Praktikum 7
Histogram and Matching

1. Brightness, Contrast & Histogram

Program berikut ini menunjukkan algoritma *brightness* (kecerahan), *contrast* (kontras) dan visualisasi histogram dari sebuah gambar.

```
#include <cv.h>
#include <highgui.h>
#include <stdio.h>

char file_name[] = "image.jpg";

int _brightness = 100;
int _contrast = 100;

int hist_size = 64;
float range_0[]={0,256};
float* ranges[] = { range_0 };
IplImage *src_image = 0, *dst_image = 0, *hist_image = 0;
CvHistogram *hist;
uchar lut[256];
CvMat* lut_mat;

/* brightness/contrast callback function */
void update_brightcont( int arg )
{
    int brightness = _brightness - 100;
    int contrast = _contrast - 100;
    int i, bin_w;
    float max_value = 0;

    /*
     * The algorithm is by Werner D. Streidt
     * (http://visca.com/ffactory/archives/5-99/msg00021.html)
     */
    if( contrast > 0 )
    {
        double delta = 127.*contrast/100;
        double a = 255./(255. - delta*2);
        double b = a*(brightness - delta);
        for( i = 0; i < 256; i++ )
        {
            int v = cvRound(a*i + b);
            if( v < 0 )
                v = 0;
            if( v > 255 )
                v = 255;
            lut[i] = (uchar)v;
        }
    }
    else
    {
        double delta = -128.*contrast/100;
        double a = (256.-delta*2)/255.;
        double b = a*brightness + delta;
        for( i = 0; i < 256; i++ )
        {
            int v = cvRound(a*i + b);
            if( v < 0 )
                v = 0;
            if( v > 255 )
                v = 255;
            lut[i] = (uchar)v;
        }
    }
    cvLUT( src_image, dst_image, lut_mat );
    cvShowImage( "image", dst_image );

    cvCalcHist( &dst_image, hist, 0, NULL );
    cvZero( dst_image );
    cvGetMinMaxHistValue( hist, 0, &max_value, 0, 0 );
    cvScale( hist->bins, hist->bins, ((double)hist_image->height)/max_value, 0 );
    /*cvNormalizeHist( hist, 1000 );*/

    cvSet( hist_image, cvScalarAll(255), 0 );
    bin_w = cvRound((double)hist_image->width/hist_size);

    for( i = 0; i < hist_size; i++ )
        cvRectangle( hist_image, cvPoint(i*bin_w, hist_image->height),
                     cvPoint((i+1)*bin_w, hist_image->height -
cvRound(cvGetReal1D(hist->bins,i))),
                     cvScalarAll(0), -1, 8, 0 );

    cvShowImage( "histogram", hist_image );
}


int main( int argc, char** argv )
{
    // Load the source image. HighGUI use.
    src_image = cvLoadImage( argc == 2 ? argv[1] : file_name, 0 );

    if( !src_image )
    {
        printf("Image was not loaded.\n");
        return -1;
    }

    dst_image = cvCloneImage(src_image);
    hist_image = cvCreateImage(cvSize(320,200), 8, 1);
    hist = cvCreateHist(1, &hist_size, CV_HIST_ARRAY, ranges, 1);
    lut_mat = cvCreateMatHeader( 1, 256, CV_8UC1 );
    cvSetData( lut_mat, lut, 0 );

    cvNamedWindow("image", 0);
    cvNamedWindow("histogram", 0);

    cvCreateTrackbar("brightness", "image", &_brightness, 200, update_brightcont);
    cvCreateTrackbar("contrast", "image", &_contrast, 200, update_brightcont);

    update_brightcont(0);
    cvWaitKey(0);

    cvReleaseImage(&src_image);
    cvReleaseImage(&dst_image);

    cvReleaseHist(&hist);

    return 0;
}
```

Petunjuk praktikum:

- Jelaskan konsep brightness dan contrast pada program diatas.
- Jelaskan hubungan penambahan ataupun pengurangan nilai brightness dan contrast dengan bentuk histogramnya.

2. Histogram Equalization

Program berikut ini menunjukkan algoritma *histogram equalization* pada sebuah gambar.

```
#include <cv.h>                                 //main OpenCV header
#include <highgui.h>                            //GUI header

int main()
{
        // Set up images
        const char* name = "Histogram Equalization";
        IplImage* img = cvLoadImage("image.jpg", 0);
        IplImage* out = cvCreateImage( cvGetSize(img), IPL_DEPTH_8U, 1 );

        // Show original
        cvNamedWindow( "Original", 1) ;
        cvShowImage( "Original", img );

        // Perform histogram equalization
        cvEqualizeHist( img, out );

        // Show histogram equalized
        cvNamedWindow( name, 1) ;
        cvShowImage( name, out );

        cvWaitKey();
        cvReleaseImage( &img );
        cvReleaseImage( &out );
        return 0;
}
```

Petunjuk praktikum:

- Jelaskan konsep Histogram Equalize pada program diatas.
- Jelaskan perbedaan hasil sebuah gambar dengan menggunakan Histogram Equalize jika dibandingkan dengan Brightness dan Contrast pada percobaan sebelumnya.

