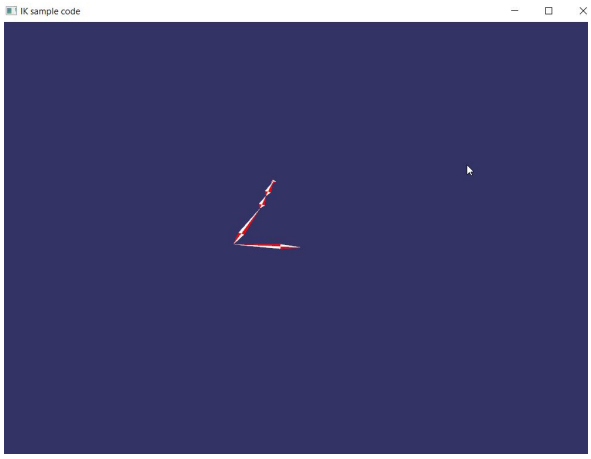

2D Inverse Kinematics

Contents of Folder:

- Shaders
- Exe file
- Main cpp file

User Interface Controls:

There are no special user interface controls for this project. One simply has to open "inverseKinematics.exe" then click on the desired placement of the target. It should be noted that there is no guarantee that the program will find a way to reach the target.

Success of the IK Implementation:

The success of the IK implementation was mixed to say the least. Sometimes the program would effortless go for the target immediately, sometimes the program would take a long time before finally being able to hit the target, a lot of the the time the arm system would never reach the target at all.

There were several interesting patterns that the arms tended to make. One of the patterns is what I call the "orbit" pattern, the end effector would practically orbit around the target, never being able to actually hit the target (the gif named "hitwithstruggle" is also a variant of this pattern). Another interesting pattern was the "outofrange" pattern, which is where you click outside of the arm's range. In this case, the arm flails in random directions (although the direction of the flailing generally tended towards the target). Of course as mentioned before, sometimes the arm just froze, or seemingly flailed randomly.

The large changes vs small changes question has no clear answer. Sometimes small changes don't matter in the speed of the arm being able to hit its target. However, there was a scenario when the arm is really quick at being able to hit its target. If the arm has ALREADY been able to hit its target, additional small modifications are easily reached by the arm. This phenomena is shown in the "snappy" gif. There were also instances when I put the target

extremeley close to the end effector (in a state where a prior target has NOT been reached) and it would have an extremeley hard time trying to reach that target.

Applications in Animation:

Ultimately, I have a hard time seeing this being used in professional animation. The end poses of the arms when they have already hit the target are somewhat similar to how humans grab objects, but the actual animation of the arm moving around is very unrealisitic.

Additional Comments:

Because the problem is using the inverse function on matrices, it has a very big problem with handling 0's. This mean that there had to be exceptions put into the program to make sure it ran smoothly. In my implementation, I made it so that the IK algorithm wouldn't run for the first few loops (which is where the 0's occur while the program updated the position of the arm's joints, the position of the arm's started at $x = 0$, $y = 0$).

It should be noted that this problem didn't happen when I had just implemented the Jacobian Transpose, but the Jacobian Pseudoinverse that is implemented in the program required this workaround.