Real-time FPGA image processing and robotic navigation

Rosen P. Spirov, Neli S. Grancharova, Georgi A. Angelov

This article presents the FPGA vision and robotic system on Altera's De2 board. The object detection algorithm implemented in FPGA was based on feature detection and image filtering. A software-based algorithm was independently developed and examined in MATLAB to evaluate its performance and verify its effectiveness. Furthermore, to detect the identities of the objects in the current frame beginning from the identities of the objects in the previous frame, a graph matching algorithm is performed. The experimental data are processed with recurrent Kalman filter. Difficult to obtain reliably eliminate global movement as error correlation is present in the image frame and the inadequacy of the subject and the background helps to produce errors. Although the system can be furthered improved to obtain better results, overall the project was a success as it enabled any inputted face to be accurately detected and tracked.

Обработка на изображения с FPGA в реално време и навигация за робот (Росен П. Спиров, Нели С. Грънчарова, Георги А. Ангелов). *Тази статия представя FPGA навигационна роботизирана система на базата на борда Алтера DE2. Алгоритъмът за откриване обекти се изпълнява в FPGA и се базира на откриване на признаци и филтриране на изображението. Софтуерно-базираният алгоритъм е разработен за независимо изследване в MATLAB, за да се оцени и провери неговата ефективност. Освен това, за да се открият и идентифицират обекти в текущия кадър, се започва от признаци на обектите от предишния кадър и се съставя граф от алгоритъм за съвпадение. Експерименталните данни се обработват с рекурентен Калманов филтър. Трудността е да се получи надеждно премахване на глобалното движение като грешка при корелацията, която присъства в кадъра на изображението и неразличимостта на обекта и фона спомага за получаването на грешки. Въпреки, че системата може да бъде значително подобрявана, за да се получават по-добри резултати, като цяло проектът е успешен, тъй като дава възможност всеки въведен образ да бъде точно разпознат и проследен.*

I. Introduction

The goal of this article is to create an FPGA (Field Programmable Gate Arrays) system to detect and track an object in real time. The overall setup will include a Verilog and VHDL program, an Altera DE2 board, a camera, and a VGA monitor.

The object detection algorithm implemented here is based on feature detection and image filtering, as shown in fig.1. After the object region is detected, its location is determined by calculating the centroid of neighbouring feature pixels [1].

A software-based algorithm is independently developed and examined in MATLAB to evaluate its performance and verify its effectiveness. However, it was infeasible to implement the same algorithm in Verilog due to the limitations of the language. Hence, several stages of the algorithm were modified.

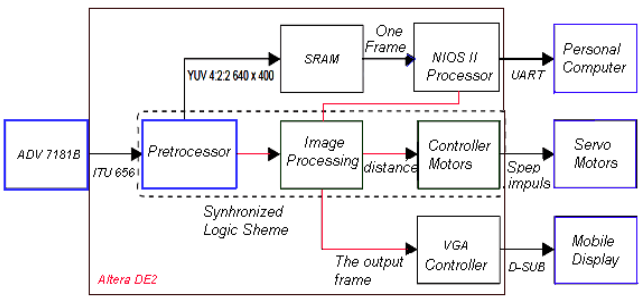

Fig.1. Block diagram of the system.

The experimental results proved the accuracy and effectiveness of the hardware real time

implementation as the algorithm is able to handle varying types of input video frame. The GPIO outputs of the board controlled by impulses IC drivers, switching power supply circuits of stepper motors, as shown in fig.2.

Fig.2. The experimental robot.

All calculation is performed in real time. Although the system can be further~~ed~~ improved to obtain better results, overall the project is a success as it enable~~d~~ any inputted face to be accurately detected and tracked.

II. Analysis

1. The method and the software algorithm

Different approaches to detect and track dynamic objects, including feature-based, appearance-based, and color-based have been actively researched and published in literature [2]. The feature-based approach detects a dynamic's object based on dynamic object features, such as human eyes and nose. Because of its complexity, this method requires lots of computing and memory resources. Although compared to other methods this one gives higher accuracy rate, it is not suitable for power-limited devices.

