Name: **SID:** **LAB Section:**

Lab 8 - Part 1: Linear interpolation

---

1. Given a continuous linear function f, defined between points P0 and P1, write the value of f(P) in terms of f(P0), f(P1) and P0, P1, and P.

f(P)=

Let u=|P-P0| / |P1-P0| rewrite f(P) in terms of f(P0), f(P1) and u.

f(P)= _______ f(P0) + ______ f(P1)

This equation can be generalized to any two values K0 and K1 with an interpolation parameter u, to find values in-between:

Replace f(P1) with K1 and f(P0) with K0.

K(u)= _______ K0 + _________ K1

2. Using the equation above find u and the interpolated value in the questions below.

a. An object is positioned at (1,6) coordinates on time t=12 and at (7,-12) on t=18. Find the position of the object at t=14.

u =

position(t=14) = ________ (1,6) + ________ (7,-12)

=

b. The point P0 = (0,7) on the line L has the RGB color (1,0,0.2). The point P1 = (-10, 17) on L is colored with (0.2,1,0.2). Find the color of the point P = (-2,9) on L if the line is smoothly colored with linear interpolation.

u =

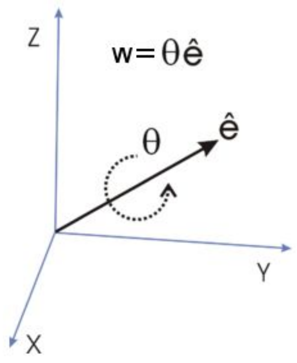
color(P) = _________ (1,0,0.2) + ________ (0.2,1,0.2)

=

Lab 8 - Part 2: Rotation Representations

3D rotations has 3 degrees of freedom and can be represented in various forms such as axis-angle, rotation matrices and quaternions.

Axis-Angle: The rotation is represented as a multiplication of a vector $\hat{e}$ that stores the axis of rotation and a scalar θ for the angle around it: $w = \theta\hat{e}$. This 4D representation is restricted to 3 degrees of freedom by constraining $\hat{e}$ to a unit vector.



https://en.wikipedia.org/wiki/Axis-angle_representation
(google "axis angle representation")

Given a vector w representing a rotation in the axis-angle format, e.g., $w = \theta\hat{e}$, find the rotation axis and angle that it represents in terms of w:

Axis of rotation = ____

Angle of rotation = ____

Rotation-matrix: The rotation is represented as a 3x3 matrix. The elements in this matrix representation are dependent on each other as can be seen in the explicit form below where R is the rotation matrix that represents a rotation around unit vector **u** with angle, θ.

$$R = \begin{bmatrix} \cos\theta + u_x^2(1-\cos\theta) & u_x u_y(1-\cos\theta) - u_z\sin\theta & u_x u_z(1-\cos\theta) + u_y\sin\theta \\ u_y u_x(1-\cos\theta) + u_z\sin\theta & \cos\theta + u_y^2(1-\cos\theta) & u_y u_z(1-\cos\theta) - u_x\sin\theta \\ u_z u_x(1-\cos\theta) - u_y\sin\theta & u_z u_y(1-\cos\theta) + u_x\sin\theta & \cos\theta + u_z^2(1-\cos\theta) \end{bmatrix}.$$

Quaternions are an extension to the complex numbers with a 3 dimensional imaginary vector. A quaternion is represented at the form:

q = a + b**i** + c**j** + d**k**   --OR--
q = (s,v) where s is the scalar real part and v is the imaginary vector, i.e. s=a, v=[a,b,c].

