A picture containing text

Description automatically generated

All function in Euler Angle Splines implemented, both linear and cubic interpolation for the euler angles.

Text

Description automatically generated

All function in orientation representation implemented. Note for the quat::FromRotation, the algorithm is used as the one from Orientation Representation -Van Verth and Bishop.

Graphical user interface, text, application

Description automatically generated

All function implemented or quaternion Splines. Note some cases such as from start point angle (0, 0, 0) to next key anlge (-180, 0, 0) may have different path from the demo. After discussion with Professor Lane in office hour, the reasons are illustrated as following.

1. quat::FromRotation in the demo code is using algorithm from Orientation Representation -Van Verth and Bishop as well. However, it does not care for the orientation from cases such as 0 to 180 degrees. Thus, there will be sudden direction shift from key2 (-180, 0, 0) to key3 in the demo (which is not optimal in a cubic interpolation). In submission, the code takes consideration of previous orientation, so such direction bump will not occur from one key to another.
2. Also in the code submission, there will be obvious orientation tipping from cases such as , which does not occurs in the demo. The comparison is shown in the Figure 3.1a, Figure 3.1b.

A screenshot of a video game

Description automatically generated with medium confidence

Figure 3.1a: Demo code of

A screenshot of a computer

Description automatically generated with medium confidence

Figure 3.1b: Submitted code of

Notice the tipping of the lateral orientation of the quaternion interpolation. The reason that occurs is because of the different ways of interpolating the edge cases at start segment and end segment. In demo code, it seems the code simply isolate the segment 0 and end segment as special cases and set the for start segment equals ­­ and for end segment equals ­­. Since the two cases are isolated, it will be just a simple cubic interpolation with only too keys . Thus there is no tipping and causes bump as well.

However, in the submitted code, the cases at start and end segment are considered as extended segments with additional neighbor keys, where , . Thus, it will be a full interpolation of . In this case it will be

If we illustrate in graph, it will be as

t

x

-180

y

t

60

Figure 3.2: Curve demonstration of interpolation

Notice the red squared area, where is the tipping occurs.,

Check:

If the explanation is valid, it means tipping can also occur in demo code. We experiment with a segment sequence of in a middle of a continuous splines. As expected, the tipping occurs as shown in Figure 3.3.

A screenshot of a video game

Description automatically generated

Figure 3.3: Tipping Also occurs in demo.

Due to the orientation of camera, it’s the same amount of tipping as the submitted code but just seen from different angle.

Text

Description automatically generated with medium confidence

Cubic B-Spline fully implemented; it shows the same result as Hermite curve as expected.