OpenCV--直方图与匹配

直方图的基本数据结构。

```
/** should be used as a parameter only,
   it turns to CV_HIST_UNIFORM_FLAG of hist->type */
#define CV_HIST_UNIFORM       1
typedef struct CvHistogram
{
    int    type;
    CvArr*  bins;
    float  thresh[CV_MAX_DIM][2];  /**< For uniform
histograms.                      */
    float** thresh2;                /**< For non-uniform
histograms.                  */
    CvMatND mat;                    /**< Embedded matrix header for array
histograms. */
}
CvHistogram;
```

结构相对而言比较简单，内部数据很多都被存储于cvMatND结构中。可以用下列函数来创建一个直方图。

```
/**************************************************************************
**************\
*                                  Histogram
functions                                   *
\**************************************************************************
**************/
/** @brief Creates a histogram.
The function creates a histogram of the specified size and returns a
pointer to the created
histogram. If the array ranges is 0, the histogram bin ranges must be
specified later via the
function cvSetHistBinRanges. Though cvCalcHist and cvCalcBackProject may
process 8-bit images
without setting bin ranges, they assume they are equally spaced in 0 to 255
bins.
@param dims Number of histogram dimensions.
@param sizes Array of the histogram dimension sizes.
@param type Histogram representation format. CV_HIST_ARRAY means that the
histogram data is
represented as a multi-dimensional dense array CvMatND. CV_HIST_SPARSE
means that histogram data
is represented as a multi-dimensional sparse array CvSparseMat.
@param ranges Array of ranges for the histogram bins. Its meaning depends
on the uniform parameter
value. The ranges are used when the histogram is calculated or
backprojected to determine which
histogram bin corresponds to which value/tuple of values from the input
image(s).
```

```
@param uniform Uniformity flag. If not zero, the histogram has evenly
spaced bins and for every
\f$0<=i<cDims\f$ ranges[i] is an array of two numbers: lower and upper
boundaries for the i-th
histogram dimension. The whole range [lower,upper] is then split into
dims[i] equal parts to
determine the i-th input tuple value ranges for every histogram bin. And if
uniform=0 , then the
i-th element of the ranges array contains dims[i]+1 elements:
\f$\texttt{lower}_0,
\texttt{upper}_0, \texttt{lower}_1, \texttt{upper}_1 = \texttt{lower}_2,
...
\texttt{upper}_{dims[i]-1}\f$ where \f$\texttt{lower}_j\f$ and
\f$\texttt{upper}_j\f$ are lower
and upper boundaries of the i-th input tuple value for the j-th bin,
respectively. In either
case, the input values that are beyond the specified range for a histogram
bin are not counted
by cvCalcHist and filled with 0 by cvCalcBackProject.
*/
CVAPI(CvHistogram*)  cvCreateHist( int dims, int* sizes, int type,
                                   float** ranges CV_DEFAULT(NULL),
                                   int uniform CV_DEFAULT(1));
```

变量dims表示直方图包含的维度。sizes变量必须是整数数组，数组长度等于dims。数组中的每一个元素表示分配给对应维度的bin个数。type既可以是cv_hist_array，用来表示使用密集多维矩阵结构存储多维直方图，也可以是CV_HIST_SPARSE,当数据以稀疏矩阵（CVSparseMat）方式存储时。变量ranges可以是两个形式中的任意一个形式。对均匀直方图来说，ranges是浮点数对构成的数组，数组的个数等于维数。而对非均匀直方图来说，则用包含分割非均匀bin的数据所构成的数组来代替均匀直方图所使用的数据对。如果某维度有N个bin，那么这个子数组有N+1个元素。每个数组的值均起始于最低bin的底边，终结于最高bin的顶边。布尔类型变量uniform说明直方图是否有均匀的bin，因此也说明了ranges的值该如何解析，如果设置为非0值，则直方图是均匀的，也可以设置ranges为NULL，这时意味着ranges是未知的，他们可以在后面使用特殊函数cvSetHistBinRanges来设置。

```
/** @brief Sets the bounds of the histogram bins.
This is a standalone function for setting bin ranges in the histogram. For
a more detailed
description of the parameters ranges and uniform, see the :ocvCalcHist
function that can initialize
the ranges as well. Ranges for the histogram bins must be set before the
histogram is calculated or
the backproject of the histogram is calculated.
@param hist Histogram.
@param ranges Array of bin ranges arrays. See :ocvCreateHist for details.
@param uniform Uniformity flag. See :ocvCreateHist for details.
*/
CVAPI(void)  cvSetHistBinRanges( CvHistogram* hist, float** ranges,
                                int uniform CV_DEFAULT(1));
```

