Image Completion Using Interpolation Techniques

A Thesis submitted by

Aditya Mishra
M20MA201

in partial fulfillment of the requirements for the award of the degree of
Master of Science in Mathematics

॥ त्वं ज्ञानमयो विज्ञानमयोऽसि ॥

Indian Institute of Technology Jodhpur

May 2022

Abstract

The aim of this thesis is to present the methods applied in an image in order to complete the image by using interpolation techniques.The completion of Image presented in this thesis are of two types , first is , if we have done the Up scaling Operation to an image with having a desire to resizing an image. Since this process is not so much easy to handle we have to take help of the One Dimensional interpolation techniques to understand the way to approach and find the the desirable curve or continuous function to recover the function points which had disappeared during resampling ,these whole processes we will learn into the First chapter. After that we will study about some basic concepts of 2-D interpolation in order to learn the image interpolation in 2nd chapter.In this chapter we will learn about the Frequency domain ,Transformation Kernel and Two dimensional filtering. After having the basic ideas of Prerequisite of Two Dimensional Interpolation , we will jump into the Two dimensional Interpolation where we will see the different kinds of interpolation techniques and their different kinds of interpolating kernels , such as Nearest Neighbor Interpolation , Bilinear Interpolation , Bicubic Interpolation. we will also implement these Techniques in MATLAB in order to complete image Pixels first in the upscaled image and Secondly we will implement these technique in order to recover the distorted image . we will compare the results of different Techniques visually.

Declaration

I hereby declare that the work presented in this thesis, submitted to Indian Institute of Technology, Jodhpur in partial fulfillment of the requirements for the award of degree of Master in Science, is a bonafide record of the research work carried out under the supervision of Associate Professor Dr Gaurav Bhatnagar of Indian Institute of Technology, Jodhpur. I confirm that this Master's thesis is my own work and I have documented all sources and materials used. This thesis was not previously presented to another examination board and has not been published.

Aditya Mishra
M20MA201
M.Sc.-M.Tech. Student

Certificate

This is to certify that the thesis entitled "Image Completion Using Interpolation Techniques" being submitted by Aditya Mishra to the Indian Institute of Technology Jodhpur, India, for the partial fulfilment of the award of degree of Master of Science in Mathematics is a record of bonafide research work carried out by him under my supervision during the last one year. This thesis is mostly a brief study of existing results available in the literature. To the best of my knowledge, this thesis have not been submitted for the award of any other degree/diploma anywhere. In my opinion, the thesis fulfils the requirement for the award of the degree.

Dr. Gaurav Bhatnagar
Department of Mathematics
Indian Institute of Technology Jodhpur
Karwar 342037, India

Acknowledgements

I thank my supervisor Dr. Gaurav Bhatnagar for allowing and helping me to explore the vast area of research in Digital Image processing, especially image Completion. In this process, I have learned many technical and non-technical aspects of professional research. I am grateful to all involved in this thesis for their help, patience, time and the constant reminder to be perfect in little things I do each day.

Aditya Mishra
M20MA201
M.Sc.-M.Tech Student

Contents

1
Introduction to Image Inpainting

Image inpainting is the process of restoring the images or the damaged area in the images in such a manner that the restoration or the change is undetectable. A long time ago this process was done by the professional artists. now this process is extended it's limits and reached to the images , photographs and the videos also.Application of Inpainting are very vast and spread to the field as restoration of an image from scratches , text and unwanted objects.Image inpainting is the process of restoring the images or the damaged area in the images in such a manner that the restoration or the change is undetectable.

There are several methods to deal with inpainting problems using the concept of Deep Learning and Machine Learning but Under this thesis we will introduce the method of interpolation techniques to deal the inpainting problems.

Fig. 1 Scratch removal and text removal examples of image inpainting

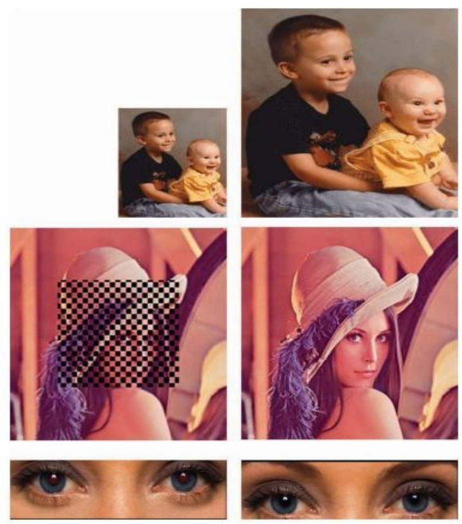


Fig. 2 Digital image zoom-in (top), image compression (middle), and red eye removal (bottom) examples of image inpainting

Image inpainting for object removal

2
Interpolation Techniques in One Dimension

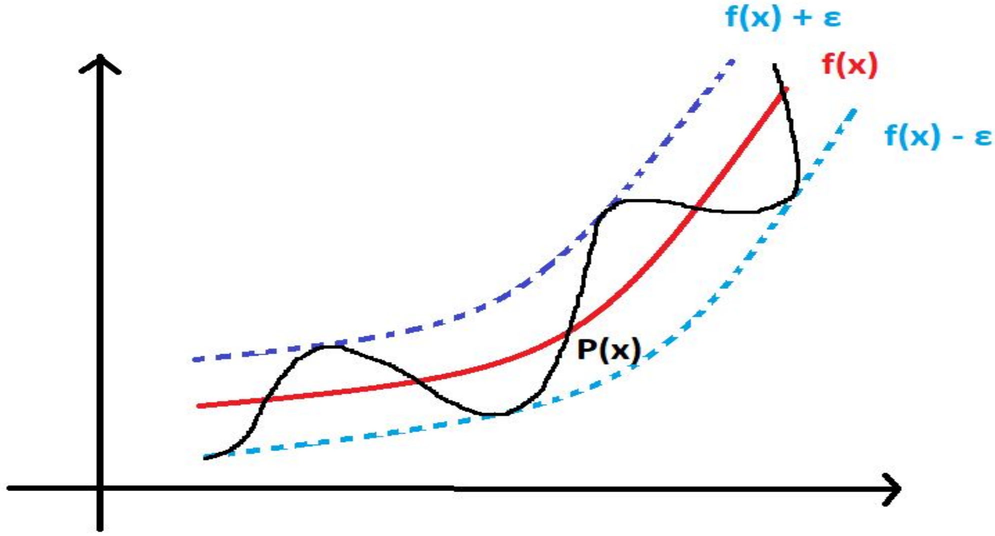
In the practical scenario , On the basis of data set whenever we have to find the value at some intermediate point of the given range , we use the interpolation Techniques. These interpolation Techniques may calculate the approximate function satisfying the data set Or can directly find the value at some intermediate point of the given range.

2.1 Mathematical definition of Interpolation :

Definition 2.1.1. *Let $v = f(u), u_0 \leqslant u \leqslant u_\eta$, let us consider that continuous function $f(u)$ is single valued , then the output of $f(u)$ by the certain given values of u, say $u_0, u_1, \ldots, u_n$ can certain given values of u, say $u_0, u_1, \ldots, u_n$ can easily be computed and tabulated.*

*but, now set of given data points $(u_0, v_0), (u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)$ reassure the relation $v = f(u)$, here the equation of $f(u)$ is not studied . now the task is to investigate a simpler function, say $\psi(u)$, such that $f(u)$ and $\psi(u)$ satisfy the set of tabulated points. so this whole process is known as **Interpolation**.*

Now here as we are using the polynomial function as a result of interpolation , so for the clarification for the estimation of not known function as of polynomial, and we know that , if $f(u)$ is continuous in $u_0 \leqslant u < u_n$, then by any $\delta > 0$, there exist a polynomial $P(u)$ such that $|f(u) - P(u)| < \delta \forall u \in (u_0, u_n)$



2.2 Preliminaries

Before coming to the main Problem of interpolation we have to understand the type of problem, so there are two kind of interpolation problem ,one is Equally spaced Problem and Second is Unequally spaced problem.

2.3 Finite Differences

Let us consider a table of values $(u_i, v_i), i = 1, 2, 3...., n$ of a fn $v = f(u)$ the next data point of u are equally spaced, as $u_i = u_o + ih, i = 1, 2, 3...., n$ where h is step size. In order to recover the value of $f(u)$ for some value of u lying inside the given range $x_0 \leqslant x \leqslant x_n$, now let us learn about some the popular based on the difference method of equal step size.

2.3.1 Forward difference

Let us consider the values $v_o, v_1, v_2, ..., v_n$ are function values of the points $u_o, u_1, u_2, ..., u_n$ respectively , then $v_1 - v_0, v_2 - v_1, ..., v_n - v_{n-1}$ is denoted by first forward differences of v .denoted by $\Delta v_o, \Delta v_1, ..., \Delta v_{n-1}$. where Δ is called the Forward difference Operator and $\Delta v_o, \Delta v_1, ..., \Delta v_{n-1}$ are called the first forward difference.In the same way second forward differences denoted by $\Delta^2 v_o, \Delta^2 v_1, ..., \Delta^2 v_{n-1}$. Similarly third forward difference and fourth forward difference. The fallowing table shows the forward difference of all order can be formed.

u	v	Δ	Δ^2	Δ^3	Δ^4	Δ^5	Δ^6
u_0	v_0						
		Δv_0					
u_1	v_1		$\Delta^2 v_0$				
		Δv_1		$\Delta^3 v_0$			
u_2	v_2		$\Delta^2 v_1$		$\Delta^4 v_0$		
		Δv_2		$\Delta^3 v_1$		$\Delta^5 v_0$	
u_3	v_3		$\Delta^2 v_2$		$\Delta^4 v_1$		$\Delta^6 v_0$
		Δv_3		$\Delta^3 v_2$		$\Delta^5 v_1$	
u_4	v_4		$\Delta^2 v_3$		$\Delta^4 v_2$		
		Δv_4		$\Delta^3 v_3$			
u_5	v_5		$\Delta^2 v_4$				
		Δv_5					
u_6	v_6						

2.3.2 Backward difference

Let us consider the values $v_o, v_1, v_2, ..., v_n$ are function values of the points $u_o, u_1, u_2, ..., u_n$ respectively , then $v_1 - v_0, v_2 - v_1, ..., v_n - v_{n-1}$ are called the differences of v .denoted by $\nabla v_1, \nabla v_2, ..., \nabla v_n$. where ∇ is called the Backward difference Operator and $\nabla v_o, \nabla v_1, ..., \nabla v_n$ are called the first Backward difference. Similarly the second forward differences are denoted by $\nabla^2 v_o, \nabla^2 v_1, ..., \nabla^2 v_n$. Similarly third Backward difference and fourth Backward difference. The fallowing table shows the Backward difference of all order can be formed.

u	v	∇	∇^2	∇^3	∇^4	∇^5	∇^6
u_0	v_0						
u_1	v_1	∇v_1					
u_2	v_2	∇v_2	$\nabla^2 v_2$				
u_3	v_3	∇v_3	$\nabla^2 v_3$	$\nabla^3 v_3$			
u_4	v_4	∇v_4	$\nabla^2 v_4$	$\nabla^3 v_4$	$\nabla^4 v_4$		
u_5	v_5	Δv_5	$\nabla^2 v_5$	$\nabla^3 v_5$	$\nabla^4 v_5$	$\nabla^5 v_5$	
u_6	v_6	∇v_6	$\nabla^2 v_6$	$\nabla^3 v_6$	$\nabla^4 v_6$	$\nabla^5 v_6$	$\nabla^6 v_6$

2.3.3 Central difference

Let us consider the values $v_o, v_1, v_2, ..., v_n$ are function values of the points $u_o, u_1, u_2, ..., u_n$ respectively , then $v_1 - v_0, v_2 - v_1, ..., v_n - v_{n-1}$ are called the differences of v .denoted by $\delta v_{1/2}, \delta v_{3/2}, ..., \delta v_{(n-1)/2}$. where δ is called the Central difference Operator and $\delta v_{1/2}, \delta v_{3/2}, ..., \delta v_{(n-1)/2}$ are called the first Central difference. Similarly the differences of first central difference are called second central differences and denoted by $\delta^2 v_1, \delta^2 v_2, ..., \delta^2 v_{n-1}$. Similarly third forward difference and fourth forward difference. The fallowing table shows the forward difference of all order can be formed.

