НЕЙРОІНФОРМАТИКА ТА ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ

НЕЙРОИНФОРМАТИКА И ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ

NEUROINFORMATICS AND INTELLIGENT SYSTEMS

UDC 004.9

Romanuke V. V.

D. Sc., Professor of Khmelnitskiy National University, Ukraine

PARAMETRIZATION OF THE OPTICAL FLOW CAR TRACKER WITHIN MATLAB COMPUTER VISION SYSTEM TOOLBOX FOR VISUAL STATISTICAL SURVEILLANCE OF ONE-DIRECTION ROAD TRAFFIC

A computer vision problem is considered. The prototype is the optical flow car tracker within MATLAB Computer Vision System Toolbox, tracking cars in one-direction road traffic. For adapting the tracker to work with other problems of moving cars stationary-camera-detection, having different properties (video length, resolution, velocity of those cars, camera disposition, prospect), it is parametrized. Altogether there are 19 parameters in the created MATLAB function, fulfilling the tracking. Eight of them are influential regarding the tracking results. Thus, these influential parameters are ranked into a nonstrict order by the testing-experience-based criterion, where other videos are used. The preference means that the parameter shall be varied above all the rest to the right side of the ranking order. The scope of the developed MATLAB tool is unbounded when objects of interest move near-perpendicularly and camera is stationary. For cases when camera is vibrating or unfixed, the parametrized tracker can fit itself if vibrations are not wide. Under those restrictions, the tracker is effective for visual statistical surveillance of one-direction road traffic.

Keywords: computer vision, optical flow, one-direction road traffic, car tracker, MATLAB function parametrization, visual statistical surveillance.

NOMENCLATURE

CAMS is a continuously adaptive mean shift;

CVST is Computer Vision System Toolbox™;

KLT is Kanade-Lucas-Tomasi;

MCSCD is a moving cars stationary-camera-detection;

MNF are motion numerical features;

OFCT is an optical flow car tracker;

VSS is visual statistical surveillance;

A is an algorithm used to compute optical flow;

a_{blob} is a ratio between area of the blob and area of the bounding box;

$b_{\max}$ is a maximum blob area in pixels;

$b_{\min}$ is a minimum blob area in pixels;

b_{offset} is a border offset in plotting motion vectors;

$c_{\max}$ is a maximum number of blobs in the input image;

d_{col} is a step through horizontal axis, when coordinates are generated for plotting motion vectors;

d_{frame} is a number of frames between reference frame and current frame;

d_{line} is a distance between centers of structuring element members at opposite ends of the line;

d_{row} is a step through vertical axis, when coordinates are generated for plotting motion vectors;

F_0 is a set of OFCT attributes;

$\bar{F}$ is a narrowed and ranked set of relevant OFCT parameters;

g_{motion} is a motion vectors gain;

$N_{\text{moment}}(t)$ is a number of cars intersecting an appropriate region at a moment t;

$N_{\text{total}}(t)$ is a total number of cars intersected the line all over the video (first) frames;

r_{blob} is a marginal ratio in classifying the blob as a car;

r_{factor} is a frame scaling percentage;

T is a total number of frames;

t is a moment (a frame);

v_{th} is a velocity threshold, computed from the matrix of complex velocities;

w is a width (in pixels) of a square structuring element;

$x_{\max}$ is maximal deviation ratio of the bounding box through horizontal axis;

y_{end} is a value of ordinate in the frame, where tracking ends;

DOI 10.15588/1607-3274-2015-3-5

$y_{\max}$ is maximal deviation ratio of the bounding box through vertical axis;

y_{start} is a value of ordinate in the frame, where tracking starts;

α_{line} is an angle of the line as measured in a counterclockwise direction from the horizontal axis;

γ is a binary classification factor;

θ is a time unit.

INTRODUCTION

Computer vision is an inseparable and high-promising part of automation. This is a very huge scientific field, included methods for acquiring, processing, analyzing, and understanding multi-dimensional data from the real world in order to produce decisions as numerical or symbolic information [1, 2]. Particularly, these data are images and frames from video sequences, views from cameras, or plane projections from scanners. Computer vision efficiently uses utilities and facilities of applied mathematics, machine learning and artificial intelligence, image and signal processing [1, 3, 4]. Being scientific-technological discipline, computer vision renders its theories and models to the construction of computer vision systems. Such systems mainly are designed for controlling industrial processes, autonomous vehicle navigation, detecting events for VSS, organizing imaged and databased information, analyzing and modeling topographical environments, computer-human interaction [1, 5, 6].

The being described application areas employ a few contemporary general problems of computer vision, whose resolution depends on the application requirements and approaches in solving. Typically, these problems are recognition, motion analysis, scene reconstruction, image restoration. Computer vision system methods for solving them issue from multi-dimensional data acquisition, preprocessing, feature extraction, detection, segmentation, high-level processing. Eventually, the final decision required for the application is made.

Before computer vision system projection, using hardware (power sources, multi-dimensional data acquisition devices, processors, control and communication cables, wireless interconnectors, monitors, illuminators) anyway, its work must be modeled in order to heed of the application area unpredictable specificities. Up-to-date MATLAB® environment grants a powerful CVST, providing algorithms and tools for the design and simulation of computer vision and video processing systems [4, 7, 8]. CVST proposes a lot of MATLAB® functions, MATLAB System objects™, and Simulink® blocks for feature extraction, motion detection, object detection, object tracking, stereo vision, video processing, and video analysis. Its tools include video file input/output, video display, drawing graphics, and compositing. For rapid prototyping and embedded system design, CVST supports fixed-point arithmetic and C code generation. Also there are demos, showing advantages of CVST. Some of those demos are a good basis for projecting real computer vision systems. However, for doing that there sometimes are not enough evident parameters, whose values might have been adjusted for other tasks within the regarded computer vision problem class. One of the classes, demonstrated in CVST, is the optical flow object tracking [2, 3, 9, 10].

When studying methods of tracking the object and motion estimation, one of the key demos in CVST is OFCT. This demo tracks cars in a one-direction road traffic video by detecting motion using the optical flow methods [2, 11, 12]. These methods, trying to calculate the motion between two image frames which are taken at neighboring times at every voxel position, are based on local Taylor series approximations [2, 3, 13] of the image signal. They use partial derivatives with respect to the spatial and temporal coordinates. The cars are segmented from the background by thresholding the motion vector magnitudes. Then, blob analysis is used to identify the cars [1, 14, 15].

A blob is an image region in which some properties are constant or vary within a prescribed range of values. All the points in a blob can be considered in some sense to be similar to each other. Blob detection refers to mathematical methods that are aimed at detecting image regions that differ in properties, such as brightness or color, compared to areas surrounding those regions. Given some property of interest expressed as a function of position on the digital image, there are two main classes of blob detectors [14, 16, 17]. The first class is differential methods, which are based on derivatives of the function with respect to position. The second class is methods based on finding the local maxima and minima of the function.

CVST algorithms for video tracking are CAMS and KLT ones [2, 4, 18, 19]. CAMS uses a moving rectangular window that traverses the back projection of an object’s color histogram to track the location, size, and orientation of the object from frame to frame. KLT tracks a set of feature points from frame to frame and can be used in video stabilization, camera motion estimation, and object tracking applications. CVST also provides an extensible framework to track multiple objects in a video stream. It includes Kalman filtering to predict a physical object’s future location, reduce noise in the detected location, and help associate multiple objects with their corresponding tracks [2, 3, 19]. The Hungarian algorithm is for assigning object detections to tracks [20]. Blob analysis and foreground detection is used for moving object detection. Additionally, there are annotation capabilities to visualize object location and to add object label.

