say that we build the hierarchy structure, then we render each object separately using its position in the object tree to figure out the correct transformation.
Q. Can you explain why the body will walk in a circle with the code you gave in demo? No sure what's the logic and math behind the code. How to decide when to use sin and when to use cos?

A. Cosine and sine give the coordinates of points on a circle. Swapping them flips the $x$ and $y$ coordinates (reflection around $y = x$ line). This makes the point travel in the opposite direction around the circle as $\theta$ increases. It also changes the orientation of the body relative to the circle.
Your Questions
Lab 8: Robot Arm