Introduction

- Curves are one of the most essential objects to create high-resolution graphics.
- While using many small polylines allows creating graphics that appear smooth at fixed resolutions, they do not preserve smoothness when scaled and also require a tremendous amount of storage for any high-resolution image.
- Curves can be stored much easier, can be scaled to any resolution without losing smoothness, and most importantly provide a much easier way to specify real-world objects.

Curve generation

There are two approaches to generate a curve

1. To develop **a curve generation algorithm such as DDA** with this approach **a true curve is created**
2. **Approximate curve by a number of small straight line** segments(interpolation technique is used for this second approach)

Now we develop a DDA curve generation algorithm

- Consider a curve, **such as arc of circle** that we wish to draw
- If we know the differential equations for this curve then we can **write a DDA algorithm which will calculate the coordinates of points on the curve**
- We **can then change a intensity values of the pixel which contain these points**
- And **image of the curve will appear on the screen**

Let us step through the development of DDA (Digital Differential Analyzers) algorithm for generating arc

The Eqs. for the arc coordinates can be written in terms of the angle parameter A as

$X = R*\cos A + Xo$ (eq.1)

$Y = R*\sin A + Yo$

Where Xo and Yo is the centre of the curvature

R is the radius

To perform differential change X by dx and Y by dy and A by dA

So we will get

$dx = - R*\sin A*dA$

$dy = R*\cos A*dA$ (eq.2)

From eq(1) we will get the values of RcosA and RsinA

$R\cos A = X - X0$

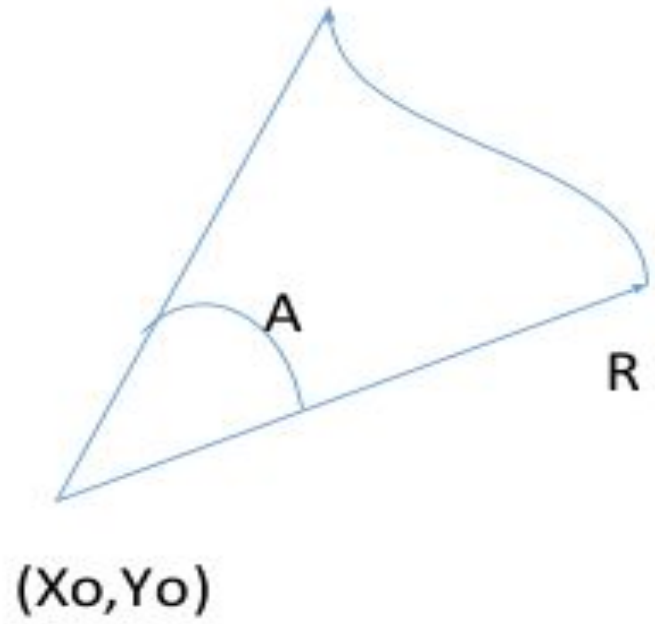
$R\sin A = Y - Y0$

If we put the values of RcosA and RsinA in eq 2 we will get

$$dx=-(Y-Yo)dA$$

$$dy=(X-Xo)dA$$

This dx and dy tell us the amount to add to an old point on the arc to get a new point



$X2=X1+dx$

$Y2=Y1+dy$

$X2=X1-(Y1-Yo)dA$

$Y2=Y1+(X1-Xo)dA$

These equations form the basis for an arc generation algorithm

- In this algorithm we start with the centre of the curvature(Xo,Yo)
- The total angle of the arc to be drawn A
- A point at which the arc drawing should begin(X,Y)
- And the intensity at which to set the pixel
- The step size of the parameter dA should be small enough not to leave gaps in the arc and small enough to give a good approximation t o a circle

INTERPOLATION

- We can draw an approximate curve if we have an array of sample points
- If curve is smooth and samples are close enough together, we can make a good guess curve
- Our guesses are not be exactly right, but it will be close enough for appearance
- We fill in portions of the unknown curve with piece of known curves which pass through the nearby sample points

