Subdivision Curves and Surfaces (1)

Hongxin Zhang and Jieqing Feng

2006-12-25

State Key Lab of CAD&CG
Zhejiang University

Contents

Refinement

- Overview
- What is a Refinement Scheme
- A Matrix Method for Refinement
- Example - A Stationary Uniform Refinement Scheme
- Example - A Non-Stationary Uniform Subdivision Scheme
- Refinement Schemes for Meshes

Overview

- Constructing Bézier curves, B-spline curves and subdivision curves from
  - a control polygon + an algorithmic = refinement
- Refinement methods for surface
  - Defined algorithmically
  - Applicable for complex control mesh
  - Simple and easy to implement.

What is a Refinement Scheme

A *refinement* process is a scheme which defines a sequence of control polygons

$$\begin{array}{c} \mathbf{P}_0,\ \mathbf{P}_1,\ \ldots,\ \mathbf{P}_n \\ \mathbf{P}_0^1,\ \mathbf{P}_1^1,\ \ldots,\ \mathbf{P}_{n_1}^1 \\ \mathbf{P}_0^2,\ \mathbf{P}_1^2,\ \ldots,\ \mathbf{P}_{n_2}^2 \\ \vdots \\ \mathbf{P}_0^k,\ \mathbf{P}_1^k,\ \ldots,\ \mathbf{P}_{n_k}^k \\ \vdots \end{array}$$

where for any $k>0$, each $\mathbf{P}_j^k$ can be written as

$$\mathbf{P}_j^k = \sum_{i=0}^{n_{k-1}} \alpha_{i,j,k} \mathbf{P}_i^{k-1}$$

What is a Refinement Scheme

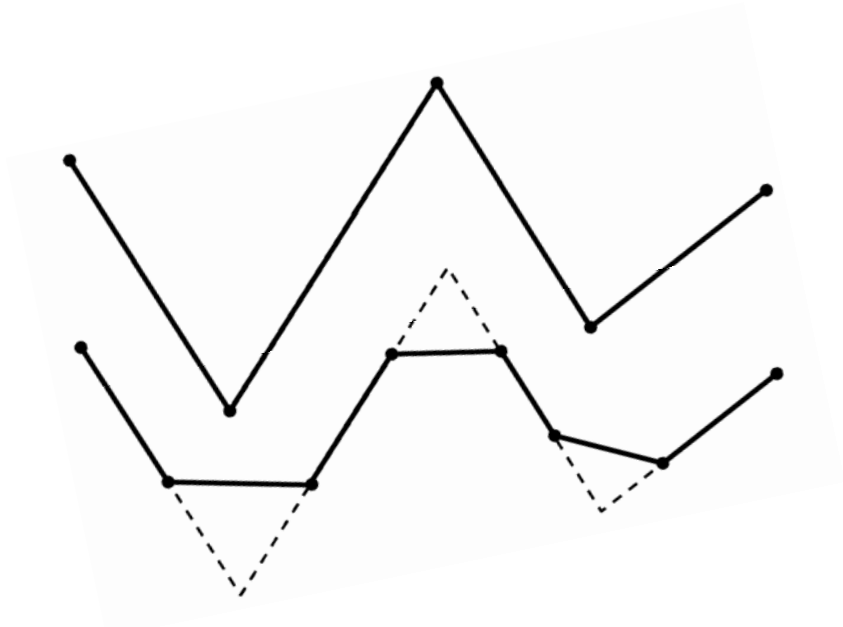
- Any element $\mathbf{P}_j^k$ can be written as a linear combination of the control points $\left\{\mathbf{P}_0^{k-1}, \mathbf{P}_1^{k-1}, \ldots, \mathbf{P}_{n_{k-1}}^{k-1}\right\}$ from the control polygon generated in the prior step
- For each fixed j and k the sequence $\{\alpha_{i,j,k}\}$ is frequently called a *mask*.
- The number of control points in each successive polygon (e.g. curve cases)
  - Increase: Chaikin's Curves
  - Decrease: Geometric Construction of Bézier Curves

Simplification of refinement scheme

- Locality Scheme: Calculate the kst level control points from the small number of k-1st level control points (most of the $\alpha_{i,j,k}$s are zero)
- Uniform Scheme (level,k): αs are independent of the level of refinement k (the scheme is basically the same at each iteration of the refinement process)
- Stationary Scheme (index, j): the mask is the same for every point of a control polygon

Example of Refinement Scheme

- Corner Cutting Scheme: If all points that result from a refinement process lie on the lines joining the points of a control polygon
  - Chaikin's Curve



A Matrix Method for Refinement

The equation of refinement

$$\mathbf{P}_j^k = \sum_{i=0}^{n_{k-1}} \alpha_{i,j,k} \mathbf{P}_i^{k-1}$$

can be written in matrix form as

$$\mathbf{P}_j^k = \begin{bmatrix} \alpha_{0,j,k} & \alpha_{1,j,k} & \cdots & \alpha_{n_{k-1},j,k} \end{bmatrix} \begin{bmatrix} \mathbf{P}_0^{k-1} \\ \mathbf{P}_1^{k-1} \\ \vdots \\ \mathbf{P}_{n_k}^{k-1} \end{bmatrix}$$

A Matrix Method for Refinement

For all control points at k^{th} level

$$\begin{bmatrix} \mathbf{P}_0^k \\ \mathbf{P}_1^k \\ \vdots \\ \mathbf{P}_{n_k}{}^k \end{bmatrix} = S_k \begin{bmatrix} \mathbf{P}_0^{k-1} \\ \mathbf{P}_1^{k-1} \\ \vdots \\ \mathbf{P}_{n_{k-1}}^{k-1} \end{bmatrix}$$

where S_k is the refinement matrix

$$S_k = \begin{bmatrix} \alpha_{0,0,k} & \alpha_{1,0,k} & \cdots & \alpha_{n_{k-1},0,k} \\ \alpha_{0,1,k} & \alpha_{1,1,k} & \cdots & \alpha_{n_{k-1},1,k} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{0,n_k,k} & \alpha_{1,n_k,k} & \cdots & \alpha_{n_{k-1},n_k,k} \end{bmatrix}$$

A Matrix Method for Refinement

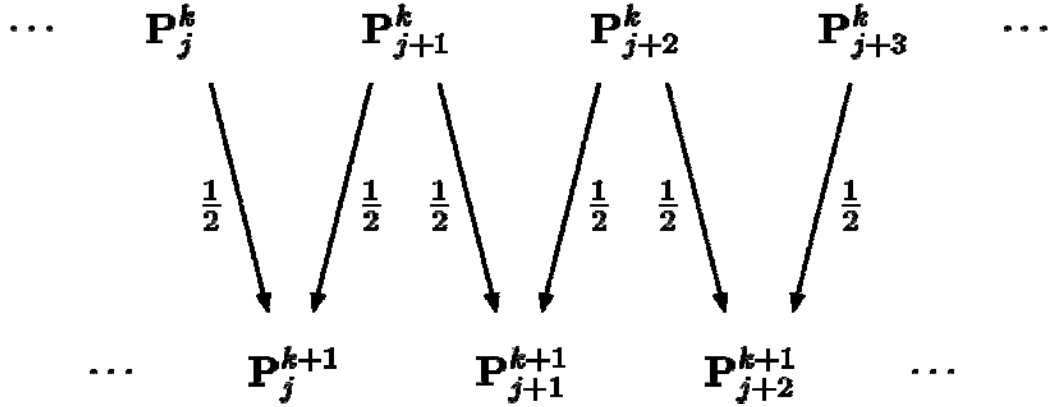
- The S_k is an $(n_k+1)\times(n_{k-1}+1)$ matrix
- In general the matrix is sparse (i.e. most of the entries being zero) with non-zero entries clustered along the diagonal

Example - A Stationary Uniform Refinement Scheme

- Given the control polygon $\{\mathbf{P}_0,\mathbf{P}_1,\ldots,\mathbf{P}_n\}$
- Define refinement scheme by the following equation

$$\mathbf{P}_j^k = \frac{1}{2}\left(\mathbf{P}_j^{k-1} + \mathbf{P}_{j+1}^{k-1}\right)$$

where $0 \le j \le n-k$ and $k=1,2,\ldots,n$. Each successive point is the midpoint of the two corresponding points in the previous control polygon.



Example - A Stationary Uniform Refinement Scheme

- The refinement procedure has following features
  - For each step, one fewer points in control polygon than those in previous step
  - Stop after n steps
  - The final control polygon has one point
- Two of the α s are $1/2$ and the remainder are zero. Thus the refinement matrix S_k is

Example - A Stationary Uniform Refinement Scheme

$$S_k = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & \cdots & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

the matrix is $k \times (k+1)$

- The complete refinement defines a point on the Bézier curve of degree n with the initial control polygon

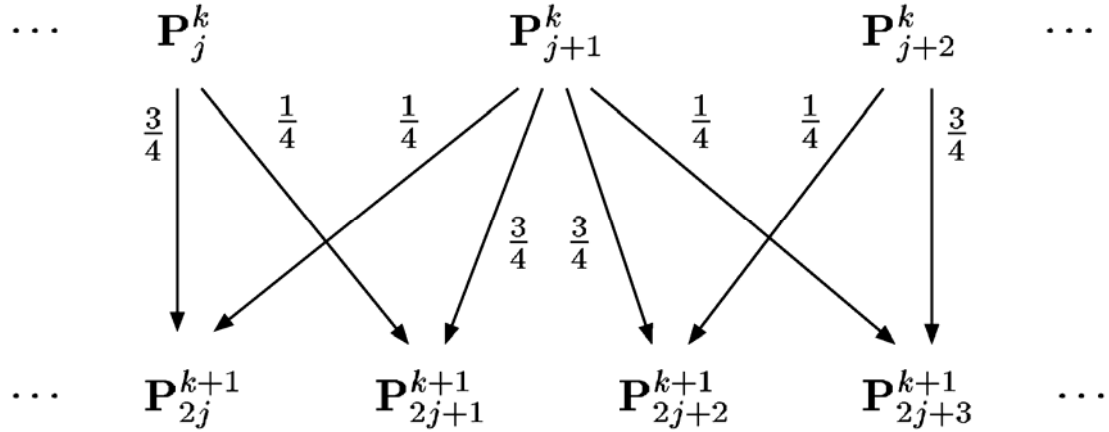
Example - A Non-Stationary Uniform Subdivision Scheme