U	v	δ	δ^2	δ^3	δ^4	δ^5	δ^6
u_0	v_0						
		$\delta v_{1/2}$					
u_1	v_1		$\delta^2 v_1$				
		$\delta v_{3/2}$		$\delta^3 v_{3/2}$			
u_2	v_2		$\delta^2 v_2$		$\delta^4 v_2$		
		$\delta v_{5/2}$		$\delta^3 v_{5/2}$		$\delta^5 v_{5/2}$	
u_3	v_3		$\delta^2 v_3$		$\delta^4 v_3$		$\delta^6 v_3$
		$\delta v_{7/2}$		$\delta^3 v_{7/2}$		$\delta^5 v_{7/2}$	
u_4	v_4		$\delta^2 v_4$		$\delta^4 v_4$		
		$\delta v_{9/2}$		$\delta^3 v_{9/2}$			
u_5	v_5		$\delta^2 v_5$				
		$\delta v_{11/2}$					
u_6	v_6						

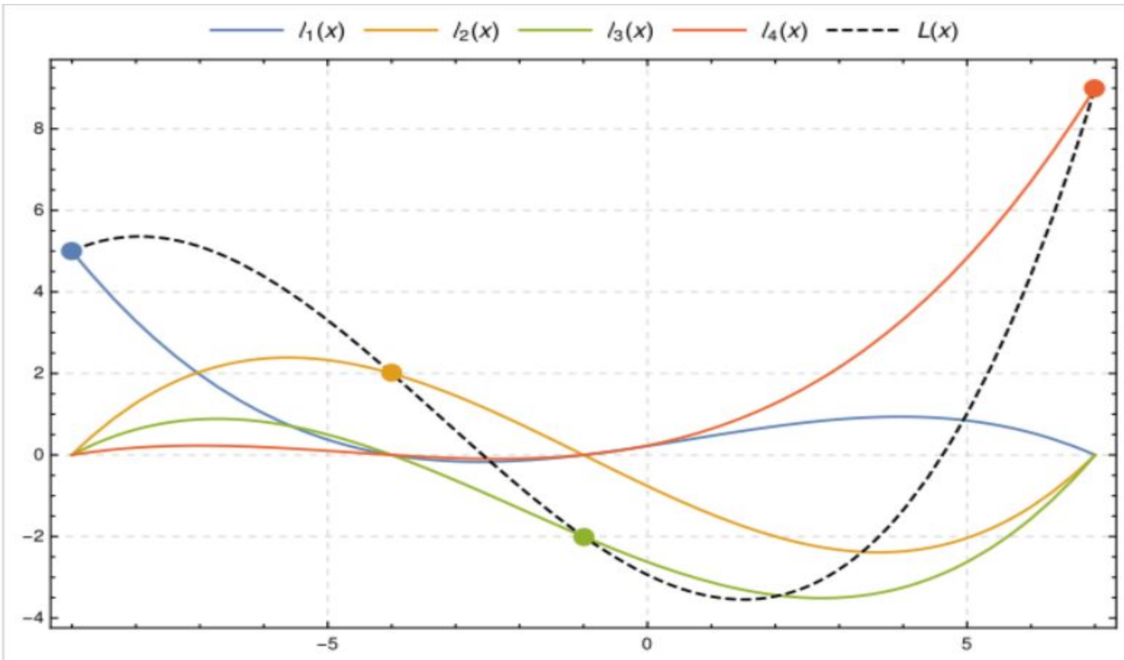
2.4 Interpolation Techniques for different step sized or Un-evenly scattered points

In the Above section I have learned about some of the interpolation method to recover the function where the sample points were equally spaced , but in real life scenario we will face the interpolation problem , where we will get the sample points , which are not equally spaced or for them step size is different for each consecutive pair. In such kind of Problems we can not use the above methods, therefore it is desirable to have the Interpolation formula for Unevenly spaced points. For such cases there are some following methods by which we can recover the function easily.

2.4.1 Lagrange's Interpolation

Let us assume $v(u)$ be acontinuous and differentiable function for $(m+1)$ times in the interval (p, q). Let us consider the $(m+1)$ points $(u_0, v_0), (u_1, v_1), \ldots, (u_m, v_m)$ where the points u's need not necessarily be equally spaced, now it is desirable to approximate a polynomial of degree n, say $\phi_m(x)$, such that

$$\phi_m(u_i) = v(u_i) = v_i, \quad i = 0, 1, \ldots, m \qquad ....(1)$$



This image shows, for four points ((−9, 5), (−4, 2), (−1, −2), (7, 9)), the (cubic) interpolation polynomial $L(x)$ (dashed, black), which is the sum of the *scaled* basis polynomials $y_0\ell_0(x)$, $y_1\ell_1(x)$, $y_2\ell_2(x)$ and $y_3\ell_3(x)$. The interpolation polynomial passes through all four control points, and each *scaled* basis polynomial passes through its respective control point and is 0 where x corresponds to the other three control points.

In front of going into the the general case ,Let's first of all work on simpler case viz., a degree one polynomial flowing through two points (u_0, v_0) and (u_1, v_1). Such a function , say $\phi_1(u)$, is easily derived to be

$$\begin{aligned}\phi_1(u) &= \frac{u-u_1}{u_0-u_1}v_0 + \frac{u-u_0}{u_1-u_0}v_1 \\ &= \alpha_0(u)v_0 + \alpha_1(u)v_1 \\ &= \sum_{i=0}^{1} \alpha_i(u)v_i \qquad ....(2)\end{aligned}$$

where

$$\alpha_0(u) = \frac{u-u_1}{u_0-u_1} \text{ and } \alpha_1(u) = \frac{u-u_0}{u_1-u_0} \qquad ....(3)$$

From (3), it is seen that

$$\alpha_0(u_0) = 1, \quad \alpha_0(u_1) = 0, \quad \alpha_1(x_0) = 0, \quad \alpha_1(x_1) = 1.$$

These relations can be expressed in a more convenient form as

$$\alpha_i(u_j) = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases} \qquad ....(4)$$

The $\alpha_i(x)$ in (2) also have the property

$$\sum_{i=0}^{1} \alpha_i(u) = l_0(u) + \alpha_1(u) = \frac{u-u_1}{u_0-u_1} + \frac{u-u_0}{u_1-u_0} = 1 \qquad \text{.........(5)}$$

The above Equation of degree one is the Lagrange polynomial flowing through two points $({}_0, v_0)$ and (u_1, v_1). In a same familiar way, the degree two Lagrange polynomial flowing through three points $(u_0, v_0), (u_1, v_1)$ and (u_2, v_2) is written as

$$\begin{aligned}\phi_2(u) &= \sum_{i=0}^{2} \alpha_i(u) v_i \\ &= \frac{(u-u_1)(u-u_2)}{(u_0-u_1)(u_0-u_2)} v_0 + \frac{(u-u_0)(u-u_2)}{(u_1-u_0)(u_1-u_2)} v_1 + \frac{(u-u_0)(u-u_1)}{(u_2-u_0)(u_2-u_1)} v_2\end{aligned} \qquad \text{.........(6)}$$

where the $\alpha_i(u)$ satisfy the conditions given in (4) and (5)

Similarly, for general case,

$$\phi_n(u) = \sum_{i=0}^{n} \alpha_i(u) v_i$$

where $\alpha_i(u)$ degree n polynomial in a variable u. Since $\phi_n(u_j) = v_j$ for $j = 0, 1, 2, \ldots, n$, and

$$\left.\begin{array}{ll} \alpha_i(u_j) = 0 & \text{if } i \neq j \\ \alpha_j(u_j) = 1 & \text{for all } j \end{array}\right\},$$

which are the same as (5). Hence $\alpha_i(u)$ may be written as

$$\alpha_i(u) = \frac{(u-u_0)(u-u_1)\ldots(u-u_{i-1})(u-u_{i+1})\ldots(u-u_n)}{(u_i-u_0)(u_i-u_1)\ldots(u_i-u_{i-1})(u_i-u_{i+1})\ldots(u_i-u_n)} \qquad \text{.........(7)}$$

If we now set

$$W_{n+1}(u) = (u-u_0)(u-u_1)\ldots(u-u_{i-1})(u-u_i)(u-u_{i+1})\ldots(u-u_n),$$

then

$$\begin{aligned}W'_{n+1}(u_i) &= \frac{d}{du}\left[W_{n+1}(u)\right]_{u=u_i} \\ &= (u_i-u_0)(u_i-u_1)\ldots(u_i-u_{i-1})(u_i-u_{i+1})\ldots(u_i-u_n)\end{aligned}$$

so that (7) becomes

$$\alpha_i(u) = \frac{W_{n+1}(u)}{(u-u_i)\, W'_{n+1}(u_i)}$$

Hence we get,

$$\phi_n(u) = \sum_{i=0}^{n} \frac{\pi_{n+1}(u)}{(u-u_i)\, W'_{n+1}(u_i)} v_i,$$

which is called Lagrange's interpolation formula.

2.4.2 Hermite Curve

In the Lagrange's Interpolation Methods , we so far used the certain number of function values , now In Hermite Interpolation Method we will be using the function value as well as the first derivative value of the the function at every given point.Such kind of Interpolation are known as Hermite Interpolation.

Let's us consider the set of points (u_i, v_i, v_i') , $i = 0, 1, \ldots, m$, now we have to find a approximated function of least degree polynomial , say $\psi_{2n+1}(u)$, such that

$$\psi_{2m+1}(u_i) = v_i \quad \text{and} \quad \psi'_{2m+1}(u_i) = v_i'; \quad i = 0, 1, \ldots, m, \qquad ......(1)$$

where the dashes are the symbol of differentiation , which is done with respect to u. The Hermite's interpolation polynomial is denoted by $\psi_{2m+1}(u)$. Since there are $(2m + 2)$ conditions and therefore we have to find the $(2m + 2)$ number of coefficients.Hence as the Lagrange's interpolation The representation of Hermite Polynomial is,

$$\psi_{2m+1}(u) = \sum_{i=0}^{m} r_i(u)v_i + \sum_{i=0}^{m} s_i(u)v_i' \qquad ......(2)$$

where $r_i(u)$ and $s_i(u)$ are polynomials of degree $(2m+1)$ in the variable u . Using conditions (1), we obtain

$$\left.\begin{array}{ll} r_i(u_j) = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases} & s_i(u) = 0, \text{ for all } i \\ r_i'(u) = 0, \text{ for all } i, & s_i'(u_j) = \left\{ \begin{array}{ll} 1, \text{ if } i = j & \\ 0, & \text{if } i \neq j. \end{array} \right\} \end{array}\right. \qquad ......(3)$$

Since $r_i(u)$ and $s_i(u)$ are polynomials in u of degree $(2m + 1)$,here $r_i(u) = A_i(u)[\alpha_i(u)]^2$ and $r_i(u) = B_i(u)[\alpha_i(u)]^2$, Now It's easy to observe that $A_i(u)$ and $B_i(u)$ are both linear functions in u. Hence we write

$$r_i(u) = (a_i u + b_i)[\alpha_i(u)]^2 \text{ and } s_i(u) = (c_i u + d_i)[\alpha_i(u)]^2 \qquad ......(4)$$

Using conditions (3) in (4), we obtain

$$\left.\begin{array}{r} a_i u_i + b_i = 1 \\ c_i u_i + d_i = 0 \end{array}\right\}$$

and

$$\left.\begin{array}{r} a_i + 2l_i'(u_i) = 0 \\ c_i = 1. \end{array}\right\}$$

From Eqs. (3.54), we deduce

$$\left.\begin{array}{ll} a_i = -2l_i'(u_i), & b_i = 1 + 2u_i\alpha_i'(u_i) \\ c_i = 1, & d_i = -u_i \end{array}\right\}$$

Hence Eqs. (3.53) become

$$\begin{aligned} r_i(u) &= \left[-2ul_i'(u_i) + 1 + 2u_i\alpha_i'(u_i)\right][\alpha_i(u)]^2 \\ &= \left[1 - 2(u-u_i)\alpha_i'(u_i)\right][l\alpha_i(u)]^2 \end{aligned}$$

and

$$s_i(u) = (u-u_i)[\alpha_i(u)]^2.$$

Using the above expressions for $r_i(u)$ and $s_i(u)$ in (2), we obtain finally

$$\psi_{2m+1}(u) = \sum_{i=0}^{m}\left[1 - 2(u-u_i)\alpha_i'(u_i)\right][\alpha_i(u)]^2 v_i + \sum_{i=0}^{m}(u-u_i)[\alpha_i(u)]^2 v_i'$$

this is required Hermite interpolation formula.

