Branch: master opencv / samples / cpp / tutorial_code / Histograms_Matching / calcHist_Demo.cpp Find file Copy path

sturkmen72 update cpp samples and tutorials 11ca1c9 on 27 Jun 2016

2 contributors

89 lines (67 sloc) 2.5 KB Raw Blame History

```
/**
 * @function calcHist_Demo.cpp
 * @brief Demo code to use the function calcHist
 * @author
 */

#include "opencv2/highgui.hpp"
#include "opencv2/imgcodecs.hpp"
#include "opencv2/imgproc.hpp"
#include <iostream>

using namespace std;
using namespace cv;

/**
 * @function main
 */
int main(int argc, char** argv)
{
  Mat src, dst;

  /// Load image
  String imageName( "../data/lena.jpg" ); // by default

  if (argc > 1)
  {
      imageName = argv[1];
  }

  src = imread( imageName, IMREAD_COLOR );

  if( src.empty() )
    { return -1; }

  /// Separate the image in 3 places ( B, G and R )
  vector<Mat> bgr_planes;
  split( src, bgr_planes );

  /// Establish the number of bins
  int histSize = 256;

  /// Set the ranges ( for B,G,R) )
  float range[] = { 0, 256 } ;
  const float* histRange = { range };

  bool uniform = true; bool accumulate = false;

  Mat b_hist, g_hist, r_hist;

  /// Compute the histograms:
  calcHist( &bgr_planes[0], 1, 0, Mat(), b_hist, 1, &histSize, &histRange, uniform, accumulate );
  calcHist( &bgr_planes[1], 1, 0, Mat(), g_hist, 1, &histSize, &histRange, uniform, accumulate );
  calcHist( &bgr_planes[2], 1, 0, Mat(), r_hist, 1, &histSize, &histRange, uniform, accumulate );

  // Draw the histograms for B, G and R
  int hist_w = 512; int hist_h = 400;
```

```
    int bin_w = cvRound( (double) hist_w/histSize );

    Mat histImage( hist_h, hist_w, CV_8UC3, Scalar( 0,0,0) );

    /// Normalize the result to [ 0, histImage.rows ]
    normalize(b_hist, b_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat() );
    normalize(g_hist, g_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat() );
    normalize(r_hist, r_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat() );

    /// Draw for each channel
    for( int i = 1; i < histSize; i++ )
    {
        line( histImage, Point( bin_w*(i-1), hist_h - cvRound(b_hist.at<float>(i-1)) ) ,
                         Point( bin_w*(i), hist_h - cvRound(b_hist.at<float>(i)) ),
                         Scalar( 255, 0, 0), 2, 8, 0  );
        line( histImage, Point( bin_w*(i-1), hist_h - cvRound(g_hist.at<float>(i-1)) ) ,
                         Point( bin_w*(i), hist_h - cvRound(g_hist.at<float>(i)) ),
                         Scalar( 0, 255, 0), 2, 8, 0  );
        line( histImage, Point( bin_w*(i-1), hist_h - cvRound(r_hist.at<float>(i-1)) ) ,
                         Point( bin_w*(i), hist_h - cvRound(r_hist.at<float>(i)) ),
                         Scalar( 0, 0, 255), 2, 8, 0  );
    }

    /// Display
    namedWindow("calcHist Demo", WINDOW_AUTOSIZE );
    imshow("calcHist Demo", histImage );

    waitKey(0);

    return 0;

}
```