Given the control polygon $\{\mathbf{P}_0,\mathbf{P}_1,\ldots,\mathbf{P}_n\}$

Define the refinement scheme by the following equation

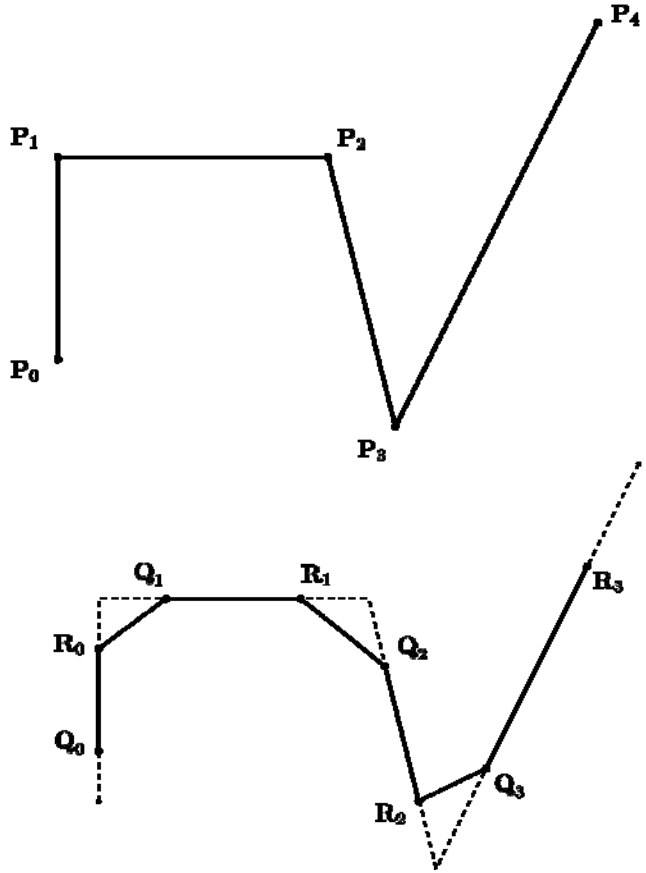
$$\mathbf{P}_{2j}^k = \frac{3}{4}\mathbf{P}_j^k + \frac{1}{4}\mathbf{P}_{j+1}^k \quad \text{and} \quad \mathbf{P}_{2j+1}^k = \frac{1}{4}\mathbf{P}_j^k + \frac{3}{4}\mathbf{P}_{j+1}^k$$

for j=0,1,2,3,....



Example - A Non-Stationary Uniform Subdivision Scheme

$$
\begin{aligned}
\mathbf{P}_0^1 &= \frac{3}{4}\mathbf{P}_0 + \frac{1}{4}\mathbf{P}_1 \\
\mathbf{P}_1^1 &= \frac{1}{4}\mathbf{P}_0 + \frac{3}{4}\mathbf{P}_1 \\
\mathbf{P}_2^1 &= \frac{3}{4}\mathbf{P}_1 + \frac{1}{4}\mathbf{P}_2 \\
\mathbf{P}_3^1 &= \frac{1}{4}\mathbf{P}_1 + \frac{3}{4}\mathbf{P}_2 \\
\mathbf{P}_4^1 &= \frac{3}{4}\mathbf{P}_2 + \frac{1}{4}\mathbf{P}_3 \\
\mathbf{P}_5^1 &= \frac{1}{4}\mathbf{P}_2 + \frac{3}{4}\mathbf{P}_3 \\
&\vdots
\end{aligned}
$$



Example - A Non-Stationary Uniform Subdivision Scheme

- Applying this refinement process to a control polygon of "length" $n+1$ gives a new control polygon of "length" $2n$.
  - length = number of segments in control polygon
- This is just Chaikin's Algorithm for curve generation. As the algorithm proceeds the number of control points gets arbitrarily large, but converges to a unique curve – Uniform Quadratic B-Spline Curve

Refinement Schemes for Meshes

- Similar methods (with much more notationally complex mathematics) exist for control meshes that result in surface generation algorithms.

- In general, the idea is the same - the refinement operation generates new control points from the control points of the previous mesh.

Chaikin's Curves

- Overview
- The Corner-Cutting Paradigm
- Chaikin's Method
- Example - How Chaikin's Algorithm Works
- Example - A Closed Curve
- Discussions

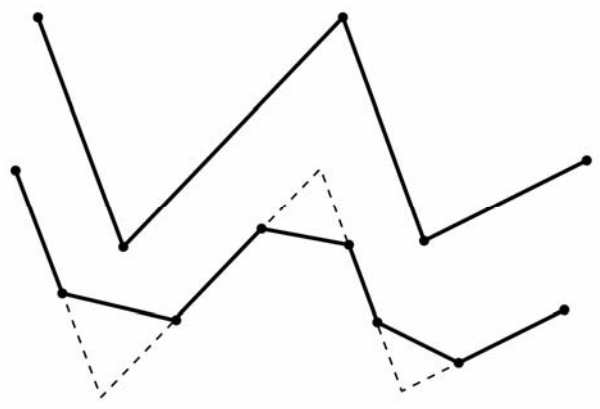
Ovweview

- In 1974, George Chaikin specified the first corner cutting or refinement algorithms to generate a curve from a set of control points, or control polygon. (http://www.cooper.edu/~george/)
- Forgotten: a quadratic uniform B-spline curve.
- His curves were generated by successive refinement of a control polygon – is now utilized to generate a wide variety of curve and surface types.

The Corner-Cutting Paradigm

- Researchers since Bézier had been working with curves generated by control polygons but had focused their analysis on the underlying analytical representation, frequently based upon Bernstein polynomials.
- Chaikin develop algorithms that worked with the control polygon directly -- so-called geometric algorithms.
  - “corner cutting” -- generates a new control polygon by cutting the corners off the original one.

The Corner-Cutting Paradigm



an initial control polygon has been refined into a second polygon (slightly offset) by cutting off the corners of the first sequence

- Clearly we could then take this second control polygon and cut the corners off it, producing a third sequence, etc. In the limit, hopefully we would have a curve. This was Chaikin's idea!

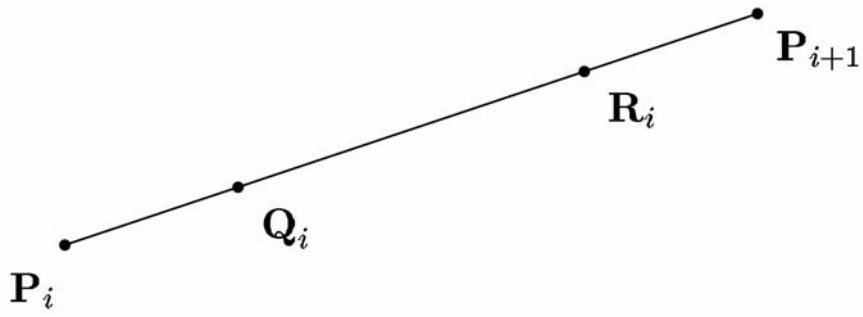
Chaikin's Method

- Chaikin utilized fixed ratios on cutting off his corners, so that they were all cut the same.

Given a control polygon $\{\mathbf{P}_0,\mathbf{P}_1,\ldots,\mathbf{P}_n\}$. we refine this control polygon by generating a new sequence of control points

$$\{\mathbf{Q}_0,\mathbf{R}_0,\mathbf{Q}_1,\mathbf{R}_1,\ldots,\mathbf{Q}_{n-1},\mathbf{R}_{n-1}\}.$$

where each new pair of points $\mathbf{Q}_i$, $\mathbf{R}i$ are to be taken to be at a ratio of ¼ and ¾ between the endpoints of line segment $\mathbf{P}_i\mathbf{P}_{i+1}$.



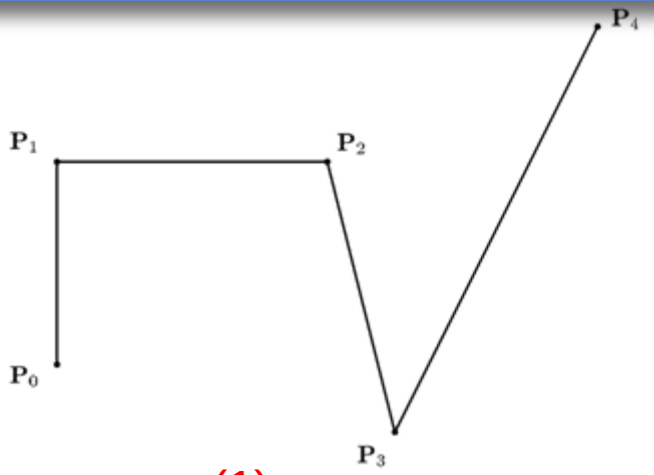
Chaikin's Method

The refinement formulae

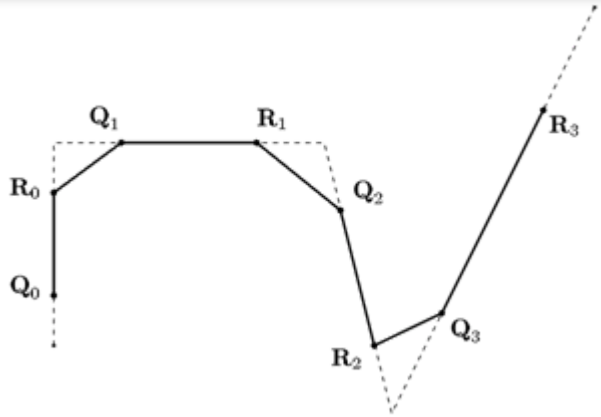
$$\mathbf{Q}_i = \frac{3}{4}\mathbf{P}_i + \frac{1}{4}\mathbf{P}_{i+1}$$
$$\mathbf{R}_i = \frac{1}{4}\mathbf{P}_i + \frac{3}{4}\mathbf{P}_{i+1}$$

- These $2n$ new points can be considered a new control polygon – a *refinement* of the original control polygon.

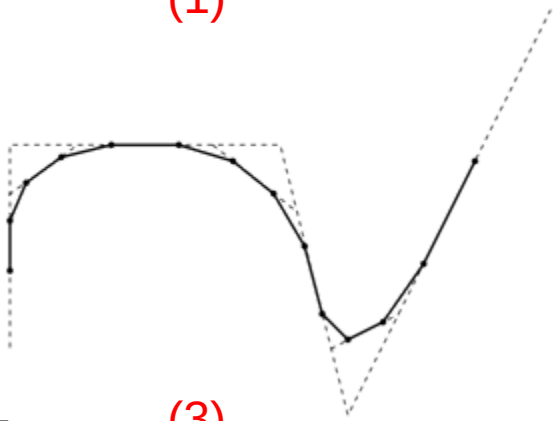
Example - How Chaikin's Algorithm Works



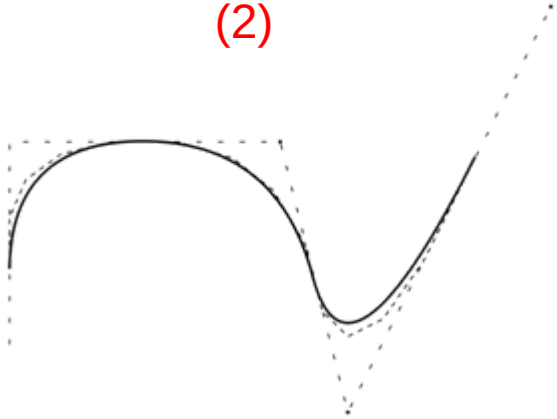
(1)



(2)