Hence, a color-based algorithm is more reasonable for applications that require low computational effort. In general, each method has its own advantages and disadvantages. More complex algorithm typically gives very high accuracy rate but also requires lots of computing resources [3]. General design stages are illustrated in next steps.

- ❖ First, the original image was converted to a different color space, namely modified YUV. Then the skin pixels were segmented based on the appropriate U range.
- ❖ Morphological filtering was applied to reduce false positives. Then each connected region of detected pixels in the image was labeled.
- ❖ The area of each labeled region was computed and an area-based filtering was applied.
- ❖ Only regions with large area were considered face regions.
- ❖ The centroid of each face region was also computed to show its location.

Converting the object pixel information to the modified YUV color space and the pixels can be segmented based on the following experimented threshold $10 < U < 64$ and $-48 < V < 16$.

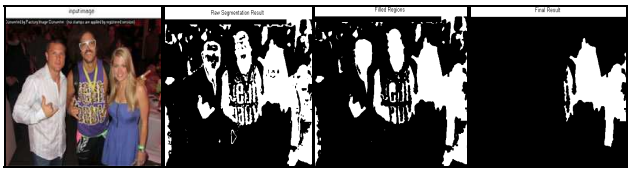

Fig.3. The MATLAB results.

As seen in fig.3, the blue channel had the least contribution to human skin color. Additionally, leaving out the blue channel would have little impact on thresholding and skin filtering. This also implies the insignificance of the V component in the YUV format. Therefore, the object detection algorithm using here was based on the U component only.

Applying the suggested threshold for the U component would produce a binary image with raw segmentation result.

The next step is applying morphological filtering including erosion and whole filling would, firstly reduce the background noise and secondly fill in missing pixels of the detected face regions [4].
The MATLAB provided built-in functions - imerode and imfill, for these two operations.

After each group of detected pixels became one connected region, connected component labeling algorithm was applied. This process labeled each connected region with a number, allowing us to distinguish between different detected regions. Filtering detected regions based on their areas would successfully remove all background noise and any skin region that was not likely to be an object. To be considered an object region, a connected group of skin pixels need to have an area of at least 26% of the largest area.

The final stage was to determine object location. The centroid of each connected labeled object region can be calculated by averaging the sum of X coordinates and Y coordinates separately. The centroid of each object region is denoted by the blue asterisk. Here the centroid of each connected region was extracted using region props

2. The Hardware Implementation Design

Each current video frame was captured by the camera and sent to the FPGA's decoder chip via a composite video cable. After the video signal was processed in different modules in Verilog, the final output passed through the VGA driver to be displayed on the VGA monitor. The hardware algorithm is shown in Fig.4 [8].

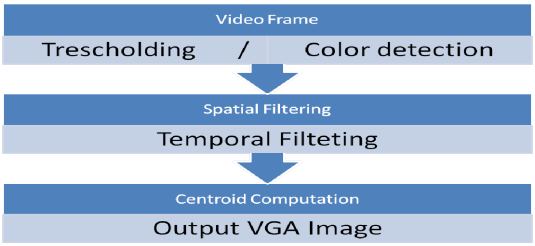

Fig.4. Hardware algorithm.

❖ Thresholding is in this step. Each input video frame was converted to a "binary image" showing the segmented raw result. Since 10-bit color was used in Verilog, adjusting the aforementioned U range yields $40 < U < 296$.

❖ Spatial Filtering is in this step. It is similar to the erosion operation used in the software algorithm. However, the structuring element used here did not have any particular shape. Instead, for every pixel *p*, their neighboring pixel in a 9x9 neighborhood is checked. If more than 75% of its neighbors are skin pixels, *p* is also a skin pixel. Otherwise *p* is a non-skin pixel. This allowed most background noise to be removed because usually noise scattered randomly through space. To examine the neighbors around a pixel, their values needed to be stored. Therefore, ten shift registers are created to buffer the values of ten consecutive rows in each frame. As seen in Fig.5, each register is 640-bit long to hold the binary values of 640 pixels in a row.