3. Back Projection and H-S Histogram

Program berikut ini menunjukkan algoritma *Back Projection* serta *H-S histogram* pada sebuah gambar.

```
#include <cv.h>
#include <highgui.h>

int main( int argc, char** argv )
{
        // Set up images
        IplImage* img = cvLoadImage("image.jpg");
        IplImage* back_img = cvCreateImage( cvGetSize( img ), IPL_DEPTH_8U, 1 );

        // Compute HSV image and separate into colors
        IplImage* hsv = cvCreateImage( cvGetSize(img), IPL_DEPTH_8U, 3 );
        cvCvtColor( img, hsv, CV_BGR2HSV );

        IplImage* h_plane = cvCreateImage( cvGetSize( img ), 8, 1 );
        IplImage* s_plane = cvCreateImage( cvGetSize( img ), 8, 1 );
        IplImage* v_plane = cvCreateImage( cvGetSize( img ), 8, 1 );
        IplImage* planes[] = { h_plane, s_plane };
        cvCvtPixToPlane( hsv, h_plane, s_plane, v_plane, 0 );

        // Build and fill the histogram
        int h_bins = 30, s_bins = 32;
        CvHistogram* hist;
        {
                int hist_size[] = { h_bins, s_bins };
                float h_ranges[] = { 0, 180 };
                float s_ranges[] = { 0, 255 };
                float* ranges[] = { h_ranges, s_ranges };
                hist = cvCreateHist( 2, hist_size, CV_HIST_ARRAY, ranges, 1 );
        }
        cvCalcHist( planes, hist, 0, 0 ); // Compute histogram
        cvNormalizeHist( hist, 20*255 ); // Normalize it

        cvCalcBackProject( planes, back_img, hist );// Calculate back projection
        cvNormalizeHist( hist, 1.0 ); // Normalize it

        // Create an image to visualize the histogram
        int scale = 10;
        IplImage* hist_img = cvCreateImage( cvSize(h_bins*scale, s_bins*scale),8,3 );
        cvZero ( hist_img );

        // populate the visualization
        float max_value = 0;
        cvGetMinMaxHistValue( hist, 0, &max_value, 0, 0 );

        for( int h = 0; h < h_bins; h++ ){
                for( int s = 0; s < s_bins; s++ ){
                        float bin_val = cvQueryHistValue_2D( hist, h, s );
                        int intensity = cvRound( bin_val * 255 / max_value );
                        cvRectangle( hist_img, cvPoint( h*scale, s*scale ),
                                                cvPoint( (h+1)*scale-1,(s+1)*scale-1 ),
                                                CV_RGB( intensity,intensity,intensity ),
                                                CV_FILLED );
                }
        }

        // Show original
        cvNamedWindow( "Source", 1) ;
        cvShowImage( "Source", img );

        // Show back projection
        cvNamedWindow( "Back Projection", 1) ;
        cvShowImage( "Back Projection", back_img );

        // Show histogram equalized
        cvNamedWindow( "H-S Histogram", 1) ;
        cvShowImage( "H-S Histogram", hist_img );

        cvWaitKey(0);

        cvReleaseImage( &img );
        cvReleaseImage( &back_img );
        cvReleaseImage( &hist_img );

        return 0;
}
```

Petunjuk praktikum:

- Jelaskan konsep Back Projection pada program diatas.
- Bagaimana cara membaca H-S Histogram pada hasil program di atas.

4. Template Matching

Program berikut ini menunjukkan algoritma *template matching* pada dua buah gambar.

```
#include <cv.h>
#include <cxcore.h>
#include <highgui.h>

int main( int argc, char** argv )
{
        IplImage *src, *templ, *ftmp[6]; // ftmp will hold results
        int i;

        // Read in the source image to be searched
        src = cvLoadImage("image.jpg");

        // Read in the template to be used for matching:
        templ = cvLoadImage("template.jpg");

        // Allocate Output Images:
        int iwidth = src->width - templ->width + 1;
        int iheight = src->height - templ->height + 1;
        for(i = 0; i < 6; ++i){
                ftmp[i]= cvCreateImage( cvSize( iwidth, iheight ), 32, 1 );
        }

        // Do the matching of the template with the image
        for( i = 0; i < 6; ++i ){
                cvMatchTemplate( src, templ, ftmp[i], i );
                cvNormalize( ftmp[i], ftmp[i], 1, 0, CV_MINMAX );
        }

        // DISPLAY
        cvNamedWindow( "Template", 0 );
        cvShowImage( "Template", templ );
        cvNamedWindow( "Image", 0 );
        cvShowImage( "Image", src );
        cvNamedWindow( "SQDIFF", 0 );
        cvShowImage( "SQDIFF", ftmp[0] );
        cvNamedWindow( "SQDIFF_NORMED", 0 );
        cvShowImage( "SQDIFF_NORMED", ftmp[1] );
        cvNamedWindow( "CCORR", 0 );
        cvShowImage( "CCORR", ftmp[2] );
        cvNamedWindow( "CCORR_NORMED", 0 );
        cvShowImage( "CCORR_NORMED", ftmp[3] );
        cvNamedWindow( "COEFF", 0 );
        cvShowImage( "COEFF", ftmp[4] );
        cvNamedWindow( "COEFF_NORMED", 0 );
        cvShowImage( "COEFF_NORMED", ftmp[5] );

        cvWaitKey(0);

        return 0;

}
```

- Jelaskan konsep Template Matching pada program diatas.
- Jelaskan tentang `SQDIFF,CCORR` dan `COEFF`.
- Apa perbedaanya `SQDIFF,CCORR` dan `COEFF` dengan setelah ditransformasi menjadi `SQDIFF_NORMED, CCORR_NORMED, COEFF_NORMED`.
- Jelaskan fungsi berikut ini beserta dengan parameter yang ada di dalamnya.
    - `cvMatchTemplate ()`
    - `cvNormalize ()`

Tugas:

Buatlah program untuk membandingkan dua bentuk objek dengan menggunakan fungsi:

- `cvMatchShapes()`

Buatlah program untuk menghitung dua jenis tanda tangan yang mirip dengan menggunakan fungsi:

- `cvCalcEMD2()`