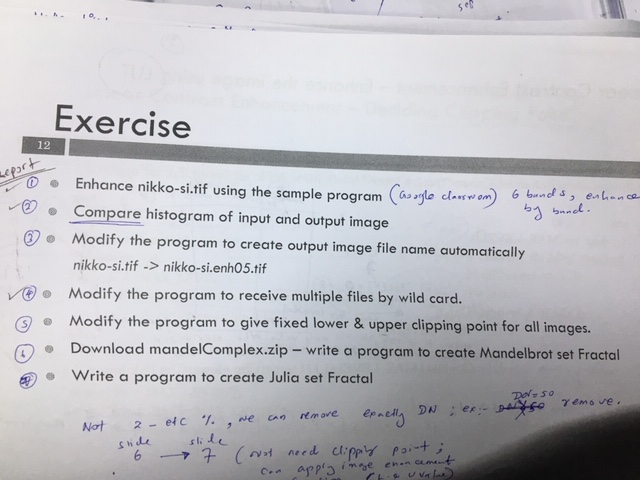
**Basic image processing- contrast enhancement**

**Objective**

To enhance nikko-si.tif image



**Input image - nikko.tiff**



**Code description**

*//Define functions image.c and image.h for handling images on memory and some image functions which are in library which link by “image.h” and “imgio.h” for handling file I/O\*/*

#include<stdio.h>

#include<math.h>

#include "image.h"

#include "imgio.h"

*/\*define maximum DN value/\**

#define MAX\_LEVEL 256

*/\*Function statements\*/*

int imageAccHist(u\_char \*\*pix, int w, int h, u\_long \*hist, u\_long \*accHist); */\*Function to calculate accumulate histogram for each band\*, \*\*pix is pointing to pointer and addree of histogram and accumulated histogram/*

int clippingPoint(int clipPercent, u\_long \*accHist, int \*lowerClip, int \*upperClip); */\*Function to calculate clipping point/\**

int imageLinearEnhance(u\_char \*\*pixI, u\_char \*\*pixO, int w, int h, int lowerClip, int upperClip); */\*Function to enhance image/\**

void usage(void){

fprintf(stderr, "usage:contrast clipping\_percent inputFile outputFile \n");

}

*/\*Main function part\*/*

int main(int argc, char \*argv[]){

*/\*three input parameters, clip percentage, input file name and output file name. percentage is integer and others are character data type \*/*

int clipPercent;

char \*imgFNameI;

char \*imgFNameO;

*/\*declare working variables such as lower and upper clip, histogram maximum values, accumulated histogram max value, image and image files, image properties (h,w,n data and nchannel)\*/*

*/\*Declare variable for lowerClip and upperClip\*/*

int lowerClip, upperClip;

*/\*Declare variable for histogram and accumulate histogram using array type\*/*

unsigned long hist[MAX\_LEVEL], accHist[MAX\_LEVEL];

IMAGE \*imgI, \*imgO;

IMAGE\_FILE \*imgFI, \*imgFO;

int w, h, nchan, data\_type;

*/\*Declare agreement, not less than four agreements, if less or more than 4 exit program\*/*

int ichan;

if (argc != 4){

usage();

exit(1);

}

*/\*Set argument to integer which is clip percent- ARGUMENT 01, clip is a integer but argument is a character, in order to change atoi is used\*/*

clipPercent = atoi(argv[1]);

*/\*Set image file name of input from argument- ARGUMENT 02\*/*

imgFNameI = argv[2];

*/\* Set image file name of output- ARGUMENT 03/\**

imgFNameO = argv[3];

*/\*checking clip percentage, if value is negative or more than 50% error, exit the program/\**

if (clipPercent < 0 || clipPercent>50){

usage();

fprintf(stderr, "contrast: clipping\_percent must be 0-50:%d\n",clipPercent);

exit(2);

}

*/\*Calling open input file function, obtain image’s w,h,nchan, and data type by read only and buffered. If file input is Null , exit the program\*/*

imgFI = image\_file\_open(imgFNameI, IMAGE\_RDONLY, IMAGE\_BUFFERED);

if (imgFI == NULL){

fprintf(stderr, "contrast: input image file open error %s\n",imgFNameI);

exit(3);