Unknown curve

Known sample points

Fit a region with a known curve

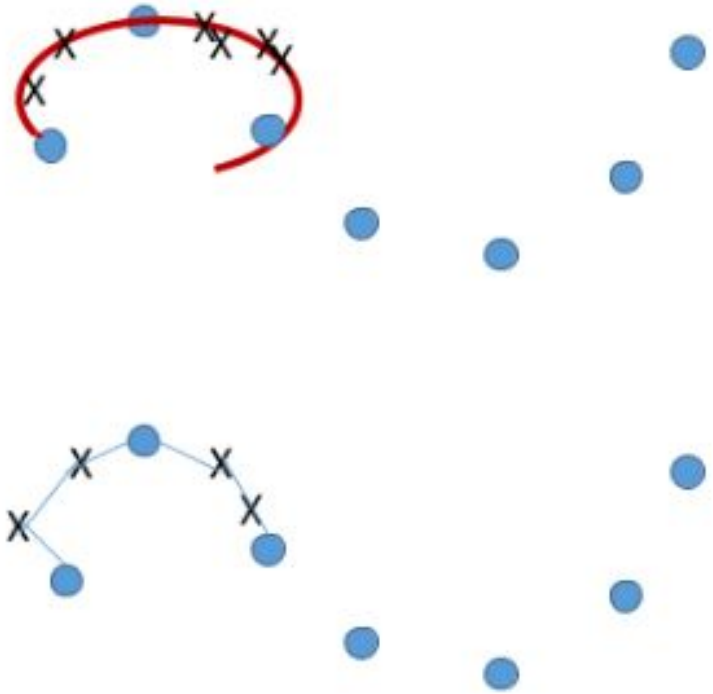


Fig:-interpolation process

- Since the unknown and unknown curve share these sample points in a local region
- Assume that the two curve in this region look pretty much alike
- We fit a portion of the unknown curve with a curve that we know
- Now we can fill in a gap between the sample points by finding the coordinates of points along the known approximating curve and connecting these points with the line segments

- All of the popular curves used in graphics are specified by parametric equations.
- Instead of specifying a function of the form y = f(x), the equations

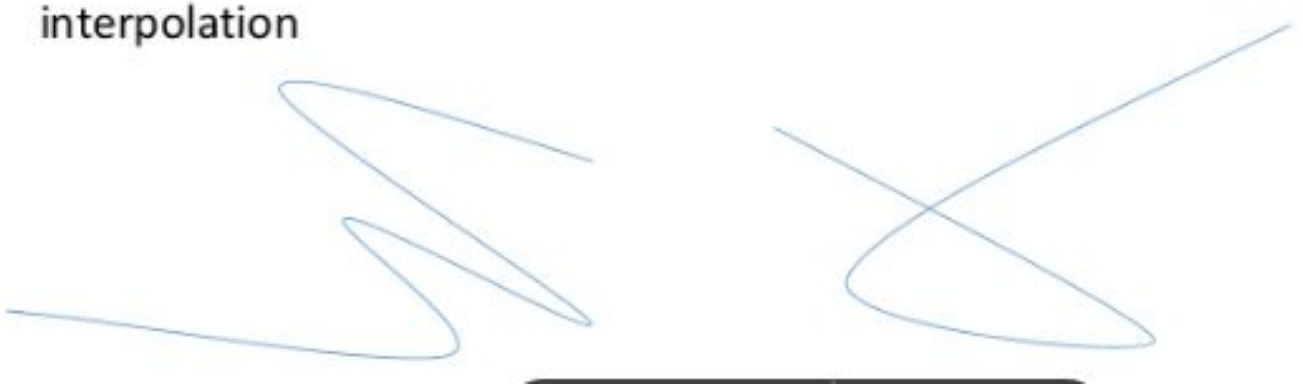
$$x = f_x(u)$$

$$y = f_y(u) \text{ and}$$

$$z = f_z(u)$$

Using parametric equations allows curves that can double back and cross themselves, which are impossible to specify in a single equation in the y = f(x) case.