Motion estimation is the process of determining the movement of blocks between adjacent video frames. CVST provides a variety of motion estimation algorithms –optical flow, block matching, template matching. These algorithms create motion vectors, which relate to the whole image, blocks, arbitrary patches, or individual pixels [21, 22]. The evaluation metrics, for finding the best match in the block and template matching, includes particularly mean-square error principle [2, 3, 21, 23, 24].

OFCT within CVST shows how moving objects are detected with a stationary camera. In series of video frames, optical flow is calculated and detected motion is shown by overlaying the flow field on top of each frame. But OFCT takes the specified series of 121 video frames, and so this demo cannot be applied outright for other moving cars videos with different frames’ number or distinct frame size. Besides, OFCT didn’t offer a numerical feature of motion results in the video frames series, except instant calculation of objects intersecting an early horizontal line at a moment.

Therefore OFCT should be parametrized for getting some needful numerical features of motion results in the video frames series, and for resolving at least slightly different tasks of MCSCD.

1 PROBLEM STATEMENT

Our goal is to view and rank the clue parameters in OFCT for parametrizing it within MATLAB CVST, what is going to be adapted for working with other MCSCD problems, having different properties (video length, resolution, velocity of those cars, camera disposition, prospect). Nominally, from the given set F_0 of OFCT attributes, we must yield a set of relevant OFCT parameters, whereupon this set is narrowed and ranked to $\bar{F}$. Formally, this is a map $\rho(F_0)=\bar{F}$, ensuring true MNF of videos. Parametrization of OFCT within MATLAB CVST by adding the MNF will allow projecting a computer vision system for VSS of one-direction road traffic. This is very important problem in organizing and optimizing the road traffic for its safety.

The successive components of the said goal are the following. Firstly, there must be structuring and algorithmization of information processing stages when one-direction road traffic is video-analyzed. Then, for VSS, MNF at OFCT windows should be added. And, eventually, the parametrized OFCT is going to be tested on another video.

2 REVIEW OF THE LITERATURE

Structurally, video information processing is divided into four items [2, 3, 8]:

1) extraction of the foreground;

2) extraction and classification of moving objects;

3) tracking trajectories of the revealed objects;

4) recognition and description of objects-of-interest activity.

Conventionally, the video foreground is of moving objects or regions. So, extraction of the foreground consists in separating moving fragments of the view from the motionless ones. These ones, being stationary objects or regions, are background of the view. Accuracy at this stage predetermines whether a computer vision problem is going to be satisfactorily solved. And nearly the best accuracy in selecting moving objects can be ensured with the optical flow methods [2, 3, 9]. The foreground extraction stage predetermines also the requirements to computational resources that may be needed at the rest three stages.

At the second stage, the extracted foreground is segmented. Each segment is a compact region whose pixels move at approximately equal velocities. Before segmentation the image is filtered for reducing noise, including impulse noise [1, 4, 25, 26]. Median filter as nonlinear digital filtering technique is usually invoked for noise reduction, running through the image entry by entry, and replacing each entry with the median of neighboring entries [26, 27]. For removing image defects (non-compactness), morphological dilatation and erosion over segments are fulfilled [1]. Subsequently, contours of the selected segments become smoother and they contain minimal quantity of spaces (gaps) within the object. Then those segmented regions, being moving objects, are classified. The classification is rough, meaning that its result is the moving object's type – a man, a car, an animal, etc.

At the third stage, the revealed objects' trajectories are tracked. For the tracking fulfillment, the one-to-one correspondence between the revealed objects on successive frames must be determined.

Finally, there are recognition and description of the revealed-and-tracked objects. In particular, for a task of MCSCD, it is VSS. Here, major MNF are number $N_{\text{moment}}(t)$ of cars intersecting an appropriate region (for instance, horizontal line) at a moment t, and total number $N_{\text{total}}(t)$ of cars intersected the line all over the video (first) frames:

$$N_{\text{total}}(t)=\sum_{\tau=1}^{t}N_{\text{moment}}(\tau) \text{ by } t=\overline{1,T} \tag{1}$$

and the total number of frames T, where the moment t corresponds to the t-th video frame. The feature $N_{\text{moment}}(t)$ is the varying number of the currently surveyed cars. The feature (1) is the aggregate of the surveillance. And for frame frequency per a time unit θ (how many frames pass per second, minute, hour, etc) there can be counted the motion T-intensity

$$\lambda(T)=\frac{\theta\cdot N_{\text{total}}(T)}{T}=\frac{\theta\cdot\sum\limits_{t=1}^{T}N_{\text{moment}}(t)}{T}, \tag{2}$$

implying how many cars intersect the line on average in the time unit θ (second, minute, hour, etc).

3 MATERIALS AND METHODS

The video in MATLAB OFCT demo has resolution 120-by-160, where the one-direction road traffic runs approximately vertically. Stages of this video processing are the said above within the processing loop to track cars along the series of 121 video frames. Initially the optical flow estimates direction and speed of motion. The optical flow vectors are stored as complex numbers, and the velocity threshold from the matrix of complex velocities is computed. Then median filter removes speckle noise introduced during thresholding and segmentation. For thinning out the parts of the road and other unwanted objects and filling holes in the blobs, there are applied morphological erosion and closing methods. After that the blob analysis method estimates the area and bounding box of the blobs, filtering out objects which cannot be cars with binary classification factor

$$\gamma=\frac{\operatorname{sign}(a_{\text{blob}}-r_{\text{blob}})+1}{2}\cdot\operatorname{sign}(a_{\text{blob}}-r_{\text{blob}}) \tag{3}$$

by $r_{\text{blob}}=0.4$, where a_{blob} is the ratio between area of the blob and area of the bounding box, and r_{blob} is the marginal ratio in classifying the blob as a car. Due to (3), if $\gamma=1$ then the blob is the car, otherwise the blob is ignored. The tracked cars are drawn around with bounding boxes.

Having analyzed the MATLAB code of OFCT demo, there in Table 1 are its input parameters.

The original MATLAB OFCT demo displays just $N_{\text{moment}}(t)$ and no MNF are returned. There are created

CVST objects to display the original video, motion vector video, the thresholded video and the final result. $N_{\text{moment}}(t)$ is displayed in the viewer named «Results» (Figure 1) in its left upper corner.

As it is well seen the demo is not ideal – there is the missed front car, without box bounding it, and $N_{\text{moment}}(73)=2$ is returned instead of $N_{\text{moment}}(73)=3$.