}

if (imgFI->data\_type != IMAGE\_CHAR){ *\*/data type is character, 8bit image with pixel 256\*/*

fprintf(stderr, "contrast: input image file must be 8 bit image %s\n",imgFNameI);

exit(3);

}

imgI = imgFI->image;

w = imgI->w;

h = imgI->h;

nchan = imgI->nchan;

data\_type = imgI->data\_type;

*/\*Allocate memory for output image. Input and putput images are supposed to be the same. Data type are same. If there is error in output image paramer , error and exit the program\*/*

imgO = image\_alloc(w, h, data\_type, nchan);

if (imgO == NULL){

fprintf(stderr, "contrast: output image file allocate error %d %d %d\n",w,h,nchan);

exit(4);

}

*/\*Print out parameters which are information for output image\*/*

printf("contrast: %s->%s%d%% w:%d h:%d nchan%d\n", imgFNameI, imgFNameO, clipPercent, w, h, nchan);

*/\*Calling functions run band by band include accumulate histogram calculate*

*function, clipping point function, and image enhancement function\*/*

for (ichan = 0; ichan < nchan; ichan++){

imageAccHist(imgI->data[ichan], w, h, hist, accHist);

clippingPoint(clipPercent, accHist, &lowerClip, &upperClip);

printf("chanl %d lower %d\n", ichan, lowerClip, upperClip);

imageLinearEnhance(imgI->data[ichan], imgO->data[ichan], w, h, lowerClip, upperClip);

}

*/\*Create output image file\*/*

imgFO = image\_file\_create(imgFNameO, IMAGE\_TRUNC, 0, imgO);

if (imgFO == NULL){

fprintf(stderr, "contrast: output image file create error %s\n",imgFNameO);

exit(5);

}

*/\*Close output image, destroy memory and close input image\*/*

image\_file\_close(imgFO);

image\_destroy(imgO);

image\_file\_close(imgFI);

return 0;

}

*/\*Function to calculate accumulate histogram for each band, unsigned long hist{0}-{256}. \*/*

int imageAccHist(u\_char \*\*pix, int w, int h, u\_long \*hist, u\_long \*accHist){

int i, j, k;

for (k = 0; k < MAX\_LEVEL; k++){ */\*histogram calculation start from 0/\**

hist[k] = 0;

}

for (i = 0; i < h; i++) */\*/then go ROW\*columns by starting from i=0;j=0.\**

for (j = 0; j < w; j++)

hist[pix[i][j]]++;

accHist[0] = hist[0];

for (k = 1; k < MAX\_LEVEL; k++)

accHist[k] = accHist[k - 1] + hist[k];

#if defined(VERBOSE)

for (k = 1; k < MAX\_LEVEL; k++)

printf("%5d\t%9d\t%9d\n", k, hist[k], accHist[k]);

#endif

return 0;

}

*/\*Function to calculate clipping point which is lower clipping point and upper*

*clipping point for image enhancement function\*/*

int clippingPoint(int clipPercent, u\_long \*accHist, int \*lowerClip, int \*upperClip){

int k;

u\_long nPix; */\*how many pixel in total is nPix and it calculate/\**

nPix = accHist[MAX\_LEVEL - 1]; */\*accumulated max\_level of hisogram = 255 = nPix= Total Pixel/\**

\*lowerClip = 0;

for (k = 0; k < MAX\_LEVEL; k++){

if ((double)accHist[k] / nPix \* 100 >= clipPercent){

\*lowerClip = k; */\*when accumulated histo is less than lowerclip percentage ;break the program/\**

break;

}

}

\*upperClip = MAX\_LEVEL - 1;

for (k = \*lowerClip; k < MAX\_LEVEL; k++){

if ((double)accHist[k] / nPix \* 100 >= 100-clipPercent){

\*upperClip = k; */\*when accumulated histo is upper than upper clip percentage ;break the program/\**

break;

