ADVANCES IN CYBER-PHYSICAL SYSTEMS

Vol. 2, No. 1, 2017

SURF FEATURES EXTRACTION IN A COMPUTER VISION SYSTEM

Volodymyr Puyda

Lviv Polytechnic National University, 12, S. Bandera str., Lviv, 79013, Ukraine
Author e-mail: *vpuyda@ukr.net*

Submitted on 19.06.2017

Abstract: **We consider a microcontroller implementation of an algorithm for extracting SURF features of an object in a video stream to be used in a specialized computer vision system.**

Key words: **computer vision, video stream, SURF features, image identification.**

INTRODUCTION

The problem of identifying objects by their images from a video stream often arises in the process of computer vision systems development. There are many methods of extracting features from images. Selecting a suitable method depends on the nature of identified objects and the specific features of computer vision system, scene complexity, technical requirements and available hardware.

PUBLICATIONS ANALYSIS

Many scientists and engineers work in the area of developing methods and algorithms of object detection and recognition by their video images, for instance, Herbert Bay, Tinne Tuytelaars, Luc Van Gool, Christopher Evans, David G.Lowe, S. Antoshchuk, P. Popovych, V. Dzhulii and others.

One of the most efficient algorithms applicable to object detecting and identifying is SURF (Speeded-Up Robust Features) [1]. SURF is used for extracting rotation- and scale-invariant features from images and describing them for the purpose of further decision making.

PROBLEM DESCRIPTION

An image 256x256-pixels in size is extracted from an input video stream: $I(i, j)$, $i \in \{0,...,255\}$, $j \in \{0,...,255\}$,. $I(i, j)$ is a monochrome image with 8-bit pixels.

Every point from the image I falls into one of the following groups: the points that belong to the object and the points that belong to the background:

$$\{(i, j)\}_{i=0,...,255,\ j=0,...255} = Obj \cup Backgr, \quad Obj \cap Backgr = \emptyset. \tag{1}$$

Fig. 1. The scene image

The problem under consideration consists in finding the object on the scene image, defining its features and describing them for identification purposes.

PROBLEM SOLUTION

In this paper, the firmware/hardware implementation of the SURF algorithm [1] for extracting and describing object features on scene images is considered.

Features extraction is based on using the Hessian matrix $H(f)$ of an image $f = f(x,y)$:

$$H(f) = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{pmatrix}. \tag{2}$$

In the discrete case, the function f is a convolution of an image I and the Gaussian kernel and the derivatives are computed by taking a convolution of I and derivatives of the kernel:

$$H(I,s) = \begin{pmatrix} L_{xx}(s) & L_{xy}(s) \\ L_{xy}(s) & L_{yy}(s) \end{pmatrix}. \tag{3}$$

Here, $L_{xx}(s)$ denotes the convolution of I and the derivative $\frac{\partial^2 g(s)}{\partial x^2}$, $L_{xy}(s)$ denotes the convolution of I and $\frac{\partial^2 g(s)}{\partial x \partial y}$, $L_{yy}(s)$ denotes the convolution of I and $\frac{\partial^2 g(s)}{\partial y^2}$ and $g(s) = g(x, y; s)$ stands for the Gaussian kernel with the parameter s :

$$g(x,y;s)=\frac{1}{2ps^2}e^{-\frac{x^2+y^2}{2s^2}}. \quad (4)$$

To decrease computation time, the SURF algorithm uses the approximation of derivatives $\frac{\partial^2 g(s)}{\partial x^2}$, $\frac{\partial^2 g(s)}{\partial x \partial y}$ and $\frac{\partial^2 g(s)}{\partial y^2}$ by rectangular kernels:

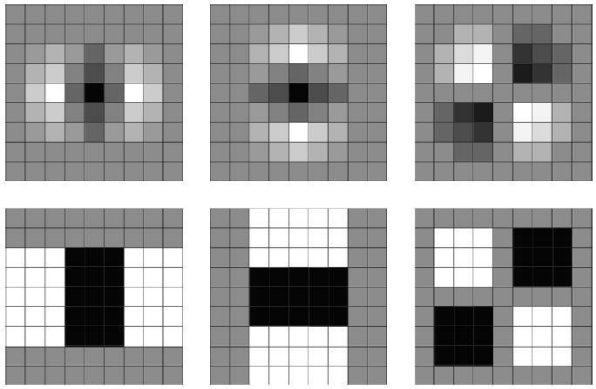


Fig. 2. Approximation of Gaussian kernel's derivatives by SURF

In the second row of Fig. 2 from the left to the right there are kernels D_{xx}, D_{yy} and D_{xy}. For D_{xx} and D_{yy}, the value of white points is -1, the value of black points is 2 and the value of grey points is 0; for D_{xy}, the value of white points is -1, the value of black points is 1 and the value of grey points is 0.