Table 1 – Parameters in MATLAB OFCT demo

MATLAB OFCT parameter name	Assignment	Math symbol	Value	Restriction
filename	name of the file, containing video T frames from a stationary camera	–	viptraffic.avi	video file with the supported extension
ReferenceFrameDelay	number of frames between reference frame and current frame, when optical flow method is applied	d_{frame}	3	$d_{\text{frame}} \in \mathbb{N}$
–	distance between the centers of the structuring element members at opposite ends of the line, when a flat linear structuring element that is symmetric with respect to the neighborhood center is created	d_{line}	5	–
–	angle (in degrees) of the line as measured in a counterclockwise direction from the horizontal axis, when a flat linear structuring element that is symmetric with respect to the neighborhood center is created	α_{line}	45	–
MinimumBlobArea	minimum blob area in pixels	$b_{\min}$	250	$b_{\min} \in \mathbb{N}\bigcup\{0\}$
MaximumBlobArea	maximum blob area in pixels	$b_{\max}$	3600	$b_{\max} \geqslant b_{\min}$, $b_{\max} \in \mathbb{N}$
MaximumCount	maximum number of blobs in the input image	$c_{\max}$	80	$c_{\max} \in \mathbb{N}$
–	width (in pixels) of a square structuring element, when morphological erosion object is created for removing portions of the road and other unwanted objects	w	2	$w \in \mathbb{N}\bigcup\{0\}$
ResizeFactor	for percentage scaling of the frame	r_{factor}	100	$r_{\text{factor}} > 0$
lineRow	value of ordinate in the frame, where tracking starts	l_{row}	22	$l_{\text{row}} \in \mathbb{N}$, $l_{\text{row}} < V$ by video resolution $V \times H$
motionVecGain	for gaining motion vectors	g_{motion}	20	$g_{\text{motion}} > 0$
borderOffset	border offset in plotting motion vectors	b_{offset}	5	$b_{\text{offset}} \in$ $\in \min\{V, H\}$
decimFactorRow	step through vertical axis, when coordinates are generated for plotting motion vectors	d_{row}	5	$d_{\text{row}} \in \overline{1, V-1}$
decimFactorCol	step through horizontal axis, when coordinates are generated for plotting motion vectors	d_{col}	5	$d_{\text{col}} \in \overline{1, H-1}$
–	ratio between area of the blob and area of the bounding box, which is marginal in classifying the blob as a car	r_{blob}	0.4	$r_{\text{blob}} \in (0; 1)$
–	scales the velocity threshold, computed from the matrix of complex velocities	v_{th}	0.5	$v_{\text{th}} > 0$

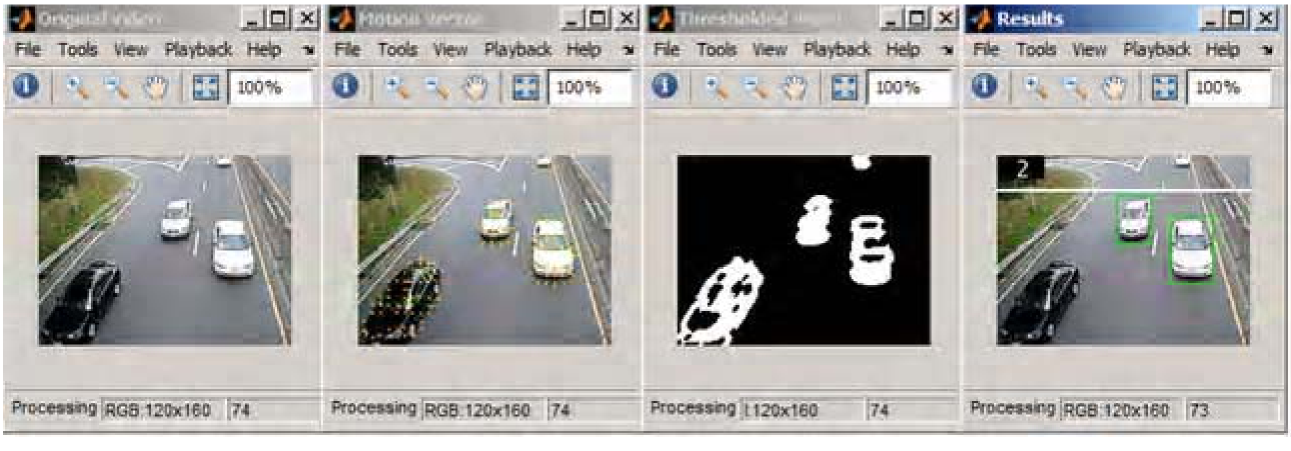

Original Video
File Tools View Playback Help
100%
Processing RGB 120x160 74
Motion Vector
File Tools View Playback Help
100%
Processing RGB 120x160 74
Thresholded Image
File Tools View Playback Help
100%
Processing 120x160 74
Results
File Tools View Playback Help
100%
2
Processing RGB 120x160 73

Figure 1 – Four viewers, visualizing the running MATLAB OFCT demo

Therefore the MATLAB OFCT demo may be modified over one or several parameters from the set

$$P = \{d_{\text{frame}}, d_{\text{line}}, \alpha_{\text{line}}, b_{\min}, b_{\max}, c_{\max}, w, \\ l_{\text{row}}, g_{\text{motion}}, b_{\text{offset}}, d_{\text{row}}, d_{\text{col}}, r_{\text{blob}}, v_{\text{th}}\} \quad (4)$$

of them. And for adding MNF at OFCT the new parameters $\{y_{\text{start}}, y_{\text{end}}\}$ are introduced, where $y_{\text{start}} = l_{\text{row}}$ and y_{end} is value of ordinate in the frame, whereupon tracking stops. Surely, $y_{\text{end}} > y_{\text{start}}$. Another two parameters being introduced are $\{x_{\max}, y_{\max}\}$ allowing to adjust the tracking to the car deviation through horizontal axis and vertical axis. Here $x_{\max}$ is maximal deviation ratio of the bounding box through horizontal axis, and $y_{\max}$ is maximal deviation ratio of the bounding box through vertical axis. Furthermore, the algorithm A used to compute optical flow ought to be specified. The available ones are algorithms of Horn –Schunck and of Lucas-Kanade. Thus, the set of parameters (Table 2)

$$P_* = \{A, y_{\text{start}}, y_{\text{end}}, x_{\max}, y_{\max}\} \quad (5)$$

is attached to the subset $P \setminus \{l_{\text{row}}\}$.