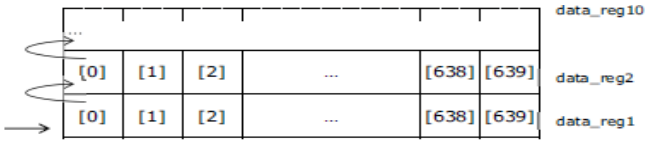

Fig.5. Ten shift registers for ten consecutive rows.

Each bit in reg1 is updated according to the X coordinate. The other registers are only updated when the X coordinate was 0. Values of reg2 to reg10 are used to examine a pixel's neighborhood.

❖ Applying temporal filtering allowed flickering to be reduced significantly. The idea of designing such a filter is borrowed from the project "Real-Time Cartoonifier" [4]. Even small changes in lighting could cause flickering and made the result displayed on the VGA screen less stable.

❖ Centroid computation. It is computed to locate the face region, because connected component labeling is not implemented as initially planned. It is infeasible to calculate the centroid for each face region separately. Although the pixels of one face region might not be connected and labeled as originally planned, simply calculating the centroid of all detected pixels still gave a good estimate for the face location. However, even if the hands are present, calculating the centroid of all detected pixels still allowed us to locate the face region. First the neighboring pixels around the centroid are checked to see if they are skin pixels. If they are, it meant the centroid accurately located the face region. However, if the neighboring pixels of the centroid are not skin pixels, it meant the centroid is somewhere in the background located between two detected face regions. To show how an object is tracked, a small box is drawn around the centroid. The box moved according to the movement of the object [5], as in fig.6.

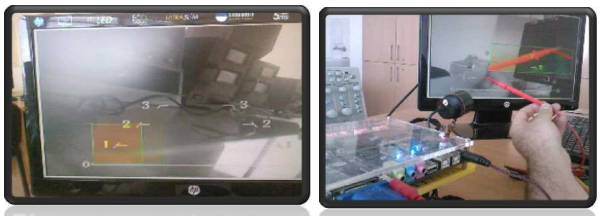

Fig.6. The object tracking.

However, if the object moved too fast, the movement of the box might become less stable. Applying temporal filtering here allowed the box to move smoothly. The implementation of the temporal filter here is slightly different from the one shown previously. The input *Xn* here is the location of the centroid before filtering. What this equation mean: ŧ with α being close to 1, current output *Yn* will be more dependent on previous output *Yn-1* than on current input. This prevented the centroid box from moving too fast when there is an abrupt change in the movement of an object, as:

$$Yn = (1 - \alpha)Xn + \alpha Yn\text{-}1$$

A clock of 27 MHz was used for the face detection and tracking algorithm. Since the timing is synchronized with the VGA clock, the VGA display is able to update within the time gap between drawing two consecutive frames [5].

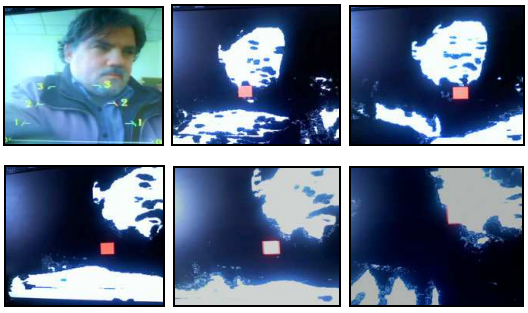

Fig.6. The FPGA result of track a moved man.

The camera is able to detect and track objects in real time. Error seemed to occur only when there is a transition from one person to two people or vice versa in the video frame. Within the lab setting, noise is very minimal and did not alter the results. As long as a person is in the camera's view, his face will be accurately detected and tracked.

III. The perspective mapping processing

The algorithm of mapping processing, based on graph pyramids [6], is shown on fig.8.