Using the above kernels, the determinant of the Hessian matrix $H(I,s)$ can be approximated using the formula

$$\det H(I,s)=D_{xx}D_{yy}-\left(0.9D_{xy}\right)^2 \quad (5)$$

Here, D_{xx}, D_{yy} and D_{xy} denote convolutions of the image I and the corresponding kernels (see Fig. 2). For a given value of s, the local maximums of $\det H(I,s)$ correspond to the features on the image I. The point (x,y) is a feature point of the image I if there is a local maximum of $\det H(I,s)$ over x, y and s at this point. The coordinates of features are found with sub-pixel accuracy using the expansion of $\det H(I,s)$ into Taylor series.

To decrease computation time, SURF uses integral images to compute the convolutions D_{xx}, D_{yy} and D_{xy}. Given an image I, the integral image I_Σ is defined by the formula

$$I_\Sigma(x,y)=\sum_{i=0}^{x}\sum_{j=0}^{y} I(x,y). \quad (6)$$

Given the integral image I_Σ, it holds

$$\sum_{x=a}^{b}\sum_{y=c}^{d} I(x,y)=I_\Sigma(b,d)-I_\Sigma(a-1,d)- \\ -I_\Sigma(b,c-1)+I_\Sigma(a-1,c-1). \quad (7)$$

In the above formula, we set $I_\Sigma(x,y)=0$ for $x<0$ or $y<0$. This formula allows to compute the sum of image values inside any rectangle using at most three operations of addition and subtraction provided that the integral image I_Σ is computed.

To gain rotation invariance, orientation is assigned to each feature. Orientation assignment is achieved using the Haar kernels:

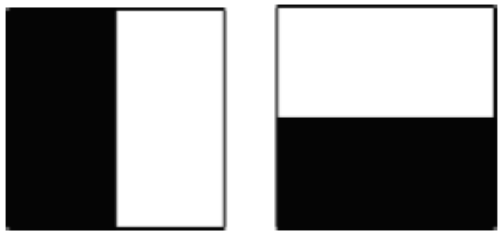


Fig. 3. Haar kernels for x and y directions

In Fig. 3 Haar kernels for *x* and *y* direction, the value of points -1 and the value of black points is 1.

To assign the orientation to a feature with the parameter s, convolutions of the image with Haar kernels of size $4s$ are computed in the circle of radius $6s$ centered at the same feature. The convolution values are multiplied by weight coefficients equal to the values of the Gaussian kernel with parameter $2.5s$ centered at the same feature. The obtained values are represented by two-dimensional points with abscissas equal to convolutions with Haar kernels for x direction and ordinates equal to convolutions with Haar kernels for y direction. For every sector with angle $p/3$ centered at the origin, a vector with abscissa equal to the sum of abscissas of all points contained in this sector and ordinate equal to the sum of ordinates of all points contained in the sector is computed. The orientation of the feature is set to the direction of the longest vector.

To describe a feature, a squared window of size $20s$ with the rotation equal to the orientation of the feature is constructed. This window is divided into 16 squares as shown in the following Fig. 4.

Inside every square, 25 uniformly distributed points are selected as shown in Fig. 4. At each of these points, convolutions of the image with Haar kernels of size $2s$ for x and y directions are computed. Let x_i and y_i, $i=1,...25$, denote the convolution values for each of the 25 points. Then each of the 16 squares has the corresponding vector

$$v_j=\left(\sum_{i=1}^{25}x_i,\sum_{i=1}^{25}y_i,\sum_{i=1}^{25}|x_i|,\sum_{i=1}^{25}|y_i|\right), \quad (8)$$

$j=1,...16$. The vectors v_j, $j=1,...16$, form a 64-dimentional vector describing the feature.

