Advanced illumination control algorithm in VHDL

Ricardo M. Sousa[a,b], Martin Wäny[b], Pedro Santos[b], Morgado-Dias [a,c]
[a] University of Madeira, Campus da Penteada 9000-390, Funchal, Portugal
[b]Awaiba Lda, Madeira Tecnopolo 9020-105, Funchal, Portugal
[c] Madeira Interactive Technologies Institute, Madeira Tecnopolo 9020-105, Funchal, Portugal

ABSTRACT

CMOS image sensor manufacturer, AWAIBA, is providing the world's smallest digital camera modules to the world market for minimally invasive surgery and one time use endoscopic equipment. Based on the world's smallest digital camera head and the evaluation board provided to it, the aim of this paper is to demonstrate an advanced fast response dynamic control algorithm of the illumination LED source coupled to the camera head, over the LED drivers embedded on the evaluation board. Cost efficient and small size endoscopic camera modules nowadays embed minimal size image sensors capable of not only adjusting gain and exposure time but also LED illumination with adjustable illumination power. The LED illumination power has to be dynamically adjusted while navigating the endoscope over changing illumination conditions of several orders of magnitude within fractions of the second to guarantee a smooth viewing experience. The algorithm is centered on the pixel analysis of selected ROIs enabling it to dynamically adjust the illumination intensity based on the measured pixel saturation level. The control core was developed in VHDL and tested in a laboratory environment over changing light conditions. The obtained results show that it is capable of achieving correction speeds under 1 s while maintaining a static error below 3% relative to the total number of pixels on the image. The result of this work will allow the integration of millimeter sized high brightness LED sources on minimal form factor cameras enabling its use in endoscopic surgical robotic or micro invasive surgery.

Keywords: Automatic Illumination Control, CMOS Image Sensors, LED Lighting, Medical Imaging, Field-Programmable Gate Arrays

1. INTRODUCTION

The purpose of the NanEye image sensor (which has a footprint of 1 mm x 1 mm) is to provide the market with the smallest digital camera for minimally invasive surgery and single use medical endoscopy equipment. Currently, endoscopy is a relatively painless medical procedure, and as such it is widely used to inspect the interior cavities of the human body [1]. In many of these applications, it is required that a light source be integrated with the camera to provide an adequate illumination to the observed environment [2], [3]. In this particular application a cold white high brightness LED array was used as a light source. In automatic illumination control systems, the light source is connected to a dedicated processor with the purpose of maintaining an ideal lighting level through the procedure [3]. The intensity of the LED lighting and the exposure time must be dynamically adjusted within short time periods to ensure a smooth viewing experience. Simultaneously, the endoscope is exposed to highly variable lighting environments as the medical operator navigates internal organs such as the esophagus, stomach or the respiratory tract [2], [3]. On this application, a FPGA platform was used to implement an algorithm for real-time lighting control, by acting on fast response LED drivers coupled to the light source. This enabled a hardware level control architecture which greatly reduces feedback, processing time, and control command exchange latency.

Changing the lighting intensity and changing the exposure time of the camera have the same effect on the image. However, changing the lighting level can be seen as a coarse adjustment and varying the exposure time as a fine tuning. As a result, the light level adjustment should be handled with care [4]. Because on the NanEye camera the aperture on the lens is fixed, there is no limitation of the exposure time on the low side, however the same cannot be said on the high side due to the fact that too long exposure times may lead to motion blur [4]. In the case of the application here described, the level of exposure was considered fixed.

The most straightforward approach is the use of a Proportional Integral Differential (PID) or Proportional Integral (PI) type controller. However, to guarantee a pleasant viewing experience, this should have a very smooth reaction such that the lighting intensity is adjusted in a gentle and gradual manner. This implies a high time constant achieved by setting a low proportional gain (K_P) and a high integral factor (K_I). Yet, due to resource constraints on the FPGA development platform, we have sought to develop a control module that minimized the amount of logic used to implement the control core.

2. PROPOSED TOPOLOGY

The topology of this proposal is presented in Fig. 1. This closed loop system adjusts the lighting intensity based on the grayscale pixel values obtained from the camera's video stream (here referred to as Pixel Data). The analysis of each frame is synchronized by the line period signal, Line Valid (LVAL), and the frame period signal, Frame Valid (FVAL). The topology is composed of a controller block (designated Illumination Control), a digitally controlled LED source, the target scene filmed by the camera and exposed to the light source, and the digital camera in conjunction with the deserializer core which is represented as a single block designated as Sensor.