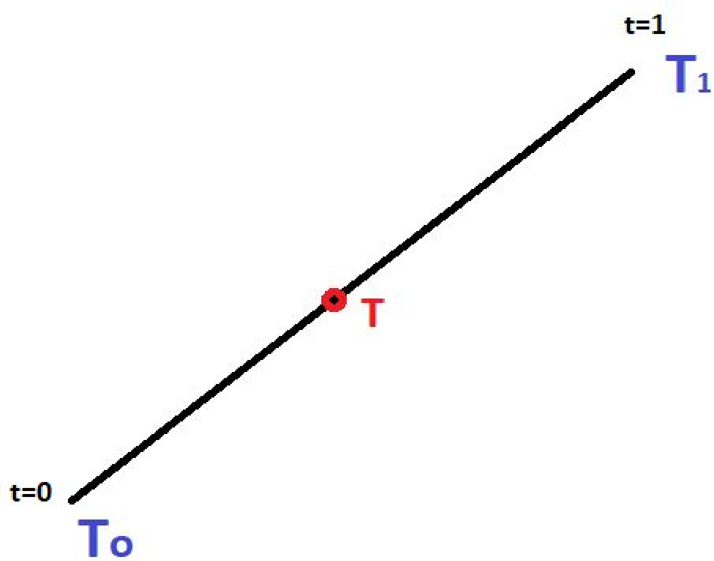
2.4.3 Bezier Curves

Bezier curves are represented as parametric equations. A parameter, p, is used to calculate the value of the variables, eg.,

$$\begin{aligned} u(p) &= (1-p)u_0 + pu_1 \\ (p) &= (1-p)v_0 + pv_1 \end{aligned}$$

where $0 \leqslant p \leqslant 1$. Let $T_0 = (u_0, v_0), T_1 = (u_1, v_1)$ and $T = (u, v)$ then

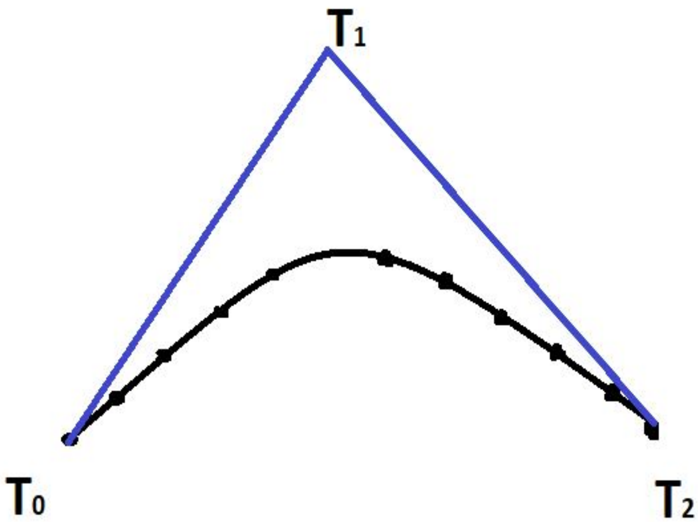
$$T(p) = (1-p)T_0 + pT_1.$$



A Bezier Curve is a parametric curve that is used to draw smooth lines. Named after **'Pierre Bezier'** who used them for designing cars at Renaults; actually - invented by Paul de Casteljau 3 year earlier while working for Citrorn. Common application includes CAD Software, 3D modelling and type faces. Translation can be easily applied to the control points.

Derivation of a Quadratic Bezier Curve

Q_0 and Q_1 lie on the lines $T_0 \longrightarrow T_1$ and $T_1 \longrightarrow T_2$. The point on the Bezier curve lies on the line $Q_0 \rightarrow Q_1$



Q_0 and Q_1 are points on the lines $T_0 \rightarrow T_1$ and $T_1 \rightarrow T_2$

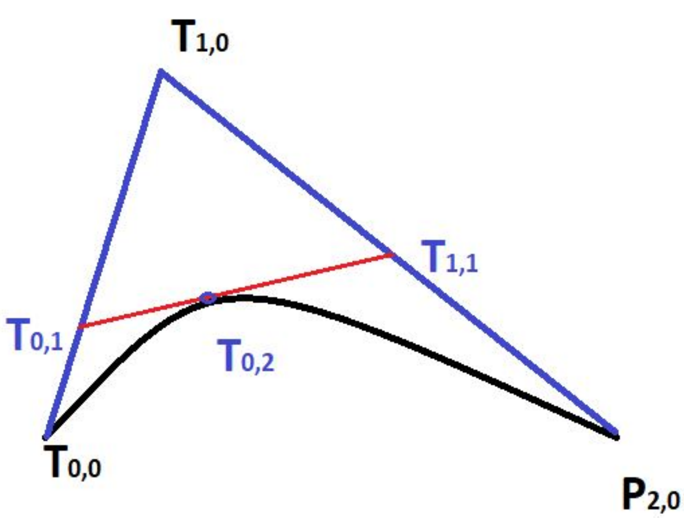
$$Q_0 = (1-p)T_0 + pT_1$$
$$Q_1 = (1-p)T_1 + pT_2$$

There is a point $C(t)$ on the Bezier curve on the line $\theta_0 \rightarrow Q_1$

$$C(t) = (1-p)Q_0 + pQ_1$$

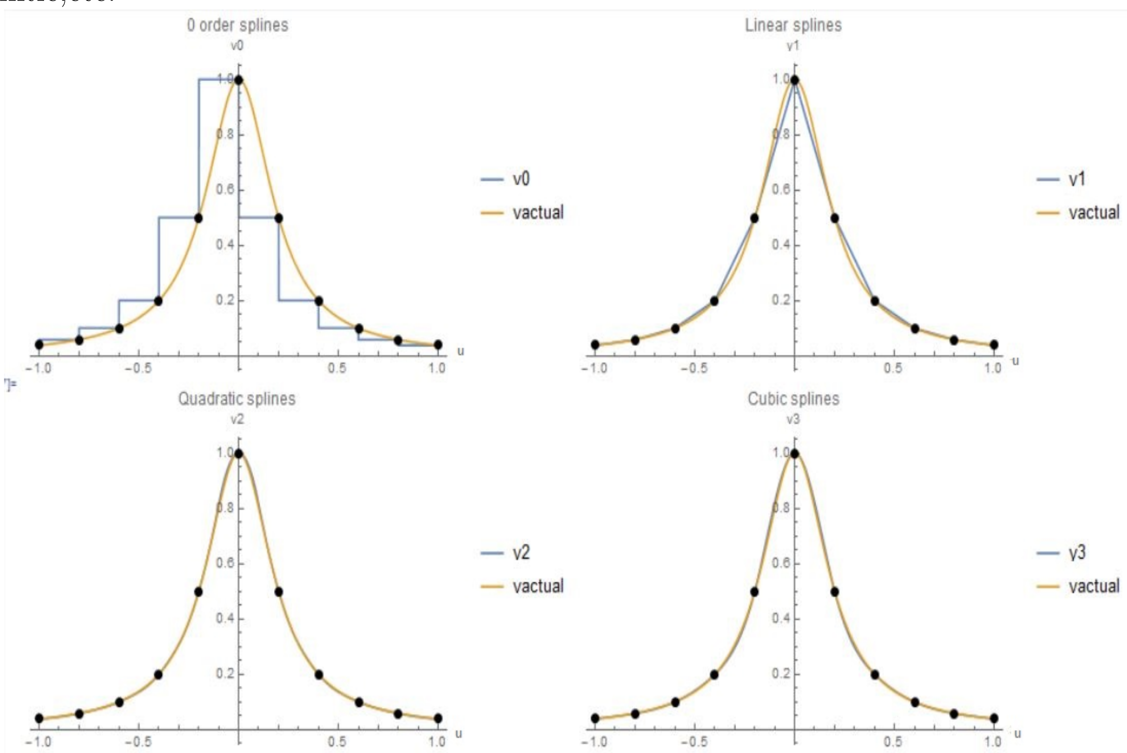
combining gives

$$C(t) = (1-p)^2T_0 + 2p(1-p)T_1 + p^2T_2$$



2.5 Spline Interpolation

In the above methods , we have so far discussed the technique to find the polynomial for recovering the function by the given set of data points, but now it is noticed that a low order polynomial approximation in each subinterval provides a better approximation to the tabulated function than fitting a single high order polynomial to the entire range. these connecting piecewise function are called Spline Function. The meeting points of two splines is called the Knots.These polynomial may be different types as Linear, Quadratic ,Cubic, Quintic,etc.



2.5.1 Linear Spline

Let the given data points be

$$(u_i, v_i)\,, \quad i = 0, 1, 2, \ldots, n$$

where

$$p = u_0 < u_1 < u_2 < \cdots < u_n = q$$

and let

$$k_i = u_i - u_{i-1}, \quad i = 1, 2, \ldots, N.$$

Also, let $\alpha_i(u)$ is the one degree defined in the interval $[u_{i-1}, u_i]$. so, $\alpha_i(x)$ denotes a straight line passing through the points (u_{i-1}, v_{i-1}) and (u_i, v_i). Hence , it can be written as

$$\alpha_i(u) = v_{i-1} + m_i\,(u - u_{i-1}) \qquad .....(1)$$

where

$$m_i = \frac{v_i - v_{i-1}}{u_i - u_{i-1}}$$

Setting $i = 1, 2, \ldots, n$ successively in (1), we obtain different splines of degree one valid in the subintervals 1 to n, respectively. It is easily seen that $\alpha_i(u)$ is continuous at both the end points.

2.5.2 Quadratic Spline

Let us consider the set of data points as

$$(u_i, v_i), \quad i = 0, 1, 2, \ldots, N$$

where

$$p = u_0 < u_1 < u_2 < \cdots < u_N = q$$

and let

$$k_i = u_i - u_{i-1}, \quad i = 1, 2, \ldots, N.$$

, let $\alpha_i(u)$ be the approximating quadratic spline function $v = f(u)$ in the closed interval $[u_{i-1}, u_i]$, here $u_i - u_{i-1} = h_i$. Let $\alpha_i(u)$ and $\alpha_i'(u)$ are continuous function in $[u_0, u_n]$ and let

$$\alpha_i(u_i) = v_i, \quad i = 0, 1, 2, \ldots, N.$$

Since $\alpha_i(u)$ is a quadratic in $[u_{i-1}, u_i]$, so $\alpha_i'(u)$ is a linear function and therefore

$$\alpha_i'(u) = \frac{1}{k_i}\left[(u_i - u)\, m_{i-1} + (u - u_{i-1})\, m_i\right]$$

where

$$m_i = \alpha_i'(u_i).$$

Integrating (3.83) with respect to u, we obtain

$$\alpha_i(u) = \frac{1}{k_i}\left[-\frac{(u_i - u)^2}{2} m_{i-1} + \frac{(u - u_{i-1})^2}{2} m_i\right] + c_i,$$

where we have to find the value of c_i . Putting $u = u_{i-1}$ in (3.85), we get

$$c_i = v_{i-1} + \frac{1}{k_i}\frac{k_i^2}{2} m_{i-1} = v_{i-1} + \frac{k_i}{2} m_{i-1}.$$

Hence (3.85) becomes:

$$\alpha_i(u) = \frac{1}{k_i}\left[-\frac{(u_i - u)^2}{2} m_{i-1} + \frac{(u - u_{i-1})^2}{2} m_i\right] + v_{i-1} + \frac{k_i}{2} m_{i-1}.$$

In (3.86), the m_i are to be determine. To find out the m_i,So by the continuity of the function since it is already given that first derivatives is continuous. By the continuity of the function $\alpha_i(u)$ at $u = u_i$, we must have

$$\alpha_i(u_i-) = \alpha_{i+1}(u_i+)$$

From (3.86), we obtain

$$\begin{aligned}\alpha_i\left(u_i-\right) &= \frac{h_i}{2}m_i + v_{i-1} + \frac{h_i}{2}m_{i-1}\\ &= \frac{h_i}{2}\left(m_{i-1}+m_i\right)+v_{i-1}\end{aligned}$$

Further,

$$\alpha_{i+1}(u) = \frac{1}{h_{i+1}}\left[-\frac{\left(u_{i+1}-u\right)^2}{2}m_i + \frac{\left(u-u_i\right)^2}{2}m_{i+1}\right] + v_i + \frac{h_{i+1}}{2}m_i$$

and therefore

$$\alpha_{i+1}\left(u_i+\right) = -\frac{h_{i+1}}{2}m_i + v_i + \frac{h_{i+1}}{2}m_i = v_i.$$

Equality of (3.88) and (3.89) produces the recurrence relation

$$m_{i-1}+m_i = \frac{2}{h_i}\left(v_i - v_{i-1}\right), \quad i = 1,2,\ldots,n$$

Differentiating (3.86) twice with respect to u, we obtain

$$\alpha_i''(u) = \frac{1}{h_i}\left(-m_{i-1}+m_i\right)$$

or

$$\alpha_1''\left(u_1\right) = \frac{1}{h_1}\left(m_1 - m_0\right).$$

Hence, we have the additional condition as

$$m_0 = m_1.$$

Therefore, Eqs. (3.90) and (3.91) can be determined for m_i, which when substituted in (3.86) gives the required quadratic spline.