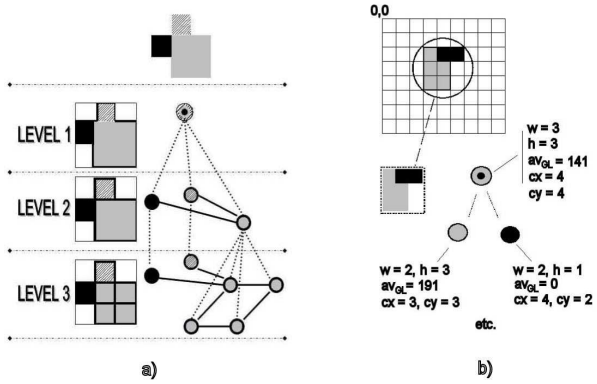

Fig.8. The graph pyramid: a) the hierarchical structure; b) the nodes attributes.

The objects are examined at different, increasing, resolutions until it is possible to separate the multiple regions in the objects belonging to the occlusion. Furthermore, to detect the identities of the objects in the current frame beginning from the identities of the objects in the previous frame, a graph matching algorithm is performed.

The basic idea is to provide the new incoming information into the current background image [7]. Therefore transformation from pixel coordinates to real world coordinates is required, as it is shown in fig.9.

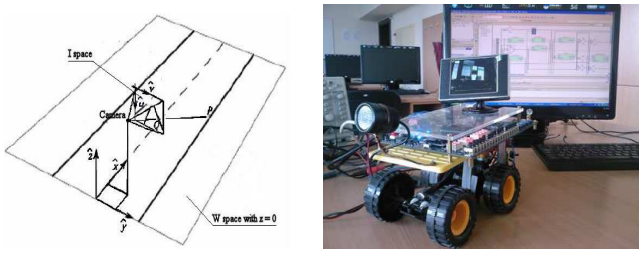

Fig.9. The Mapping Processing on FPGA Robot.

That is an inverse of each intermediate node. It represents a sub region of the whole region, and its attributes are the position and size of the sub region bounding box and average colour of that sub region.

IV. The Kalman filter implementation

The experimental data are processed with recurrent Kalman filter. The determination of the matrix *K (k+1)* requires the use of a matrix nxn, were *n* is the projects of vectors of measurements *z(k)*.

The important feature of the filter is represented in that the matrix *K(κ+1)* is not depend on the vectors of measurements *{z(1),...z(k+1)}* and the interference *{w(1),...,w(k+1)}*.

Therefore, it can be calculated with a priority and its elements to be stored until the start of the processing of information. Then, in the process of treatment with the vector of measurements *z(k+1)* it is calculated the matrix *K(k+1)* making a new assessment $\hat{x}(k+1)$. This enables to perform processing of another measurement in real time. To see the model of the movement of a dynamic object set with expressions (8) and (9) it is important to know that the process *{w(k), k=0,1,2,...}* is Gaussian white series with zero mathematical expectation and constant dispersion *Q*. And the process $\{\eta(K+1), K=0,1,...\}$ is white Gaussian series with ~~and~~ average zero dispersion R. Summary Kalman filter equations will acquire type:

$$\hat{x}(k+1) = E\hat{x}(k) + K(k+1)\left[z(k+1) - E\hat{x}(k)\right] \quad (1)$$

$$K(k+1) = \frac{EV_\varepsilon(k) + Q}{EV_\varepsilon(k) + Q + R} \quad (2)$$

$$V_\varepsilon(k+1|k) = EV_\varepsilon(k) + Q \quad (3)$$

$$V_\varepsilon(k+1) = \frac{R(EV_\varepsilon(k) + Q)}{EV_\varepsilon(k) + Q + R} \quad (4)$$

where the initial conditions is $\hat{x}(0) = m_x(0)$. They are assigned specific values of parameters such as: *E=[1]*, $V_\varepsilon(0) = 100$, *Q=25, R=15*.

The previous equations will change as follows:

$$V_\varepsilon(k+1|k) = V_\varepsilon(k) + 25 \quad (5)$$

$$K(k+1) = \frac{V_\varepsilon(k) + 25}{V_\varepsilon(k) + 40} \quad (6)$$

$$V_\varepsilon(k+1) = \frac{15.(V_\varepsilon(k) + 25)}{V_\varepsilon(k) + 40} = 15.K(k+1), \text{ к= } 0,1... \quad (7)$$

In table 1, the values of scalar variables for the first seven steps are given.