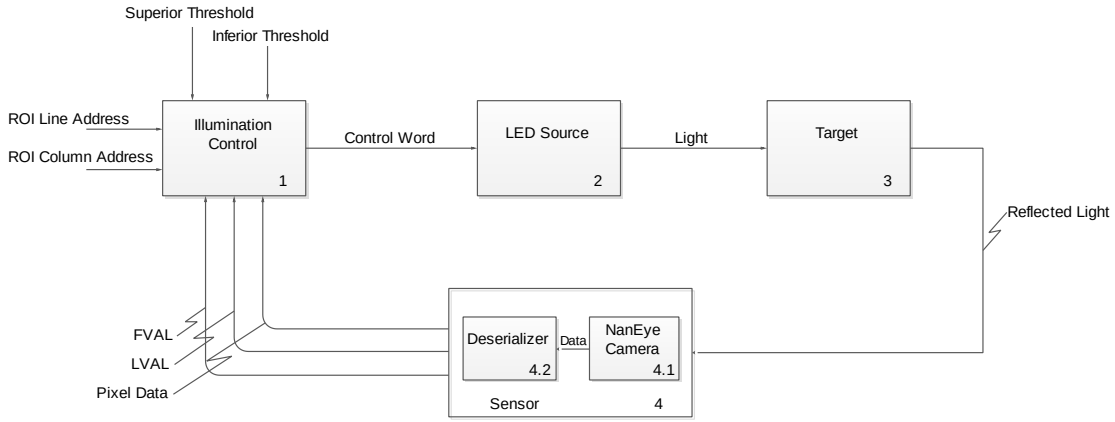

Figure 1 - Block diagram of the illumination control system.

On the Illumination Control block an image histogram is built and analysed for each received frame considering a select Region of Interest (ROI) defined by the Line and Column Address inputs. The optimum illumination level is defined by using the Superior and Inferior Threshold inputs as references. The output is a control word used to adjust the light intensity emitted by the LEDs coupled to a LED driver. In turn, this light source illuminates a scene/object/plane, here referred to as Target. Since there is a multitude of scenes which the camera can be exposed to, all of them with different characteristics in terms of light reflection and natural lighting, the illumination algorithm needs to be independent from the target.

3. ILLUMINATION CONTROL ALGORITHM

3.1. Region of interest

One of the implemented features was the ability to define ROIs in the captured image, allowing the illumination adaptation to be focused on a restricted region of the captured scene. In practice this means restricting the number of pixels analyzed by the algorithm, to a region defined by the user. To realize this, two counters were used, one designated Line Counter and the other Column Counter. Note on Fig. 2 that two line addresses (Initial Line Address and End Line Address) and two more column addresses (Initial Column Address and End Column Address) are used to outline the intended ROI. The counters are then incremented from 0 to 249 (NanEye camera has a 250 x 250 pixel resolution), activating the Enable Line and Enable Column signals when their count is within the limits defined by ROI addresses. Combining both enables a Pixel Evaluation Enable is generated. As the frame is received this signal only allows the evaluation of the pixels contained in the high period of the latter.

3.2. Illumination control algorithm

In order to render the illumination control algorithm independent from the target scene, a histogram based image analysis was used (Fig. 3). The histogram shows, from left to right, the level of dark, medium and white tones on the image (level of illumination or brightness), and from the top down, the amount of image pixels distributed throughout these regions. By setting low and high side reference levels, Inferior Threshold and Superior Threshold, it is possible to control the light intensity that the camera is exposed to. Two counters are used to accumulate the number of pixels which are exposed to a light intensity below the Inferior Threshold and the number of pixels which are exposed to a light intensity above the Superior Threshold. By controlling the amount of pixels which is above the upper reference level, using the lower level as an auxiliary, it is possible to maintain an adequate lighting level.

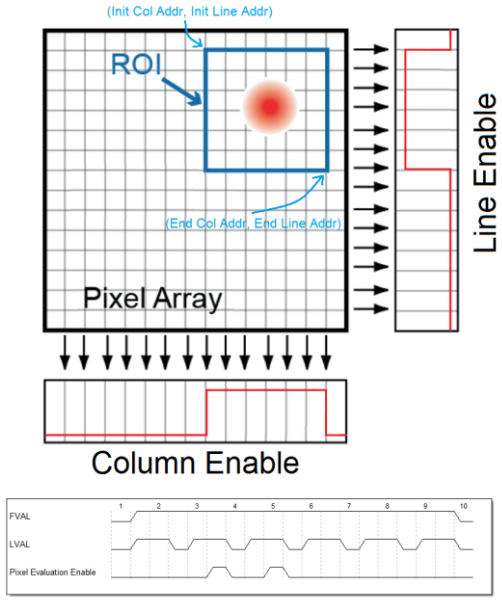

Figure 2 – ROI definition.