A unit quaternion can be used to represent a 3D rotation. See this article for more details: http://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation

Briefly, a rotation with angle θ around the unit axis vector $\vec{e}$ can be represented by the following quaternion:

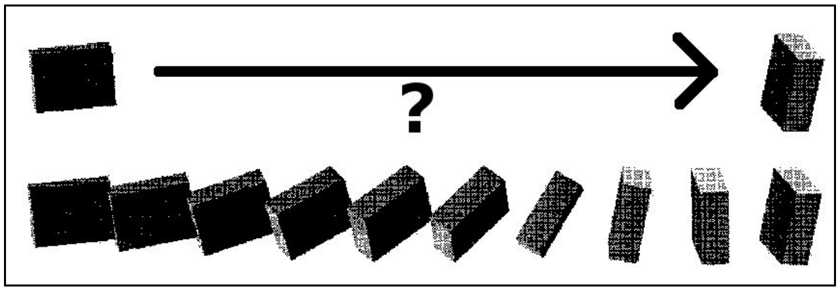
$$q = (s, v) = (cos(\theta/2),\ sin(\theta/2)\ \vec{e}\ )$$

Given a unit quaternion **(s,v)** find the rotation axis and angle that it represents in terms of **s** and **v**:

Axis of rotation = ____

Angle of rotation = ____

Lab 8 - Part 3: Rotation interpolation and Slerp



In this part, we will investigate the interpolation of rotations for different representations, and try to formulate the intermediate rotations given a starting and ending rotation.

1. Interpolating axis-angle with linear interpolation:

Given two rotations in axis-angle representations, w0 and w1, and the interpolation parameter u. Compute the interpolated axis-angle, w, using linear interpolation formulation from part 1.

w =

2. Interpolating rotation-matrix with linear interpolation:

Given two rotations in rotation matrix representations, R0 and R1, and the interpolation parameter u. Compute the interpolated rotation-matrix, R, using linear interpolation formulation from part 1.

R =

3. **Spherical linear interpolation** (**Slerp**): is an quaternion interpolation method that is mainly used for interpolating 3D rotation. Read the wikipedia page: https://en.wikipedia.org/wiki/Slerp and answer the questions below:

Given the interpolation value s and the quaternions q0 and q1 representing the start and the end rotations, respectively, write the Slerp formulation to find the interpolated quaternion q.

q = Slerp(q0,q1,u) = ____________________________________

Select true (T) or false (F)

T / F : The formulation above always follows the shortest path between two quaternions.

T / F : The formulation above always follows the path between two quaternions in constant speed.

T / F : A quaternion represents a valid rotation only if the magnitude of the quaternion is 1.

Lab 8 - Part 4: Code

Download the skeleton code on iLearn and follow the instructions below to complete the lab.

1. Implement the following functions using the formulations in **Part 2.** Note that the quaternion and axis_angle classes uses radians while the input to the function is in degrees.

- **to_angle_and_axis** in quaternion.cpp
- **from_angle_and_axis** in quaternion.cpp
- **to_angle_and_axis** in axis_angle.cpp
- **from_angle_and_axis** in axis_angle.cpp

2. Implement the **Interpolate_Position** function in application.cpp using the formulations in **Part 1.**

Compile and run your code. The center of the plane should follow a smooth line, you may ignore the skips/errors in the rotation of the plane for now.

3. Implement the following functions using the formulations in **Part 3.**

- **Interpolate_Axis_Angle** in application.cpp
- **Interpolate_Rotation_Matrix** in application.cpp
- **slerp** in quaternion.cpp

Make sure that the slerp implementation always uses the shortest path!

4. Test your methods in different scenarios (keys 1-4) and make sure the plane follows a smooth transformation from left to right in all test cases.

5. Briefly write down the differences/issues of the outputs generated by the 3 methods

Rotation-Matrix:

Angle-Axis:

Quaternion (Slerp):

6. (optional) Change the mode by pressing 'm' and make the plane move in its front direction in this mode by changing the code (search 'head' or 'TODO' in application.cpp to find the block that is required to be changed). Note: press 'm' again to switch back to comparison mode.

Hints:
(1) create a vector d that points in the forward direction of the plane at its natural pose (d=(1,0,0)
(2) implement the rotate function in quaternion.cpp that rotates a vector with a quaternion represented rotation.
(3) rotate d and add to position after multiplying it with a speed value.