- By using this eqs we can find out the functions which can be used for interpolation



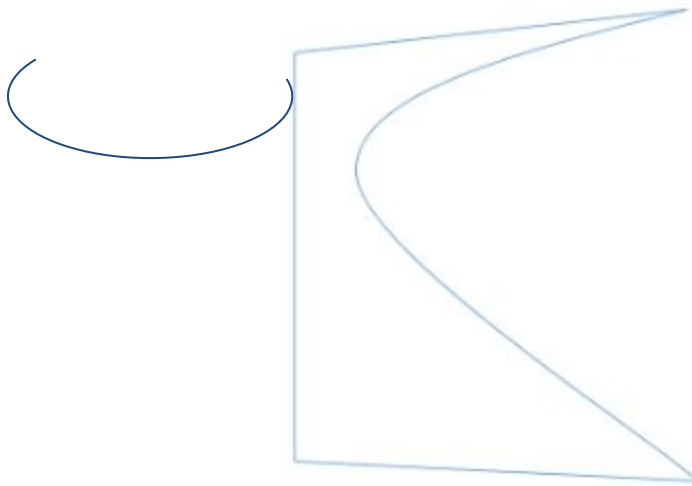
Bezier curve

- Cubic Bezier curve is adequate for most graphics application
- The cubic Bezier curve requires four control points
- These points completely specifies the curve
- Unlike b-splines curve, we cannot add additional point and smoothly extend the cubic bezier curve,but we pick four more points and construct a second curve which can be attached to the first
- The curve begins at the first control point and ends on the fourth.
- If we want to connect two bezier curve
- We just make the first control point of the second curve match the last control point of its first curve
- At the start of the curve ,it is tangent to the line connecting the first and second control points,likewise at he end of the curve ,it is tangent to line connecting third and fourth control points
- If we want to join the two bezier curve smoothly ,we must arrange for the third and fourth control points of the first curve to be on the same line as the first and second control points of the second curve

- Equations for Bezier curve are as follows
- $X = X4u^3 + X3u^2(1-u) + X2u(1-u)^2 + X1(1-u)^3$
- $Y = Y4u^3 + Y3u^2(1-u) + Y2u(1-u)^2 + Y1(1-u)^3$
- $Z = Z4u^3 + Z3u^2(1-u) + Z2u(1-u)^2 + Z1(1-u)^3$

- IN THIS EQS AS U GOES FROM 0 TO 1,THE CURVE GOES FORM FIRST TO FOURTH CONTROL POINT

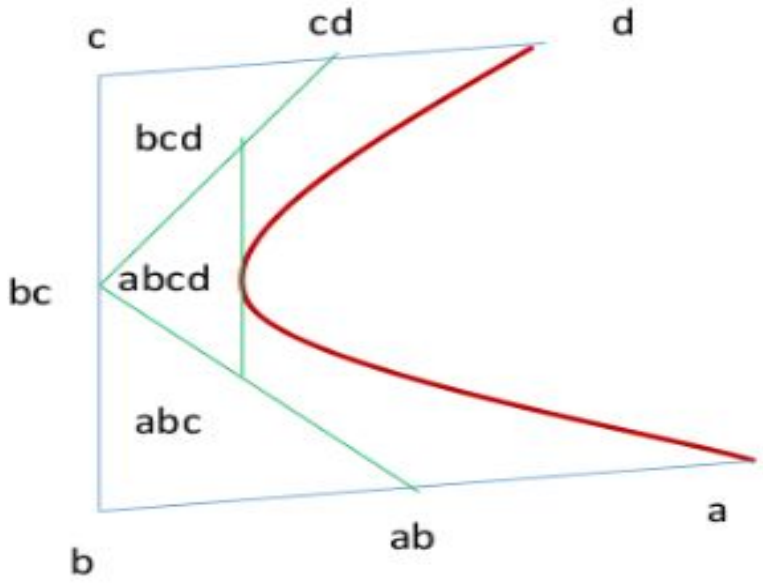
- Without using this Esq. we can also construct the Bezier curve
- A curve can be constructed simply by taking the midpoints
- Ex we have drawn the lines connecting four control points a,b,c,d
- We have determine the midpoints of these line segment ab,bc,cd
- We connect these line segment and find their midpoints abc and bcd
- Finally we connect these two points and find the midpoint abcd
- Now the point abcd is on the bezier curve