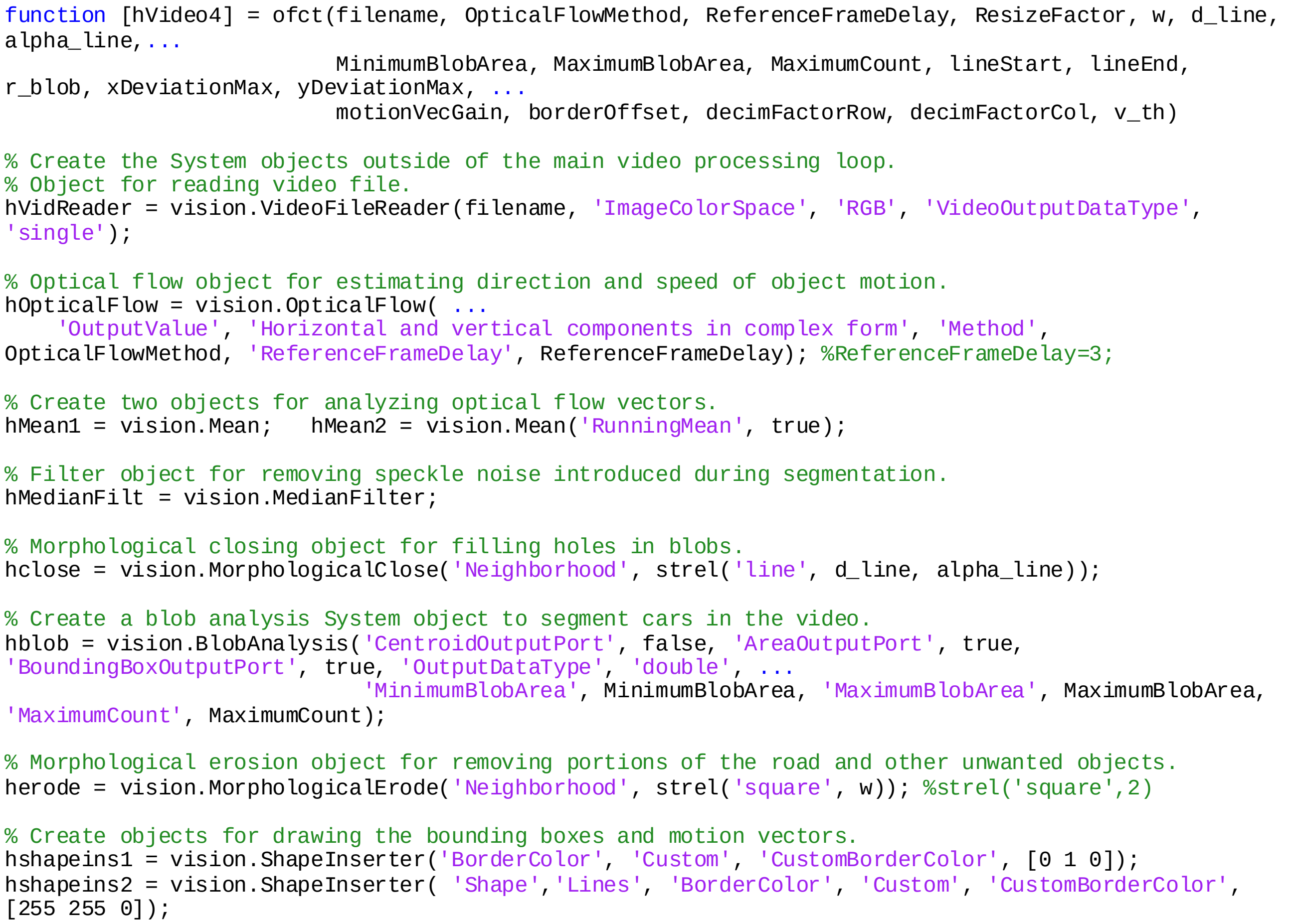
The modified OFCT code screenshot is in Figure 2. Having been made as a MATLAB function, it works with other MCSCD problems. But primarily it should to be adjusted to the problem, slightly changing 19 parameters (name of the file is not reckoned in) in sets (4) and (5). These parameters are input in the following order:

$$\{A, d_{\text{frame}}, r_{\text{factor}}, w, d_{\text{line}}, \alpha_{\text{line}}, b_{\min}, b_{\max}, c_{\max}, y_{\text{start}}, \\ y_{\text{end}}, r_{\text{blob}}, x_{\max}, y_{\max}, g_{\text{motion}}, b_{\text{offset}}, d_{\text{row}}, d_{\text{col}}, v_{\text{th}}\}$$

and the video file name is input at the front of them.

Table 2 – Adjustable parameters attached to the MATLAB OFCT demo

MATLAB OFCT parameter name	Assignment	Math symbol	Restriction
OpticalFlowMethod	algorithm used to compute optical flow	A	either algorithm of Horn-Schunck (string name «Horn-Schunck») or of Lucas-Kanade (string name «Lucas-Kanade»)
lineStart	value of ordinate in the frame, where tracking starts	y_{start}	$y_{\text{start}} \in \mathbb{N}$, $y_{\text{start}} < V$ by video resolution $V \times H$
lineEnd	value of ordinate in the frame, where tracking ends	y_{end}	$y_{\text{end}} \in \mathbb{N}$, $y_{\text{end}} > y_{\text{start}}$, $y_{\text{end}} < V$
xDeviationMax	maximal deviation ratio of the bounding box through horizontal axis	$x_{\max}$	$x_{\max} \in (0; 1)$
yDeviationMax	maximal deviation ratio of the bounding box through vertical axis	$y_{\max}$	$y_{\max} \in (0; 1)$



```
function [hVideo4] = ofct(filename, OpticalFlowMethod, ReferenceFrameDelay, ResizeFactor, w, d_line,
alpha_line,...
                          MinimumBlobArea, MaximumBlobArea, MaximumCount, lineStart, lineEnd,
r_blob, xDeviationMax, yDeviationMax, ...
                          motionVecGain, borderOffset, decimFactorRow, decimFactorCol, v_th)

% Create the System objects outside of the main video processing loop.
% Object for reading video file.
hVidReader = vision.VideoFileReader(filename, 'ImageColorSpace', 'RGB', 'VideoOutputDataType',
'single');

% Optical flow object for estimating direction and speed of object motion.
hOpticalFlow = vision.OpticalFlow( ...
    'OutputValue', 'Horizontal and vertical components in complex form', 'Method',
OpticalFlowMethod, 'ReferenceFrameDelay', ReferenceFrameDelay); %ReferenceFrameDelay=3;

% Create two objects for analyzing optical flow vectors.
hMean1 = vision.Mean;   hMean2 = vision.Mean('RunningMean', true);

% Filter object for removing speckle noise introduced during segmentation.
hMedianFilt = vision.MedianFilter;

% Morphological closing object for filling holes in blobs.
hclose = vision.MorphologicalClose('Neighborhood', strel('line', d_line, alpha_line));

% Create a blob analysis System object to segment cars in the video.
hblob = vision.BlobAnalysis('CentroidOutputPort', false, 'AreaOutputPort', true,
'BoundingBoxOutputPort', true, 'OutputDataType', 'double', ...
                            'MinimumBlobArea', MinimumBlobArea, 'MaximumBlobArea', MaximumBlobArea,
'MaximumCount', MaximumCount);

% Morphological erosion object for removing portions of the road and other unwanted objects.
herode = vision.MorphologicalErode('Neighborhood', strel('square', w)); %strel('square',2)

% Create objects for drawing the bounding boxes and motion vectors.
hshapeins1 = vision.ShapeInserter('BorderColor', 'Custom', 'CustomBorderColor', [0 1 0]);
hshapeins2 = vision.ShapeInserter( 'Shape','Lines', 'BorderColor', 'Custom', 'CustomBorderColor',
[255 255 0]);
```

Figure 2 – The modified MATLAB OFCT code, made as MATLAB function «ofct» with 19 adjustable parameters

The flaw concerning the 73-th frame of the video in original MATLAB OFCT demo is remedied by launching the modified MATLAB OFCT code as follows (from the MATLAB Command Window prompt):

```
ofct('viptraffic.avi', 'Horn-Schunck', 1,
  150, 2, 5, 45, 400, 3600, 40, 22, 130,
    0.35, 0.3, 0.7, 20, 5, 5, 5, 0.5).
```

Now the previously missed front car is captured and $N_{\mathrm{moment}}(73) = 3$ is returned as well (Figure 3). And there are no missed or defectively-tracked cars anymore.