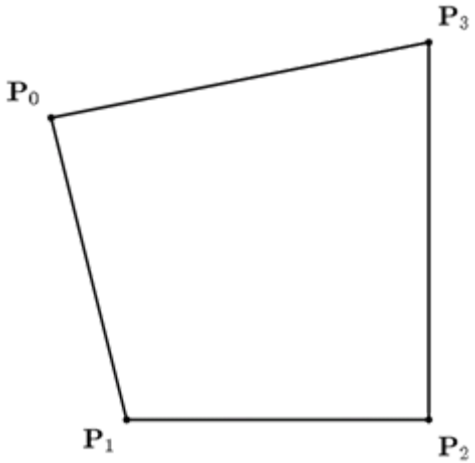


(3)

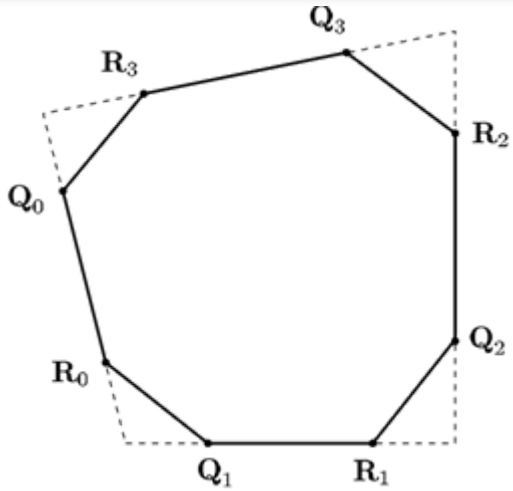


(4)

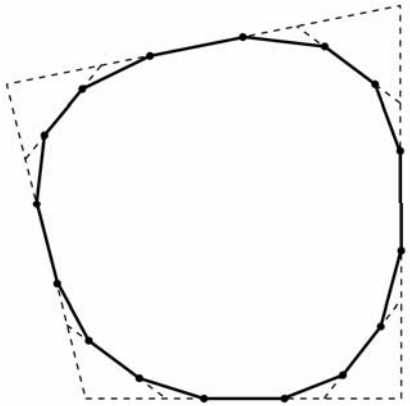
Example - A Closed Curve



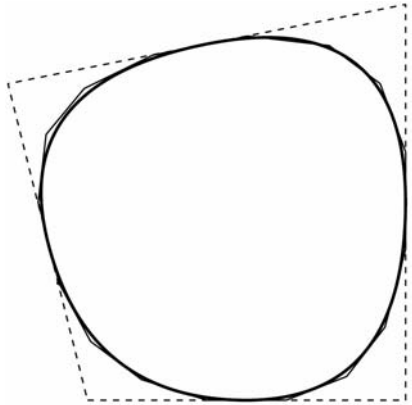
(1)



(2)



(3)



(4)

Discussions

- For different type of control polygons
  - Open control polygon: $(n+1)$ vertices → $2n$ vertices
  - Close control polygon: n vertices → $2n$ vertices
- For graphics purposes, we will stop after a number of refinements and approximate the curve by connecting the points of the resulting control polygon by straight lines.
- The idea is unique in that the underlying mathematical description (uniform quadratic B-spline curve) is ignored in favor of a geometric algorithm that just selects new control points along the line segments of the original control polygon.

Quadratic Uniform B-Spline Curve Refinement

- Overview
- The Matrix Equation for the Quadratic Uniform B-Spline Curve
- Splitting and Refinement
- The General Refinement Procedure

Overview

- Subdivision Curves: the refinement methods are based upon the binary subdivision of uniform B-spline curves
- The refinement method for a quadratic uniform B-spline curve ≡ Chaikin's Algorithm

The Matrix Equation for the Quadratic Uniform B-Spline Curve

Given a set of control points $\{\mathbf{P}_0,\mathbf{P}_1,\ldots,\mathbf{P}_n\}$, the quadratic uniform B-spline curve $\mathbf{P}(t)$ defined by these control points can be defined in n-1 segments by the n-1 equations

$$\mathbf{P}(t) = \begin{bmatrix} 1 & t & t^2 \end{bmatrix} M \begin{bmatrix} \mathbf{P}_k \\ \mathbf{P}_{k+1} \\ \mathbf{P}_{k+2} \end{bmatrix}$$

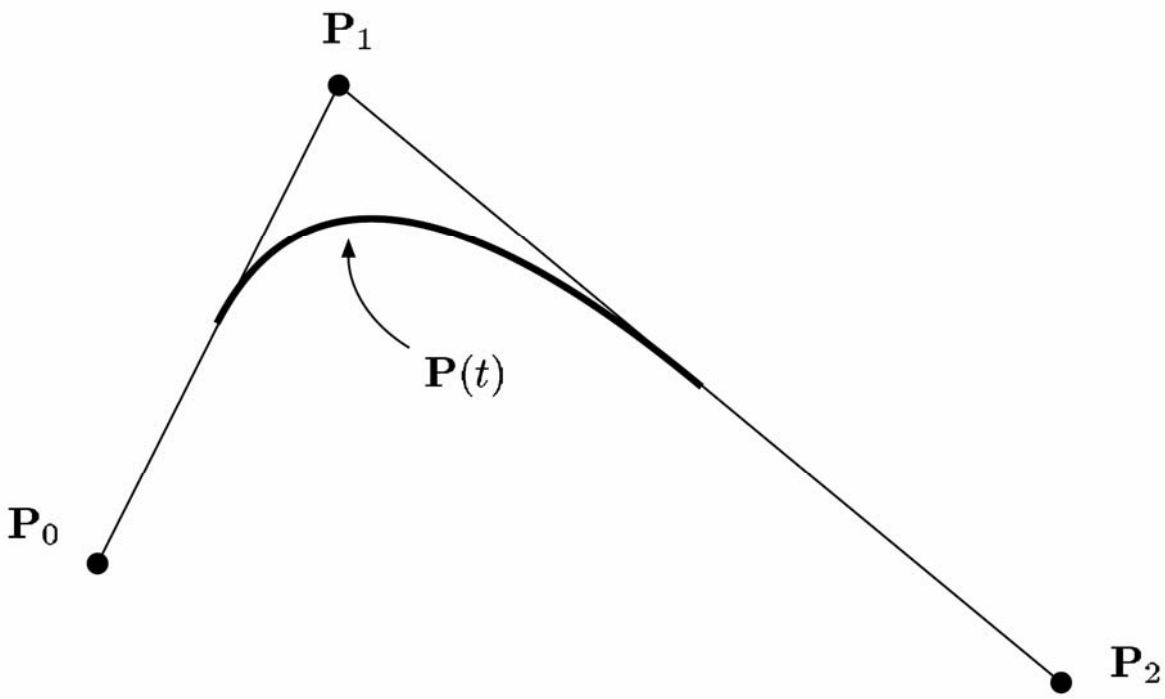
for k=0,1,…,n-2, and $0 \le t \le 1$, and where

$$M = \begin{bmatrix} 1 & 1 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{bmatrix}$$

The matrix M, when multiplied by $[1 \;\; t \;\; t^2]$ defines the quadratic uniform B-spline blending functions

Splitting and Refinement

Studying the binary subdivision of a quadratic uniform B-spline curve $\mathbf{P}(t)$ defined by the control polygon $\{\mathbf{P}_0,\mathbf{P}_1,\mathbf{P}_2\}$



Splitting and Refinement

We can perform a binary subdivision of the curve, by applying one of two splitting matrices

$$S^L = \frac{1}{4}\begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \end{bmatrix} \qquad S^R = \frac{1}{4}\begin{bmatrix} 1 & 3 & 0 \\ 0 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix}$$

to the control polygon. (When applied to the control polygon S^L gives the first half of the curve, and S^R gives the second half.)

Several of the control points for the two subdivided components are the same. Thus, we can combine these matrices, creating a 4×3 matrix as below

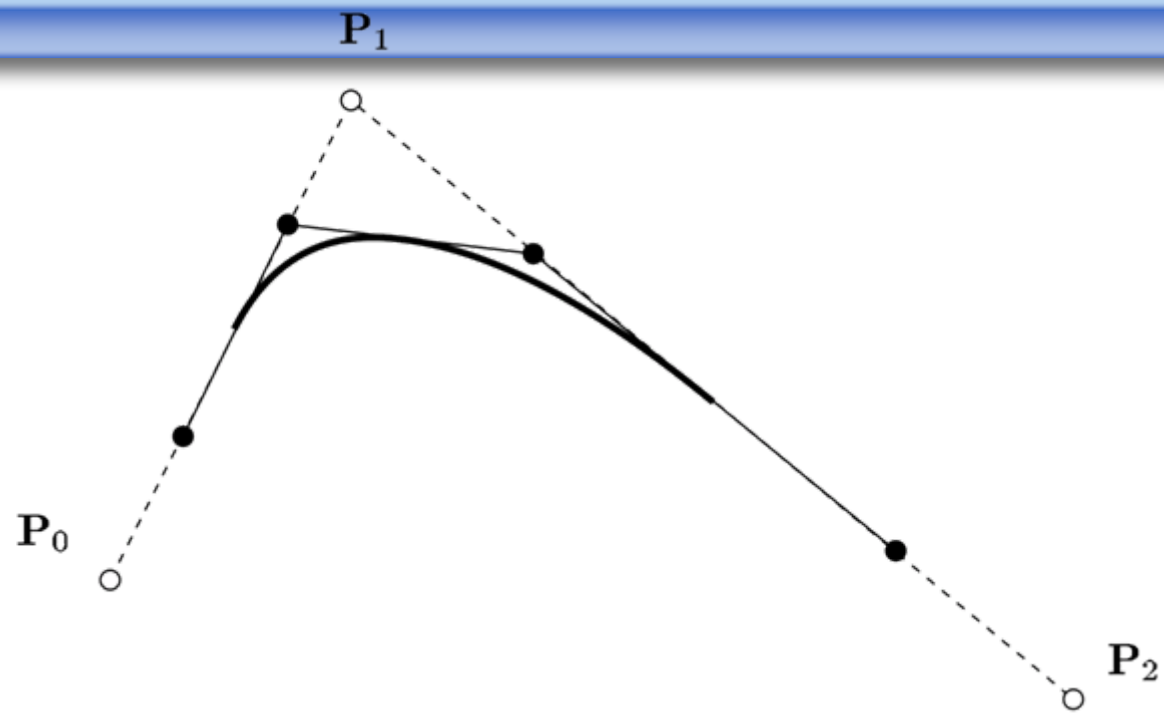
Splitting and Refinement

$$R = \frac{1}{4}\begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix}$$

apply it to a control polygon as follows

$$\begin{bmatrix} \mathbf{P}_0^1 \\ \mathbf{P}_1^1 \\ \mathbf{P}_2^1 \\ \mathbf{P}_3^1 \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix}\begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \end{bmatrix} = \begin{bmatrix} \frac{3}{4}\mathbf{P}_0 + \frac{1}{4}\mathbf{P}_1 \\ \frac{1}{4}\mathbf{P}_0 + \frac{3}{4}\mathbf{P}_1 \\ \frac{3}{4}\mathbf{P}_1 + \frac{1}{4}\mathbf{P}_2 \\ \frac{1}{4}\mathbf{P}_1 + \frac{3}{4}\mathbf{P}_2 \end{bmatrix}$$

Splitting and Refinement



Subdivision at 1/4 and 3/4 along each of the lines of the control polygon. These are the same points as are developed in Chaikin's method

The General Refinement Procedure

The general procedure is

1. Given a control polygon,
2. Constructing new points along each edge of the original polygon at a distance of 1/4 and 3/4 between the endpoints of the edge.
3. Assembling these points into a new control polygon,
4. Useing it as input to another refinement operation, generating a new control polygon (repeating step 2 and 3)
5. Continue this process until a refinement is reached that accurately represents the curve to a desired resolution.