2.5.3 Cubic B-Spline

Let's assume the data points $\Delta_n : p = u_0 < u_1 < \cdots < u_{n-1} < u_n = q$ on a given interval $[p,q]$ and let $hk = \frac{q-p}{n}$ be the step size of the partition. So the spline of degree k is known as Given piecewise polynomial function s on the interval $[a,b]$ if $s \in C^{l-1}[a,b]$ and s is a polynomial of degree at most l on each subinterval $[u_i, u_{i+1}]$. Now the degree zero B-splines are defined by

$$P_i^0(u) = \begin{cases} 1 & \text{if } u_i \le u < u_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

and

$$P_i^k(u) = \left(\frac{u-u_i}{u_{i+k}-u_i}\right)P_i^{k-1}(c)$$

Cubic B-spline:

$$
P_i^3(u) = \begin{cases}
\frac{(u-u_i)^3}{(u_{i+3}-u_i)(u_{i+2}-u_i)(u_{i+1}-u_i)} & \text{if } xu_i \leq u < u_{i+1}, \\
\frac{(u-u_i)^2(u_{i+2}-u)}{(u_{i+3}-u_i)(u_i-u_i)(u_{i+2}-u_{i+1})} + \frac{(u-u_i)(u_{i+3}-u)(u-u_{i+1})}{(u_{i+3}-u_i)(u_{i+3}-u_{i+1})(u_{i+2}-u_{i+1})} & \\
+\frac{(u-u_{i+1})^2}{(u_{i+4}-u_{i+1})(u_{i+3}-u_{i+1})(u_{i+2}-u_{i+1})} & \text{if } u_{i+1} \leq u < u_{i+2}, \\
\frac{(u-u_i)(u_{i+3}-u)^2}{(u_{i+3}-u_i)(u_{i+3}-u_{i+1})(u_{i+3}-u_{i+2})} + \frac{(u_{i+4}-u)^2(u-u_{i+2})}{(u_{i+4}-u)(u-u_{i+1})(u_{i+3}-u_{i+1})(u_{i+3}-u_{i+2})} & \\
+\frac{(u_{i+4}-u_{i+1})(u_{i+4}-u_{i+2})(u_{i+3}-u_{i+2})}{(u_{i+4}-u)^3} & \text{if } u_{i+2} \leq u < u_{i+3}, \\
\underline{(u_{i+4}-u_{i+3})} & \text{if } u_{i+3} \leq u < u_{i+4}, \\
0 & \text{otherwise.}
\end{cases}
$$

The knots in cubic B-Spline $u_i, u_{i+1}, u_{i+2}, u_{i+3}, u_{i+4}$.The cubic B-spline does vanish with leaving some points on the interval $[u_i, u_{i+4})$. This condition applies for all B-splines. In fact, $P_i^k(u) = 0$ if $u \notin [u_i, u_{i+k+1})$, otherwise $P_i^k(u) > 0$ if $u \in (u_i, u_{i+k+1})$.

In order to represent this function of degree 3 , we write P_i instead of P_i^3. Here the Knots are equally spaced. Hence, after joining four extra knots, we consider that $\Delta : u_{-2} < u_{-1} < u_0 < u_1 < \cdots < u_{n-1} < u_n < u_{n+1} < u_{n+2}$ is a set of uniform data points. Letting $h = u_{i+1} - u_i$ for any $0 \leq i \leq n$, we define the uniform cubic B-spline $P_i(u)$ as

$$
P_i(u) = \frac{1}{6k^3} \begin{cases}
(u - u_{i-2})^3 & \text{if } u_{i-2} \leq u < u_{i-1} \\
-3(u - u_{i-1})^3 + 3k(u - u_{i-1})^2 + 3k^2(u - u_{i-1}) + k^3 & \text{if } u_{i-1} \leq u < u_i \\
-3(u_{i+1} - u)^3 + 3k(u_{i+1} - u)^2 + 3k^2(u_{i+1} - u) + k^3 & \text{if } u_i \leq u < u_{i+1} \\
(u_{i+2} - u)^3 & \text{if } u_{i+1} \leq u < u_{i+2} \\
0 & \text{otherwise.}
\end{cases}
$$

let us consider $k = 1$, then nature in bounded interval $[-2, 2]$, we have the following

$$
P_0(u) = \frac{1}{6} \begin{cases}
(u + 2)^3 & \text{if } -2 \leq u < -1 \\
4 - 6u^2 - 3u^3 & \text{if } -1 \leq u < 0 \\
4 - 6u^2 + 3u^3 & \text{if } 0 \leq u < 1 \\
(2 - u)^3 & \text{if } 1 \leq u < 2 \\
0 & \text{otherwise}
\end{cases}
$$

Now, Let's formulate cubic B-spline method for the second-order linear

$$
(u) + \left(\frac{u_{i+k+1} - u}{u_{i+k+1} - u_{i+1}} \right) P_{i+1}^{k-1}(u),
$$

for $i = 0, \pm 1, \pm 2, \pm 3, \ldots$. P_i^k function is defined by above equations are called *B*-splines of degree k.

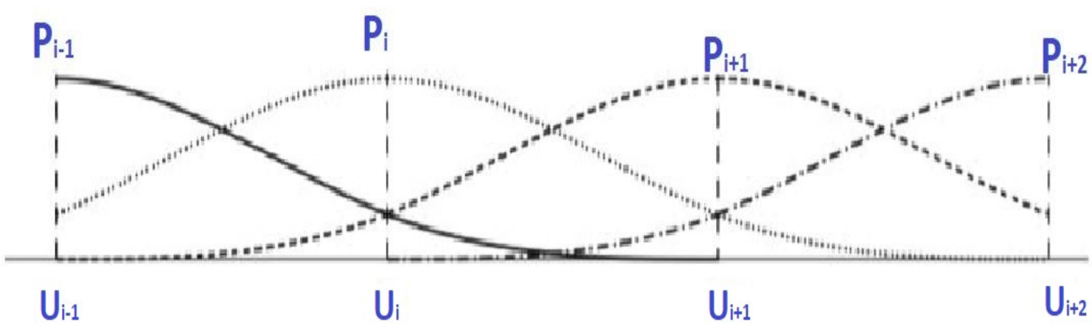
With the help of recurrence relation and considering the partition Δ_n

(a) Linear B-spline:

$$P_i^1(u) = \begin{cases} \frac{u-u_i}{u_{i+1}-u_i} & \text{if } u_i \leq u < u_{i+1} \\ \frac{u_{i+2}-u}{u_{i+2}-u_{i+1}} & \text{if } u_{i+1} \leq u < u_{i+2} \\ 0 & \text{otherwise.} \end{cases}$$

(b) Quadratic B-spline:

$$P_i^2(u) = \begin{cases} \frac{(u-u_i)^2}{(u_{i+2}-u_i)(u_{i+1}-u_i)} & \text{if } u_i \leq u < u_{i+1} \\ \frac{(u-u_i)(u_{i+2}-u)}{(u_{i+2}-u_i)(u_{i+2}-u_{i+1})} + \frac{(u_{i+3}-u)(u-u_{i+1})}{(u_{i+3}-u_{i+1})(u_{i+2}-u_{i+1})} & \text{if } u_{i+1} \leq u < u_{i+2}, \\ \frac{(u_{i+3}-u)^2}{(u_{i+3}-u_{i+1})(u_{i+3}-u_{i+2})} & \text{if } u_{i+2} \leq u < u_{i+3}, \\ 0 & \text{otherwise.} \end{cases}$$



3
Prerequisite For Image Interpolation

The digital Image is stored in the digital computer in the form of Two dimensional Array $P \times Q$ array $f = [f(p,q)]$, where $1 \leqslant p \leqslant P$ representing the rows and $1 \leqslant q \leqslant Q$ representing the columns where p and q are non negative Integers. when we stored a image in a digital computer then it's class create in MATLAB shows it uint8 . here 8 denotes the number of bits of binary , required for storage at a given quantization level . Let's talk about the Binary Image , which contains only two pixel values which is Black and white . In case of Gray scale Image there are 256 (counted from 0 to 255) different gray levels ranging from lowest intensity i.e. Black to highest intensity i.e.White.

In this chapter we will learn about some Important Terms and Concept which will be very useful during the image interpolation and These Concepts are such as Frequency Domain , Transform Kernel , Two dimensional Filtering.

3.1 Frequency Domain

In order to manipulate and do processing in digital images ,the mathematical basic concepts are derived from the Basic concepts of Linear Systems , the theory of integral transformation.on the digital image $f[p,q]$, All of these techniques can be applied . In particular the integrable image vector space defined with a basis of the space $\mathcal{L}^2\left(\mathbb{Z}_P \times \mathbb{Z}_Q\right)$ as

$$\mathbf{S} = \{\mathbf{S}_{a,b}\}, \qquad \text{with } 0 \leqslant a < P, 0 \leqslant b < Q, \text{ and } a, b \in \text{ real number},$$

where

$$\mathbf{S}_{a,b}[p,q] = \begin{cases} 1, & \forall [a,b] = [p,q], \\ 0, & \text{otherwise.} \end{cases}$$

Now the image is represented as

$$f = \sum_{a=0}^{P-1} \sum_{b=0}^{Q-1} \sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} f[p,q]\mathbf{S}_{a,b}[p,q].$$

this above representation of an image particularly called as frequency domain. now it's basis is represented as $\mathbf{T}_{a,b}$ with $0 \leqslant a < P$, $0 \leqslant b < Q$, here

$$\mathbf{T}_{a,b}[p,q] = e^{2\pi j\left(\frac{ap}{P} + \frac{bq}{Q}\right)}.$$

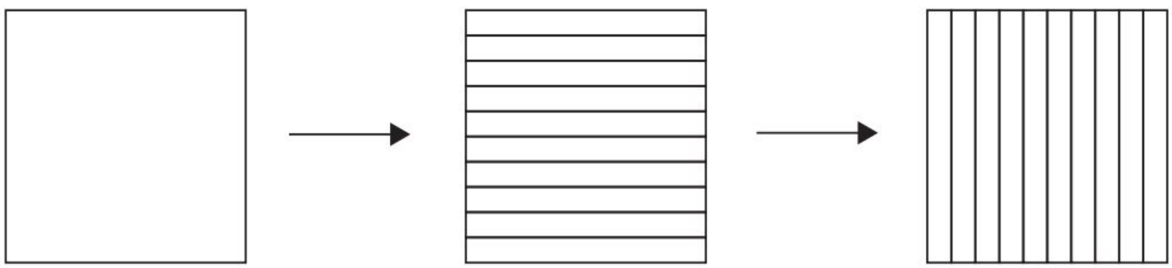
When the transformation is done from the domain of space to the domain of frequency ,it is called as the forward Two dimensional discrete Fourier transform (2D DFT) and the reverse process is called as the inverse transformation , it can be represented mathematically as,

$$F[u,v] = \tfrac{1}{PQ}\textstyle\sum_{p=0}^{P-1}\sum_{q=0}^{Q-1} f[p,q]e^{-2\pi j\left(\frac{p}{P}+\frac{pv}{Q}\right)}, \qquad \begin{array}{l} u = 0,1,\ldots,P-1 \\ v = 0,1,\ldots,Q-1 \end{array}$$

$$f[p,q] = \textstyle\sum_{u=0}^{Q-1}\sum_{v=0}^{Q-1} F[u,v]e^{2\pi j\left(\frac{pw}{P}+\frac{pv}{Q}\right)}, \qquad \begin{array}{l} p = 0,1,\ldots,P-1 \\ q = 0,1,\ldots,Q-1 \end{array}$$

here u and v are frequencies in spatial domain. Note that

$$\exp\left(2\pi j\left(\frac{pu}{P}+\frac{qv}{Q}\right)\right) = \exp\left(2\pi j\frac{pu}{P}\right)\exp\left(2\pi j\frac{qv}{Q}\right)$$



