User Manual: Histograms of Optical Flow Orientation and Magnitude - C++ Library

Rensso Victor Hugo Mora Colque
Carlos Antonio Caetano Junior
SSIG

This document presents a brief tutorial for HOM descriptor. The code was developed using C++ language and OpenCV library.

Introduction

The HOM library is a tool based in Histograms of Optical Flow Orientation and Magnitude descriptor [1]. It can be downloaded from GitHub. The main goal of the project is gives an alternative tool to describe an image sequence (video).

Structure

In the link folder you will find the code for HOM descriptor and a simple example of use. The HOM descriptor is in *HOMDes.h*, this file contains the main class **OFBasedDescriptorMO** and other support functions. The example will be found in *main cpp*.

OFBasedDescriptorMO Class

This class contains the main mechanism of the project. It has two constructors, you may set the variables using a file or introducing them through constructor function. Once the object is instantiated you can use the Describe function to obtain the feature vector or feature vectors. We have some general definitions:

```
typedef std::pair< cv::Mat_<float>, cv::Mat_<float> >
DesparMat;

typedef std::vector<DesparMat>
DesvecParMat;

typedef std::vector<cutil_grig_point>
CuboTypeCont;

typedef cv::Mat_<float>
HistoType;

typedef std::pair<DesvecParMat, CuboTypeCont>
DesInData;//input data type

typedef std::vector<HistoType>
DesOutData;//output data type
```

Describe Function This function computes the feature vectors. It receives a *DesInData* type, in other words, it receives a pair composes by a vector with optical flow images (orientation and magnitude), and a vector of regions. Thus, in first element (pair vector) we have the optical flow information. The second one is a vector than contains a rectangles, in the code the struct is named as *cuboid_dim*, basically it is represented by two points: (x_i, y_i) and (x_f, y_f) (main diagonal). You decide what regions you want to describe, for instance, if you want to describe the whole image, you can set one region with the image dimensions. The output (*DesOutData*) is a histogram vector, the size of this vector is given by the number of regions that have been set

in the input.

The Example You will find an example code in *descriptor.cpp* file.

```
#include "HOMDes.h"

int _tmain(int argc, char ** argv)
{
        int cuboid_width = 30,
        cuboid_height= 30;
        //Optical flow class extractor, you may choose other
        OpticalFlowOCV ofOCV;
        cv::Mat img1 = cv::imread("043.tif"),
        img2 = cv::imread("045.tif"),
        img3 = cv::imread("047.tif");
        OFdataType in{ img1, img2, img3 };
        OFvecParMat outOf;
        ofOCV.compute(in, outOf);
        //-------------------------------------------------
        DesInData input;
        std::vector<cutil_grig_point> grid;
        grid = grid_generator(img1.rows, img1.cols,
        cuboid_width, cuboid_height,
        cuboid_width, cuboid_height );
        input.first = outOf;
        input.second = grid;
        DesOutData output(grid.size());
        //-------------------------------------------------
        OFBasedDescriptorMO Descriptor( /*orientbin*/4,
        /*magnitudebin*/6,
        /*MaxMagnitude*/15,
        /*Magnitude thr*/0.1);
        Descriptor.Describe(input, output);

        return 0;
}
```

For this specific case, we created a regular grid over the frame. The cuboid dimensions is $30 \times 30 \times 3$, the tow first dimensions correspond to grid cell dimensions (width and height), the third corresponds to the depth in time, it means, the time range. Actually this value is given by the number of optical information pair in the input vector.

As we aforementioned, the descriptor uses optical flow information. In this case we use OpenCV pyramid optical flow, however, we use some support functions to set the descriptor input vector. You may use another optical flow technique, but remember the input is a vector pair, where the first element corresponds to orientation values and the second to magnitude. After that, we create a single grid, already using a support function. All this functions will be find in the *HOMDes.h*. Finally, the output contains each histogram corresponding to each rectangle in the second item in the input vector. In case of you want to describe some particular points, you would create the rectangle centered in those points and called the HOM descriptor.

Final Remarks This code is available to make research, please do not forget cite us. If you have some doubts, suggestions or bugs, please mail to *rensso@dcc.umfg.br* or *carlos.caetano@dcc.ufmg.br*. We hope this code help you.

Bibliografía

[1] R. V. H. M. Colque, C. A. C. Junior, and W. R. Schwartz, "Histograms of optical flow orientation and magnitude to detect anomalous events in videos," in *28th SIBGRAPI Conference on Graphics, Patterns and Images, SIBGRAPI 2015, Salvador, Bahia, Brazil, August 26-29, 2015*, 2015, pp. 126–133. [Online]. Available: http://dx.doi.org/10.1109/SIBGRAPI.2015.21