A cubic Bezier spline

The points a,ab,abc,abcd are the control points of the first section
And the points abcd,bcd,cd and d are the control points for the second section
With the midpoint we can find a point on the curve and also split the curve into two pieces complete with control point

- Bezier curve satisfy the convex hull property
- This means that the curve lies within the boundary formed by an elastic band stretched over all four control points
- This property is useful in clipping tests



B Splines

- We have seen a Langrange Interpolation program using blending functions.
- Let's examine some of its inadequacies.
- One thing to notice is that the sum of the blending functions is not 1 at every value of u.
- The blending functions were designed to sum to 1 at integer values of u , but not at fractional values.
- What does this mean?
- Suppose that all sample points had the same x value,

  X = X0.so the approximating curve is,

$$X = \sum_{i=1}^{n} Xi\ Bi(u)$$

which for Xi = X0,

$$X = X0 \sum_{i=1}^{n} Bi(u)$$

- We will get a flat behavior .
- Here, each section of the curve is connected to the next section at a sample point, but slopes of the two sections need not match at this point.
- This means that there can be corners at the sample points and that we may not have a completely smooth curve.
- Finally, pulling the curve all the way over to pass through one of the sample points, we had to reduce the effect of all the other sample points to zero.
- The control of the curve by a sample point pulsing in and out as we move along in u.

- A more natural behavior might be to have a sample point's control vary smoothly from zero far away from the point to a maximum near the point.

- We can do this , if we don't try to force the curve through the sample point, but rather gently pull it into the neighborhood of the sample point.

- This result will be a curve which follows the general contours indicated by the sample points but may not actually pass through the points themselves.

- A set of blending function which take this approach and also always sum to 1 are called "B Splines".

- They generate curve sections which have continuous slopes so that they fit together smoothly.

Fractals

- The objects occuring in nature often have rough ,jagged,random edges
- Attempting to draw things like mountains,trees,rivetrs or lightning bolt directly with lines or polygons requires a lot of specification
- It is desirable to let the machine do the work of specifying the jagged lines
- We would like to give the machine two endpoints and let the machine draw a jagged line between them
- Not only the rough line but it will closely approximate the behavior of nature
- Recently the branch of mathematics has been developed algorithm to describes such rough structure.they are called fractals
- A rough or fragmented geometric shape that can be subdivided in parts, each of which is (at least approximately) a reduced/size copy of the whole.

- Fractals are very complex pictures generated by a computer from a single formula.
- They are created using iterations.
- This means one formula is repeated with slightly different values over and over again, taking into account the results from the previous iteration.

Generation of Fractals

- Fractals can be generated by repeating the same shape over and over again as shown in the following figure. In figure a shows an equilateral triangle. In figure b we can see that the triangle is repeated to create a star-like shape. In figure c we can see that the star shape in figure b is repeated again and again to create a new shape.
- We can do unlimited number of iteration to create a desired shape. In programming terms, recursion is used to create such shapes.

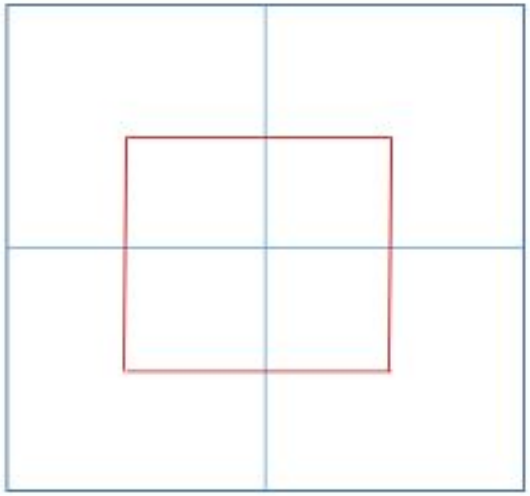
(a) Zeroth Generation (b) First Generation (c) Second Generation