Hereinafter, we will test the OFCT parametrized under (4) and (5) for seeing how MATLAB function "ofct" performs on other MCSCD problems for VSS of one-direction road traffic. But the part of empirical adjustment is omitted. The omission cause is that the adjustment is not routine.

4 EXPERIMENTS

For testing the parametrized OFCT, diverse videos containing one-direction road traffic have been explored. It is noteworthy that the road view is not always straight perpendicular. Figure 4 shows that cars are successfully tracked when they move non-perpendicularly having different velocities and accelerations. At that, the input arguments of the being invoked parametrized OFCT are just slightly different. For instance, there are invocations:

```
ofct('test1.avi', 'Horn-Schunck', 1, 100,
2, 5, 45, 200, 4000, 40, 22, 250, 0.5, 0.3,
          0.6, 20, 1, 1, 1, 0.1)
```

and

```
ofct('test2.avi', 'Horn-Schunck', 1, 100,
2, 5, 45, 200, 8000, 40, 22, 250, 0.3, 0.3,
          0.8, 20, 1, 1, 1, 0.4)
```

at the MATLAB Command Window prompt.

While testing, some parameters from the set $\{P \setminus \{l_{\mathrm{row}}\}\} \cup P_*$ are not varied at all. And variation of some parameters does not influence on the tracking visibly. Particularly, algorithm of Lucas-Kanade for computing optical flow is non-effective. Such non-influential parameters are in the set

$$\{A,\, d_{\mathrm{frame}},\, w,\, d_{\mathrm{line}},\, \alpha_{\mathrm{line}},\, c_{\max},\, g_{\mathrm{motion}},\, b_{\mathrm{offset}},\, d_{\mathrm{row}},\, d_{\mathrm{col}},\, v_{\mathrm{th}}\}.$$

On the contrary, values of parameters

$$\{r_{\mathrm{factor}},\, b_{\min},\, b_{\max},\, y_{\mathrm{start}},\, y_{\mathrm{end}},\, r_{\mathrm{blob}},\, x_{\max},\, y_{\max}\} \quad (6)$$

influence on the tracking much. Mostly, it is sensitive to r_{factor}, $b_{\min}$, $b_{\max}$, r_{blob}, $x_{\max}$, $y_{\max}$. Reasonable values of y_{start} and y_{end} depend rather than on video resolution and the road disposition.

5 RESULTS

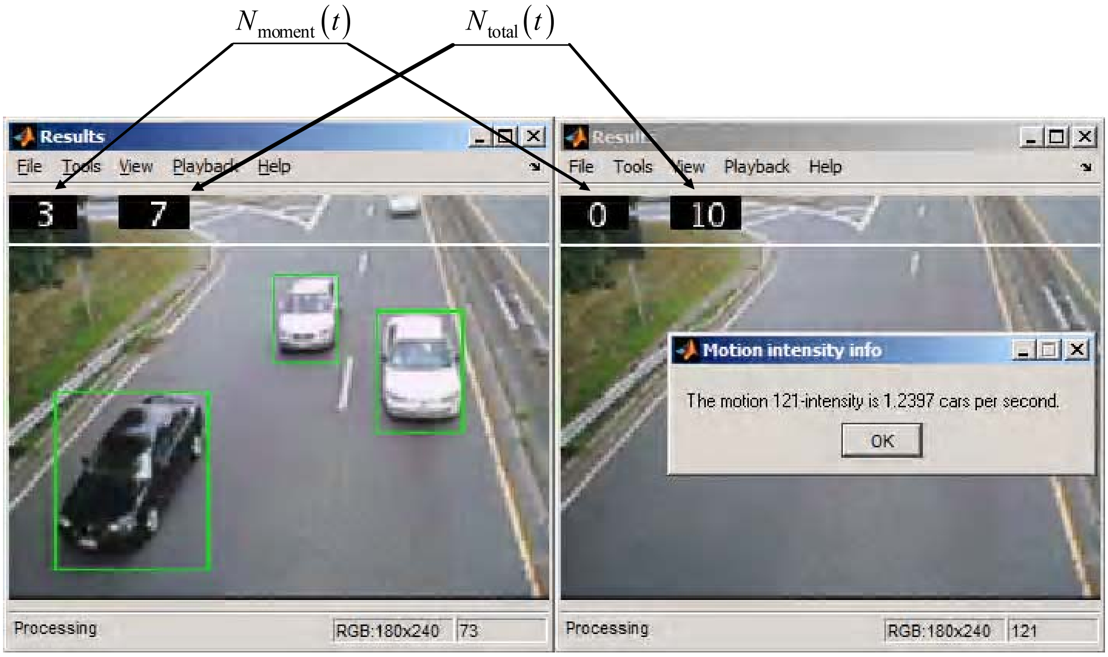
Guided by the testing experience, the influential parameters (6) are ranked as

$$(x_{\max} \sim y_{\max}) \succ (b_{\min} \sim b_{\max}) \succ r_{\mathrm{blob}} \succ r_{\mathrm{factor}} \succ (y_{\mathrm{start}} \sim y_{\mathrm{end}}) \quad (7)$$

pointing to their importance. The nonstrict order (7) means that the couple $\{x_{\max}, y_{\max}\}$ should (is advised to) be adjusted first. Then goes the couple $\{b_{\min}, b_{\max}\}$ defining the size of blob area. Naturally, the marginal ratio r_{blob} in (3) for classifying the blob as a car is near to $\{b_{\min}, b_{\max}\}$. The frame scaling r_{factor} is more important to set adjusted before putting $\{y_{\mathrm{start}}, y_{\mathrm{end}}\}$ because is global-like.

6 DISCUSSION

The parametrized OFCT should be adjusted for every new MCSCD problem. Fundamentally, the scope of this MATLAB tool is unbounded when objects of interest move



The motion 121-intensity is 1.2397 cars per second.

Figure 3 – The fourth viewer «Results», visualizing MNF $N_{\mathrm{moment}}(t)$ and (1) by the message box with (2) for MCSCD problem on video «viptraffic.avi» from MATLAB OFCT demo

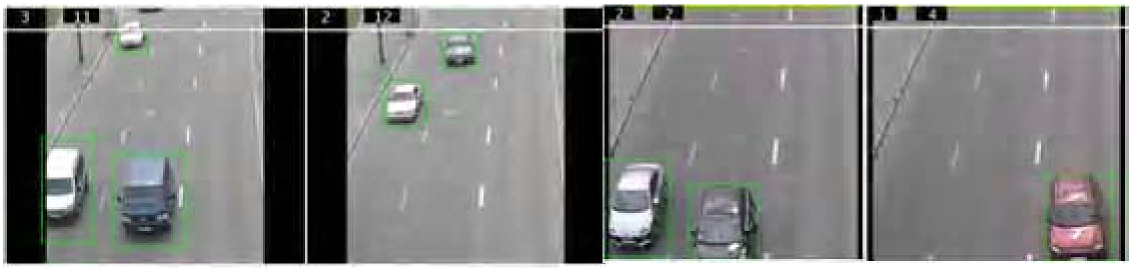


Figure 4 – Snapshots off the viewer «Results», visualizing the running OFCT by MATLAB function «ofct» on other MCSCD problems for VSS of one-direction road traffic

near-perpendicularly (but not horizontally or close to that) and camera is stationary. Nonetheless, adjustment even on the foremost couple $\{x_{\max}, y_{\max}\}$ by (7) may take substantial time. And tracking any vehicles is handled harder.