(a) (b) (c)

3.2 Kernel of the Transformation

The Forward Fourier transformation is mathematically represented as

$$F[u,v] = \frac{1}{PQ}\sum_{p=0}^{P-1}\sum_{q=0}^{Q-1} f[p,q]r[p,q,u,v],$$

and the inverse Forward Fourier transformation is mathematically represented as

$$f[p,q] = \sum_{u=0}^{P-1}\sum_{v=0}^{Q-1} F[u,v]s[p,q,u,v]$$

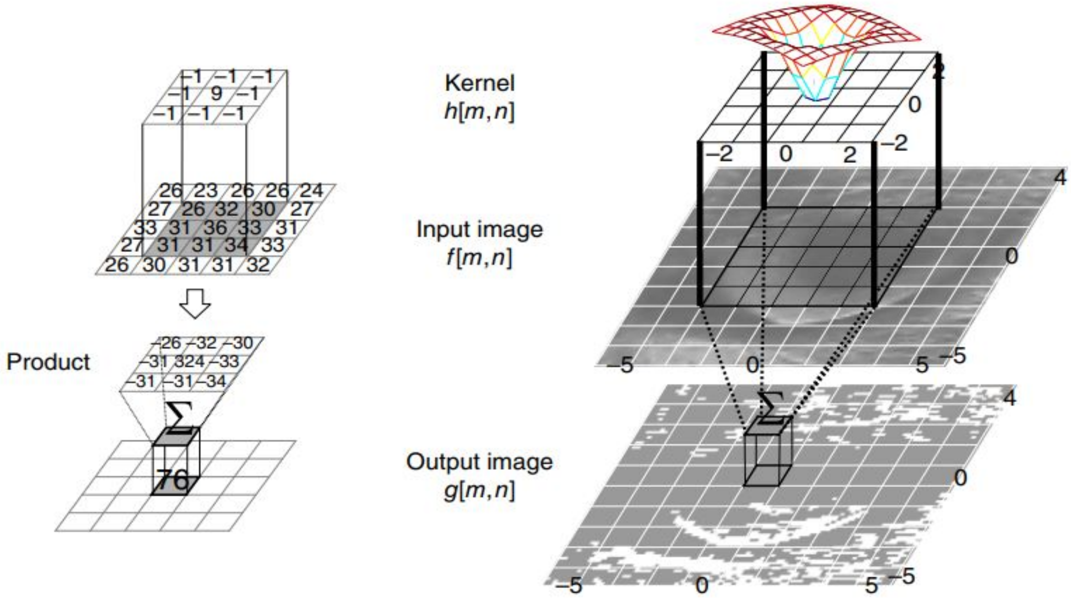
here $r[p,q,u,v]$ and $s[p,q,u,v]$ are called as the kernels of the forward and inverse transformation , respectively. If Kernel $r[p,q,u,v] = r_1[p,u]_2[q,v]$, thus the transformation are said to be separable. In addition.

3.3 Two Dimensional Filtering

The operation in between $f[p,q]$ (the image) and the two dimensional filter $k[p,q]$ is called the Two Dimensional Filtering and hence it generates the filtered image $g[p,q]$ by using the following convolution equation.

$$g[p,q] = (f \otimes h)[p,q] = \sum_{a=P_1}^{P_2} \sum_{b=Q_1}^{Q_2} f[p-a, q-b]h[a,b]$$

Here the size of the convolution is dependent on two things,the first is the data points set size and the second one is the kernel of the transformation. In this case, the filter kernel spans $([P_1, P_2] \times [Q_1, Q_2])$. The geomatrical view of the operation of 2D image convolution can be understood by the Figure given below.



4
Image Interpolation using Two Dimensional Interpolation

4.1 Introduction

In the Practical Scenario when we resize an Image , it does not simply increases the resolution of the image. The Process *increasing the resolution* or resizing it ,is called as "Up Sampling" and this is also known as Image Interpolation. There are two important steps for Up Sampling ,and the first is Signal Approximation and the second one is Resampling.

In *Signal Approximation* ,it is desirable to find a continuous function using the Original points ,where the Original points will lie on the same position as in the previous one.

In *Resampling* we find the value of approximated point by the help of that derived continuous function and increase the density of the original data set.

Since In the Image we do the same work for resizing it , hence image Up-Sampling is done by using the interpolation techniques so it is called as the Image Interpolation.There are mainly two steps of digital image interpolation : Resampling as discussed above , the original image $f(p, q)$ is expanded to the image lattice it formed a intermediate image $\hat{g}(p, q)$ .Then we apply a transformation to create a better image $g(p, q)$ with the help of low pass filter.In this process we remove all kinds of noises which are unwanted in an image.

4.1.1 Mathematical representation of Resampling

$$\hat{g}[p, q] = \begin{cases} f[p, q], & [p, q] \in \Gamma \uplus \Lambda \\ 0, & [p, q], \in \Gamma \notin \Lambda \end{cases}$$

where Λ is low resolution grid and Γ is low resolution grid.

When we increase the density of sampling grid or matrix , there are some Pixels remains left or get NULL value , then we have to apply the low pass filter $k(p, q)$ .Depend upon the quality of Image there are Different options for the filter $k(p, q)$ are taken in practical scenario .

After resampling we get image $\hat{g}(p, q)$, which is convolved with the filter $k(p, q)$. The Rconstructed Process can be Formulated

$$g[p, q] = (\hat{g} \otimes k)[p, q] = \sum_{a=P_1}^{P_2} \sum_{b=Q_1}^{Q_2} \hat{g}[p - a, q - b]k[a, b]$$

Here the size of the convolution is calculated by both the data point matrix and the size of the kernel of the transformation or filter. After the Convolution , we will get the pixels value which were left during the resampling. Since we are using the Interpolation Technique in Filtering process so the Filter is called the Interpolation Filter. In The Image Interpolation

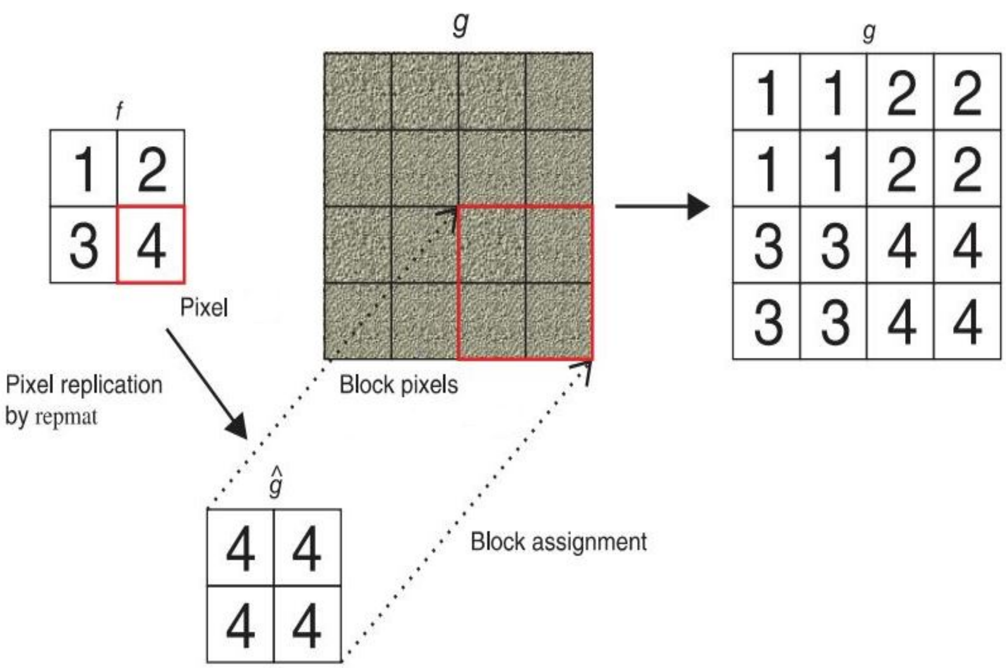
, the Interpolation Filter is the most important aspect responsible for the image quality.Now In the next section we will discuss about the different types of the Interpolation Technique where the Kernel for the each technique .

4.2 Nearest Neighbor Technique

In this Technique we replicate the grayscale value from the nearest pixel of its neighbourhood , because of the replication of the pixel value this technique is also called as Pixel Replication Technique because we replicate the value of nearest pixel which have the value derived from the original value. The Kernel of this technique can be mathematically represented as

$$k[p,q] = \begin{cases} 1, & 0 \leqslant |q| < t-1, \quad 0 \leqslant |p| < t-1 \\ 0, & \text{elsewhere.} \end{cases}$$

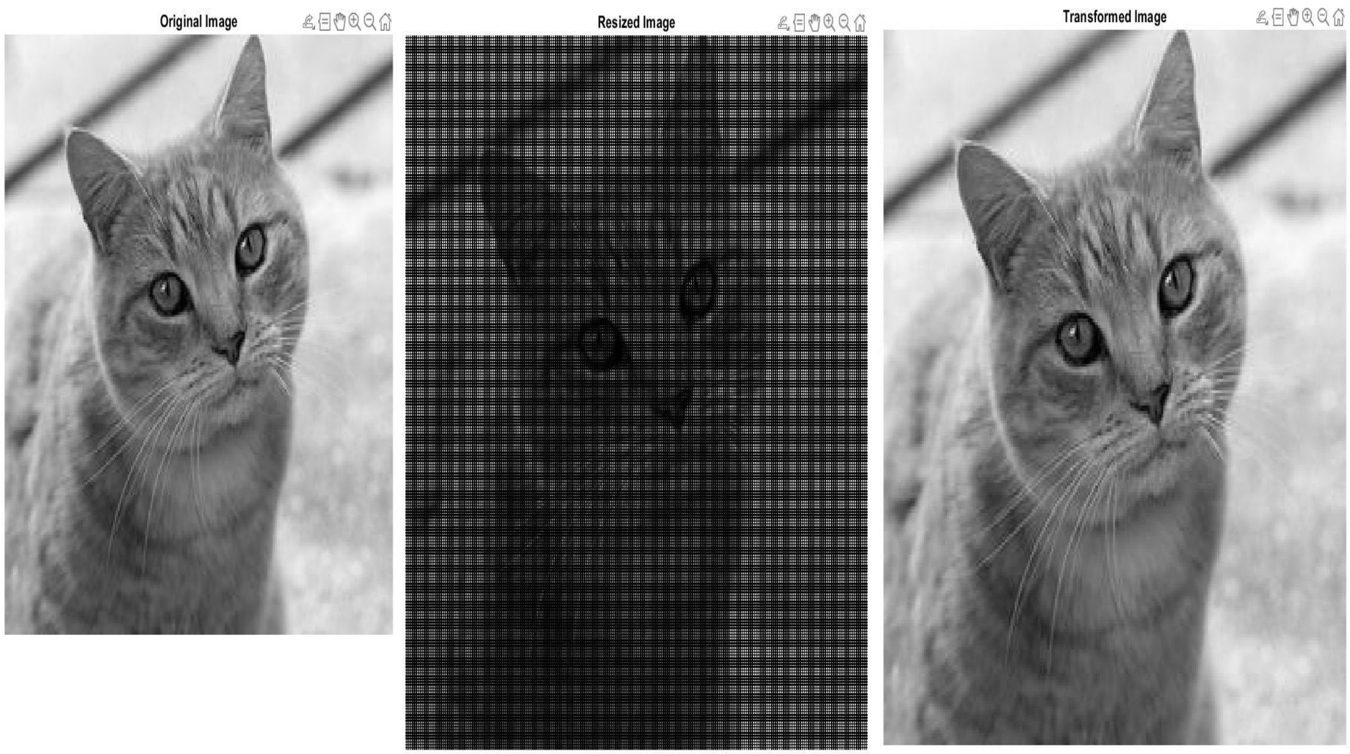
here the value of the $t = 2$ and the size of kernel is is 2×2 rectangular matrix of data points. In The Nearest Neighbor interpolation first we resample the data value from the image matrix area to the expanded matrix area and the declare the zero value to the empty pixels which will be the Intermediate Image $\hat{g}[p,q]$ then we use the nearest neighbor kernel to filter the Image.the process of nearest neighbor technique can be shown in the fallowing figure.



4.2.1 MATLAB Implementation for recovering a Up Sampled Image by factor 2 By Nearest Neighbor Technique

Steps involved in Matlab Codes :

1. Read the Gray scaled image
2. Upscale the image by factor 2.
3. Go to the 2 by 2 matrix having three empty pixel and one pixel having the value.
4. Give all three pixels the same value of the pixel which was having the value.
5. Repeat the 3rd and then 4th step to full region of the image.
6. At last covering the full region of image we will get the final output.