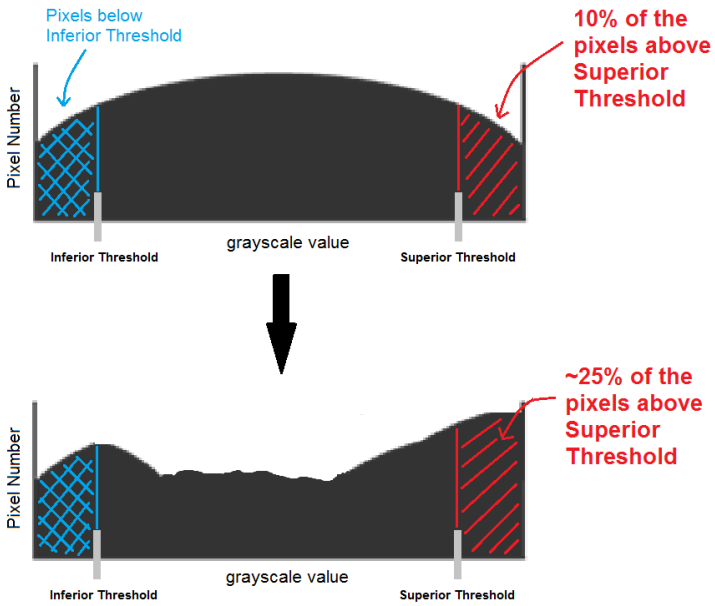

Figure 3 – Reference thresholds defined on the image histogram.

Unfortunately there is no formula for the optimum illumination level on a given scene. So, on this application it was defined that in order to have a correctly illuminated image, 25% of the total number of pixels in a ROI would have to be above the Superior Threshold. The system would then have to gradually adjust the voltage level applied to the DAC incorporated on the LED Driver so that the illumination intensity corresponded to this percentage, by moving the histogram to the right as exemplified on Fig. 3. Since the ROI may have variable sizes, an error of 1000 pixels in relation to the 25% may be perfectly acceptable in a region of 50000 pixels but the same may not be true for a region comprised of 10000 pixels. Consequently, the error will always have to be measured as a percentage of the total number of pixels in a ROI and not in absolute values. In this way, we get a target value that is always referenced to the total number of pixels in the ROI.

A feedbacked system like the one described requires a discrete compensator (PI or PID) to accelerate the convergence of the lighting adjustment without compromising system stability. However, some of the filter parameters have non integer values, which due to the nature of the operation implies the use of floating point math. The latter consumes a considerable amount of resources on the FPGA, and may require several clock cycles to generate results [5]–[7]. Moreover the presence of integral and derivate components implies a sampling time division. One way to simplify this operation is to keep the number of samples per second as a power of 2, which reduces the division to a simple bit-shift operation [5]. However due to the time sampling dependence with the camera's frame rate, this condition could not be guaranteed.

To circumvent this problem a control algorithm to replace a typical PID was developed. This algorithm is illustrated in the flowchart of Fig. 7. Two counters are used to accumulate the number of pixels above the top reference, Superior Threshold Count, and the number of pixels below the lower reference, Inferior Threshold Count .The value of each of these counters is compared with the relative percentage and the system corrects by acting on the LED driver's DAC. The correction is in the downward direction (negative correction) if the number of pixels above the Superior

Threshold is greater than 25%, and ascending if lower (positive correction). The magnitude of the correction is dependent on the error in relation to the figure of 25%. As such, it is higher for large deviations, and smaller for minor deviations. The implemented mechanism can be described as

$$\begin{aligned}
&Sup\ Thold\ Count > 75\% && \Rightarrow Correction = -x\\
&Sup\ Thold\ Count < 75\% \ \wedge\ Sup\ Thold\ Count > 50\% && \Rightarrow Correction = -y\\
&Sup\ Thold\ Count > 25\% \ \wedge\ Sup\ Thold\ Count < 50\% && \Rightarrow Correction = -z\\
&Sup\ Thold\ Count < 25\% \ \wedge\ Sup\ Thold\ Count > 6.25\% && \Rightarrow Correction = \ z\\
&Sup\ Thold\ Count > 6.25\% \wedge \ \mathrm{Inf}\ Thold\ Count \ < 50\% && \Rightarrow Correction = \ y\\
&Sup\ Thold\ Count < 6.25\% \wedge \ \mathrm{Inf}\ Thold\ Count \ > 50\% && \Rightarrow Correction = \ x
\end{aligned} \tag{1}$$