The General Refinement Procedure

The limit of the sequence of control polygons generated by the refinement procedure converges to a quadratic uniform B-spline curve

- The general refinement procedure is developed from the binary subdivision of a uniform B-spline curve
- The control points of the refined polygon are unions of those of the subdivided curves

Chaikin's curve is a quadratic uniform B-spline curve!

Cubic Uniform B-Spline Curve Refinement

- Overview
- The Matrix Equation for the Cubic Uniform B-Spline Curve
- Splitting and Refinement
- The General Refinement Procedure

Overview

- Developing the refinement method for a cubic uniform B-spline curve
- The refinement algorithm can be specified in a different manner which eventually allows us to use eigenanalysis and directly calculate points on the curve.

The Matrix Equation for the Cubic Uniform B-Spline Curve

Given a set of control points $\{\mathbf{P}_0,\mathbf{P}_1,\ldots,\mathbf{P}_n\}$, the cubic uniform B-spline curve $\mathbf{P}(t)$ defined by these control points can be defined in n-2 segments by the n-2 equations

$$\mathbf{P}(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} M \begin{bmatrix} \mathbf{P}_k \\ \mathbf{P}_{k+1} \\ \mathbf{P}_{k+2} \\ \mathbf{P}_{k+3} \end{bmatrix}$$

for k=0,1,…,n-3, and $0 \le t \le 1$, and where

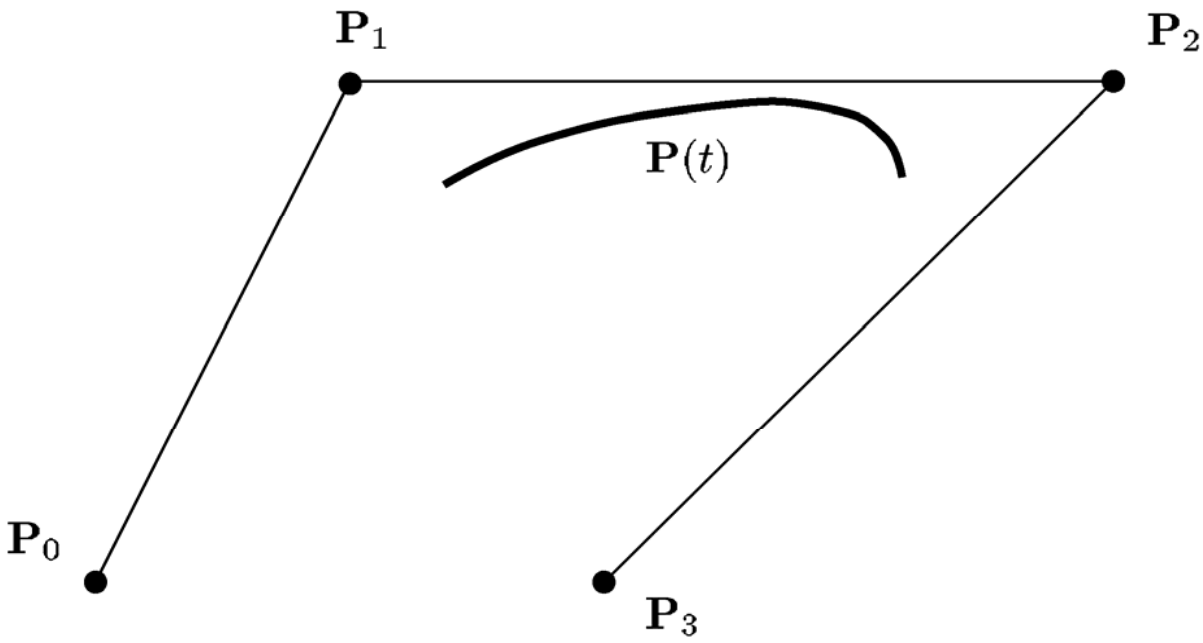
The Matrix Equation for the Cubic Uniform B-Spline Curve

$$M = \frac{1}{6}\begin{bmatrix} 1 & 4 & 1 & 0 \\ -3 & 0 & 3 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

The matrix M, when multiplied by $[1 \;\; t \;\; t^2 \;\; t^3]$ defines the cubic uniform B-spline blending functions

Splitting and Refinement

Studying the binary subdivision of a cubic uniform B-spline curve $\mathbf{P}(t)$ defined by the control polygon $\{\mathbf{P}_0,\mathbf{P}_1,\mathbf{P}_2,\mathbf{P}_3\}$



Splitting and Refinement

We can perform a binary subdivision of the curve, by applying one of two splitting matrices

$$S^L = \frac{1}{8}\begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \end{bmatrix} \qquad S^R = \frac{1}{8}\begin{bmatrix} 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \\ 0 & 0 & 4 & 4 \end{bmatrix}$$

to the control polygon. (When applied to the control polygon S^L gives the first half of the curve, and S^R gives the second half.)

Several of the control points for the two subdivided components are the same. Thus, we can combine these matrices, creating a 5×4 matrix as below

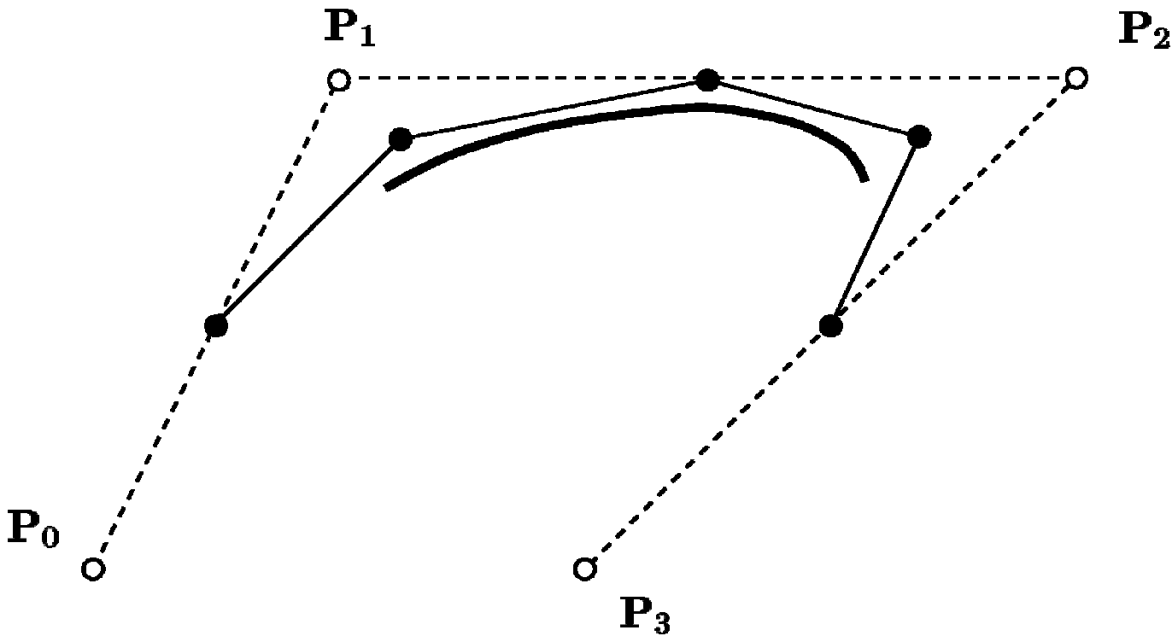
Splitting and Refinement

$$\frac{1}{8}\begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \\ 0 & 0 & 4 & 4 \end{bmatrix}$$

apply it to a control polygon as follows

$$\begin{bmatrix} \mathbf{P}_0^1 \\ \mathbf{P}_1^1 \\ \mathbf{P}_2^1 \\ \mathbf{P}_3^1 \\ \mathbf{P}_4^1 \end{bmatrix} = \frac{1}{8}\begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \\ 0 & 0 & 4 & 4 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix}$$

Splitting and Refinement



generating a new control polygon which serves as the refinement of the original. The five control points of this new control polygon specify the two subdivided halves of the curve - and therefore specify the curve itself

The General Refinement Procedure

- If examining the rows of 5×4 matrix used in the refinement, we see that they have two distinct forms.

$$\begin{bmatrix} \mathbf{P}_0^1 \\ \mathbf{P}_1^1 \\ \mathbf{P}_2^1 \\ \mathbf{P}_3^1 \\ \mathbf{P}_4^1 \end{bmatrix} = \frac{1}{8} \begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \\ 0 & 0 & 4 & 4 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix}$$

Edge point

Vertex point

- Classifying the points of the refinement as vertex and edge points. This classification makes the description of the refinement process quite straightforward.

Vertex and Edge Points

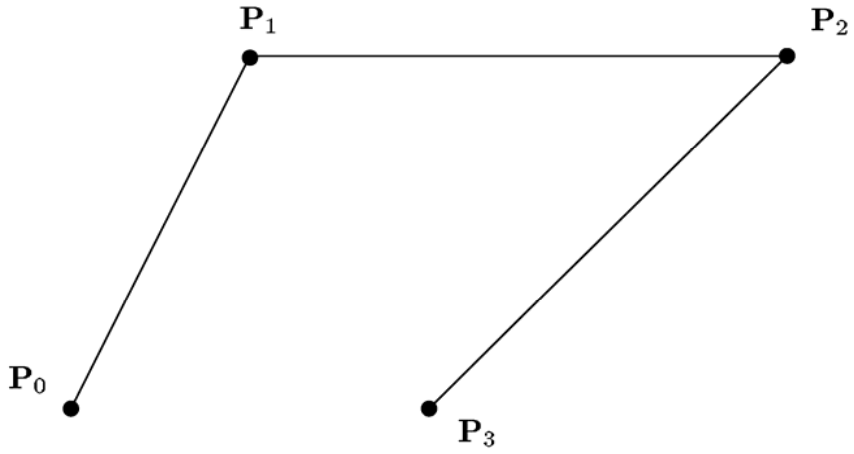
- Overview
- Classifying the Refinement Points
- An Example of the Refinement Algorithm