Table 1

к	Vt{k + l\k)	K(к + 1)	V(к + 1)
0		—	100.00
1	125.00	0.893	13.393
2	38.393	0.719	10.786
3	35.786	0.705	10,570
4	35.570	0,703	10.549
5	35.551	0.703	10,549
6	35.549	0.703	10,549
7	35.549	0.703	10,549

The last rows of the table show the existence of limits for these values. Limit of $\overline{V}_\varepsilon$ is obtained by assigning a value of $V_\varepsilon(k+1)=V_\varepsilon(k)=\overline{V}_\varepsilon$ in the last expression. Then:

$$V_\varepsilon(k)(V_\varepsilon(k)+40)=15V_\varepsilon(k)+375; \quad (8)$$

$$\overline{V}_\varepsilon^2+25.\overline{V}_\varepsilon-375=0 \quad (9)$$

The graphical data results are shown respectively as follows in fig.10:

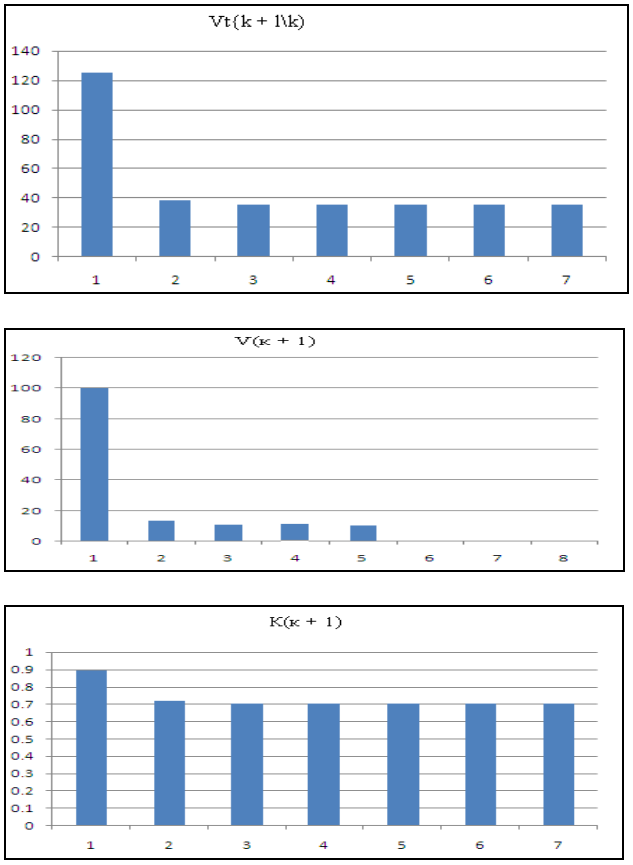

Fig.10. Graphical representation of data.

Since $\overline{V}_\varepsilon \geq 0$ is assumed that there is only one root of this quadratic equation, and that is $\overline{V}_\varepsilon = 10{,}55$. Then the estimated rate of transmission of the Kalman filter can be determined as follows:

$$\overline{K}=\frac{\overline{V}_\varepsilon+25}{\overline{V}_\varepsilon+40}=0{,}703 \quad (10)$$

Kalman filter is established in the state $k \geq 5$ and is described by the equation:

$$\hat{x}(k+1)=\hat{x}(k)+0.703\,(z(k+1)-\hat{x}(k)) \\ =0.297\,\hat{x}(k)+0.703\,z(k+1) \quad (11)$$

This equation illustrates very well the contribution to the "new" assessment $\hat{x}(k+1)$ in the "old" assessment $\hat{x}(k)$ and the new measurement $z(k+1)$. The error and deviation is calculated from the reference values and can be determined as follows:

$$\Delta x=\frac{1}{K}\sum_{i=1}^{K}\left|x_{експер}-x_{расч}\right| \quad (12)$$

where K - is the number of measurements. It is similarly determine Δy.

