# REAL-TIME OBJECT'S CENTER CALCULATION 

Todor Stoyanov Djamiykov*, Nikolay Petrov Nenov*, Marin Berov Marinov*, Volker Zerbe**<br>*Technical University of Sofia, Department of Electronics, P.O. Box 43, BG-1756-Sofia, Bulgaria, e-mail: nenov@mail.bg<br>**Technical University of Ilmenau, Institute of Technical and Theoretical Computer Science, P.O. Box 100565, D-98684 Ilmenau, Germany, e-mail: zerbe @ theoinf.tu-ilmenau.de

Keywords: FPGA, real-time data processing, optical sensors
In the current paper is presented a modify algorithm for object's center calculation achieved from CMOS with resolution of $1024 x 768$ pixels and 10-bits digital output. Used hardware is based on FPGA family SPARTAN II by Xilinx capable of 100 MHz system operation speed.

Object's center calculation of digital image is wide used in many areas nowadays. This problem grows huge when there is a need for high data accuracy and calculation time is short. Contemporary state of programmable logic devices reveals many opportunities for real-time data processing. At the same time there was made right choice of algorithms that are embedded into programmable device.

In the paper are shown results from accuracy based on mathematical modeling of algorithm. In the same time algorithm implementation with VHDL and SPARTAN II shows great performance for frame center calculation by frame speed of 20ms. This speed in most cases is high. For practice applications is used average value of multiple frames in valuable working conditions of 1-2s.

## 1. INTRODUCTION

Finding a particular object and computing the coordinates of its center is widely used in many areas as localization, seismology, tuning and qualifying of optical systems, different production processes, etc. The object might be either a physical object or a light spot. The spot might be projected over a screen, which is observed with a video sensor or projected directly over the surface of the sensor. The independent systems for calculating object's center working in real time are the main point of interest.

The functional diagram of one of those systems is shown on fig.1. Its purpose is to measure the absolute coordinates X and Y of the object and then they are equaled to the relative coordinates $X^{\prime}$ and $Y^{\prime}$ of the measuring system. The object is observed through a lens which focuses the stage over the optical sensor and through a predefined algorithm is calculated if there is an object and with what coordinates is it.


Fig. 1

## 2. SOLUTIONS OF THE PROBLEM

The strange thing in the computation of Xc and Yc is that at the moment of reading them from the sensor, there should be a calculation if the data belongs to the object, and if they don't belong then an array should be reserved in the memory with the size of the image and in that array the data should be stored and later, the needed calculations to be done. There are two frequently used algorithms for computation of object's center coordinates:

Algorithm 1: In this algorithm the threshold of sensitivity Eth is used. Consecutive samples of the image are taken, and the first element with brightness higher than Eth shows the existence of an object, and this element is taken for the center of the object. The result of the algorithm for one-dimensional array is shown on fig. 2.


Fig. 2. Result of the algorithm for one-dimensional array
The first element with brightness higher than threshold Eth which will be taken for center of the object is Xc1, but the real center of the object, as seen on fig. 2, is Xc2. This algorithm is convenient when used in real time calculations and there is no need to store the entire frame. A drawback is that it is inaccurate. Another inconvenience is the fact that, there are production defective elements in the video sensor, which, when scanned return a maximum brightness. The first defective element will lead to wrong calculations of the coordinates.

Algorithm 2: In this algorithm the threshold of sensitivity Eth is used again. The computation of the center of the object involves all the elements with brightness higher than Eth. This algorithm is much more precise than the previous one but it requires more calculations and is not suitable for real time environments. The whole frame should be stored in memory. Then, Eth should be calculated for each element,
the elements belonging to the object put aside and at the end, the center of the object and its brightness will be computed using the following formulas:

$$
\begin{equation*}
X c=\frac{\sum_{i=1}^{p} X i E i}{\sum_{i=1}^{p} E i} \quad Y c=\frac{\sum_{i=1}^{p} Y i E i}{\sum_{i=1}^{p} E i} \quad E c=\frac{\sum_{i=1}^{p} E i}{p} \tag{1}
\end{equation*}
$$

where: Xc, Yc - coordinates of the object, Ec - brightness of the center of the object, p - number of elements in the object.
The information from then sensor is transmitted serial pixel by pixel till the whole line is scanned. Then the next line is scanned till the end of the frame. In order to use this algorithm in real time applications the information for one frame should be stored in memory and in case of resolution of 1024 by 768 the amount of memory will be 768 KB. During the sampling of the next frame, the coordinates of the object should be calculated and the calculations should be finished till the end of the sampling, and during this time the next sample should be stored in another 768 KB . This leads to a total of $1,5 \mathrm{MB}$ of memory which is too much for a mobile device.