4.3 Bilinear Interpolation

Bilinear is also very well known technique because of it's simplicity when we apply the low pass filtering operation . Bilinear Technique is derived from the one dimensional linear interpolation technique. In the Bilenar Interpolation first we use the approximation of the straight line along the data set in array in it's row ,then we use the approximation of the straight line along the data set in array in it's columns.

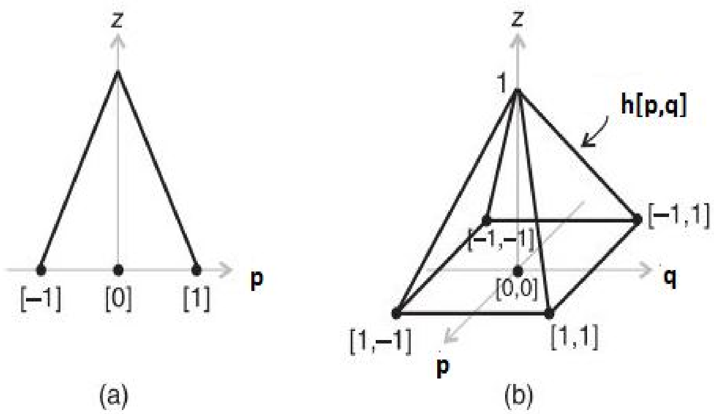


Figure The plot of (a) linear interpolation kernel and (b) bilinear interpolation kernel in spatial domain.

In One Dimensional Interpolation , Kernel can be mathematically formulated as

$$h[p] = \begin{cases} 1 - |p|, & 0 \leqslant |p| < 1, \\ 0, & 1 \leqslant |p|. \end{cases}$$

This Kernel is a triangular function. The function is continuous but not differentiable. First we have to find the intermediate result $\hat{g}[p]$

$$\hat{g}[p] = \begin{cases} f[p], & p = 2m - 1 \\ 0, & \text{otherwise} \end{cases}$$

Now the interpolated result is given by

$$g[p] = (1 - \Delta n)\hat{g}[p-1] + \Delta n \hat{g}[p+1]$$

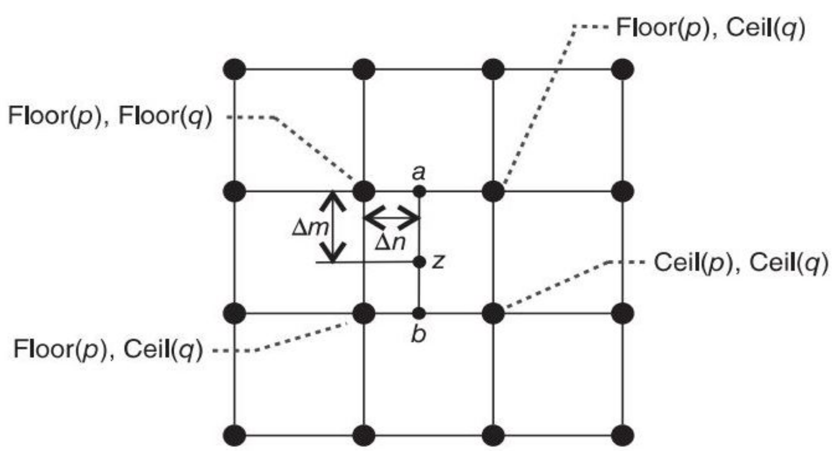


Figure Spatial weighting map of the bilinear interpolation for the pixel *z* located at (*p*, *q*).

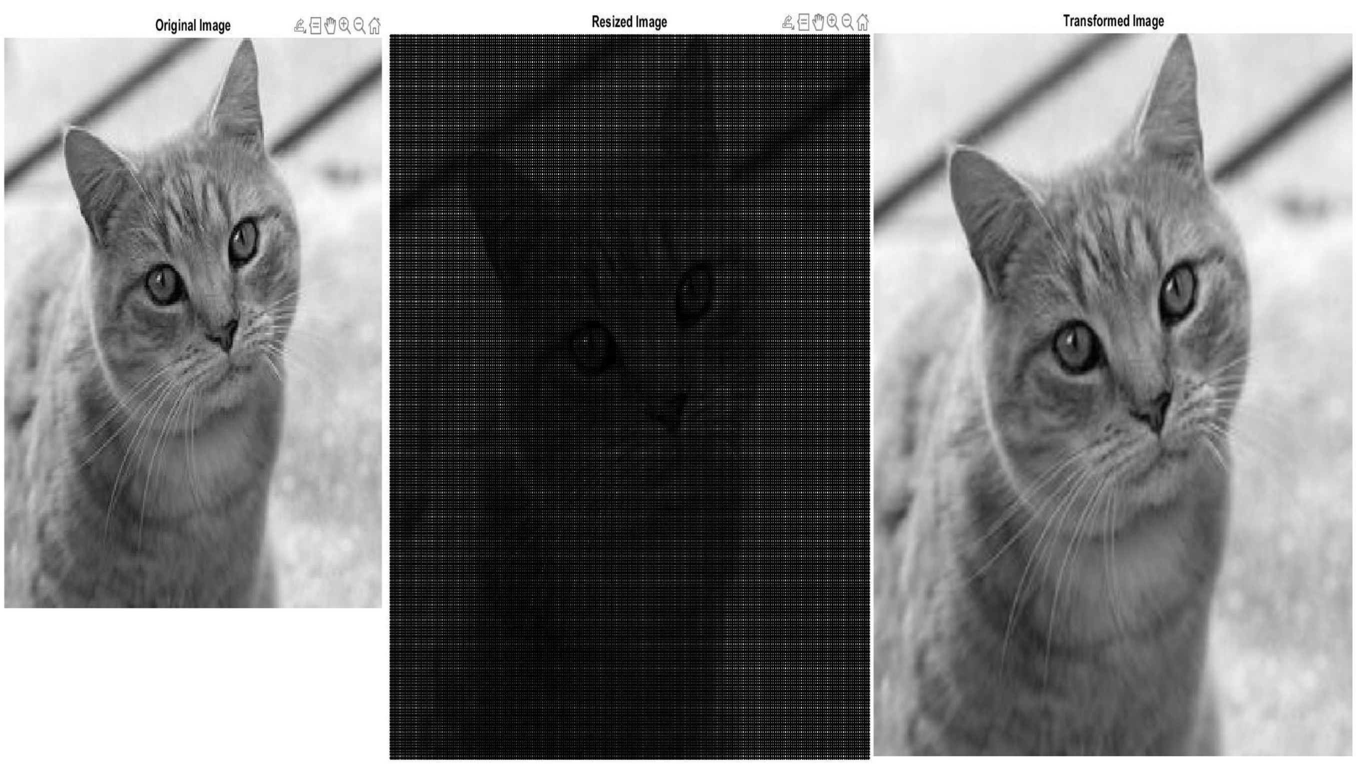
this is basically a weighted Sum using two pixels which are nearest to that particular pixel, where $1 - \Delta n$ denotes the weighting factor on $\hat{g}[p-1]$ and Δn denotes the weighting factor on $\hat{g}[p+1]$

Now , it is very easy to derive the two dimensional interpolation by One dimensional linear interpolation technique , which can be clearly seen by the above figure.

4.3.1 MATLAB Implementation for recovering a Up Sampled Image by factor 3 By Bilinear Interpolaion Technique

Steps involved in Matlab Codes :

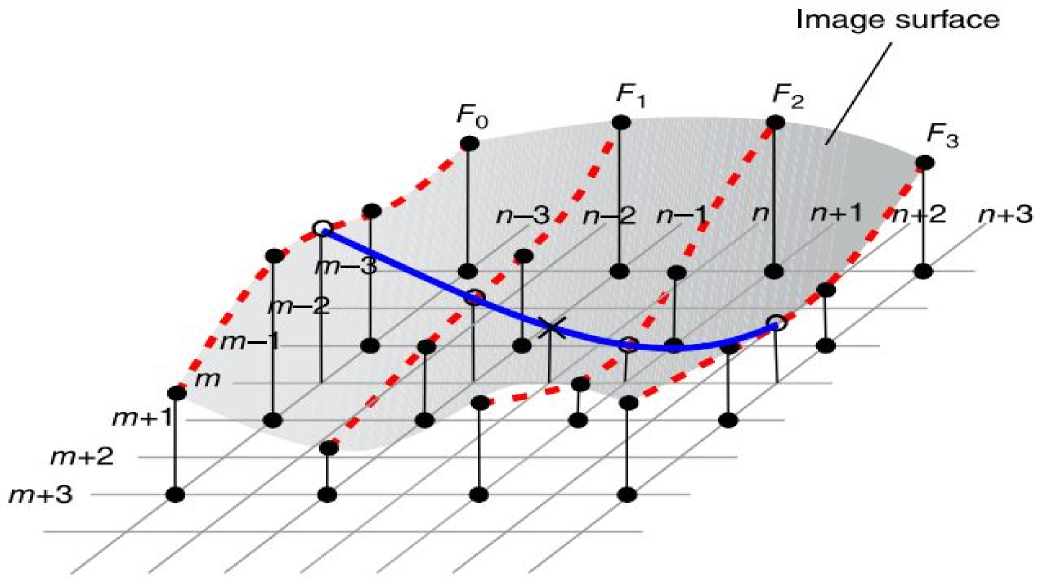
1. Read the Gray scaled image
2. Upscale the image by factor 2.
3. Go to the 4 by 4 matrix with four corners pixels having the value.
4. By the two known pixels of 1st row of matrix, calculate all pixels in between these two by linear interpolation.
5. Do the same process of step 4 for last row of the matrix.
6. In the same matrix by 1st row and 4th row apply linear interpolation and calculate value of each pixels column wise.
7. Repeat the 3rd , 4th , 5th and 6th step to full region of the image.
8. At last covering the full region of image we will get the final output.



4.4 Bicubic Interpolation

Bicubic Interpolation is more better interpolation than the Bilinear Interpolation. In the case of Bicubic Interpolation the kernel size is 4×4. This higher order interpolation gives the far better result that the nearest neighbor technique and the bilinear technique and the result contains the sharp edges which looks very good visually for the user as output.

As done before in the previous section , the Bicubic Interpolation can also be derived easily by the One Dimensional Cubic Interpolation . It can be done by one dimensional cubic spline interpolation for the row and then by one dimensional cubic spline interpolation along the column.



By the help of neighboring 16 pixels it can be clearly seen that the we have find the value of unknown pixel.

With the help of kernel of One Dimensional cubic spline interpolation , we can derive the Bicubic Interpolation Kernel and it's transformation , this is normally a combination of the piece wise cubic polynomials. Cubic spline interpolation shown in the following.

$$k[p] = \begin{cases} (t+2)|p|^3 - (t+3)|p|^2 + 1, & |p| \leqslant 1, \\ t|p|^3 - 5t|p|^2 + 8t|p| - 4a, & 1 < |p| < 2, \\ 0, & \text{otherwise}, \end{cases}$$

where t is normally found in between -0.5 and -0.75. The geomatrical functions with $t = -0.5, -1$, and -2 are drawn below in the diagram.

Figure Basic function of cubic convolution.

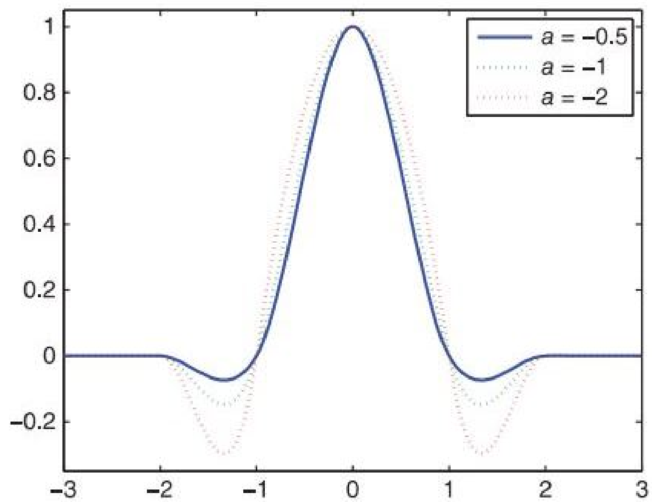
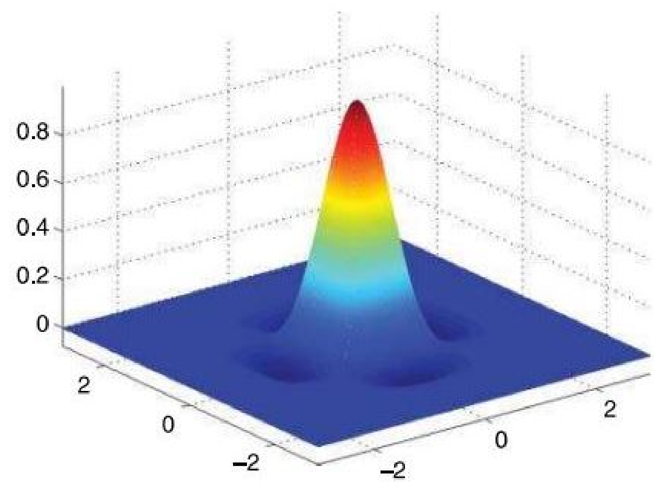


Figure The 2D bicubic interpolation kernel in spatial domain.