Overview

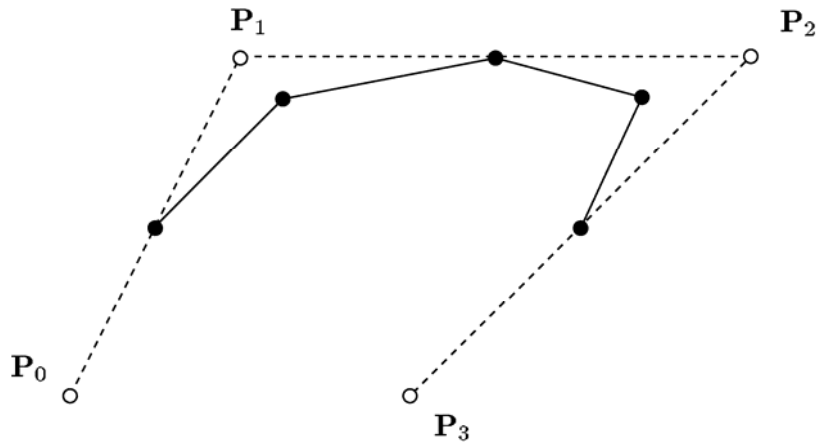
- The refinement process defined by the cubic uniform B-spline curve generates a sequence of control polygons that converges to the curve.
- Each of refinement points can be classified as either a “vertex point” or an “edge point” and methods can be specified to calculate each point.
- This procedure allows us to directly calculate points on the limit curve without going through the refinement

Classifying the Refinement Points

Suppose we are given a control polygon $\{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3\}$



Initial Control Points



Refinement Control Points

$\{\mathbf{E}_0, \mathbf{V}_0, \mathbf{E}_1, \mathbf{V}_1, \mathbf{E}_2\}$

Edge points $\{\mathbf{E}_0, \mathbf{E}_1, \mathbf{E}_2\}$: three points on the edges

Vertex points $\{\mathbf{V}_0, \mathbf{V}_1\}$: two points close to original points

Classifying the Refinement Points

Combining and applying the splitting matrices that generate the binary subdivision of the curve

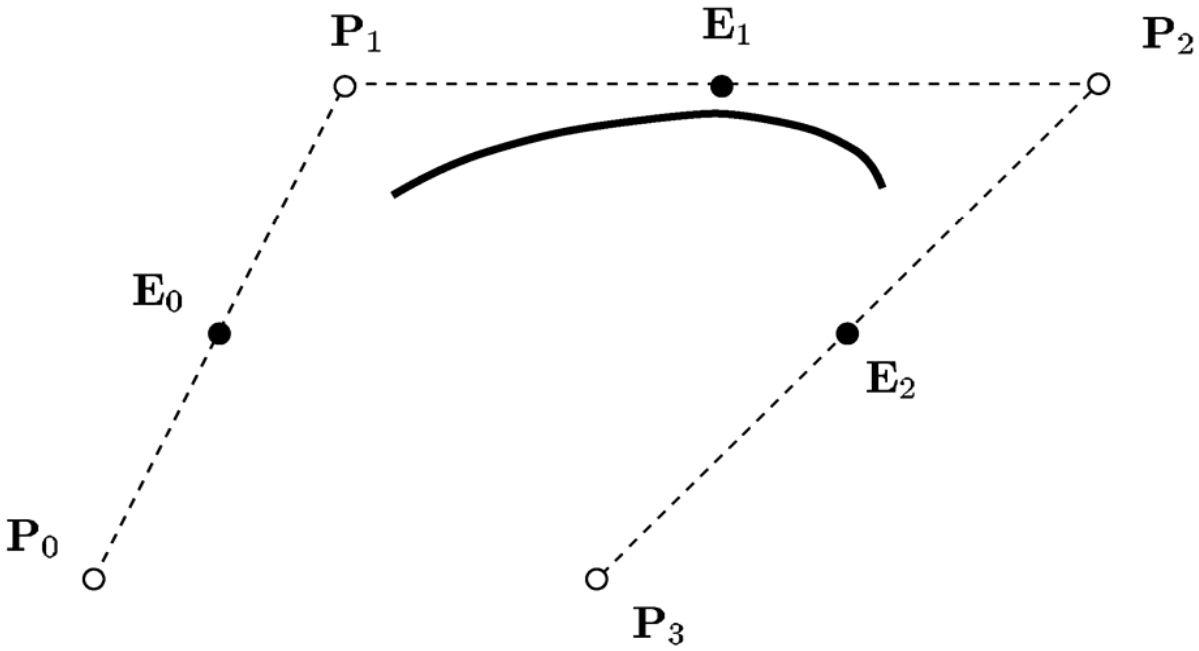
$$\begin{bmatrix} \mathbf{E}_0 \\ \mathbf{V}_0 \\ \mathbf{E}_1 \\ \mathbf{V}_1 \\ \mathbf{E}_2 \end{bmatrix} = \frac{1}{8} \begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \\ 0 & 0 & 4 & 4 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix}$$

So the edge points are calculated by

$$\begin{aligned} \mathbf{E}_0 &= \frac{1}{8}(4\mathbf{P}_0 + 4\mathbf{P}_1) \\ &= \frac{\mathbf{P}_0 + \mathbf{P}_1}{2} \end{aligned} \qquad \begin{aligned} \mathbf{E}_1 &= \frac{1}{8}(4\mathbf{P}_1 + 4\mathbf{P}_2) \\ &= \frac{\mathbf{P}_1 + \mathbf{P}_2}{2} \end{aligned} \qquad \begin{aligned} \mathbf{E}_2 &= \frac{1}{8}(4\mathbf{P}_2 + 4\mathbf{P}_3) \\ &= \frac{\mathbf{P}_2 + \mathbf{P}_3}{2} \end{aligned}$$

Classifying the Refinement Points

The edge points are the midpoints of the line segments connecting the original control points.



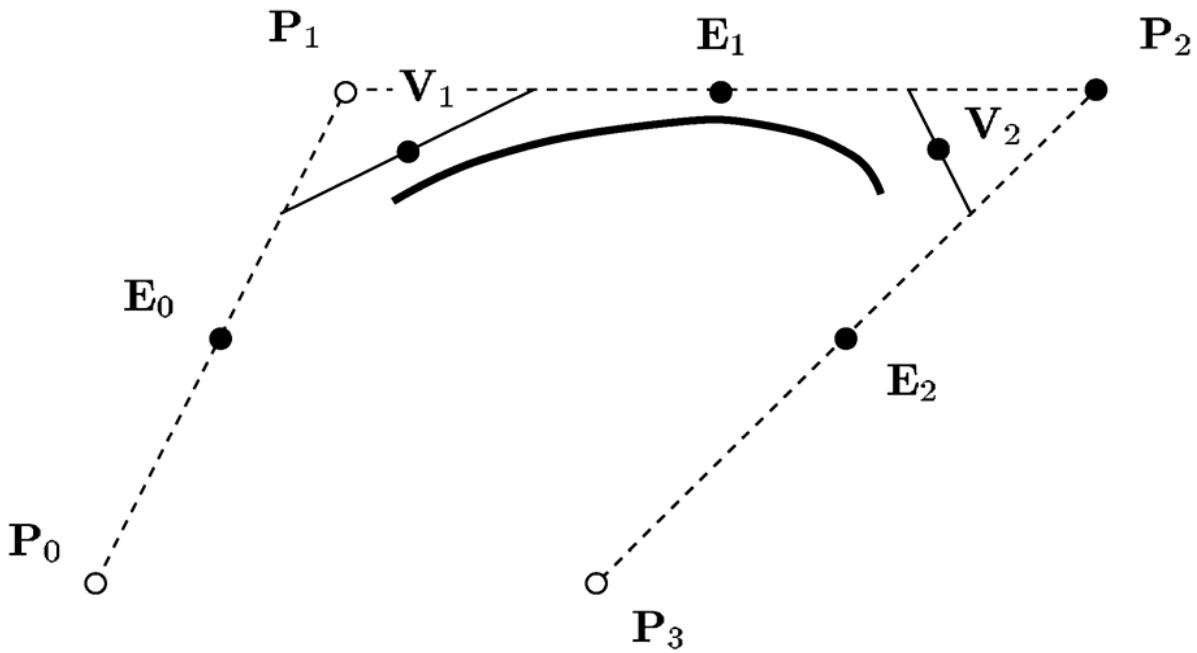
Classifying the Refinement Points

The vertex points $\mathbf{V}_0$ and $\mathbf{V}_1$ are calculated by

$$
\begin{aligned}
\mathbf{V}_0 &= \frac{1}{8}(\mathbf{P}_0 + 6\mathbf{P}_1 + \mathbf{P}_2) \\
&= \frac{1}{8}((\mathbf{P}_0 + \mathbf{P}_1) + 4\mathbf{P}_1 + (\mathbf{P}_1 + \mathbf{P}_2)) \\
&= \frac{1}{4}(\frac{\mathbf{P}_0 + \mathbf{P}_1}{2} + 2\mathbf{P}_1 + \frac{\mathbf{P}_1 + \mathbf{P}_2}{2}) \\
&= \frac{1}{4}(\mathbf{E}_0 + 2\mathbf{P}_1 + \mathbf{E}_1) \\
&= \frac{\frac{\mathbf{E}_0+\mathbf{P}_1}{2} + \frac{\mathbf{P}_1+\mathbf{E}_1}{2}}{2}
\end{aligned}
$$