VSS of bigger vehicles causes a new problem. Trucks having long trailers may be split into a few blobs and thus the big long vehicle is tracked as two or more. Another great problem is that the rectangular bounding box sometimes disappears for a frame or two and then appears again. This effect may cause a fail of MNF calculation.

For cases when camera is vibrating or unfixed, the parametrized OFCT can fit itself if vibrations are not wide. However, the influential parameters (6) and their order (7) may become incomplete. Wider ranges of the camera vibration will require either to re-rank the elements of the set (6) or to re-select influential parameters from the set $\{P\backslash\{l_{\text{row}}\}\}\cup P_*$. The frame scaling r_{factor} will probably become crucial exceeding both r_{blob} and $\{b_{\min}, b_{\max}\}$.

CONCLUSIONS

The testing-experience-based criterion of ranking OFCT parameters has allowed to reduce the set of 19 non-arranged elements down to eight ones (6) ordered as (7). The advantage (preference) means that the parameter (of the couple) shall be varied above all the rest (to the right side of the ranking order). At that, there is no preference inside of the couples $\{x_{\max}, y_{\max}\}$, $\{b_{\min}, b_{\max}\}$, $\{y_{\text{start}}, y_{\text{end}}\}$, and their elements are likely to be varied simultaneously.

The adjustment is that naive heuristic optimization of values in (6) giving true $\{N_{\text{moment}}(t)\}_{t=1}^{T}$ and MNF (1) for an MCSCD problem. This is possible owing to parametrization of the OFCT within MATLAB CVST whose corollary is the ranking (7). Consequently, MCSCD problems are solved via MATLAB function «ofct». These primitives are indispensable to projecting a computer vision system for VSS of one-direction road traffic and ensuring its safety.

For general VSS of one-direction road traffic, the parametrized MATLAB function «ofct» is not going to be used straight off. The explanation lies in that the objects-of-interest activity is the cars' movement which must be almost perpendicular, and camera disposition ought to hang over the road (hanging not low). Hence, the promising research is in adapting the developed MATLAB tool for tracking vehicles of any form and size, moving in one-direction under arbitrarily disposed camera.

ACKNOWLEDGEMENTS

The work is technically supported by the Parallel Computing Center at Khmelnitskiy National University (http://parallelcompute.sourceforge.net).

REFERENCES

1. Parker J. R. Algorithms for Image Processing and Computer Vision / J. R. Parker. – Indianapolis : Wiley, 2011. – 480 p.
2. Klette R. Concise Computer Vision. An Introduction into Theory and Algorithms / R. Klette. – London : Springer, 2014. – 429 p.
3. Forsyth D. A. Computer Vision. A Modern Approach / D. A. Forsyth, J. Ponce. – New Jersey : Pearson, 2012. – 761 p.
4. Sonka M. Image Processing, Analysis, and Machine Vision / M. Sonka, V. Hlavac, R. Boyle. – Toronto : Thomson, 2008. – 829 p.
5. Mohr J. A computer vision system for rapid search inspired by surface-based attention mechanisms from human perception / J. Mohr, J.-H. Park, K. Obermayer // Neural Networks. – 2014. – Vol. 60. – P. 182–193. DOI: http://dx.doi.org/10.1016/j.neunet.2014.08.010
6. Cyganek B. Hybrid computer vision system for drivers' eye recognition and fatigue monitoring / B. Cyganek, S. Gruszczycski // Neurocomputing. – 2014. – Vol. 126. – P. 78 – 94. DOI: http://dx.doi.org/10.1016/j.neucom.2013.01.048
7. Park M.-W. Construction worker detection in video frames for initializing vision trackers / M.-W. Park, I. Brilakis // Automation in Construction. – 2012. – Vol. 28. – P. 15–25. DOI: http://dx.doi.org/10.1016/j.autcon.2012.06.001
8. Balasubramanian A. Utilization of Robust Video Processing Techniques to Aid Efficient Object Detection and Tracking / A. Balasubramanian, S. Kamate, N. Yilmazer // Procedia Computer Science. – 2014. – Vol. 36. – P. 579 – 586. DOI: http://dx.doi.org/10.1016/j.procs.2014.09.057
9. Bhattacharyya S. High-speed target tracking by fuzzy hostility-induced segmentation of optical flow field / S. Bhattacharyya, U. Maulik, P. Dutta // Applied Soft Computing. – 2009. – Vol. 9, Issue 1. – P. 126–134. DOI: http://dx.doi.org/10.1016/j.asoc.2008.03.012
10. Cortical surface shift estimation using stereovision and optical flow motion tracking via projection image registration / [S. Ji, X. Fan, D. W. Roberts, A. Hartov, K. D. Paulsen] // Medical Image Analysis. – 2014. – Vol. 18, Issue 7. – P. 1169–1183. DOI: http://dx.doi.org/10.1016/j.media.2014.07.001
11. A self-adaptive optical flow method for the moving object detection in the video sequences / [Y. Xin, J. Hou, L. Dong, L. Ding] // Optik – International Journal for Light and Electron Optics. – 2014. – Vol. 125, Issue 19. – P. 5690–5694. DOI: http://dx.doi.org/10.1016/j.ijleo.2014.06.092
12. Xiong J.-Y. An Improved Optical Flow Method for Image Registration with Large-scale Movements / J.-Y. Xiong, Y.-P. Luo, G.-R. Tang // Acta Automatica Sinica. – 2008. – Vol. 34, Issue 7. – P. 760 – 764. DOI: http://dx.doi.org/10.3724/SP.J.1004.2008.00760