where $|x| > |y| > |z|$. As exemplified in (1), if Superior Threshold Count is greater than 75% of the Pixel Number, the correction is in the downward direction and with high amplitude. On the other hand, if it is greater than 25% and less than 50 %, the correction is still in the downward direction but with a lower amplitude. The inferior threshold count is also used in combination with the superior count in order to achieve the same effect but on the ascending direction. The percentage values used in (1), are obtained by a simple bit shift operation of the Pixel Number word as shown in Fig. 4, minimizing resource consumption.

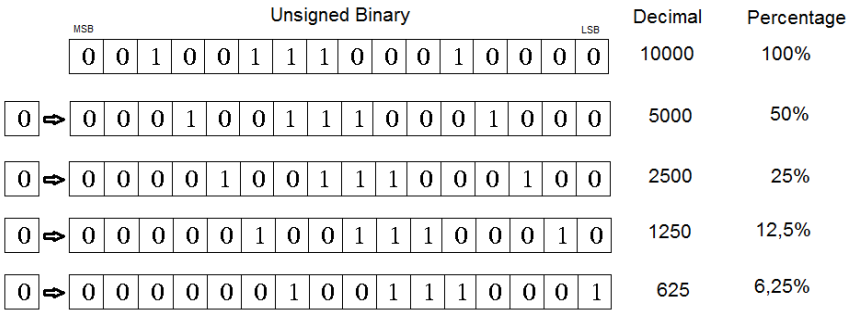

Figure 4 – Logical displacement of the Pixel Number word to generate percentage values.

The use of these percentage levels is justified by the need to add a differential component to the algorithm without having to measure the error in absolute values. Specifically, the differential factor (here designated as $Diff\ Factor$) assumes a value based on the percentage difference, e_{diff}, between the $Sup\ Thold\ Count\ (N)$ and the $Sup\ Thold\ Count\ (N-1)$. With N the current count and $N-1$ the previous count. This difference can be interpreted as

$$e_{diff}(N) = \begin{cases}
|Sup\ Thold\ Count\ (N) - Sup\ Thold\ Count\ (N-1)| < 1\% & \Rightarrow Diff\ Factor = 26\\
|Sup\ Thold\ Count\ (N) - Sup\ Thold\ Count\ (N-1)| < 3\% & \Rightarrow Diff\ Factor = 18\\
|Sup\ Thold\ Count\ (N) - Sup\ Thold\ Count\ (N-1)| < 6\% & \Rightarrow Diff\ Factor = 14\\
|Sup\ Thold\ Count\ (N) - Sup\ Thold\ Count\ (N-1)| < 12\% & \Rightarrow Diff\ Factor = 10\\
|Sup\ Thold\ Count\ (N) - Sup\ Thold\ Count\ (N-1)| > 12\% & \Rightarrow Diff\ Factor = 1
\end{cases} \tag{2}$$

By using this $Diff\ Factor$ as a multiplicative element in the correction, it is possible to significantly accelerate the error correction speed. However, this comes at a cost of stability. So, to ensure an appropriate behavior and a more rapid stabilization it was necessary to also add an integral component to the error correction. In discrete systems this can be done by using a simple cumulative sum, again avoiding the use of more complex operations [8]. In Fig. 5 this concept is exemplified. On this implementation this solution was implemented by means of a sum of variables, in order to avoid the risk of overflow/underflow of the sum.