}

}

return 0;

}

*/\*Function to enhance image by cutting at lower clipping point and upper clipping*

*point and then straight the value to 0-255\*/*

int imageLinearEnhance(u\_char \*\*pixI, u\_char \*\*pixO, int w, int h, int lowerClip, int upperClip){

int i, j, k;

u\_char lut[MAX\_LEVEL];

for (k = 0; k < MAX\_LEVEL; k++){

if (k <= lowerClip){ *\*/until lut[k]=0\*/*

lut[k] = 0;

}

else if (k >= upperClip){

lut[k] = MAX\_LEVEL - 1; *\*/until lut[k]=Max\*/*

}

else{

lut[k] = (double)(MAX\_LEVEL - 1) / (upperClip - lowerClip)\*(k - lowerClip);

}

}

for (i = 0; i < h; i++){ *\*/now check pixel by pixel , LUT values in each I and j\*/*

for(j = 0; j < w; j++){

pixO[i][j] = lut[pixI[i][j]];

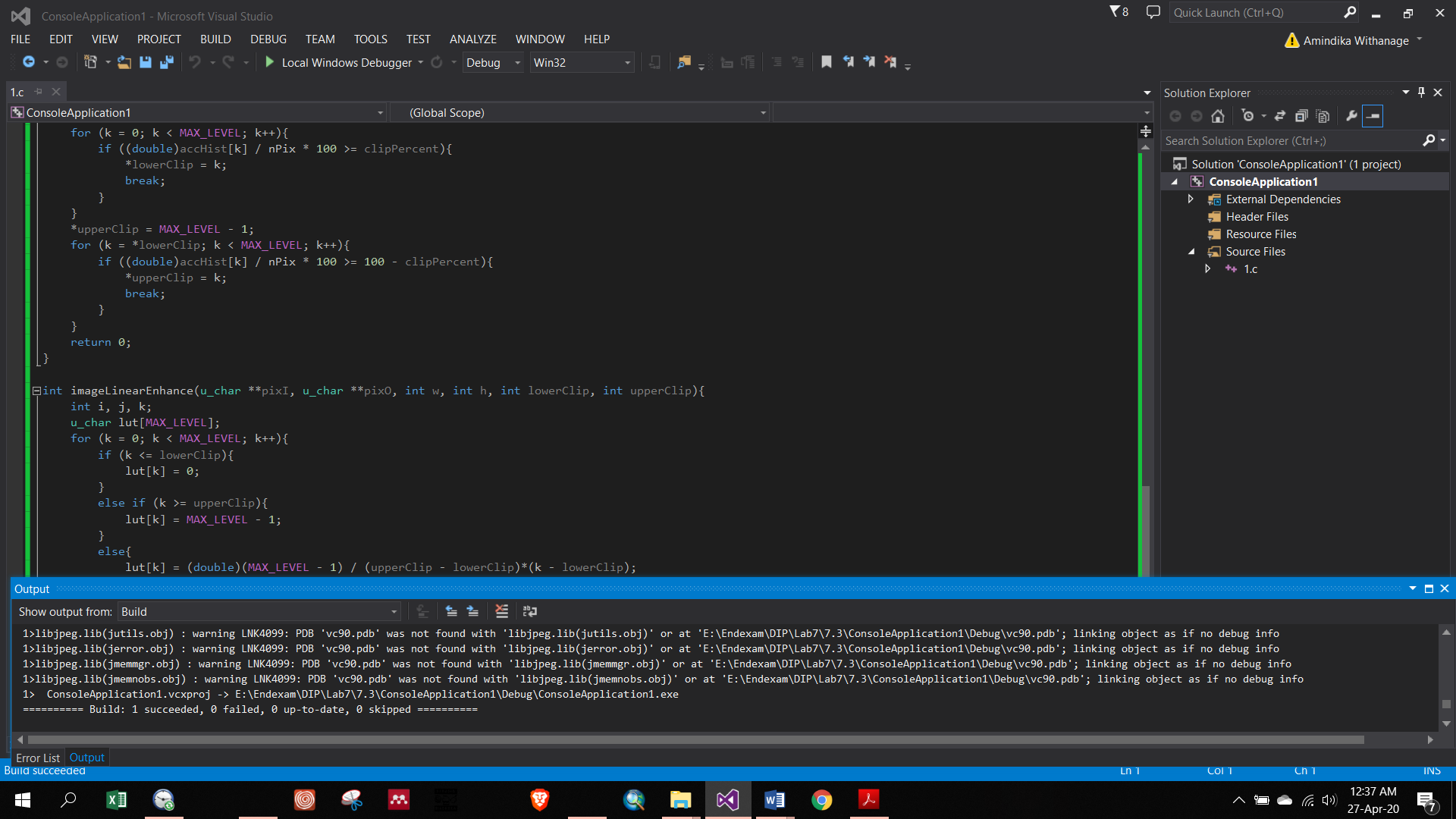
}

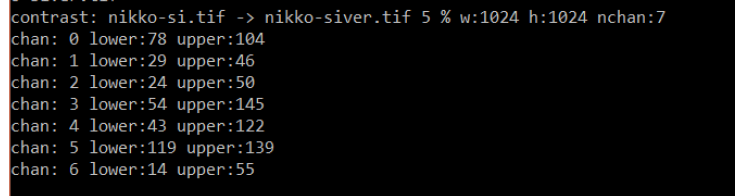
}

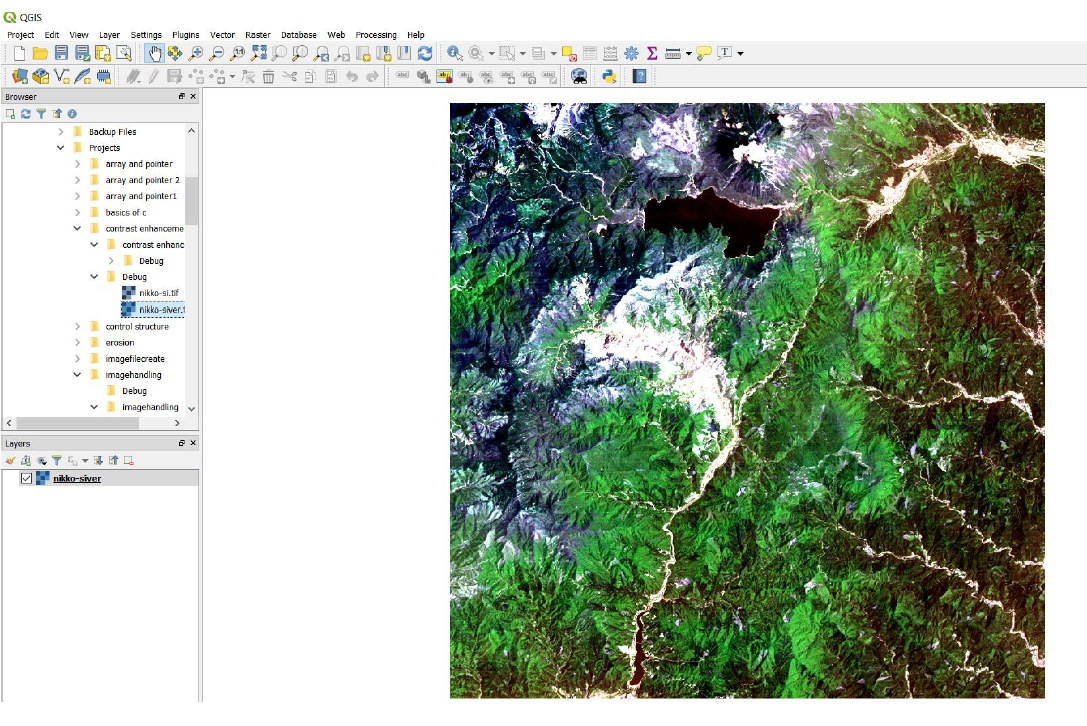
return 0;

}

**Output**







**Figure 1: Tiff output after changing histogram**