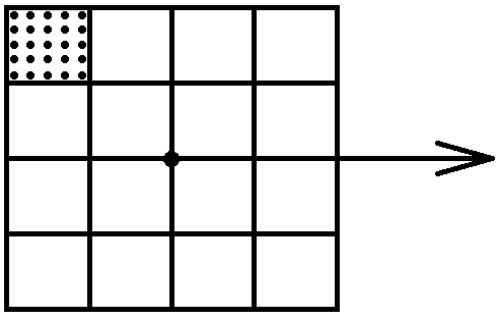


Fig. 4. Constructing a feature descriptor

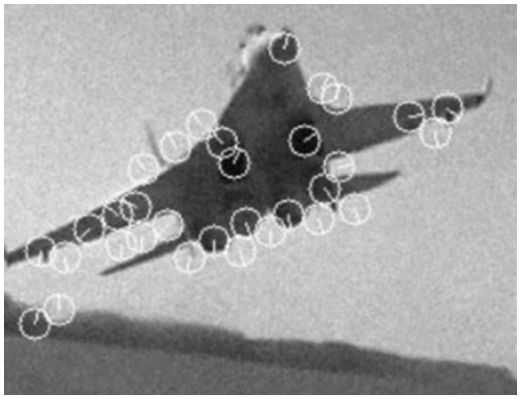


Fig. 5. The features of a 256x256-pixel image with a given value of s

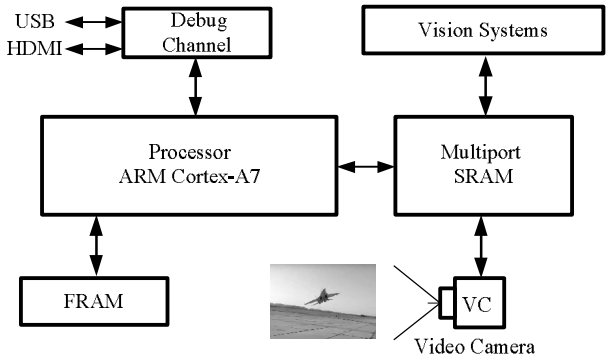


Fig. 6. Structural scheme of a special processor for feature extraction

Algorithm debugging was done on the board based on a four-kernel ARM Cortex A7 microprocessor with frequency 1.6GHz. Scene images are input to RAM from the 8-bit grayscale video camera through a special processor. The image size is 640x480 pixels. Using a grayscale camera allowed to minimize memory consumption and increase the speed of method implementation. From each video frame, a 256x256-pixel sub-image is selected for feature extraction. 1GB DDR3 RAM is used to store data during algorithm computations. FRAM is used to store algorithm parameters and template data. FRAM is based on 1MB FM25V10 microprocessors; it supports fast read/write operations through the SPI interface and can store data when the power is off. The debug channel includes USB and HDMI interfaces. USB2.0 is used for connecting system devices and uploading firmware to the processors. HDMI is used for displaying the debug data.

In the following figure, there is a structural scheme of a special processor for extracting features from 256×256 images using the SURF algorithm. The special processor is based on a four-kernel ARM Cortex A7 microprocessor.

A special processor for feature detection on complex scenes can be implemented using more powerful hardware, e.g. the eight-kernel TMS320C6678 with TI's KeyStone architecture or other similar processors.

CONCLUSIONS

The SURF algorithm for extracting features from grayscale images can be implemented using modern microprocessors in different computer vision systems. The average computation time for describing a 256x256-pixel image does not exceed the duration of one frame in PAL systems.

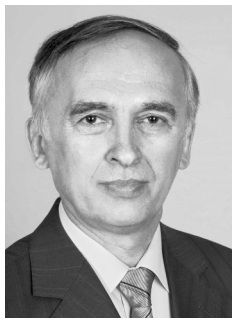
REFERENCES

[1] H. Baya, A. Essa, T. Tuytelaarsb, L. Van Goola, Speeded-Up Robust Features (SURF), Computer Vision and Image Understanding 110 (2008), no. 3, 346–359.

[2] V. Puyda. Implementation of the algorithm for finding object coordinates in a grayscale video stream. Visnyk of Lviv Polytechnic National University "Computer Systems and Networks", 2015, No. 830, 136-140.

[3] Pat. 52535 U Ukraine, G 06 K 9/00. A method for automatic identification of objects by their contours / V. Puyda and M. Oleksiv; patent applicant and patent owner "Lviv Polytechnic National University". – № u201003306, 25.08.2010, Bul. № 16.

[4] M. Oleksiv and V. Puyda A computer vision system for plane identification based on neural networks. Visnyk of Lviv Polytechnic National University "Computer Systems and Networks", 2009, No. 638, 61-64



Volodymyr Puyda, Ph. D., Associate Professor at Computer Engineering Department at Lviv Polytechnic National University. In 1975–1984, he was developing hardware and algorithms for real-time image recognition in computer vision systems. In 1985–1990, he developed and supported serial production of PC01 "Lviv". Also, at that time he was developing different process and control systems based on special processors and personal computers. Since 1991, he has been developing firmware and hardware for automated tracking systems, firmware and hardware for DSP-based data processing and control systems, firmware and hardware for measurement systems.