13. Wolfinger R. D. Two Taylor-series approximation methods for nonlinear mixed models / R. D. Wolfinger, X. Lin // Computational Statistics & Data Analysis. – 1997. – Vol. 25, Issue 4. – P. 465 – 490. DOI: http://dx.doi.org/10.1016/S0167-9473(97)00012-1
14. Automated detection of microaneurysms using scale-adapted blob analysis and semi-supervised learning / [K. M. Adal, D. Sidibé, S. Ali, E. Chaum, T. P. Karnowski, F. Mériaudeau] // Computer Methods and Programs in Biomedicine. – 2014. – Vol. 114, Issue 1. – P. 1–10. DOI: http://dx.doi.org/10.1016/j.cmpb.2013.12.009
15. Leitner R. Real-time classification of polymers with NIR spectral imaging and blob analysis / R. Leitner, H. Mairer, A. Kercek // Real-Time Imaging. – 2003. – Vol. 9, Issue 4. – P. 245–251. DOI: http://dx.doi.org/10.1016/j.rti.2003.09.016
16. Ferraz L. A sparse curvature-based detector of affine invariant blobs / L. Ferraz, X. Binefa // Computer Vision and Image Understanding. – 2012. – Vol. 116, Issue 4. – P. 524–537. DOI: http://dx.doi.org/10.1016/j.cviu.2011.12.002
17. Sclaroff S. Active blobs: region-based, deformable appearance models / S. Sclaroff, J. Isidoro // Computer Vision and Image Understanding. – 2003. – Vol. 89, Issues 2–3. – P. 197–225. DOI: http://dx.doi.org/10.1016/S1077-3142(03)00003-1
18. Mean shift based gradient vector flow for image segmentation / [H. Zhou, X. Li, G. Schaefer, M. E. Celebi, P. Miller] // Computer Vision and Image Understanding. – 2013. – Vol. 117, Issue 9. – P. 1004–1016. DOI: http://dx.doi.org/10.1016/j.cviu.2012.11.015
19. Bagherpour P. Upper Body Tracking Using KLT and Kalman Filter / P. Bagherpour, S. A. Cheraghi, M. bin Mohd Mokji // Procedia Computer Science. – 2012. – Vol. 13. – P. 185–191. DOI: http://dx.doi.org/10.1016/j.procs.2012.09.127
20. Jonker R. Improving the Hungarian assignment algorithm / R. Jonker, T. Volgenant // Operations Research Letters. – 1986. – Vol. 5, Issue 4. – P. 171–175. DOI: http://dx.doi.org/10.1016/0167-6377(86)90073-8
21. A Comparison of Different Block Matching Algorithms for Motion Estimation / [R. Yaakob, A. Aryanfar, A. A. Halin, N. Sulaiman] // Procedia Technology. – 2013. – Vol. 11. – P. 199–205. DOI: http://dx.doi.org/10.1016/j.protcy.2013.12.181
22. Park C.-S. Level-set-based motion estimation algorithm for multiple reference frame motion estimation / C.-S. Park // Journal of Visual Communication and Image Representation. – 2013. – Vol. 24, Issue 8. – P. 1269–1275. DOI: http://dx.doi.org/10.1016/j.jvcir.2013.08.008
23. Frasca P. On the mean square error of randomized averaging algorithms / P. Frasca, J. M. Hendrickx // Automatica. – 2013. – Vol. 49, Issue 8. – P. 2496–2501. DOI: http://dx.doi.org/10.1016/j.automatica.2013.04.035
24. Perez F. L. An improved mean-square weight deviation-proportionate gain algorithm based on error autocorrelation / F. L. Perez, F. das Chagas de Souza, R. Seara // Signal Processing. – 2014. – Vol. 94. – P. 503–513. DOI: http://dx.doi.org/10.1016/j.sigpro.2013.06.030
25. Qiu Z. A new feature-preserving nonlinear anisotropic diffusion for denoising images containing blobs and ridges / Z. Qiu, L. Yang, W. Lu // Pattern Recognition Letters. – 2012. – Vol. 33, Issue 3. – P. 319–330. DOI: http://dx.doi.org/10.1016/j.patrec.2011.11.001
26. Toprak A. Impulse noise reduction in medical images with the use of switch mode fuzzy adaptive median filter / A. Toprak, İ. Güler // Digital Signal Processing. – 2007. – Vol. 17, Issue 4. – P. 711–723. DOI: http://dx.doi.org/10.1016/j.dsp.2006.11.008
27. Sebastiani G. A Bayesian approach for the median filter in image processing / G. Sebastiani, S. Stramaglia // Signal Processing. – 1997. – Vol. 62, Issue 3. – P. 303–309. DOI: http://dx.doi.org/10.1016/S0165-1684(97)00131-X

Article was submitted 26.01.2015.
After revision 02.02.2015.

Романюк В. В.
Д-р техн. наук, профессор, Хмельницкий национальный университет, Украина

ПАРАМЕТРИЗАЦИЯ ОТСЛЕЖИВАТЕЛЯ ЛЕГКОВЫХ АВТОМОБИЛЕЙ ПО ОПТИЧЕСКОМУ ПОТОКУ В MATLAB COMPUTER VISION SYSTEM TOOLBOX ДЛЯ ВИЗУАЛЬНОГО СТАТИСТИЧЕСКОГО НАБЛЮДЕНИЯ ОДНОСТОРОННЕГО ДОРОЖНОГО ДВИЖЕНИЯ

Рассматривается задача компьютерного зрения. Прототипом является отслеживатель легковых автомобилей по оптическому потоку в MATLAB Computer Vision System Toolbox, который отслеживает автомобили в одностороннем дорожном движении. Для приспособления отслеживателя к работе с другими задачами обнаружения движущихся легковых автомобилей неподвижной камерой, имеющих разные параметры (длительность видео, разрешение, скорость легковых автомобилей, расположение камеры, обзор), он параметризируется. В созданной MATLAB-функции, выполняющей отслеживание, всего насчитывается 19 параметров. Восемь из них оказывают решающее влияние на результаты отслеживания. Эти влияющие параметры соответственно ранжируются в некий нестрогий порядок по критерию на основании опыта тестирования с использованием других видео. Предпочтение означает то, что параметр должен изменяться первым перед другими в правой части порядка ранжирования. Возможности разработанного MATLAB-средства неограниченны при условии, когда соответствующие объекты осуществляют движение, близкое к перпендикулярному, и камера является неподвижной. В случаях, когда камера вибрирует или не закреплена, параметризированный отслеживатель способен подстраиваться, когда диапазон вибраций незначителен. При этих ограничениях отслеживатель эффективен для визуального статистического наблюдения одностороннего дорожного движения.

Ключевые слова: компьютерное зрение, оптический поток, одностороннее дорожное движение, отслеживатель легковых автомобилей, параметризация MATLAB-функции, визуальное статистическое наблюдение.

Романюк В. В.
Д-р техн. наук, професор, Хмельницький національний університет, Україна

ПАРАМЕТРИЗАЦІЯ ВІДСТЕЖУВАЧА ЛЕГКОВИХ АВТОМОБІЛІВ ЗА ОПТИЧНИМ ПОТОКОМ У MATLAB COMPUTER VISION SYSTEM TOOLBOX ДЛЯ ВІЗУАЛЬНОГО СТАТИСТИЧНОГО СПОСТЕРЕЖЕННЯ ОДНОСТОРОННЬОГО ДОРОЖНЬОГО РУХУ

Розглядається задача комп'ютерного зору. Прототипом є відстежувач легкових автомобілів за оптичним потоком у MATLAB Computer Vision System Toolbox, що відстежує автомобілі в односторонньому дорожньому русі. Для пристосування відстежувача до роботи з іншими задачами виявлення легкових автомобілів у русі нерухомою камерою, що мають різні параметри (тривалість відео, роздільність, швидкість легкових автомобілів, розташування камери, огляд), він параметризується. У створеній MATLAB-функції, що виконує відстеження, всього налічується 19 параметрів. Вісім з них є вельми впливовими на результати відстежування. Ці впливові параметри відповідно ранжуються у деякий нестрогий порядок за критерієм на основі досвіду тестування з використанням інших відео. Перевага означає те, що параметр має змінюватись першим за решту у правій частині порядку ранжування. Можливості розробленого

MATLAB-засобу необмежені за умови, коли відповідні об'єкти здійснюють рух, близький до перпендикулярного, і камера є нерухомою. У випадках, коли камера вібрує або не закріплена, параметризований відстежувач здатен підлаштовуватись, якщо діапазон вібрацій є незначним. За цих обмежень відстежувач є ефективним для візуального статистичного спостереження одностороннього дорожнього руху.

Ключові слова: комп'ютерний зір, оптичний потік, односторонній дорожній рух, відстежувач легкових автомобілів, параметризація MATLAB-функції, візуальне статистичне спостереження.