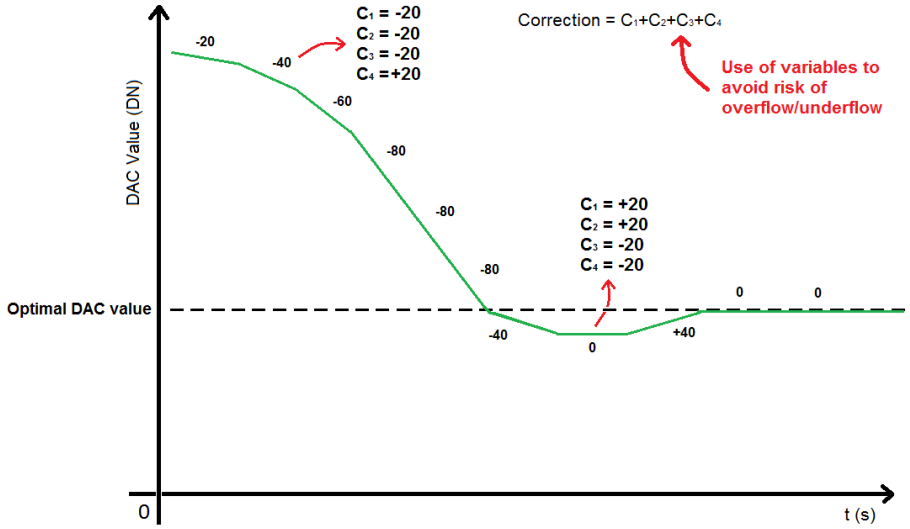

Figure 5 – Integral component of the error correction.

The resulting control word written to the LED driver can then be written has

$$DAC\ Word = DAC\ Word + Correction \times Diff\ Factor \tag{3}$$

where $Correction$ is given as the sum of the previous adjustments,

$$Correction(k) = C_1(k) + C_2(k-1) + C_3(k-2) + C_4(k-3) \tag{4}$$

and k represents the current iteration. To obtain the integrative effect, the variables are logically shifted at each iteration has shown on Fig. 6. To further improve the stabilization speed a mechanism was included to perform a reset on the sum when the optimal DAC value was crossed (25% figure), essentially performing a reset of the filter whenever $C_1 = \pm 1 \vee C_1 = 0$.

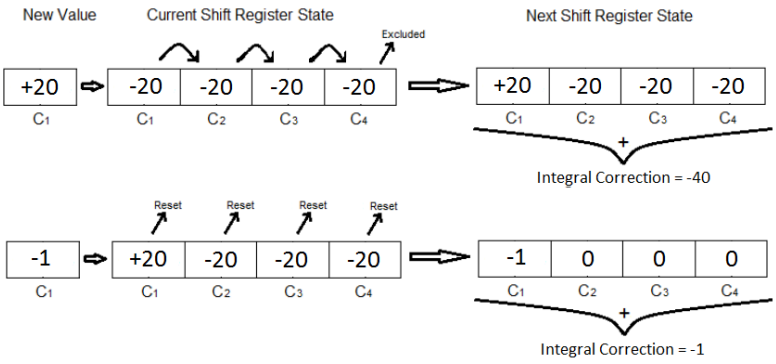

Figure 6 – Correction Shift Register with reset mechanism.

Two status signals are used, Above Flag and Below Flag, to indicate if the sequence and amplitude of previous corrections were on the downwards or upwards direction, allowing the system to detect high amplitude oscillations. This decreases the stabilization time also avoiding oscillations that might become visible to the user in the form of a flickering light source. This process and all the control method described in this section is illustrated in the flowchart of Fig. 7.