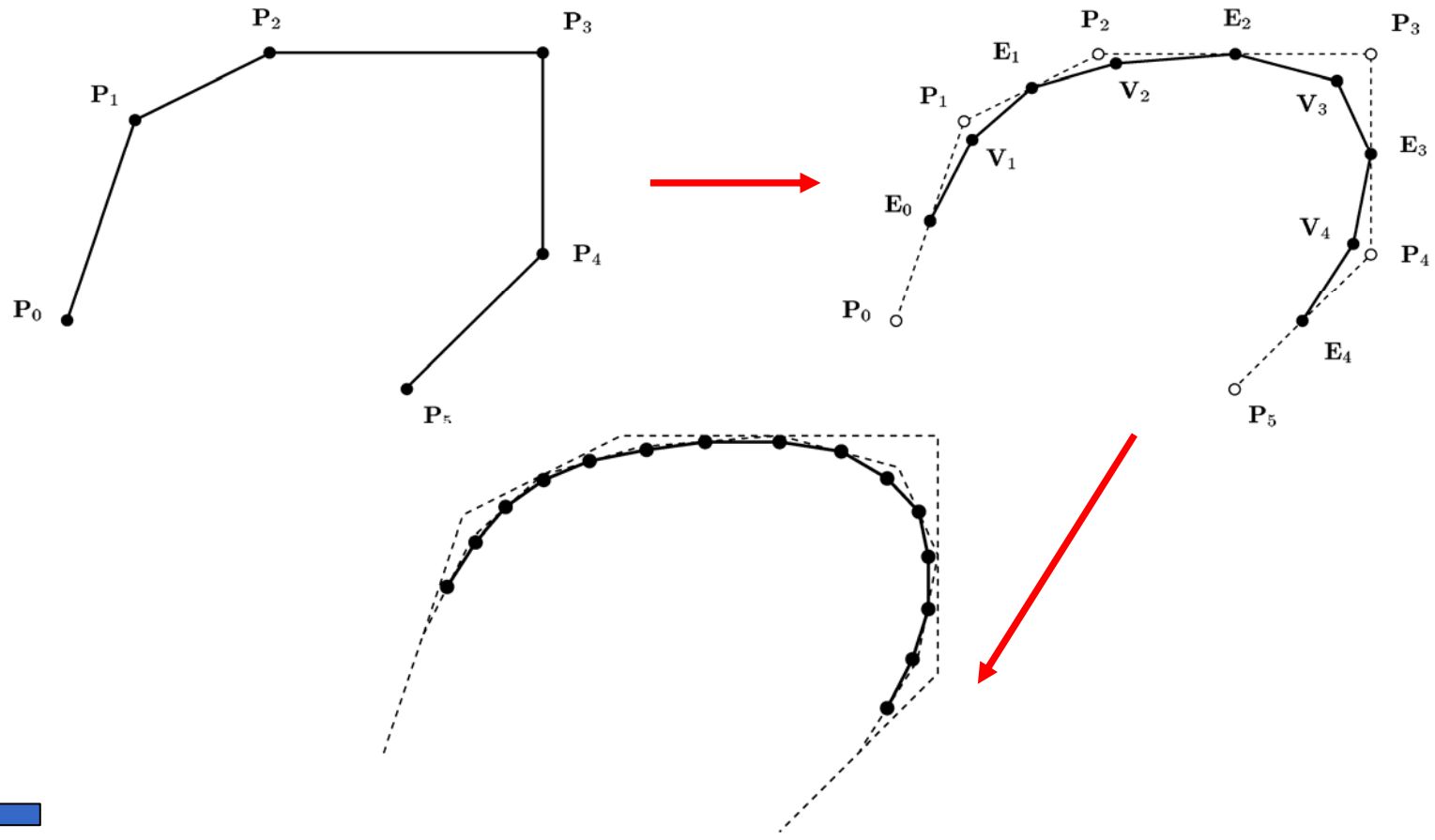
$$
\begin{aligned}
\mathbf{V}_1 &= \frac{1}{8}(\mathbf{P}_1 + 6\mathbf{P}_2 + \mathbf{P}_3) \\
&= \frac{1}{8}((\mathbf{P}_1 + \mathbf{P}_2) + 4\mathbf{P}_2 + (\mathbf{P}_2 + \mathbf{P}_3)) \\
&= \frac{1}{4}(\frac{\mathbf{P}_1 + \mathbf{P}_2}{2} + 2\mathbf{P}_2 + \frac{\mathbf{P}_2 + \mathbf{P}_3}{2}) \\
&= \frac{1}{4}(\mathbf{E}_1 + 2\mathbf{P}_2 + \mathbf{E}_2) \\
&= \frac{\frac{\mathbf{E}_1+\mathbf{P}_2}{2} + \frac{\mathbf{P}_2+\mathbf{E}_2}{2}}{2}
\end{aligned}
$$

Classifying the Refinement Points



The vertex points are the midpoint of the line segment that joins the respective midpoints of the two line segments from the vertex point to the respective edge points.

An Example of the Refinement Algorithm



Developing a Matrix Equation for Refinement

- Overview
- Refinement at a Vertex Point
- So How Do You Do Refinement In General ?
- Summary

Overview

Refinement of the cubic uniform B-spline curve can be written as

$$\begin{bmatrix} \mathbf{P}_0^1 \\ \mathbf{P}_1^1 \\ \mathbf{P}_2^1 \\ \mathbf{P}_3^1 \\ \mathbf{P}_4^1 \end{bmatrix} = \frac{1}{8} \begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \\ 0 & 0 & 4 & 4 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix}$$

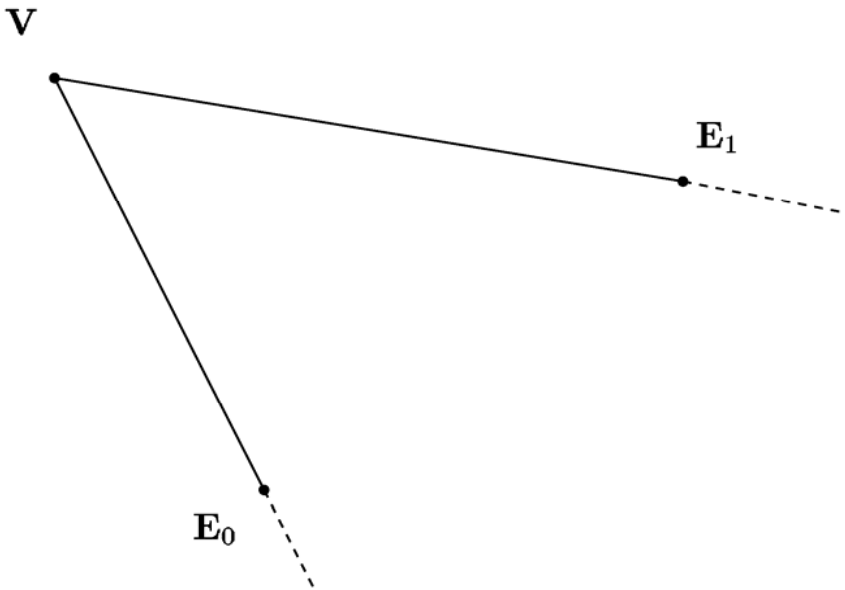
where $\{\mathbf{P}_0,\mathbf{P}_1,\mathbf{P}_2,\mathbf{P}_3\}$ is the original control polygon and $\{\mathbf{P}_0^1, \mathbf{P}_1^1, \mathbf{P}_2^1, \mathbf{P}_3^1, \mathbf{P}_4^1\}$ is the result of the refinement.

Overview

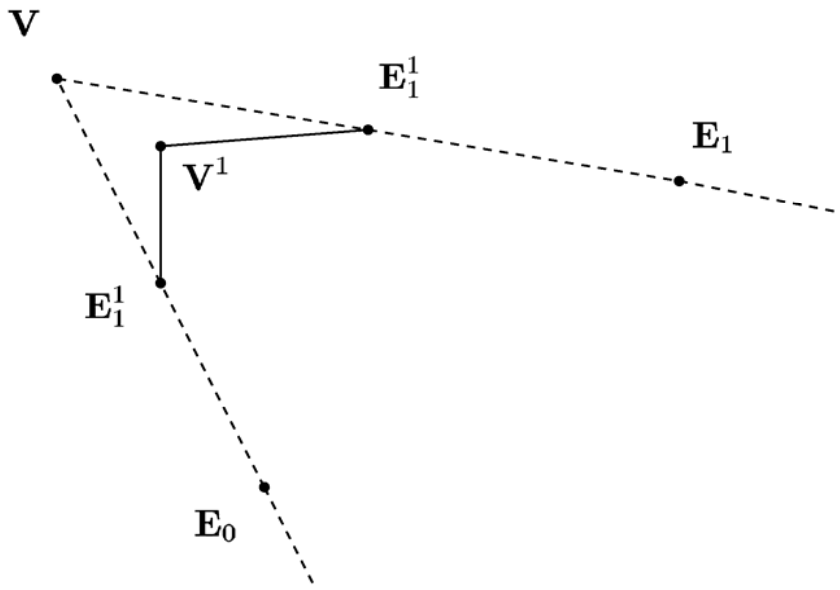
- We can classify points of the refinement as vertex points and edge points
- Here we examine what happens to a particular control point (vertex point) under this refinement procedure. Examining this closely, we can develop an alternate matrix equation that can also be used to represent the refinement procedure

Refinement at a Vertex Point

Focusing on a single vertex point (**V**) of the control polygon and the two points ($\mathbf{E}_0$, $\mathbf{E}_1$) adjacent to **V**



Original vertex points (**V**) and two neighbor points ($\mathbf{E}_0$, $\mathbf{E}_1$)



After refinement

Refinement at a Vertex Point

- In the refinement procedure, we can use these three points to create a new vertex point $\mathbf{V}^1$ and two new edge points $\mathbf{E}_0{}^1$ and $\mathbf{E}_1{}^1$

- We can write this in vector form (we note that the matrices are applied to vectors of points, and therefore the operations must be affine).

$$\begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix} \quad \text{is refined into} \quad \begin{bmatrix} \mathbf{V}^1 \\ \mathbf{E}_0^1 \\ \mathbf{E}_1^1 \end{bmatrix}$$

Refinement at a Vertex Point

$$\begin{bmatrix} \mathbf{V}^1 \\ \mathbf{E}_0^1 \\ \mathbf{E}_1^1 \end{bmatrix} = \begin{bmatrix} \frac{1}{4}\left(\frac{\mathbf{V}+\mathbf{E}_0}{2} + 2\mathbf{V} + \frac{\mathbf{V}+\mathbf{E}_1}{2}\right) \\ \frac{\mathbf{V}+\mathbf{E}_0}{2} \\ \frac{\mathbf{V}+\mathbf{E}_1}{2} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{8}\left(\mathbf{E}_0 + 6\mathbf{V} + \mathbf{E}_1\right) \\ \frac{\mathbf{V}+\mathbf{E}_0}{2} \\ \frac{\mathbf{V}+\mathbf{E}_1}{2} \end{bmatrix}$$

$$= \frac{1}{8}\begin{bmatrix} 6 & 1 & 1 \\ 4 & 4 & 0 \\ 4 & 0 & 4 \end{bmatrix}\begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix}$$

where matrix A

$$A = \frac{1}{8}\begin{bmatrix} 6 & 1 & 1 \\ 4 & 4 & 0 \\ 4 & 0 & 4 \end{bmatrix}$$

is called *refinement matrix*

So How Do You Do Refinement In General ?

Given a set of control points $\{\mathbf{P}_0,\mathbf{P}_1,\ldots,\mathbf{P}_n\}$, we the vectors

$$\begin{bmatrix}\mathbf{P}_1\\ \mathbf{P}_0\\ \mathbf{P}_2\end{bmatrix}, \begin{bmatrix}\mathbf{P}_2\\ \mathbf{P}_1\\ \mathbf{P}_3\end{bmatrix}, \ldots, \begin{bmatrix}\mathbf{P}_{n-1}\\ \mathbf{P}_{n-2}\\ \mathbf{P}_n\end{bmatrix}$$

apply the *refinement matrix* to each vector, giving new vertex and edge points

$$\begin{bmatrix}\mathbf{V}_0\\ \mathbf{E}_0\\ \mathbf{E}_1\end{bmatrix}, \begin{bmatrix}\mathbf{V}_1\\ \mathbf{E}_1\\ \mathbf{E}_2\end{bmatrix}, \ldots, \begin{bmatrix}\mathbf{V}_{n-2}\\ \mathbf{E}_{n-2}\\ \mathbf{E}_{n-1}\end{bmatrix}$$

These new points are then assembled into new control polygon

$$\{\mathbf{E}_0, \mathbf{V}_0, \mathbf{E}_1, \mathbf{V}_1, \ldots, \mathbf{V}_{n-3}, \mathbf{E}_{n-2}, \mathbf{V}_{n-2}, \mathbf{E}_{n-1}\}$$

Summary

- It is possible to write the refinement process for cubic subdivision curves in a matrix form which focuses on the action of the refinement on a single control point. In this case, we can calculate new vertex points and edges points just by applying the refinement matrix to the vertex-edge-point vector.
- We note that this procedure does take more processor time than the refinement based upon binary subdivision. However, it will enable us to analyze the refinement operation further and generate a procedure to directly calculate a point on the curve without resulting to refinement.

