1 Curves and Surfaces

1.1 Bézier curves of degree 1

In class, we have studied cubic Bézier curves. In this question, we will simplify it to degree 1 polynomials. This case is rather trivial but will allow us to assess your understanding of splines.

The degree 1 Bernstein basis is defined as:

\[ B_1 = t \]
\[ B_2 = (1 - t) \]

How many control points are needed for a degree-1 Bézier curve? [1]

What is the basis matrix for degree-1 Bézier curves, if the power basis is \((1, t)^T\)? [2]

Does the curve interpolate or approximate its control points? [2]

What can you say about the tangent at 0 and 1?

You do not need to provide derivations. [2]
Recall that the DeCasteljeau construction allows us to subdivide a Bézier curve into two Bézier curves by taking a succession of middle points. What is the corresponding construction for degree-1 Bézier splines? [ /5]

1.2 Bézier surfaces of degree 1

We now consider the extension to bi-parametric 3D surfaces $S(u,v)$ defined as a tensor product of degree-1 Bézier curves.

How many control points are needed for such a degree-1 surface? [ /3]

2 Transformations

2.1 Normal transformation

In this question, we consider standard linear coordinates, not homogenous coordinates, and no translation. If a 3D object is linearly transformed by the following matrix:

\[
\begin{pmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0.5
\end{pmatrix}
\]
What is the matrix that gives the normal transformation? Do not worry about the final normalization. [ /3]

If a 3D object is linearly transformed by the following matrix:

\[
\begin{pmatrix}
1 & 0 & 0 \\
0 & 0 & 1 \\
0 & 0.5 & 0
\end{pmatrix}
\]

What is the matrix that gives the normal transformation? Do not worry about the final normalization. [ /5]

2.2 Rotations

How many degrees of freedom for a rotation around the origin in 2D? [ /1]

How many degrees of freedom for a rotation around the origin in 3D? [ /2]
2.3 Skinning

The skinning or SSD equation for the transformation of a vertex can be given by

\[ p'_i = \sum_j w_{ij} T_j B_j^{-1} p_i \]

What does \( j \) index? [ /1]

What is \( B_j \) and why is it needed? [ /4]

Which term(s) vary over time and need to be updated for each frame of the animation? [ /2]

What is the problem if \( \sum_j w_{ij} \neq 1 \)? No need for a proof. [ /3]

3 Animation

3.1 Particle systems

For a system of \( N \) particles in 3D, how big is the state vector \( X \) passed to an ODE solver? [ /1]

In a good implementation of particle systems, who is responsible for the computation of forces? The particle system or the ODE solver? [ /1]
What happens if we forget the diagonal springs for cloth simulation? [ /3]

3.2 Collision Detection

We want to compute the collision between a single 3D point (e.g. a particle) and a bounding sphere hierarchy. The collision method will be called at the root node of the hierarchy. The Node class has the following methods already implemented: Node::radius(), Node::center(), and Node::children(). You are encouraged to use pseudocode and to assume you can traverse all elements of a list using a foreach keyword and that you have access to a good Point3D class.

Write the predicate boolean Node::collide(Point3D pt). [ /8]
3.3 ODE

Write the general equation for $x(t + h)$ for the implicit Euler solver, for a generic single-variable $x$.

Recall that the trapezoid method is the one that does a first (temporary) Euler step, reads the force and takes the average of the force at this temporary location and at the origin. Write the equation for $x(t + h)$ using the trapezoid method and our favorite equation: $x(t) = -kx(t)$. 