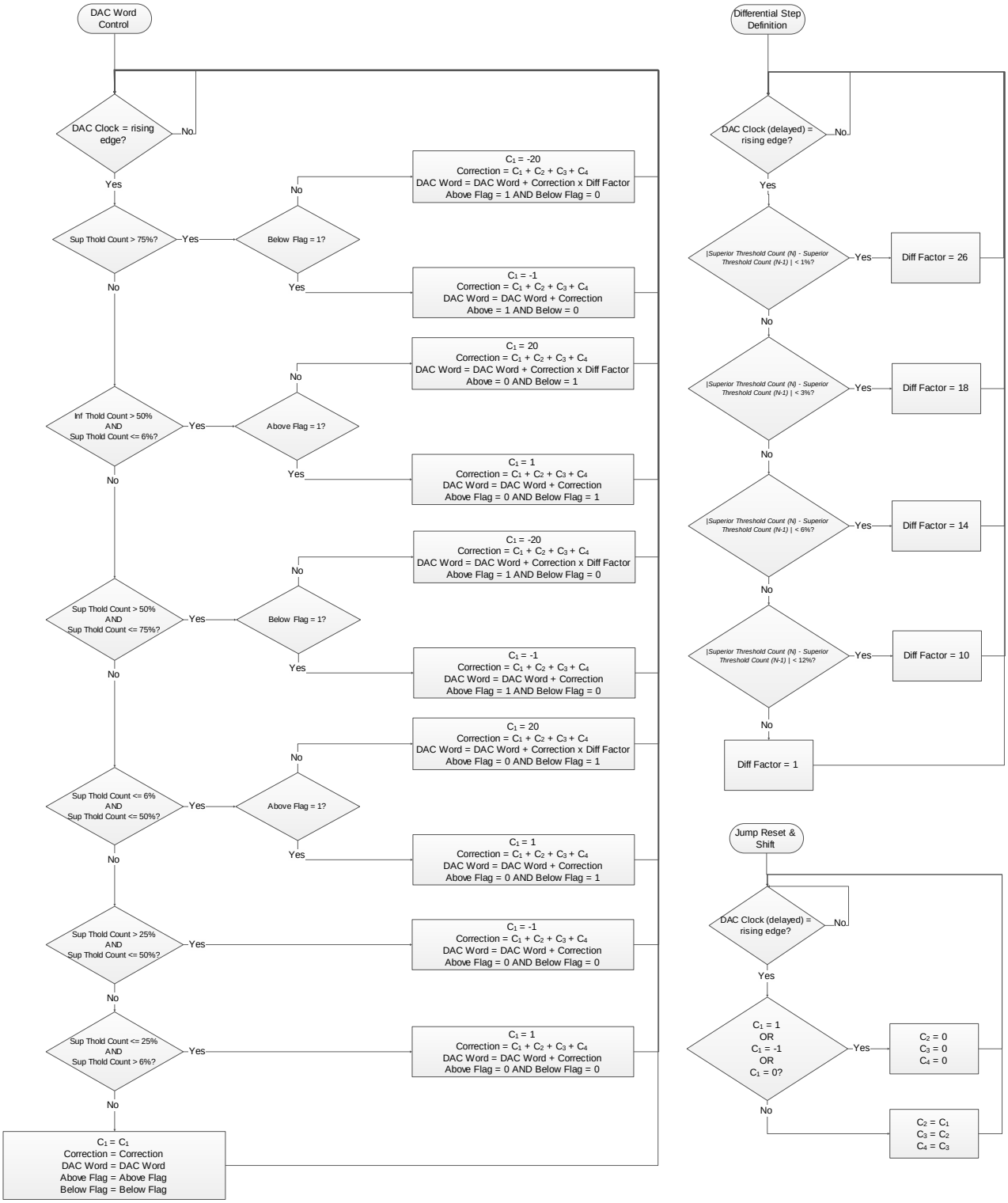

Figure 7 – Flowchart of the illumination control algorithm.

4. RESULTS

The results will be presented in two sections: the illumination level adjustment and the resources used to implement the control algorithm.

4.1. Illumination level adjustment

The control core was tested in laboratory for performance evaluation. In order to exclude the vignetting affected pixels at the edges of the image, a standard ROI of 200 x 200 pixels was considered. Since the pixel output, on the NanEye camera is in 10 bit format, the threshold values can range from 0 to 1023. For the test exemplified on Fig. 8 and 9, a Superior Threshold of 600 and an Inferior Threshold of 300 were considered. Note, however, that the pixel output after color reconstruction is reduced to 8 bit format.

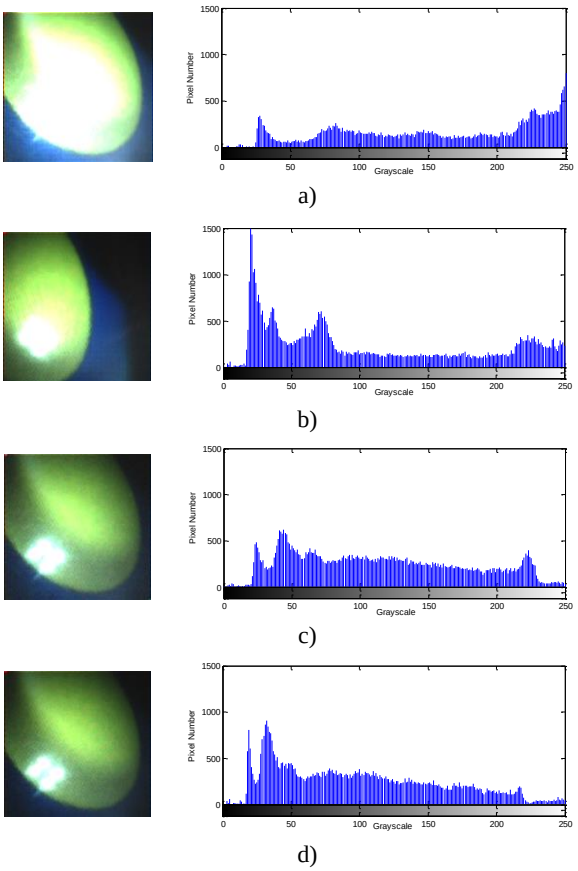

Figure 8 – Illumination adjustment to an optimum level (1).