REFERENCES

1. Parker J. R. Algorithms for Image Processing and Computer Vision. Second Edition, 2011, Wiley Publishing, Inc., 480 p.
2. Klette R. Concise Computer Vision: An Introduction into Theory and Algorithms, 2014, Springer, 429 p.
3. Forsyth D. A., Ponce J. Computer Vision. A Modern Approach. Second Edition, 2012, Pearson, 761 p.
4. Sonka M., Hlavac V., Boyle R. Image Processing, Analysis, and Machine Vision. Third Edition, 2008, Thomson, 829 p.
5. Mohr J., Park J.-H., Obermayer K. A computer vision system for rapid search inspired by surface-based attention mechanisms from human perception, *Neural Networks*, 2014, Vol. 60, pp. 182–193. http://dx.doi.org/10.1016/j.neunet.2014.08.010
6. Cyganek B., Gruszczycski S. Hybrid computer vision system for drivers' eye recognition and fatigue monitoring, *Neurocomputing*, 2014, Vol. 126, pp. 78–94. DOI: http://dx.doi.org/10.1016/j.neucom.2013.01.048
7. Park M.-W., Brilakis I. Construction worker detection in video frames for initializing vision trackers, *Automation in Construction*, 2012, Vol. 28, pp. 15–25. DOI: http://dx.doi.org/10.1016/j.autcon.2012.06.001
8. Balasubramanian A., Kamate S., Yilmazer N. Utilization of Robust Video Processing Techniques to Aid Efficient Object Detection and Tracking, *Procedia Computer Science*, 2014, Vol. 36, pp. 579–586. DOI: http://dx.doi.org/10.1016/j.procs.2014.09.057
9. Bhattacharyya S., Maulik U., Dutta P. High-speed target tracking by fuzzy hostility-induced segmentation of optical flow field, *Applied Soft Computing*, 2009, Vol. 9, Issue 1, pp. 126–134. DOI: http://dx.doi.org/10.1016/j.asoc.2008.03.012
10. Ji S., Fan X., Roberts D. W., Hartov A., Paulsen K. D. Cortical surface shift estimation using stereovision and optical flow motion tracking via projection image registration, *Medical Image Analysis*, 2014, Vol. 18, Issue 7, pp. 1169–1183. DOI: http://dx.doi.org/10.1016/j.media.2014.07.001
11. Xin Y., Hou J., Dong L., Ding L. A self-adaptive optical flow method for the moving object detection in the video sequences, *Optik – International Journal for Light and Electron Optics*, 2014, Vol. 125, Issue 19, pp. 5690–5694. DOI: http://dx.doi.org/10.1016/j.ijleo.2014.06.092
12. Xiong J.-Y., Luo Y.-P., Tang G.-R. An Improved Optical Flow Method for Image Registration with Large-scale Movements, *Acta Automatica Sinica*, 2008, Vol. 34, Issue 7, pp. 760–764. DOI: http://dx.doi.org/10.3724/SP.J.1004.2008.00760
13. Wolfinger R. D., Lin X. Two Taylor-series approximation methods for nonlinear mixed models, *Computational Statistics & Data Analysis*, 1997, Vol. 25, Issue 4, pp. 465–490. DOI: http://dx.doi.org/10.1016/S0167-9473(97)00012-1
14. Adal K. M., Sidibé D., Ali S., Chaum E., Karnowski T. P., Mériaudeau F. Automated detection of microaneurysms using scale-adapted blob analysis and semi-supervised learning, *Computer Methods and Programs in Biomedicine*, 2014, Vol. 114, Issue 1, pp. 1–10. DOI: http://dx.doi.org/10.1016/j.cmpb.2013.12.009
15. Leitner R., Mairer H., Kercek A. Real-time classification of polymers with NIR spectral imaging and blob analysis, *Real-Time Imaging*, 2003, Vol. 9, Issue 4, pp. 245–251. DOI: http://dx.doi.org/10.1016/j.rti.2003.09.016
16. Ferraz L., Binefa X. A sparse curvature-based detector of affine invariant blobs, *Computer Vision and Image Understanding*, 2012, Vol. 116, Issue 4, pp. 524–537. DOI: http://dx.doi.org/10.1016/j.cviu.2011.12.002
17. Sclaroff S., Isidoro J. Active blobs: region-based, deformable appearance models, *Computer Vision and Image Understanding*, 2003, Vol. 89, Issues 2–3, pp. 197–225. DOI: http://dx.doi.org/10.1016/S1077-3142(03)00003-1
18. Zhou H., Li X., Schaefer G., Celebi M. E., Miller P. Mean shift based gradient vector flow for image segmentation, *Computer Vision and Image Understanding*, 2013, Vol. 117, Issue 9, pp. 1004–1016. DOI: http://dx.doi.org/10.1016/j.cviu.2012.11.015
19. Bagherpour P., Cheraghi S. A., M. bin Mohd Mokji. Upper Body Tracking Using KLT and Kalman Filter, *Procedia Computer Science*, 2012, Vol. 13, pp. 185–191. DOI: http://dx.doi.org/10.1016/j.procs.2012.09.127
20. Jonker R., Volgenant T. Improving the Hungarian assignment algorithm, *Operations Research Letters*, 1986, Vol. 5, Issue 4, pp. 171–175. DOI: http://dx.doi.org/10.1016/0167-6377(86)90073-8
21. Yaakob R., Aryanfar A., Halin A. A., Sulaiman N. A Comparison of Different Block Matching Algorithms for Motion Estimation, *Procedia Technology*, 2013, Vol. 11, pp. 199–205. DOI: http://dx.doi.org/10.1016/j.protcy.2013.12.181
22. Park C.-S. Level-set-based motion estimation algorithm for multiple reference frame motion estimation, *Journal of Visual Communication and Image Representation*, 2013, Vol. 24, Issue 8, pp. 1269–1275. DOI: http://dx.doi.org/10.1016/j.jvcir.2013.08.008
23. Frasca P., Hendrickx J. M. On the mean square error of randomized averaging algorithms, *Automatica*, 2013, Vol. 49, Issue 8, pp. 2496–2501. DOI: http://dx.doi.org/10.1016/j.automatica.2013.04.035
24. Perez F. L., F. das Chagas de Souza, Seara R. An improved mean-square weight deviation-proportionate gain algorithm based on error autocorrelation, *Signal Processing*, 2014, Vol. 94, pp. 503–513. DOI: http://dx.doi.org/10.1016/j.sigpro.2013.06.030
25. Qiu Z., Yang L., Lu W. A new feature-preserving nonlinear anisotropic diffusion for denoising images containing blobs and ridges, *Pattern Recognition Letters*, 2012, Vol. 33, Issue 3, pp. 319–330. DOI: http://dx.doi.org/10.1016/j.patrec.2011.11.001
26. Toprak A., İ. Gü ler Impulse noise reduction in medical images with the use of switch mode fuzzy adaptive median filter, *Digital Signal Processing*, 2007, Vol. 17, Issue 4, pp. 711–723. DOI: http://dx.doi.org/10.1016/j.dsp.2006.11.008
27. Sebastiani G., Stramaglia S. A Bayesian approach for the median filter in image processing, *Signal Processing*, 1997, Vol. 62, Issue 3, pp. 303–309. DOI: http://dx.doi.org/10.1016/S0165-1684(97)00131-X