4.4.1 MATLAB Implementation for recovering a Up Sampled Image by factor 3 By Bicubic Interpolaion Technique

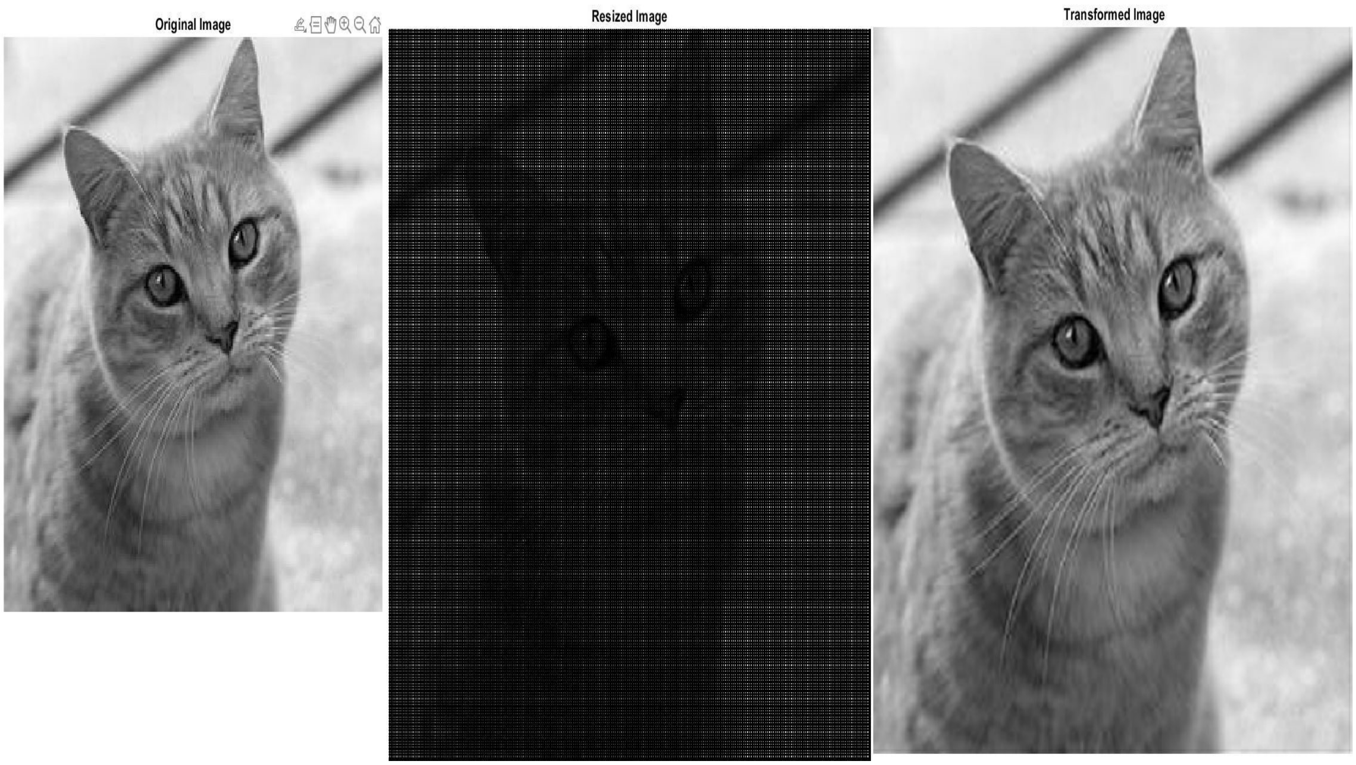
Steps involved in Matlab Codes :

1. Read the Gray scaled image
2. Upscale the image by factor 2.
3. Go to the matrix with four known pixels in row and four known pixels in column pixels having the value.
4. By the four known pixels in each row of matrix, calculate all pixels among these pixels by cubic interpolation.

5. In the same matrix using the same way as in 4th step , calculate the pixels in column wise manner.

6. Repeat the 2rd , 3rd and 4th step to full region of the image.

7. At last covering the full region of image we will get the final output.



4.5 Inpainting Problem

In the above section we did the image completion of the Upscaled image , which may be a kind of inpainting problem but now we are using our interpolation techniques in the distorted image , in which some part of image is distorted by white intensity or in a image in some part is deleted by any means. Our main goal is to recover this part of image by interpolation techniques. Let's jump into the the interpolation techniques

4.5.1 Steps Using Nearest Neighbor Technique :

1.) Read the image.

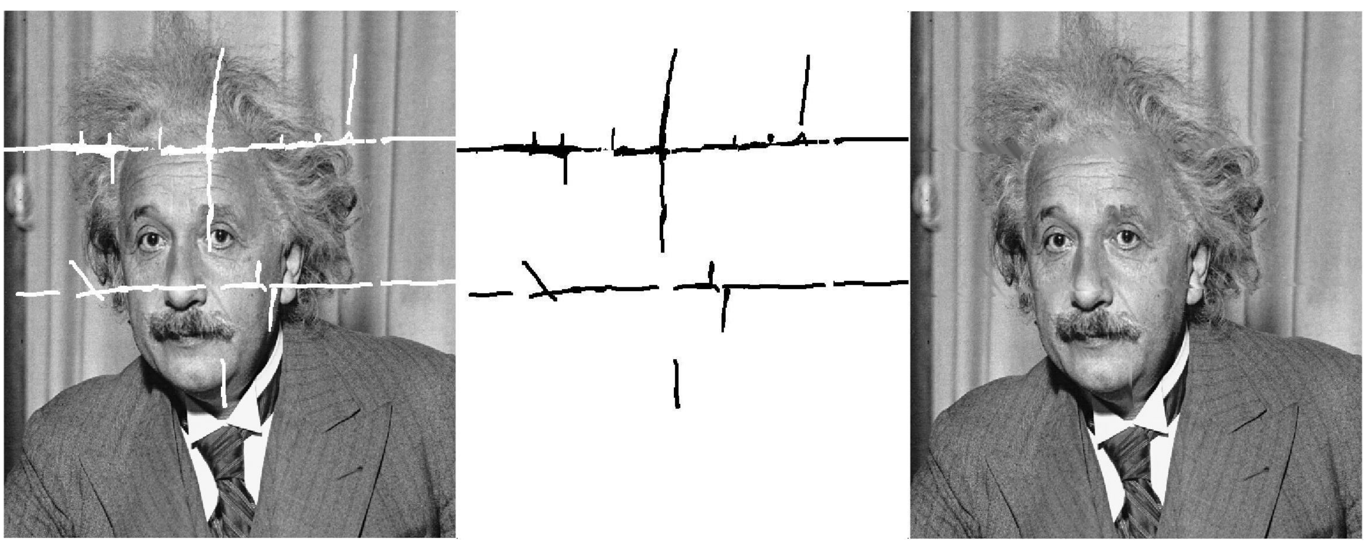
2.) Search for the white intensity (I=255) pixels in image row wise.

3.) Check if that white intensity pixel is a part of image or a part of distortion. if it is a part of image then delete that pixel from the distorted pixels.

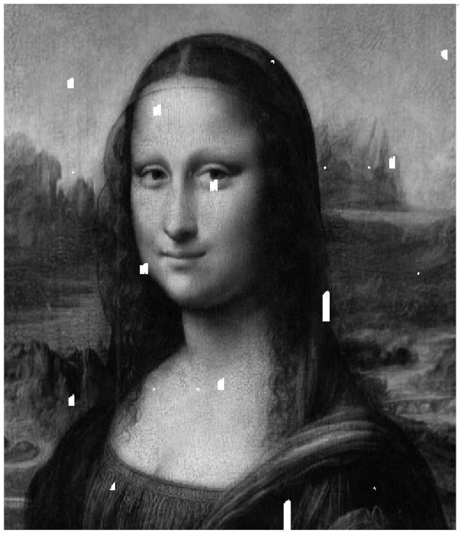
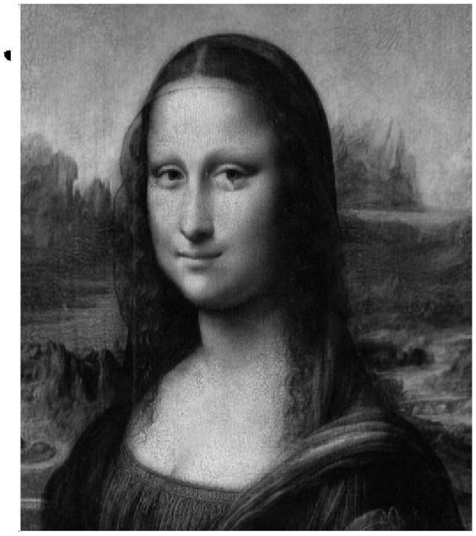
4.) Go to the Pixel having $I = 255$ and if this Pixel having coordinate $g(i, j)$ then take the average of the pixels $g(i, j-1), g(i-1, j-1)$ and $g(i-1, j)$.

5.) Repeat the same process of step 4 for each pixels of intensity $I = 255$.

6.) Hence we will get the final Recovered image.



for observing the quality of recovered image we
Brique(g)= 29.610882979264375
niqe(g)=5.215548622230775
piqe(g)=40.867374151827880

for observing the quality of recovered image we
Brique(g)= 19.788253128022660
niqe(g)= 3.057596551099227
piqe(g)= 15.944211669671619

4.5.2 Steps Using Bilinear Interpolation Technique :

1.) Read the image.

2.) Search for the white intensity (I=255) pixels in image row wise.

3.) Check if that white intensity pixel is a part of image or a part of distortion. if it is a part of image then delete that pixel from the distorted pixels.

4.) Go to the Pixel having $I = 255, g(i,j)$ and search in right direction to the pixel for the next pixel having value $g(i,q)$. Let's distance from the current pixel and the next pixel with value in right direction is c_1.Simailarly search in the down $g(p,j)$ to the current pixel and let's say it's distance is c_2.

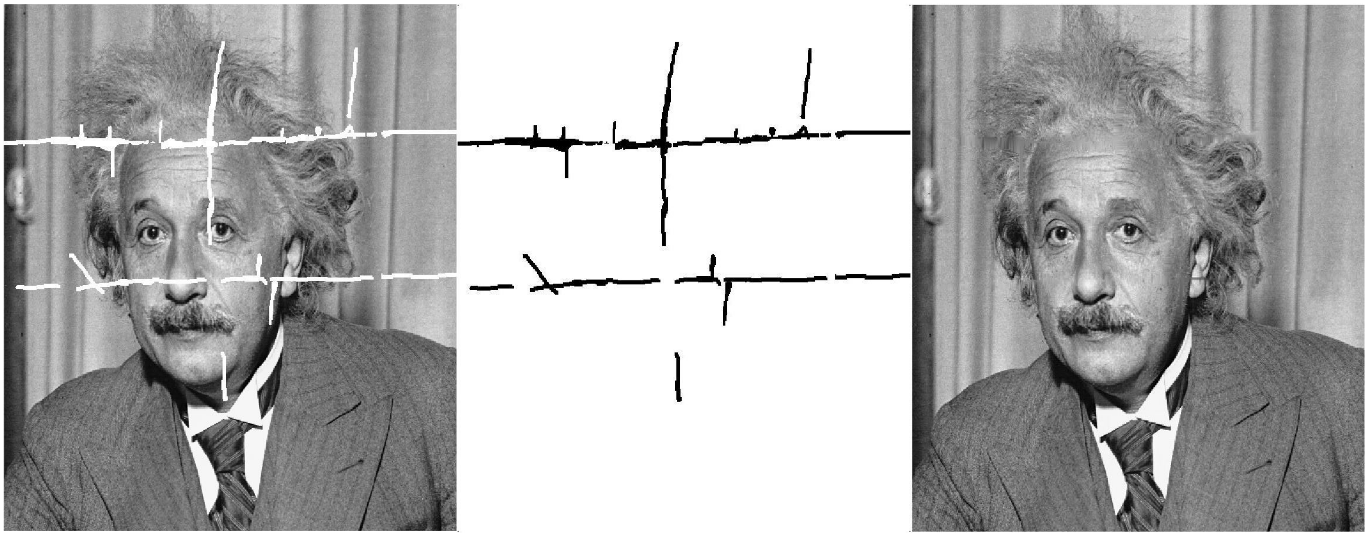
5.) This step may be devided into two parts

- If $c_1 > c_2$ use the pixels $g(i-1,j)$ and $g(p,j)$ and the linear interpolation in order to find all the pixels in between them.

