CS 410, Fall 2017, First Third of Semester Review
September 21, 2017

The following is not an exhaustive list of what has been covered, and to be very explicit, the First Midterm may cover material discussed in class not been mentioned below. That said, hopefully the following will be a helpful as you review the broad topics and techniques we have covered so far in CS410.

1. The mathematics of 3-D modeling, both of objects and cameras, rests squarely upon a solid working understanding of the following 3 terms: scalar, vector and point. Your understanding of computer graphics at this point means you're able to succinctly define each of these 3 mathematical concepts.

2. The concept of a vector space is straightforward, for the sake of this class you may define it simply as the space of all vectors which may be defined using a fixed number of scalars, e.g. 2 scalars for 2D and 3 scalars for 3D. The concept of an affine space is somewhat more difficult to hang onto, but you now know the one thing that affine spaces have that the vector spaces do not, and you can use it to remind yourself about the difference between these 2 concepts.

3. You now appreciate that the dot product has significance for geometric modeling at many levels. At the most pragmatic level, you certainly know how to compute the dot product between 2 vectors. At a much deeper level, the dot product represents the transition from affine spaces to Euclidean spaces. You're now in a position to explain why.

4. You have now seen how basis vectors play a truly fundamental role in how we express the geometry of object models and cameras. Here's a thought question you should be ready to tackle. In order to span a space, is it absolutely essential that basis vectors be orthogonal? Another way to think about this question is as follows, if it is not a requirement that basis vectors be orthogonal, why does it appear that we place such a high premium on selecting orthogonal basis vectors in practice.

5. You are now aware that there are 2 rather distinct ways of introducing the concept of rotation in the 2-D plane. Most references (e.g. Wikipedia) simply state the rotation matrix in terms of the cosine and sine of an angle theta. There is nothing incorrect about this approach, but it does not really help one understand what is actually taking place when a 2-D point is rotated. Fortunately, you can now take your knowledge of the dot product and derive a 2-D rotation matrix without ever having to touch a cosine or a sine.

6. There are many ways to motivate homogeneous coordinates, fortunately you are now in a position to quickly provide a pragmatic motivation in terms of chaining together, or perhaps one should say composing, sequences of transformations. Further, if making this argument to another it is best to illustrate using concrete examples of commonly named types of 2-D transformations.

7. It goes without saying that at this point in the course you can write down examples of 2-D rotation, translation, scaling, non-uniform scaling, and sheer matrices - all in homogeneous coordinates. Just as important, if you are presented with a set of 2-D points drawn before and after the application of one or several of these transformations, you can now work out by looking at the relative positions of the points pre- and post-transformation which operation or combinations of operations were carried out.

8. Perhaps one the odder aspect of learning computer graphics is the need first to learn about Euler angles and subsequently to learn why in practice they are virtually never the best way of capturing the rotation of an object or a camera. Fortunately, you can both explain what Euler angles are and why they are not heavily used.

9. If you are asked over lunch in a job interview to compute the cross product between the vectors (1, 2) and (2, 3), be certain that you would know why your reaction should be one of puzzled amusement, aware of the fact that you are being baited into possibly revealing a basic misunderstanding of geometry.

10. There are 2 ways you should have the cross product committed to memory. The 1st is to be able to explain in English the geometric interpretation of the resultant vector. The 2nd is the algebraic approach in which you can actually compute the cross product of 2 vectors.

11. If one were to say there's something special about the cross product between 2 unit-length and perpendicular vectors, you fortunately could quickly explain what is special about this case.

12. In 3 dimensions as opposed to 2 dimensions, it is a bit harder to make up matrices that represent valid 3-D rotations. That said, it is not all that hard, and you now know how. You should be able to explain by example the rules governing the construction of a valid rotation matrix.

13. Just as computer graphics folks don't typically think much of Euler angles, many of us have a fondness for the axis angle approach to specifying 3-D rotations. Fortunately, you can now illustrate this by showing how to rotate 45° about the vector (2, 2, 1).

14. Our lectures on modeling cameras have used phrases such as “bear in the box” and “pinhole room”. What underlying critical concept rests beneath the illustrations that have in turn given rise to these phrases?

15. The 2 term “frustum" is important to camera modeling and you are able to explain this geometric construct in your own words and with your own simple drawings. Doing so includes the critical concepts of near and far clipping planes.

16. Intersecting two parametric line segments is a very useful first example of how parametric forms of objects makes computation of intersections ‘easy’. You now understand in the large the approach as well as how to mechanically carry out the procedure. Indeed, in some cases where the solution of 2 equations in 2 unknowns might simplify, you could work an example by hand.

17. In ray tracing the first major formulation shoots one ray per pixel. As introduced in CS410, this means it is essential to be able to enumerate in 3D world coordinates the 3D position of individual pixels. Given a camera specification, you can derive the necessary formulas and associated code to accomplish this task.

18. Also using a camera specification, you understand and can explain the process of generating rays, one per pixel, given a camera specification.

19. There is a kind of word problem you can now easily solve: a space ship leaves earth in a direction U, how far has it traveled when it passes closest to a planet at position P. Assume the earth is the origin.

20. There is a closed form efficient way to compute the intersection between a ray and a triangle in 3D. You can explain all aspects of this clever algorithm.

21. There is a clever way of determining the intersection of a sphere and a ray. Since you have already implemented this as part of your programming assignments, you're aware that solving a general problem without the aid of a calculator would be annoying. That said, the construction of the algorithm is elegant and important, and you certainly could work out a concrete numerical example if the constituent components of the example were well enough chosen to make the ensuing arithmetic straightforward.