Curve Interpolation Library

Overview

Library contains spline and Bézier functions for creating a smooth 3d curves inspired by three.js

CurveInterpolations

Static utility class providing methods for working with curves

Static methods

- `Vector3 CatmullRom(float t, Vector3 p0, Vector3 p1, Vector3 p2, Vector3 p3)`
  t -- interpolation weight.
  p0, p1, p2, p3 - the points defining the spline curve.

  Returns a point on 3d spline curve using the Catmull-Rom algorithm. (Centripetal Catmull Rom - wiki)

- `QuadraticBezier (float t, Vector3 p0, Vector3 p1, Vector3 p2, Vector3 p3)`
  t -- interpolation weight.
  p0, p1, p2 -- the starting, control and end points defining the curve.

  Returns a point on quadratic bezier curve, defined by a startpoint, endpoint and a single control point. (wikipedia)

- `Vector3 CubicBezier (float t, Vector3 p0, Vector3 p1, Vector3 p2, Vector3 p3)`
  t -- interpolation weight.
  p0, p1, p2, p3 -- the starting, control(twice) and end points defining the curve.

  Returns a point on cubic bezier curve, defined by a start point, endpoint and two control points. (wikipedia)

BaseCurve

An abstract base class for creating a Curve object that contains methods for interpolation.

methods

- `Vector3 GetPoint (float t)`
  t - A position on the curve. Must be in the range [0, 1].

  Returns a vector for a given position on the curve.

  `Vector3 GetPointAt (float u)`
  u - A position on the curve according to the arc length. Must be in the range [0, 1].

  Returns a vector for a given position on the curve according to the arc length.

  `Vector3[] GetPoints (int divisions)`
  divisions -- number of pieces to divide the curve into. Default is 5.

  Returns a set of divisions + 1 points using GetPoint(t).

  `Vector3[] GetSpacedPoints ( int divisions )`
  divisions -- number of pieces to divide the curve into. Default is 5.

  Returns a set of divisions + 1 equi-spaced points using GetPointAt(u).

  `float GetLength ()`
  Get total curve arc length.

  `float[] GetLengths (int divisions)`
  Get list of cumulative segment lengths.

  `void UpdateArcLengths ()`
  Update the cumlative segment distance cache.

  `float GetUtoTmapping ( float u, float distance )`
  Given u in the range (0 .. 1), returns t also in the range (0 .. 1). u and t can then be used to give you points which are equidistant from the ends of the curve, using .getPoint.

  `Vector3 GetTangent ( float t )`
  t - A position on the curve. Must be in the range [0, 1].

  Returns a unit vector tangent at t. If the derived curve does not implement its tangent derivation, two points a small delta apart will be used to find its gradient which seems to give a reasonable approximation.

`Vector3 GetTangentAt ( float u )`
u - A position on the curve according to the arc length. Must be in the range [0, 1].

Returns tangent at a point which is equidistant to the ends of the curve from the point given in GetTangent.

`(Vector3[] tangents, Vector3[] normals, Vector3[] binormals) ComputeFrenetFrames(int segments, bool closed)`
Generates the Frenet Frames. Requires a curve definition in 3D space.

CubicBezierCurve

Contains a smooth 3d cubic bezier curve, defined by a start point, endpoint and two control points.

methods

- `Vector3 GetPoint (float t)`
  Overrides the method in BaseCurve.
  Returns a point on Cubic Bezier spline.

SplineCurve

Contains a smooth 3d spline curve from a series of points using the Catmull-Rom algorithm.

constructor

- `SplineCurve(Vector3[] points, bool closed = false, SplineType curveType = SplineType.Centripetal, float tension = 0.5f)`

  points - An array of Vector3 points
  closed - Whether the curve is closed. Default is false.
  curveType - Type of the curve. Default is centripetal.
  tension - Tension of the curve. Default is 0.5.

methods

- `Vector3 GetPoint (float t)`

  Overrides the method in BaseCurve.
  Returns a point on Cubic Bezier spline.