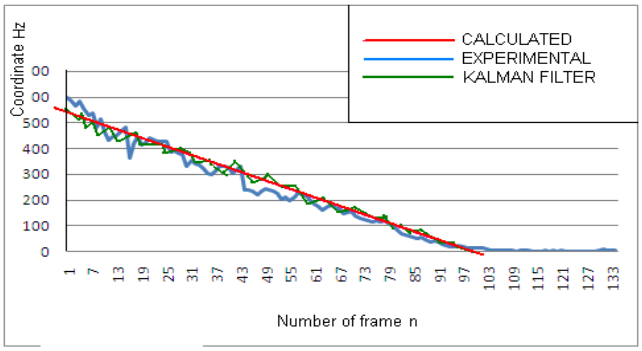

Fig.11. A result of determining the objects coordinates in the direction of the axis OX.

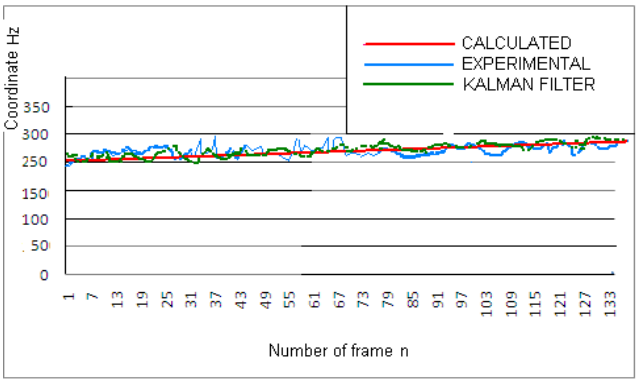

Fig.12. A result of determining the objects coordinates in the direction of axis OY.

The size of the dependence of the error in determining the coordinates of the center of the object is apparent from the graph on fig.12.

After treatment with a Kalman, the filter error in determining the coordinates of the center of the object in the medium is less than 7%.

It is possible to obtain even results, which show no movement, but it actually exists. Therefore, it must cluster of vectors to be high-density area of interest. Difficult to obtain reliably eliminate global movement as error correlation is present in the image frame and the inadequacy of the subject and the background helps to produce errors.

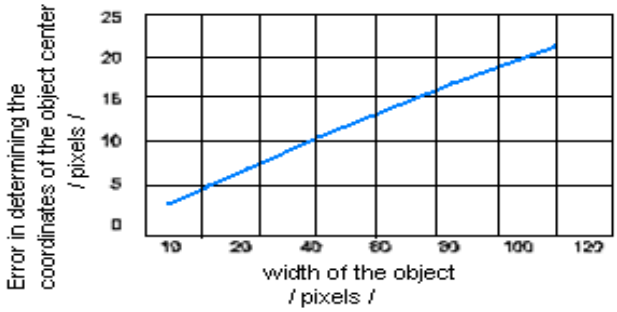

Fig.13. The error in determining the coordinates of the center of the site.

In the experiments of detection of dynamic object, which is defined as the ratio of the amount properly identified items to total shots in the sequence containing dynamic object, assessment of the probability of false alarm upon detection of dynamic object is calculated. That is defined as the ratio of the amount of occurrences of false objects to the total footage in the sequence containing dynamic object. The experimental results are illustrated in table 2.

Table 2

Experimental results

Probability of detection		False alarm probability	
statistical background	dynamic background	statistical background	dynamic background
0,98	0,98	0,06	0,26

Based on experimentally obtained results it is made perfect simulation of the Kalman filter in tools of Quartus II, shown in figure14.

The design of the Kalman filter loan is from FPGA Altera CycloneII EP2C35F672 [9] resource (only ALUT 4168 and 241 registers). The used general system resources are about 49% as represented in table 3.