如果想要进行重用直方图，则需要用到cvClearHist方法来清零，或者使用cvReleaseHist方法来释放内存空间。

```
/** @brief Releases the histogram.
The function releases the histogram (header and the data). The pointer to
the histogram is cleared
by the function. If \*hist pointer is already NULL, the function does
nothing.
@param hist Double pointer to the released histogram.
*/

CVAPI(void)  cvReleaseHist( CvHistogram** hist );

/** @brief Clears the histogram.
The function sets all of the histogram bins to 0 in case of a dense
histogram and removes all
histogram bins in case of a sparse array.
@param hist Histogram.
*/
CVAPI(void)  cvClearHist( CvHistogram* hist );
```

一旦直方图被释放，则会导致指向直方图的指针会被置为NULL。其次还有一个函数是依据已给出的数据创建一个直方图：

```
/** @brief Makes a histogram out of an array.
The function initializes the histogram, whose header and bins are allocated
by the user.
cvReleaseHist does not need to be called afterwards. Only dense histograms
can be initialized this
way. The function returns hist.
@param dims Number of the histogram dimensions.
@param sizes Array of the histogram dimension sizes.
@param hist Histogram header initialized by the function.
@param data Array used to store histogram bins.
@param ranges Histogram bin ranges. See cvC

reateHist for details.
@param uniform Uniformity flag. See cvCreateHist for details.
*/
CVAPI(CvHistogram*)  cvMakeHistHeaderForArray(
                            int  dims, int* sizes, CvHistogram* hist,
                            float* data, float** ranges CV_DEFAULT(NULL),
                            int uniform CV_DEFAULT(1));
```

这种情况下，hist指向的是CVHistogram数据结构体指针，data是指向的直方图bins大小为size[0]*size[1]…*size[dims-1]的区域指针。注意，data是浮点数指针。返回为我们输入的hist值一样。与cvCreateHist方法不同，没有type变量。所有由cvMakeHist创建的都是密集型直方图。

- 访问直方图

访问直方图数据的方式有很多种。最直接的是使用OpenCV访问数据。有对应的query方法，也存在对应的get方法。也可以直接获取最大最小值。

```
/** @brief Finds the minimum and maximum histogram bins.
The function finds the minimum and maximum histogram bins and their
positions. All of output
arguments are optional. Among several extremas with the same value the ones
with the minimum index
(in the lexicographical order) are returned. In case of several maximums or
minimums, the earliest
in the lexicographical order (extrema locations) is returned.
@param hist Histogram.
@param min_value Pointer to the minimum value of the histogram.
@param max_value Pointer to the maximum value of the histogram.
@param min_idx Pointer to the array of coordinates for the minimum.
@param max_idx Pointer to the array of coordinates for the maximum.
*/
CVAPI(void)  cvGetMinMaxHistValue( const CvHistogram* hist,
                                   float* min_value, float* max_value,
                                   int* min_idx CV_DEFAULT(NULL),
                                   int* max_idx CV_DEFAULT(NULL));
```

- 直方图操作

首先来看一些反复用到的基本数据结构，自然要用它来做一些有趣的事情。首先来看一些反复用到的基本操作，然后转到为了更特殊的任务而用到的一些更复杂的特性。当处理直方图时，一般会向直方图不同的bin中累积信息。完成信息的累积后，通常希望使用归一化的形式直方图。

cvNormalizeHist是归一化使用的函数。

```
/** @brief Normalizes the histogram.
The function normalizes the histogram bins by scaling them so that the sum
of the bins becomes equal
to factor.
@param hist Pointer to the histogram.
@param factor Normalization factor.
*/
CVAPI(void)  cvNormalizeHist( CvHistogram* hist, double factor );
```

其中factor是下限阈值，当直方图小于这个阈值时，会变成0。

cvCopyHist可以用于copy直方图数据。

```
/** @brief Copies a histogram.
The function makes a copy of the histogram. If the second histogram pointer
\*dst is NULL, a new
histogram of the same size as src is created. Otherwise, both histograms
must have equal types and
sizes. Then the function copies the bin values of the source histogram to
the destination histogram
and sets the same bin value ranges as in src.
```

```
@param src Source histogram.
@param dst Pointer to the destination histogram.
*/
CVAPI(void)  cvCopyHist( const CvHistogram* src, CvHistogram** dst );
```

cvCalcHist用于计算直方图。

```
/** @brief Copies a histogram.
The function makes a copy of the histogram. If the second histogram pointer
\*dst is NULL, a new
histogram of the same size as src is created. Otherwise, both histograms
must have equal types and
sizes. Then the function copies the bin values of the source histogram to
the destination histogram
and sets the same bin value ranges as in src.
@param src Source histogram.
@param dst Pointer to the destination histogram.
*/

CVAPI(void)  cvCopyHist( const CvHistogram* src, CvHistogram** dst );

/** @brief Calculates bayesian probabilistic histograms
   (each or src and dst is an array of _number_ histograms */

CVAPI(void)  cvCalcBayesianProb( CvHistogram** src, int number,
                                CvHistogram** dst);
```

