**Practical No 12.C**

**Implementation of Basic Intensity Transformation- Power law Transformation.**

**Aim: Write a program to implement a Basic Intensity Transformation using Power law Transformation algorithm.**

**Theory:**

There are further two transformation is power law transformations, that include nth power and nth root transformation.

These transformations can be given by the expression: s=cr^γ

This symbol γ is called gamma, due to which this transformation is also known as gamma transformation. Variation in the value of γ varies the enhancement of the images. Different display devices / monitors have their own gamma correction, that’s why they display their image at different intensity. This type of transformation is used for enhancing images for different type of display devices. The gamma of different display devices is different. For example, Gamma of CRT lies in between of 1.8 to 2.5, that means the image displayed on CRT is dark.

Correcting gamma. s=cr^γ s=cr^ (1/2.5)

The same image but with different gamma values has been shown here.

For example

**Gamma = 10**



**Gamma = 8**



**Gamma = 6**



**Conclusion: We have implemented Basic Intensity Transformation using Power law Transformation algorithm.**

**Code:**

#include<iostream.h>

#include<fstream.h>

#include<conio.h>

#include<string.h>

#include<math.h>

struct pix

{

unsigned char b,g,r;

}pixel;

class powerlaw

{

int L;

char Header[54];

float gamma;

ifstream in;

ofstream out,out1;

public:

void process()

{

L=255;

char infile[]="cat.bmp";

char outfile[]="output2.bmp";

char imdata[]="logdata.rtf";

in.open(infile,ios::in|ios::binary);

in.read(( char\*)(&Header),sizeof(Header));

out.open(outfile,ios::out| ios::binary);

out.write(( char\*)(&Header),sizeof(Header));

out1.open(imdata, ios::out);

cout<<"Enter gamma value (0 To 1):"; cin>>gamma;

while(!in.eof())

{

in.read((char \*)(&pixel),sizeof(pixel));

out1<<"ORIGINAL : " <<(int)pixel.r<<" , "<<(int)pixel.g<<" , "<<(int)pixel.b<<endl; pixel.r=pow(pixel.r,gamma); pixel.g=pow(pixel.g,gamma); pixel.b=pow(pixel.b,gamma);

out.write((char \*)(&pixel),sizeof(pixel));

out1<<"MODIFIED : " <<(int)pixel.r<<" , "<<(int)pixel.g<<" , "<<(int)pixel.b<<endl;

}

in.close();

out.close();

}

};

int main()

{

clrscr();

powerlaw p;

p.process();

getch();

}

**Output:**

****

** **

**Before Transformation After Transformation**