With the help of Mathcad, a mathematical analyze of the method was made using an image of laser spot (fig. 3) and the following results were determined:


Fig. 3. Image resolution: $308 \times 245$, Coordinates of the object: $\mathrm{X}=147 ; \mathrm{Y}=127$
This algorithm is the base of the algorithm for calculating of coordinates in real time applications. The average brightness of the whole image from the previous frame is used as a threshold.

$$
\begin{equation*}
E t h=\frac{\sum_{i=1}^{k} E i}{k} \tag{2}
\end{equation*}
$$

where: Eth - brightness threshold, k - number of pixels in the image.
The decision for a given pixel if it belongs to the object is made with the following inequality:
(3) $\mathrm{E}>$ Eth

In the modified algorithm, at the moment of scanning of the element a decision is taken if it belongs to the object and if it conforms to condition (3), $\Sigma \mathrm{XiEi}$ and $\Sigma \mathrm{Ei}$ are calculated. At the end of each line of the frame the following formulas are used:

$$
\begin{equation*}
X c_{\text {row }}=\frac{\sum_{i=1}^{n} X i E i}{\sum_{i=1}^{n} E i} \quad E c_{\text {row }}=\frac{\sum_{i=1}^{n} E i}{n} \tag{4}
\end{equation*}
$$

where: $\mathrm{Xc}_{\text {row }}$ - center of the object for the corresponding line, $\mathrm{Ec}_{\text {row }}$-brightness of $\mathrm{Xc}_{\text {row, }} \mathrm{n}$ - number of elements in the line conforming to (3)

At the end of the line, $\mathrm{Xc}_{\text {row }}$ and $\mathrm{Ec}_{\text {row }}$ are stored in the corresponding elements in two one-dimensional arrays with a size equaled to the number of lines in one frame - 1024. This method occupies only 2 KB of memory.

At the end of the frame the real center of the object is computed using the data from the arrays:

$$
\begin{equation*}
X c=\frac{\sum_{i=1}^{m} X i_{\text {row }} E i_{\text {row }}}{\sum_{i=1}^{m} E i_{\text {row }}} \quad Y_{C}=\frac{\sum_{i=1}^{m} i . E i_{\text {row }}}{\sum_{i=1}^{m} E i_{\text {row }}} \quad E c=\frac{\sum_{i=1}^{m} E i_{\text {row }}}{m} \tag{5}
\end{equation*}
$$

where: Xc, Yc - coordinates of the center of the object, Ec - brightness of the center of the object, m - number of stored elements in the array.

During the time of scanning the image, parallel to the algorithm for calculating the coordinates of the center of the object, Eth is calculated using (2), because Eth will be used in the next frame. With this calculation an adaptation of algorithm towards changes in the brightness of the stage is achieved. The algorithm of work is shown on fig. 4.


Fig. 4. The algorithm of work

## 3. HARDWARE

A board was made, and for the calculations is used FPGA SPARTAN II from XILINX. SPARTAN II is programmed via JTAG interface and can be reprogrammed while on the board. The hierarchy of the system is shown on fig. 5 . Software is written in VHDL (Very High Integrated Circuit Hardware Description Language).


Fig. 5. The Structure of the system

## 4. REFERENCES

[1] Djamiykov, T.; Marinov, M.; Lesichkov, I.; Milushev, M.: Internet-basierte Erfassung von optischen Daten. Proc. 47. Internationales Wissenschaftliches Kolloquium, Ilmenau, Deutschland, 23-27.09.2002, S. 368.
[2] Howarth, D.: Performing Data Acquisition over the Internet. Sensors Magazine, January 1998.
[3] Schrüfer, E.: Signalverarbeitung. Hanser Verlag, München, 1990.
[4] Sze, S.: Semiconductor Sensors, John Wiley, New York, 1994.
[5] Yamasaki, H. K. Takahashi.: Advanced intelligent sensing system using sensor fusion. Proceedings of the 1992 International Conference on Industrial and Electronic Instrumentation and Automation. Vol. 1, 1993.