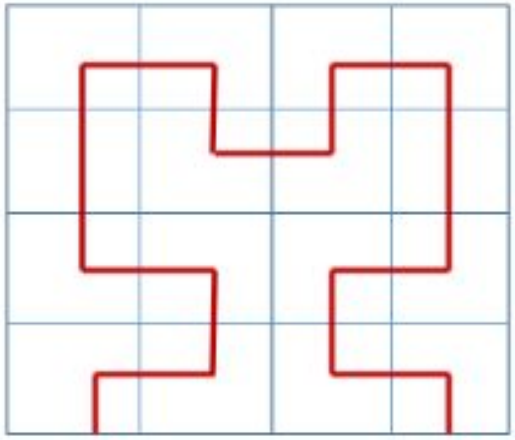
- Lets consider the curve called the **triadic koch curve**
- Begin with line segment
- Divide it into thirds and replace the centre third by two adjacent sides of the equilateral triangle
- This gives us a curve which begins and ends at the same place as the original segment but is built of 4 equal length segments each 1/3 the original length

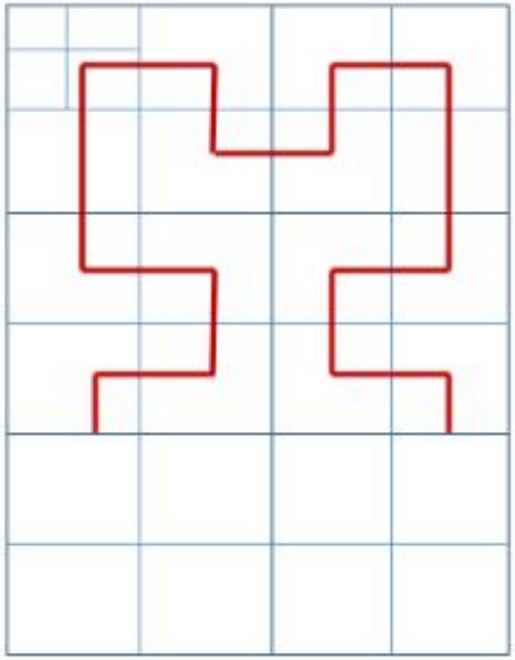
Replacement for a line segment for the triadic koch curve

- So the new curve has 4/3 the length of the original segment
- Repeat the step for each of the four segment
- The curve has gained more wiggles and its length is now 16/9 times the original
- Imagine repeating the replacements indefinitely
- Since each repetition increases the length by a factor of 4/3
- The length of the curve will be finite, but it is folded in lots of tiny wiggles

- Lets consider a rather strange construction call a **Peano curve or space filling curve**
- The particular Peano curve is called **Hilbert's curve**
- This curve can be built by the following successive approximation
- Begin with square
- First approximation will be divide square into 4 quadrants and draw the curve which connects the centre points of each
- The second approximation will be further subdivide each of the quadrants and connect the centre of each of these finer divisions before moving to the next major quadrant
- The third approximation again subdivides it again connects the centre of the finest level before moving to the next level of detail







- In this process the curve never crosses itself
- At each subdivision, the curve fills with the smaller quadrants ,but never crosses into an area where it already exists
- The curve is arbitrarily closed to every point in the square
- The curve passes through the points on a grid, which becomes twice as fine with each subdivision and there is no limit to the subdivision
- The length of the curve is infinite
- Withy each subdivision length increases by a factor of 4
- Since we imagine no limit to the subdivision
- So no limit to the length

- So we construct a curve which is topologically equivalent to the line Dt=1
- But the curve has been folded or twisted so we can find the fractal dimension of the curve

- At each subdivision the scale changes by 2,but the length changes by 4
- For square it takes 4 curves of half scale to build the full sized object
- So the dimension D is given by
- $4=2^D$
- must be D=2
- The Hilbert curve has topological dimension 1 but the fractal dimension 2
- It is the line so folded that it behaves like a two dimensional object