Acceleration of Blob Detection in a Video Stream using Hardware

by
Alexander Bochem,
Rainer Herpers and Kenneth B. Kent

TR 10-201, May 26, 2010

Faculty of Computer Science
University of New Brunswick
Fredericton, NB, E3B 5A3
Canada

Phone: (506) 453-4566
Fax: (506) 453-3566
Email: fcs@unb.ca
http://www.cs.unb.ca

Contents

List of Figures

List of Tables

Abstract

This report presents the implementation and evaluation of a computer vision problem on a Field Programmable Gate Array (FPGA). This work is based upon [5] where the feasibility of application specific image processing algorithms on a FPGA platform have been evaluated by experimental approaches. The results and conclusions of that previous work builds the starting point for the work, described in this report. The project results show considerable improvement of previous implementations in processing performance and precision. Different algorithms for detecting Binary Large OBjects (BLOBs) more precisely have been implemented. In addition, the set of input devices for acquiring image data has been extended by a Charge-Coupled Device (CCD) camera. The main goal of the designed system is to detect BLOBs in continuous video image material and compute their center points.

This work belongs to the MI6 project from the Computer Vision research group of the University of Applied Sciences Bonn-Rhein-Sieg[1]*. The intent is the invention of a passive tracking device for an immersive environment to improve user interaction and system usability. Therefore the detection of the users position and orientation in relation to the projection surface is required. For a reliable estimation a robust and fast computation of the BLOB's center-points is necessary.*

This project has covered the development of a BLOB detection system on an Altera DE2 Development and Education Board with a Cyclone II FPGA. It detects binary spatially extended objects in image material and computes their center points. Two different sources have been applied to provide image material for the processing. First, an analog composite video input, which can be attached to any compatible video device. Second, a five megapixel CCD camera, which is attached to the DE2 board. The results are transmitted on the serial interface of the DE2 board to a PC for validation of their ground truth and further processing. The evaluation compares precision and performance gain dependent on the applied computation methods and the input device, which is providing the image material.

[1] http://www.cv-lab.inf.fh-brs.de

1 Introduction

The usability of the software interface is a main criteria for the product acceptance by the user. A major issue in Computer Science is the improvement of information representation in interfaces and finding alternatives for user interaction. Simplifying the way a user operates with a computer helps optimize the usability and increases the benefit of digitization. The intention to close the gap between user and computer systems has lead to several different technologies. One way is the integration of computer generated data into the real world, also known as augmented reality (AR). Another opportunity is going for the integration of the user into a virtual world, such as virtual reality (VR). The connection between those two different approaches is referred two as virtuality continuum (VC) and has been defined by Paul Milgram and Fumio Kishino in 1994 [10].

The term "virtual reality" leads back to the late 30's. Antonin Artaud[2] has used this term in his book *The Theatre and Its Double (1938)* and refers to it as an environment "in which characters, objects, and images take on the phantasmogaric force of alchemy's visionary internal dramas". Today the common understanding for VR refers to three-dimensional computer-designed virtual environments, in which the user can navigate in any direction.

The common way to display a virtual environment are computer monitors, customized stereoscopic displays, also known as Head Mounted Displays (HMD), and the Cave Automatic Virtual Environment (CAVE). In general computer screens are relatively small compared to the virtual environment they display. The frame of a computer monitor causes a significant disturbance in the perception of the VR by the user. HMDs on the other hand allow a complete immersion of the virtual reality for the user. The downside is that HMDs are heavy and therefore not recommendable to wear for a longer time. The CAVE concept applies projection technology on multiple large screens, which are arranged in the form of a big cubicle [6]. The size of the cubicle usually fits several people. Based on the CAVE concept the University of Applied Sciences Bonn-Rhein-Sieg has invented a low cost mobile platform for immersive visualisations, called "Immersion Square" [8]. It consists of three back projection walls using standard PC hardware for the image processing task.

A common VR system uses standard-input devices such as keyboards and mice, or multimodal devices such as omni directional treadmills and wired gloves. The issue with those "active input devices" is the lack of information about the user and his point of interest. A computer mouse, for example, only gives information about its relative position on the desk in relation to its position on the computer monitor. In the common use-case the mouse sends data, which describes its movement in the X- and Y-direction. The coordinates cannot be used as a representative for

[2]http://en.wikipedia.org/wiki/Artaud

the user's position and orientation in the immersive environment. In addition a computer mouse lacks in the ability to navigate in a 3-dimensional space. A wired glove represents the user in the coordinate system of the virtual world, but it gives no information about the absolute position in the real world.