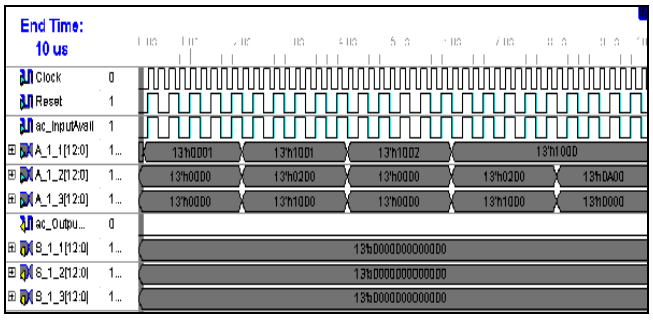

Fig.14. The simulation of Kalman filter architecture.

Table 3

Used FPGA resources by Kalman filter

Sr.No	Information	Count	% use
1	No of slice	2145 of 32640	7%
2	Slice LUTs	3626 of 32640	14%
3	Slice LUTs Used as logic	3626 of 32640	14%
4	LUT Flip-Flop pair use	4168	
5	LUT Flip-Fl. pairs	2023 of 4168	49%
6	LUT Flip-Flop pair	542 of 4168	17%
7	Fully used FF pairs	1603 of 4168	41%
8	Bonded IOBs	82 of 480	21%
9	DSP48Es	16 of 288	7%

In this work an FPGA system to detect and track an object in real time is created. In the overall setup the Verilog and VHDL program, the Altera DE2 board [9], the camera and the VGA monitor are included.

V. Conclusions

The experimental results proved the accuracy and effectiveness of the hardware in real time working. The master of this robot on board Altera De2 is R. Spirov Ph.D and all rights reserved. The implementation has shown that the algorithm is able to handle varying types of input video frame. Although the system can be furthered improved to obtain better results, overall the project was a success as it enabled any inputted face to be accurately detected and tracked.

This article was originally published in the Fourteenth International Scientific Conference on Electrical Machines, Drives and Power Systems (ELMA), Varna, Bulgaria, 2015.

REFERENCES

[1] Bertozzi, M., Fascioli, A. Vision-based intelligent vehicles: State of art and perspectives. Robotics and Autonomous Systems, 2000.

[2] Poupard, M. Road edge traking using filtering theory. Palaiseau Ecole, 2000.

[3] Sedgewick, R. Algorithms in C. Pt. 5, Graph algorithms. 3rd Edition Harlow: Addison-Wesley, 2002, 482 p.

[4] Kuhn, H.W. The Hungarian Method for the Assignment Problem. Naval Research Logistics Quarterly, 2002.

[5] Buxton, H. and S. G. Gong. Visual surveillance in a dynamic and uncertain world. Artif. Intell. vol. 78, 1995.

[6] Jolion, J.M., A. Montanvert. The Adaptive Pyramid: A Framework for 2D Image Analysis. CVGIP: Image Understanding, Academic Press. Vol. 55 – 3, 1992, pp. 339–348.

[7] Spirov, R. Practical Object Tracking System on FPGA. 5-th Intern. Conference on Communications, Electromagnetic and Medical Application - CEMA'2012, Athens, Greece, 2012, ISSN 1314-2100.

[8] Advanced Microcontroller Final Projects, online: http://people.ece.cornell.edu/land/courses/ece5760/FinalProjects.

[9] Quartus II Introduction Using VHDL Design www.altera.com

Dr. Rosen P. Spirov *– is Head of Specialized Laboratory of Electronic in Technical University of Varna, and is interested in FPGA, Image Processing and Computer Vision, Intellects System, Robotic and Navigation Electronic, etc.*

tel.:+35952383488 e-mail: rosexel@abv.bg

Eng. Neli S. Gryncharova *is Ph.D student in Technical University of Varna, and is interested in FPGA, Image Processing and Computer Vision, Intellects System, Robotic and Navigation Electronic, etc.*

tel.:+35952383488 e-mail: nelly2000@abv.bg

Eng. Georgy A. Angelov *is Ms student in Technical University of Varna, and is interested in FPGA, Robotics and Navigation Electronic, etc.*

tel.:+35952383488 e-mail: valledy@abv.bg

Received on: 19.12.2016