Eigen-Analysis for Refinement Matrices

- Overview
- The Eigenvalues of the Refinement Matrix
- Diagonalization of the Matrix
- The Inverse of the Refinement Matrix
- Summary

Overview

In the cubic case, the refinement procedure can be specified by a matrix operation

$$A = \frac{1}{8}\begin{bmatrix} 6 & 1 & 1 \\ 4 & 4 & 0 \\ 4 & 0 & 4 \end{bmatrix}$$

- The eigenvalues and eigenvectors of the refinement matrix play an important role in the analysis of subdivision curves.
- This matrix is diagonalizable and we use the eigenvalues and eigenvectors to calculate this diagonal decomposition.
- The diagonal decomposition allows an easy calculation of the inverse of the matrix.

The Eigenvalues of the Refinement Matrix

Definition of Eigenvalue:

$A\mathbf{V}=\lambda\mathbf{V}$ (right) **V—column vector**

$\mathbf{V}A=\lambda\mathbf{V}$ (left) **V—row vector**

$\lambda=1$ is a eigenvalue, $\lambda=$ ½, ¼,

$$A\begin{bmatrix}1\\1\\1\end{bmatrix}=\begin{bmatrix}1\\1\\1\end{bmatrix}$$

The Eigenvalues of the Refinement Matrix

Three right eigenvalues

$$\vec{r}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \vec{r}_2 = \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix}, \vec{r}_3 = \begin{bmatrix} -1 \\ 2 \\ 2 \end{bmatrix}$$

Three left eigenvalues

$$\begin{aligned} \vec{l}_1 &= \begin{bmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \end{bmatrix} \\ \vec{l}_2 &= \begin{bmatrix} 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \\ \vec{l}_3 &= \begin{bmatrix} -\frac{1}{3} & \frac{1}{6} & \frac{1}{6} \end{bmatrix} \end{aligned}$$

Diagonalization of the Matrix

Let L be the 3×3 matrix whose rows are the left eigenvectors of A

$$L = \begin{bmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \\ 0 & \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{3} & \frac{1}{6} & \frac{1}{6} \end{bmatrix}$$

Let R be the 3×3 matrix whose columns are the right eigenvectors of A

$$R = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 1 & 2 \\ 1 & -1 & 2 \end{bmatrix}$$

Diagonalization of the Matrix

Noting that $R=L^{-1}$

The matrix A is diagonalizable and can be written as

$$A = R\Lambda L$$

where Λ is the diagonal matrix whose diagonal elements are the eigenvalues of A.

$$\Lambda = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix}$$

The Inverse of the Refinement Matrix

Since $A = R\Lambda L$, it is easy to generate the inverse of A

$$A^{-1} = L^{-1}\Lambda^{-1}R^{-1}$$

$$= R\Lambda^{-1}L$$

$$\Lambda^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{bmatrix}$$

$$A^{-1} = \frac{1}{2}\begin{bmatrix} 4 & -1 & -1 \\ -4 & 5 & 1 \\ -4 & 1 & 5 \end{bmatrix}$$

Summary

- The eigenvalues and eigenvectors of the refinement matrix are straightforward to calculate.
- Since this matrix is well conditioned it can be diagonalized and written in form $R\Lambda L$ which will be very useful when analyzing subdivision curves.

Direct Calculation of Points on Cubic Subdivision Curves

Overview

- Given an initial control polygon we can define a refinement process that generates a sequence of control polygons from the original.
- In the limit, this sequence converges to the uniform B-spline curve specified by the original control polygon.
- By examining this refinement process from a different angle, we can specify a procedure that allows us to directly calculate points on the curve.

Direct Calculation of Points on the Curve

In the cubic case, the refinement process related to a control point and its two adjacent control points can be written in a matrix form as follows

$$\begin{bmatrix} \mathbf{V}^1 \\ \mathbf{E}_0^1 \\ \mathbf{E}_1^1 \end{bmatrix} = A \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix}$$

where the refinement matrix A is

$$A = \frac{1}{8} \begin{bmatrix} 6 & 1 & 1 \\ 4 & 4 & 0 \\ 4 & 0 & 4 \end{bmatrix}$$

Direct Calculation of Points on the Curve

The procedure creates two new edge points and a vertex point which are part of a new control polygon that represents the curve.

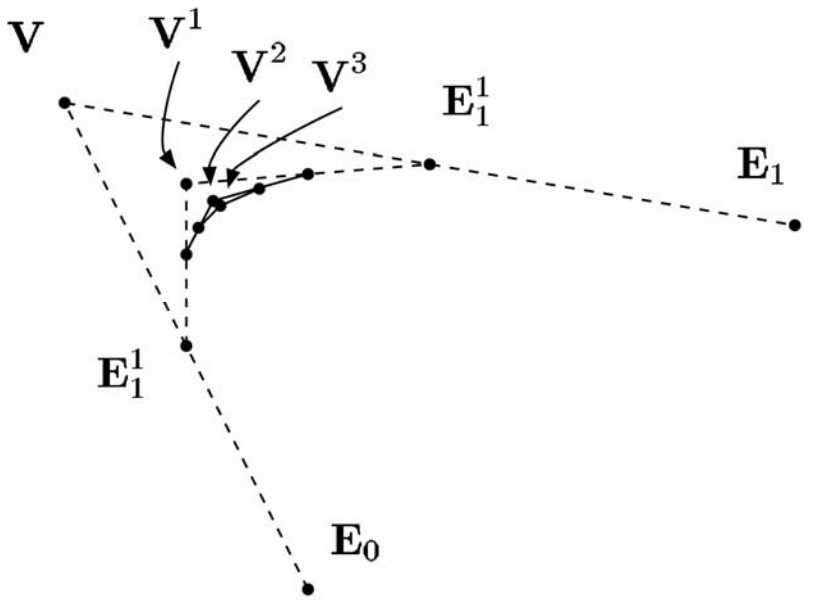
Refining the three control points again

$$\begin{bmatrix} \mathbf{V}^2 \\ \mathbf{E}_0^2 \\ \mathbf{E}_1^2 \end{bmatrix} = A \begin{bmatrix} \mathbf{V}^1 \\ \mathbf{E}_0^1 \\ \mathbf{E}_1^1 \end{bmatrix} = A^2 \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix}$$

where $A^2=AA$. k^{th} refinement means A^k. $k\rightarrow\infty$

Direct Calculation of Points on the Curve

k^{th} refinement means A^k. $k \to \infty$, obtain a point on the curve.



We call this point $\mathbf{V}^\infty$ and note that it is equal to $\lim_{k\to\infty}\mathbf{V}^k$.

Calculating the Limit Point

The limit calculation utilizes the fact that the refinement matrix A can be diagonalized

$$A = R\Lambda L$$

where

$$R = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 1 & 2 \\ 1 & -1 & 2 \end{bmatrix} \qquad \Lambda = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} \qquad L = \begin{bmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \\ 0 & \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{3} & \frac{1}{6} & \frac{1}{6} \end{bmatrix}$$

right eigenvectors of A | eigenvalues of A | left eigenvectors of A

Calculating the Limit Point

Since $R = L^{-1}$, this allows us to write

$$A^2 = (R\Lambda L)^2 = R\Lambda L R\Lambda L = R\Lambda^2 L$$

$$A^k = R\Lambda^k L$$

Calculating the Limit Point

To calculate the limit, first consider applying A k-times

$$\begin{bmatrix} \mathbf{V}^k \\ \mathbf{E}_0^k \\ \mathbf{E}_1^k \end{bmatrix} = A^k \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix}$$

$$= (R\Lambda L)^k \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix}$$

$$= R\Lambda^k L \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix}$$

$$= R \begin{bmatrix} (1)^k & 0 & 0 \\ 0 & \left(\frac{1}{2}\right)^k & 0 \\ 0 & 0 & \left(\frac{1}{4}\right)^k \end{bmatrix} L \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix}$$

Calculating the Limit Point

Now as $k \to \infty$, this approaches

$$R \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} L \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix}$$

Calculating the Limit Point

by substituting for L and R

$$
\begin{aligned}
R\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} L \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix} &= \begin{bmatrix} 1 & 0 & -1 \\ 1 & 1 & 2 \\ 1 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \\ 0 & \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{3} & \frac{1}{6} & \frac{1}{6} \end{bmatrix} \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \\ 0 & \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{3} & \frac{1}{6} & \frac{1}{6} \end{bmatrix} \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix} \\
&= \begin{bmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \\ \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \\ \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \end{bmatrix} \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix} \\
&= \frac{1}{6} \begin{bmatrix} 4 & 1 & 1 \\ 4 & 1 & 1 \\ 4 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix}
\end{aligned}
$$

Calculating the Limit Point

The vertex and edge points all converge to the same value,

$$(4\mathbf{V} + \mathbf{E}_0 + \mathbf{E}_1)/6$$

which is just the dot product of the left eigenvector of A that corresponds to the eigenvalue 1, and the original vertex-edge vector of the refinement.

$$\frac{1}{6}\begin{bmatrix} 4 & 1 & 1 \end{bmatrix}\begin{bmatrix} \mathbf{V}_i \\ \mathbf{E}_i \\ \mathbf{E}_{i+1} \end{bmatrix}$$

Examples

Consider a cubic uniform B-spline curve with control polygon $\{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3\}$, the curve can be written as

$$\mathbf{P}(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \frac{1}{6} \begin{bmatrix} 1 & 4 & 1 & 0 \\ -3 & 0 & 3 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix}$$

Consider the first step of the refinement using $\mathbf{P}_1$ as the vertex with $\mathbf{P}_0$ and $\mathbf{P}_2$ as the two respective edge points. The limit point on curve is

$$\frac{1}{6} \begin{bmatrix} 4 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_1 \\ \mathbf{P}_0 \\ \mathbf{P}_2 \end{bmatrix} = \frac{1}{6} (\mathbf{P}_0 + 4\mathbf{P}_1 + \mathbf{P}_2)$$