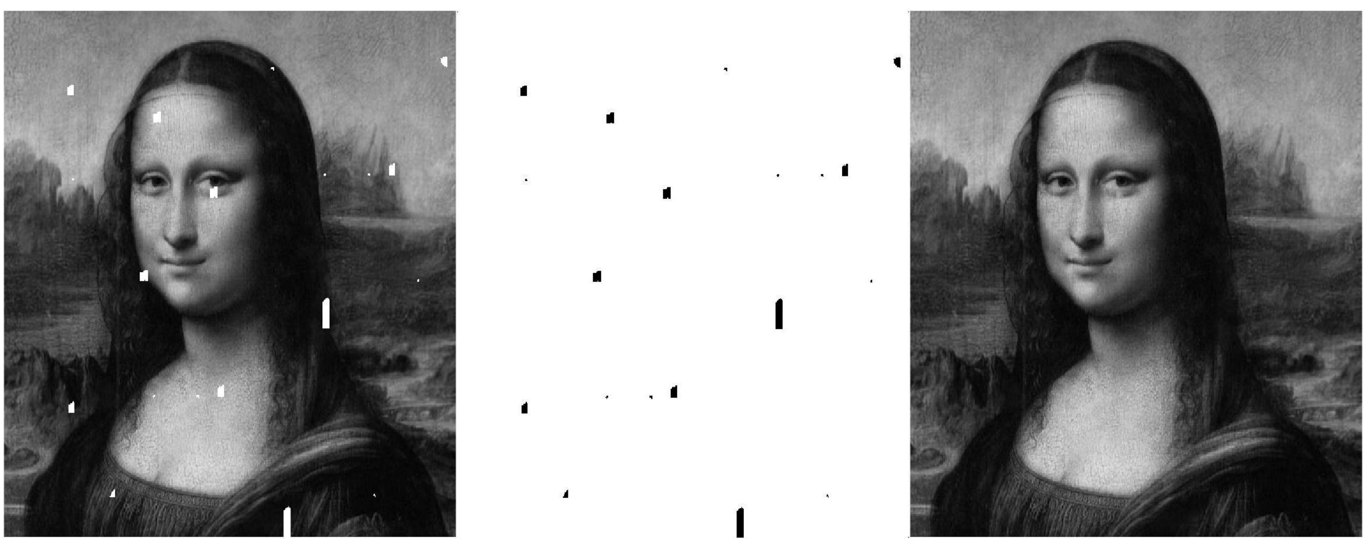
- If $c_1 \leqslant c_2$ use the pixels $g(i,j-1)$ and $g(i,q)$ and apply the linear interpolation in order to find all the pixels in between them.

6.) Do the same step 5 for the whole mask .

7.) Hence we will get the final Recovered image.



for observing the quality of recovered image we
Brique(g)= 28.880889693315627
niqe(g)= 5.358907461696010
piqe(g)= 42.039460998141855



for observing the quality of recovered image we
Brique(g)= 19.620030757367317
niqe(g)= 3.060425212435167
piqe(g)= 15.946848095843968

4.5.3 Steps Using Bicubic Interpolation Technique :

1.) Read the image.

2.) Search for the white intensity (I=255) pixels in image row wise.

3.) Check if that white intensity pixel is a part of image or a part of distortion. if it is a part of image then delete that pixel from the distorted pixels.

4.) Go to the Pixel having $I = 255, g(i, j)$ and search in right direction to the pixel for the next pixel having value $g(i, q)$. Let's distance from the current pixel and the next pixel with value in right direction is c_1.Simailarly search in the down $g(p, j)$ to the current pixel and let's say it's distance is c_2.

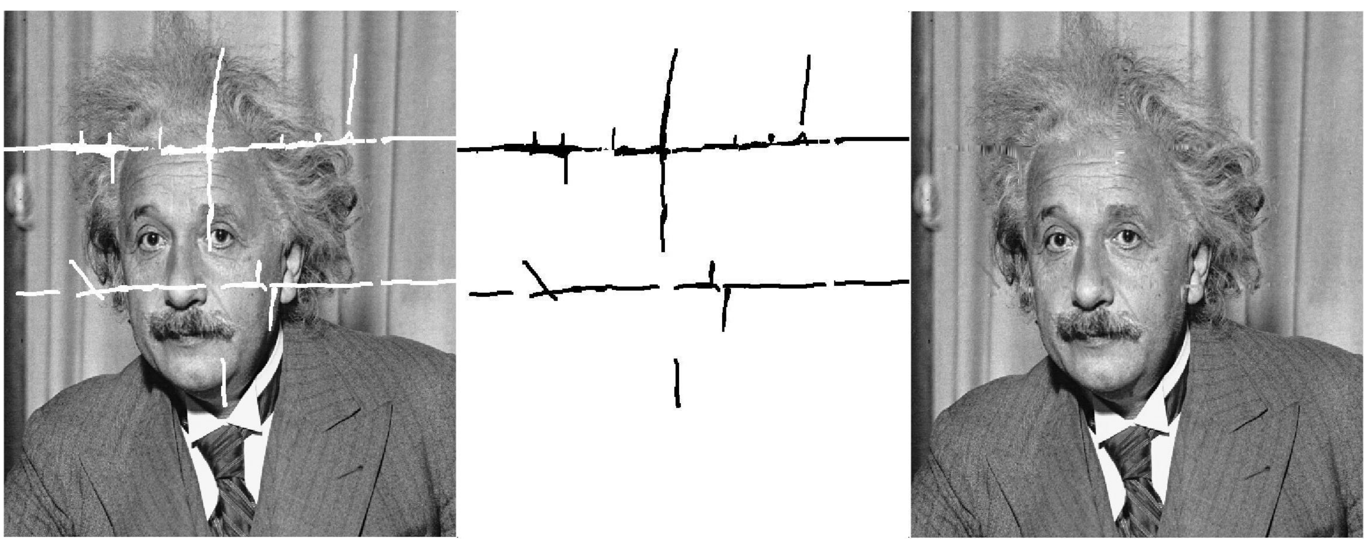
5.) This step may be devided into two parts

- If $c_1 > c_2$, use the pixels $g(i-2, j), g(i-1, j), g(p, j), g(p+1, j)$ (if $g(p+1, j)$ doesn't have value the give it the value as same as $g(p, j)$) and apply cubic interpolation in order to find all the pixels in between them.

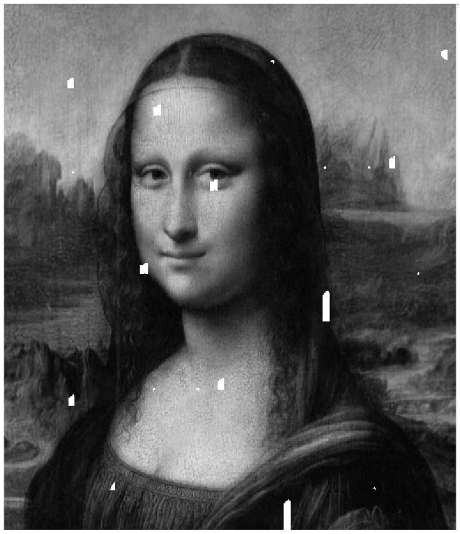
- If $c_1 \leqslant c_2$ use the pixels $g(i, j-2), g(i, j-1), g(i, q), g(i, q+1)$ (if $g(i, q+1)$ doesn't have value the give it the value as same as $g(i, q)$) and apply linear interpolation in order to find all the pixels in between them.

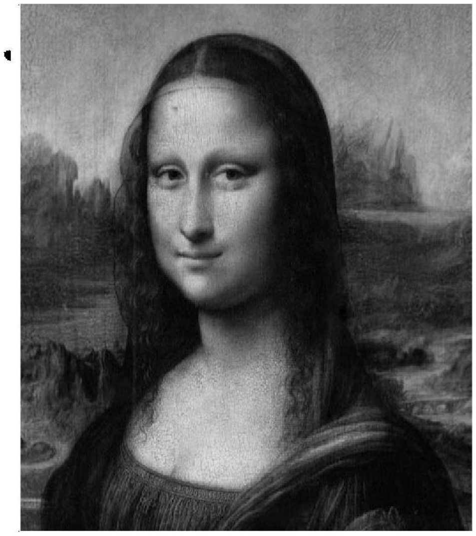
6.) Do the same step 5 for the whole mask .

7.) Hence we will get the final Recovered image.



for observing the quality of recovered image we
Brique(g)= 28.521082515071853
niqe(g)= 5.140113260054049
piqe(g)= 41.570279848314650





for observing the quality of recovered image we
Brique(g)= 19.996873347695253
niqe(g)= 3.057653394320377
piqe(g)= 15.850505583725322

Here we can see the proper meaning of these comparison factors as given below:

Brisque(A) : `brisque(A)` calculates the no-reference image quality score for image A using the Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE). `brisque` compare A to a default model computed from images of natural scenes with similar distortions. A smaller score indicates better perceptual quality. BRISQUE values typically range between 0 (Very good quality) and 150 (very bad quality). Some clean images may show high BRISQUE values based on the nature of the image.

Niqe(A) : `niqe(A)` calculates the no-reference image quality score for image A using the Naturalness Image Quality Evaluator (NIQE). niqe compares A to a default model computed from images of natural scenes. A smaller score indicates better perceptual quality.Summary of Natural Image Quality Evaluator (NIQE) scores lower means better image quality, the higher the value is not so good image quality.

Piqe(A) : piqe(A) calculates the no-reference image quality score for image A using a perception based image quality evaluator. A smaller score indicates better perceptual quality.
Excellent [0, 20]
Good [21, 35]
Fair [36, 50]
Poor [51, 80]
Bad [81, 100]
Hence from the above comparison (with the help of comparison parameters Brisque , Niqe and Piqe) we can clearly see that the Bicubic Interpolation gives the finest result in the inpainting problem discussed above.

5
Conclusion and Future Work

Under this thesis we finally conclude some of important things such that the task of image completion (which is a part of Image inpainting) can be done by Two dimensional Interpolation Techniques . Among these interpolation Techniques Bicubic Interpolation Technique gives the finest result. We are using these techniques in two kinds of Inpainting problems , first is ,in which we have to complete the up sampled image and second is to recover the image by scratches or Distortion.

In future I will be exploring the vast area of Image Inpainting . As I am an M.Sc.-M.Tech student. In my M.Tech.Program , I will be learning about Machine Learning and Deep Learning , so I will use these concepts in Image Inpainting Under my Project.

Bibliography

[1] Gonzalez, C. and Woods, R.E., Digital Image Processing, Prentice Hall, 2008

[2] Jain, A.K., Fundamentals of Digital Image Processing, Prentice Hall, 2001

[3] Sonka, M., Vaclav Hlavac, Roger Boyle, Image Processing, Analysis and Machine Vision, Thomson,2008

[4] Al Bovik, Editor Handbook of image and video processing, Academic Press, 2005.

[5] Seul, M., O'Gorman, L. and Sammon, M.J., Practical algorithms for image analysis: description, examples, and code, Cambridge University Press, 2000.

[6] Gimel'farb, G and Delmas, P., Image Processing and Analysis: a Primer, World Scientific Publishing, 2018.

[7] Kok, C. and Tam, W.-S., Digital Image Interpolation in Matlab, Wiley, 2019.

[8] Lapante Phillip A- Encyclopedia of Image Processing CRC-Press (2018)

[9] Fundamentals of Digital Image Processing A Practical Approach with Examples in Matlab by Chris Solomon, Toby Breckon

[10] Introductory methods of numerical analysis ,S.S. Sastry.