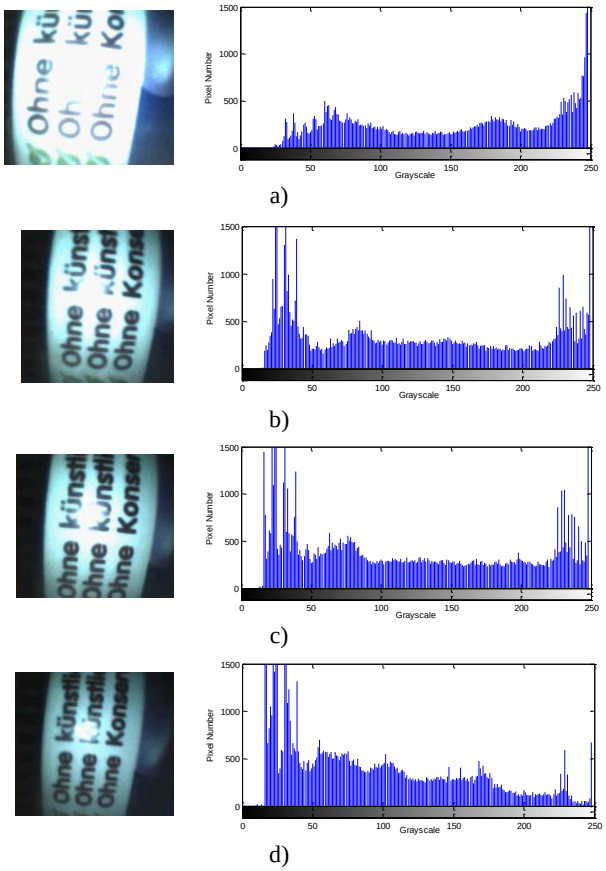

Figure 9 – Illumination adjustment to an optimum level (2).

It was possible to observe that for over illuminated images (Fig. 8 a) and 9 a)) there was a large concentration of pixels on the right end of the histogram, corresponding to high values on the grayscale. In this situation the percentage of pixels above the Superior Threshold is greater than 25%, which causes a gradual adjustment of the light intensity in the downward direction. As a consequence, the pixel concentration on the high side decreases until the percentage of bright pixels is approximately 25%, a situation portrayed in Fig. 8 d) and 9 d). To regulate the brightness level on the image, the user can simply increase or decrease the Superior Threshold value. Naturally, the pixel concentration at the left end of the histogram increases to the extent that the image becomes darker, however as long as the level of bright pixels remains around 25% the algorithm considers that the scene is properly illuminated. Further, it was found that the system correctly adjusted the lighting level regardless of the scene's color multiplicity, and the target's surface reflective properties.

To quantify and compare the obtained results, several algorithm development stages were tested for performance. These include a purely proportional unitary step correction (Stage 1), a purely proportional multi-step correction (Stage 2), a proportional multi-step correction coupled with the filter reset mechanism (Stage 3), a proportional multi-step correction coupled with an integral component (Stage 4), and finally the implemented solution, described in section 3, which is comprised of a proportional multi-step correction coupled with the filter reset mechanism plus an integral and a differential component (Stage 5). In Table 1 the most significant parameters can be observed.

Table 1 – Obtained performance for each development stage.

	Stage 1	**Stage 2**	**Stage 3**	**Stage 4**	**Stage 5**
Error	< 1%	< 12%	< 12%	< 3%	< 3%
Adjustment speed	> 10 s	2.5 s to 3.5 s	2 s to 3 s	4 s to 5 s	0,5 s to 1 s
Stability	+++	--	-	++	+[1]
Associated logic	--	-	+	++	+++

[1] Due to the differential component included on the algorithm, there is a greater risk of sporadic super elevations in relation to the optimal value, as such there is a slight decrease in stability.

It was observed that the final algorithm implementation is capable of achieving the correct illumination level, in the worst case, within 1s whilst maintaining a relative error below 3% of the total amount of pixels contained within a ROI. However, stability and associated logic are somewhat sacrificed for improved performance when compared to earlier stages.