另外，也提供了相关对比两个直方图的工具。例如cvCompareHist方法，提供了多种对比方法。

```
/** Histogram comparison methods */
enum
{
    CV_COMP_CORREL        =0,
    CV_COMP_CHISQR        =1,
    CV_COMP_INTERSECT     =2,
    CV_COMP_BHATTACHARYYA =3,
    CV_COMP_HELLINGER     =CV_COMP_BHATTACHARYYA,
    CV_COMP_CHISQR_ALT    =4,
    CV_COMP_KL_DIV        =5
};
/** Compares two histogram */
CVAPI(double)  cvCompareHist( const CvHistogram* hist1,
                              const CvHistogram* hist2,
                              int method);
```

CV_COMP_CORREL,计算得出数值越大，图像匹配度越高，完全匹配是1，完全不匹配是-1，0表示无关联。

其余对比，具体看源文档。

```
int main(int argc, const char* argv[])
{
    IplImage* src = cvLoadImage(IMAGE_LOAD_PATH);
    IplImage* hsv = cvCreateImage(cvGetSize(src), 8, 3);
    cvCvtColor(src, hsv, CV_RGB2HSV);
    IplImage* hplane   = cvCreateImage(cvGetSize(src), 8, 1);
    IplImage* splane   = cvCreateImage(cvGetSize(src), 8, 1);
    IplImage* vplane   = cvCreateImage(cvGetSize(src), 8, 1);
    IplImage* planes[] = {hplane, splane};
    cvSplit(hsv, hplane, splane, vplane, 0);
    int hbins = 30, sbins = 32;
    CvHistogram* hist;
    int   histsize[] = {
            hbins, sbins
    };
    float hranges[]  = {0, 180};
    float sranges[]  = {0, 255};
    float* ranges[] = {hranges, sranges};
    hist = cvCreateHist(2, histsize, CV_HIST_ARRAY, ranges);
    cvCalcHist(planes, hist, 0, NULL);
    cvNormalizeHist(hist, 1.00);
    int scale = 10;
    IplImage* histimage = cvCreateImage(CvSize(hbins * scale,
sbins * scale), 8, 3);
    cvZero(histimage);
    float maxValue = 0;
    cvGetMinMaxHistValue(hist, 0, &maxValue, 0, 0);
    for (int h = 0; h < hbins; h++)
    {
        for (int s = 0; s < sbins; s++)
        {
            float value     = cvGetReal2D(hist->bins, h, s);
            int   idteneity = cvRound(value * 255 / maxValue);
            cvRectangle(histimage, cvPoint(h * scale, s * scale),
                        cvPoint((h + 1) * scale - 1, (s + 1) *
scale - 1),
                        CV_RGB(idteneity, idteneity, idteneity),
CV_FILLED);
        }
    }
    cvShowImage("h-s histogram", histimage);
    cvWaitKey();
    return 1
}
```

- 反响投射

反向投射是一种记录像素点或者像素块如何适应直方图模型中的分布的方式。CV提供了CVCalcBackProject方法来查看。

```
/** @brief Calculates back project
@see cvCalcBackProject, cv::calcBackProject
```

```
*/
CVAPI(void)  cvCalcArrBackProject( CvArr** image, CvArr* dst,
                                   const CvHistogram* hist );
#define  cvCalcBackProject(image, dst, hist)
cvCalcArrBackProject((CvArr**)image, dst, hist)
```

- 匹配魔板

通过cvMatchTemplate来匹配模板，并且提供了很多种方法。

```
/** @brief Measures similarity between template and overlapped windows in
the source image
   and fills the resultant image with the measurements
@see cv::matchTemplate
*/
CVAPI(void)  cvMatchTemplate( const CvArr* image, const CvArr* templ,
                              CvArr* result, int method );
```

Method 是具体的方法参数。

- CV_TM_SQDIFF

平方差匹配法，这类方法是利用平方差进行匹配，最好匹配是0，匹配越差，匹配值越大。

- CV_TM_COCORR

这方法是采用模板和图像之间的乘法操作，所以较大的数表示匹配程度较高，0表示最坏的匹配结果。

- CV_TM_CCOEFF

相关性匹配法，这类方法是将模板对其均值的相对值与图像对其均值的相对值进行匹配，1表示完美匹配，-1表示最糟糕的匹配，0表示没有任何相关性匹配。

其次，还有上述三种方式的归一化匹配方法。具体测试代码如下：