```
#include <iostream>
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
VideoCapture cap;
cap.open(0);

double scale = 0.5;

while (1)
{
cv::Mat SrcImg;
cap >> SrcImg;
Size ResImgSiz = Size(SrcImg.cols*scale, SrcImg.rows*scale);
Mat matSrc = Mat(ResImgSiz, SrcImg.type());
//resize(srcMat,matSrc,ResImgsiz, INTER_LINEAR);INTER_LINEAR 函数插值方法
resize(SrcImg, matSrc, ResImgSiz, INTER_LINEAR);
Mat matRGB[3];
split(matSrc, matRGB);
int Channels[] = { 0 };
int nHistSize[] = { 256 };
float range[] = { 0, 255 };
const float *fHistRanges[] = { range };
Mat histR, histG, histB;
// 计算直方图
calcHist(&matRGB[0], 1, Channels, Mat(), histB, 1, nHistSize, fHistRanges, true, false);
calcHist(&matRGB[1], 1, Channels, Mat(), histG, 1, nHistSize, fHistRanges, true, false);
calcHist(&matRGB[2], 1, Channels, Mat(), histR, 1, nHistSize, fHistRanges, true, false);

// 创建直方图画布
int nHistWidth = 800;
int nHistHeight = 600;
int nBinWidth = cvRound((double)nHistWidth / nHistSize[0]);
Mat matHistImage(nHistHeight, nHistWidth, CV_8UC3, Scalar(255, 255, 255));
// 直方图归一化
normalize(histB, histB, 0.0, matHistImage.rows, NORM_MINMAX, -1, Mat());
normalize(histG, histG, 0.0, matHistImage.rows, NORM_MINMAX, -1, Mat());
normalize(histR, histR, 0.0, matHistImage.rows, NORM_MINMAX, -1, Mat());
// 在直方图中画出直方图
for (int i = 1; i < nHistSize[0]; i++)
{
line(matHistImage,
Point(nBinWidth * (i - 1), nHistHeight - cvRound(histB.at<float>(i - 1))),
Point(nBinWidth * (i), nHistHeight - cvRound(histB.at<float>(i))),
Scalar(255, 0, 0),
2,
8,
0);
line(matHistImage,
Point(nBinWidth * (i - 1), nHistHeight - cvRound(histG.at<float>(i - 1))),
Point(nBinWidth * (i), nHistHeight - cvRound(histG.at<float>(i))),
Scalar(0, 255, 0),
2,
8,
0);
line(matHistImage,
Point(nBinWidth * (i - 1), nHistHeight - cvRound(histR.at<float>(i - 1))),
Point(nBinWidth * (i), nHistHeight - cvRound(histR.at<float>(i))),
Scalar(0, 0, 255),
2,
8,
0);
}

// 显示直方图
imshow("frame", matSrc);
imshow("histogram", matHistImage);
waitKey(30);
}
}
```