4.2. Used resources

The control module was implemented on a Xilinx Spartan 3 XC3S500E FPGA. The amount of resources used on this chip are presented in Table 2. Although in the literature there are several FPGA based PID (or PI) controller implementations [5], [9]–[12], it was not possible to perform direct comparisons due to the fact that most of these implementations are aimed at completely distinct applications which require different controller implementations. Therefore, these results are presented as a reference to future implementations.

Table 2 – FPGA logic resources used by the illumination control core (Stage 5 implementation).

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Flip Flops	373	9312	4%
Number of 4 input LUTs	816	9312	8%
Number used as logic	787	-	-
Number used as a route-through	29	-	-
Number of occupied Slices	538	4656	11%
Number of BUFGMUXs	3	24	12%
Number of MULT18X 18SIOs	2	20	10%

5. CONCLUSIONS

The problem of adaptively regulating the intensity of an illumination source without the use of an external luxmeter was solved by analyzing the captured images from a camera and dynamically acting on the current intensity supplied to the light source. For this purpose, a control module was implemented on a FPGA platform to select and analyze the grayscale values of each pixel contained on a given region of interest. This allows the generation of an image histogram for each captured frame which is evaluated taking into account two reference levels. The number of pixels above and below these reference levels is determined. The algorithm modulates the light source's power supply so that the number of pixels above the upper reference level corresponds to 25% of the total number of pixels in the region of interest. This allows for an adequate illumination level to be maintained regardless of the size of the region of interest and the type of illuminated surface. Response times smaller than 1 s were achieved with error rates lower than 3%.

It was necessary to include a PID type compensator to accelerate the convergence speed while limiting the light source's oscillation. However this controller was implemented with a focus on FPGA resource use minimization. To achieve this an algorithm was created to act on the system dynamics much in the same way a typical PID does. To reduce the static error an integrative integrative component was realized in the form of a sum of variables. And to increase the convergence speed a differential component was added to the filter through a multiplicative factor based on the error percentage level. This solution enabled simple bit shift operations and nested multiplexers to replace more resource hungry divider and multiplier cores necessary to implement a discrete PID controller.

ACKNOWLEDGEMENTS

We thank all the Awaiba staff for their encouragement and support. Their technical knowledge and expertise were invaluable contributions to the inception and continued development of this work. The authors would like to acknowledge the Portuguese Foundation for Science and Technology for their support through project PEst-OE/EEI/LA0009/2011. Also acknowledged is the Funding Program + Conhecimento II: Incentive System to Research and Technological Development and Innovation of Madeira Region II, through the project Vision 3D – MADFDR – 01 – 0190 – FEDER – 000014.

REFERENCES

[1] M. Liedlgruber and A. Uhl, "Computer-aided decision support systems for endoscopy in the gastrointestinal tract: a review.," *IEEE Rev. Biomed. Eng.* vol. 4, 73–88 (2011).

[2] ECRI Institute, "Video Endoscopy Systems," *Healthc. Prod. Comp. Syst.* vol. 1, 1–7 (2007).

[3] A. Y. K. Chan, "Endoscopic Video Systems," *Biomedical Device Technology: Principles and Design*, 536–546 (2008).

[4] M. Muehlebach, "Camera Auto Exposure Control for VSLAM Applications," Swiss Federal Institute of Technology Zurich, (2010).

[5] E. Normann Naess and S. Hendseth, "Implementing Controller Strategies in FPGA," Norwegian University of Science and Technology, (2009).

[6] Xilinx, "Spartan-3E Libraries Guide for HDL Designs," vol. 617, 1–489 (2011).

[7] G. Stitt, "Are field-programmable gate arrays ready for the mainstream?," IEEE Micro vol. 31 no. 6, 58–63 (2011).

[8] J. Wells, "Generalizations of the Riemann Integral : An Investigation of the Henstock Integral," *Whitman Coll.*, 1–28 (2011).

[9] B. V Sreenivasappa and R. Y. Udaykumar, "Analysis and Implementation of Discrete Time PID Controllers using FPGA," vol. 2 no. 1, 71–82 (2010).

[10] R. Nema, R. Thakur, and R. Gupta, "Design & Implementation of FPGA Based On PID Controller," no. 2, 14–16 (2013).

[11] S. Sonoli, "Implementation of FPGA based PID Controller for DC Motor Speed Control System," vol. 2, (2010).

[12] A. Trimeche, A. Sakly, A. Mitbaa, and M. Benrejeb, "PID Controller Using FPGA Technology," *Advances in PID Control*, V. D. Yurkevich, Ed. InTech, 259–273 (2011).