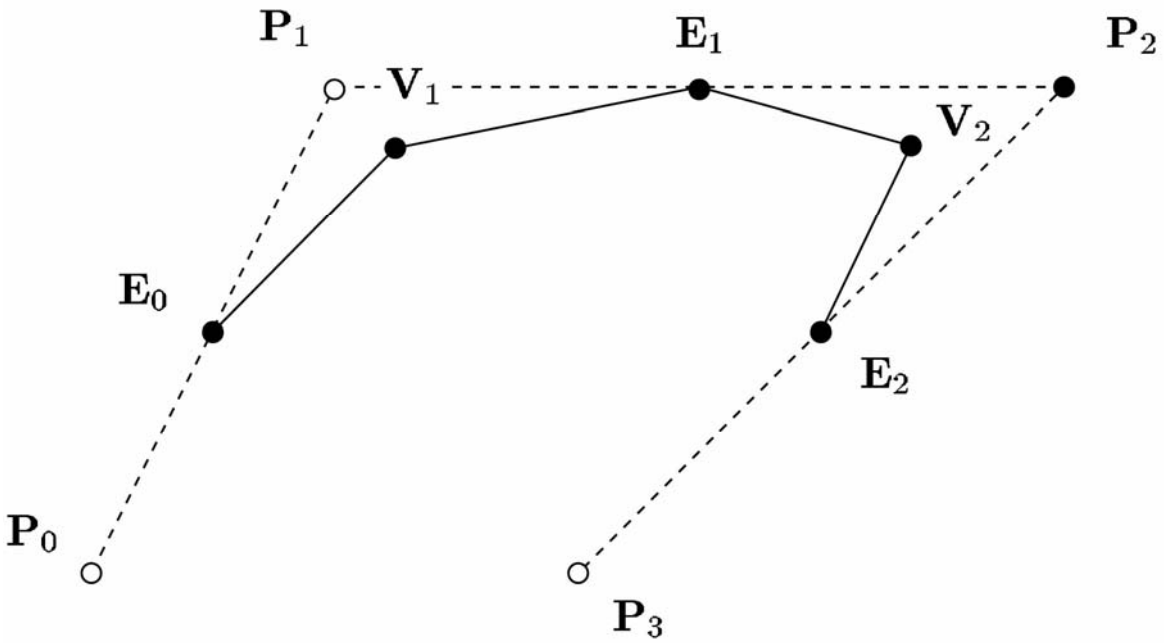
which is exactly the point $\mathbf{P}(0)$ on the curve

Examples

Consider a point on the curve using vertex $\mathbf{P}_2$ as the vertex with $\mathbf{P}_1$ and $\mathbf{P}_3$ as the two respective edge points. The limit point on curve is exactly the point $\mathbf{P}(1)$.

Examples

Perform one step refinement, generate new vertex and edge points



we consider $\mathbf{E}_1$ as the vertex point, with $\mathbf{V}_1$ and $\mathbf{V}_2$ as the respective edge points, we obtain the point on the curve

Examples

$$\begin{bmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \end{bmatrix} \begin{bmatrix} \mathbf{E}_1 \\ \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix} = \begin{bmatrix} \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \end{bmatrix} \begin{bmatrix} \frac{1}{2}(\mathbf{P}_1 + \mathbf{P}_2) \\ \frac{1}{8}(\mathbf{P}_0 + 6\mathbf{P}_1 + \mathbf{P}_2) \\ \frac{1}{8}(\mathbf{P}_1 + 6\mathbf{P}_2 + \mathbf{P}_3) \end{bmatrix}$$

$$= \frac{1}{48}(\mathbf{P}_0 + 23\mathbf{P}_1 + 23\mathbf{P}_2 + \mathbf{P}_3)$$

which is exactly $\mathbf{P}(1/2)$

Summary

- It is possible, using eigenanalysis, to formulate simple procedures that calculate directly a point on the limit curve. This, in many cases, can be used to replace the overall refinement process
- The tangent vector on the curve at this limit point can also be directly calculated, by much the same procedure.

Calculating the Tangent Vectors at a Point

- Overview
- Direct Calculation of Points on the Curve
- Tangent Vectors to the Curve
- Summary

Overview

- Based upon eigenanalysis of the refinement matrix, we can exactly calculate a point on the limit curve.
- By applying the eigenanalysis further we can also calculate directly the tangent vectors on the limit curve.
  - The eigenvector corresponding to the eigenvalue ½ that plays the primary role

Direct Calculation of Points on the Curve

Select a control point $\mathbf{V}_i$ and let $\mathbf{E}_i$ and $\mathbf{E}_{i+1}$ be the adjacent control points (thought of as edge points of the refinement). Given the refinement matrix

$$A = \frac{1}{8}\begin{bmatrix} 6 & 1 & 1 \\ 4 & 4 & 0 \\ 4 & 0 & 4 \end{bmatrix}$$

we can show that repeatedly applying A to the vector

$$\begin{bmatrix} \mathbf{V}_i \\ \mathbf{E}_i \\ \mathbf{E}_{i+1} \end{bmatrix}$$

Direct Calculation of Points on the Curve

Gives us a point $\mathbf{V}^{\infty}$ on the curve. This point can be directly calculated as the dot product of the left eigenvector of A that corresponds to the eigenvalue one, and the original vertex-edge vector of the refinement - obtaining the point

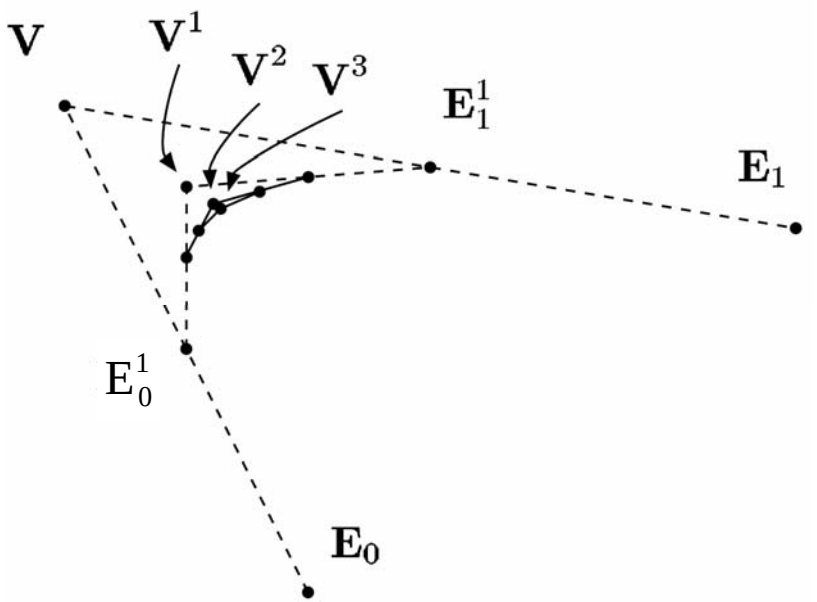
$$(4\mathbf{V} + \mathbf{E}_0 + \mathbf{E}_1)/6$$

Tangent Vectors to the Curve

Given a vertex point $\mathbf{V}$ and the two adjacent edge points $\mathbf{E}_0$ and $\mathbf{E}_1$ respectively

$$\lim_{k \leftarrow \infty} \frac{\mathbf{E}_0^k - \mathbf{V}^\infty}{||\mathbf{E}_0^k - \mathbf{V}^\infty||}$$

the limit above should converge to the unit tangent vector at the point $\mathbf{V}^\infty$



Tangent Vectors to the Curve

To calculate this quantity, we will use the diagonalization of the matrix A which states that

$$\begin{bmatrix} \mathbf{V}^k \\ \mathbf{E}_0^k \\ \mathbf{E}_1^k \end{bmatrix} = A^k \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix} = R\Lambda^k L \begin{bmatrix} \mathbf{V} \\ \mathbf{E}_0 \\ \mathbf{E}_1 \end{bmatrix}$$

Now, let $\vec{l}_1, \vec{l}_2$ and $\vec{l}_3$ be the left eigenvectors of the refinement matrix A, and $\vec{r}_1, \vec{r}_2$ and $\vec{r}_3$ be the row vectors of the matrix of right eigenvalues of the refinement matrix A. Then to calculate $\mathbf{E}_0^k$ we have

Tangent Vectors to the Curve

$$
\begin{aligned}
\mathbf{E}_0^k &= \vec{r}_2 \cdot \left[\Lambda^k L \vec{\mathbf{P}}\right] \\
&= \vec{r}_2 \cdot \left(\Lambda^k \begin{bmatrix} \vec{l}_1 \cdot \vec{\mathbf{P}} \\ \vec{l}_2 \cdot \vec{\mathbf{P}} \\ \vec{l}_3 \cdot \vec{\mathbf{P}} \end{bmatrix} \right) \\
&= \vec{r}_2 \cdot \begin{bmatrix} \vec{l}_1 \cdot \vec{\mathbf{P}} \\ \left(\frac{1}{2}\right)^k \vec{l}_2 \cdot \vec{\mathbf{P}} \\ \left(\frac{1}{4}\right)^k \vec{l}_3 \cdot \vec{\mathbf{P}} \end{bmatrix} \\
&= \vec{l}_1 \cdot \vec{\mathbf{P}} + \left(\frac{1}{2}\right)^k \vec{l}_2 \cdot \vec{\mathbf{P}} + 2\left(\frac{1}{4}\right)^k \vec{l}_3 \cdot \vec{\mathbf{P}}
\end{aligned}
$$

where we have used $\vec{r}_2 = (1,1,2)$, since $\mathbf{V}^\infty = \vec{l}_1 \cdot \vec{\mathbf{P}}$

Tangent Vectors to the Curve

We have

$$\begin{aligned}
\lim_{k\to\infty} \frac{\mathbf{E}_0^k - \mathbf{V}^\infty}{||\mathbf{E}_0^k - \mathbf{V}^\infty||} &= \lim_{k\to\infty} \left(\frac{\left(\frac{1}{2}\right)^k \vec{l}_2 \cdot \vec{\mathbf{P}} + 2\left(\frac{1}{4}\right)^k \vec{l}_3 \cdot \vec{\mathbf{P}}}{||\left(\frac{1}{2}\right)^k \vec{l}_2 \cdot \vec{\mathbf{P}} + 2\left(\frac{1}{4}\right)^k \vec{l}_3 \cdot \vec{\mathbf{P}}||} \right) \\
&= \lim_{k\to\infty} \left(\frac{\vec{l}_2 \cdot \vec{\mathbf{P}} + 2\left(\frac{1}{2}\right)^k \vec{l}_3 \cdot \vec{\mathbf{P}}}{||\vec{l}_2 \cdot \vec{\mathbf{P}} + 2\left(\frac{1}{2}\right)^k \vec{l}_3 \cdot \vec{\mathbf{P}}||} \right) \\
&= \frac{\vec{l}_2 \cdot \vec{\mathbf{P}}}{||\vec{l}_2 \cdot \vec{\mathbf{P}}||} \\
&= \frac{\mathbf{E}_0 - \mathbf{E}_1}{||\mathbf{E}_0 - \mathbf{E}_1||}
\end{aligned}$$

Tangent Vectors to the Curve

Therefore, given any vertex $\mathbf{V}$ with corresponding edge points $\mathbf{E}_0$ and $\mathbf{E}_1$, we can directly calculate the tangent vector at the limit point $\mathbf{V}^\infty$ dotting the left eigenvector that corresponds to the eigenvalue ½ by the vertex-edge vector

$$\frac{1}{2}\begin{bmatrix} 0 & 1 & -1 \end{bmatrix}\begin{bmatrix} \mathbf{V}_i \\ \mathbf{E}_i \\ \mathbf{E}_{i+1} \end{bmatrix}$$

i.e. by substracting $\mathbf{E}_{i+1}$-$\mathbf{E}_i$

Summary

It is possible, using eigenanalysis, to formulate simple procedures that calculate directly a point on the limit curve, and the tangent vector on the curve at this point. This makes this refinement procedure quite simple to use.