For immersive environments the position and orientation of the user in relation to the projection surface would allow alternative ways of user interaction. Based on this information, it would be possible to manipulate the ambiance without having the user actively use an input device. At the Bonn-Rhein-Sieg University for Applied Sciences this problem has been addressed in a project named *6 Degree-of-Freedom Multi-User Interaction-Device for 3D-Projection Environments (MI6)* by the Computer Vision research group. The projects goal is to estimate the position and orientation of the user to improve the interaction of the user with the immersive environment. Therefore a light-emitting device is developed, that creates a unique pattern of light dots. This device will be carried by the user and pointed towards the area on the projection screens of the immersive environment, where the user is looking. For the light source a laser diode in the infrared range will be used since the created light dots cannot be recognized by the human eye. By detecting the light dots on the projection screens, it is possible to estimate the position and orientation of the user in the cubicle, as shown in [13, 9]. These light dots appear as BLOBs with white and light-grey pixels on a black background in the image material. One problem is that the detection process of the BLOBs and the estimation of their center points requires a fast image processing. It is intended to realize an immediate feedback for the user for any change of direction and position in the immersive environment. Another problem is the required precision for the estimation of the BLOBs' center points. A mismatch of the BLOB's center point by a few pixels will cause a big error for the estimated position of the user. And with a growing distance between the user and the projection surface, the error becomes larger as well. With those problems in mind the intent is to develop a system which delivers high performance and high accuracy.

Computer Vision problems in general are characterized as the processing of large amounts of similar data at high speeds. This circumstance is a constant companion in Computer Vision and a major reason for the invention and further development of specialized hardware for digital image processing. A Graphic Processing Unit (GPU) is a customized processor for computations that are required for computer vision. One particular characteristic of a GPU is its ability to process data in parallel. While GPUs have a static architecture, one way to realize more customized solutions is to implement it on a FPGA. The FPGA is a grid of freely configurable logic gates, usually extended with internal memory or other specialized circuits, dependent on the application area. The main aspect of working with a FPGA is to realize algorithms or processing tasks in hardware. A "program" for an FPGA is implemented in a Hardware Description Language (HDL). For

working with HDLs the developer requires a different perspective than software engineers usually have. They have to keep in mind that every code line will be translated into a hardware circuit and therefore does not behave like a piece of C-code, which is executed sequentially on a general purpose processor (GPP). The compiler, also known as a synthesizer in the FPGA domain, parses the HDL code and identifies execution lines that can be realized in parallel circuits. The outcome is a hardware configuration file for the FPGA that could serve as a chip specification for a custom integrated circuit (IC).

Since FPGAs are well suited for fast image processing problems [7], this project's intent is to show the performance gain for the specified application task. This work covers the design, implementation, and validation of BLOB detection with different methods for the computation of their center point. In addition the pre-processing for the applied input sources are part of this work and the implementation for a serial communication module to transmit computation results on the RS232 interface to a connected Host-PC.

2 System Design

Figure 1 shows the schematic design of the complete system. The image material can be acquired on two different devices. The analog video input allows us to connect to any device with video output, like a DVD player. Therefore, the analog video input is used for precision evaluation with the recorded image material. The CCD camera has been applied for performance evaluation with real-time data, since the camera allows to run the system with higher frame rates. Each input device provides image material that requires pre-processing. The BLOB detection is designed to work with image material in RGB format. The input material from the analog video input is in YCbCr format with a resolution of 640x480. The input material from the CCD camera is in RGB format. But according to the Bayer pattern of the sensor, the frame rows need to be merged. A frame from a sensor with a resolution of 1280x960 results in a image of 640x480 pixels. Every frame in the system is processed only once.

2.1 Serial Interface

To observe the system process and evaluate its results a module for communication on the serial interface of the target board has been developed. Other hardware interfaces have been available, such as USB, Ethernet and IRDA. But the serial interface allows the smallest design with respect to protocol overhead and resource allocation on the FPGA. The RS232 controller on the DE2 board can operate on a maximum bandwidth of 120 Kbit/s [12]. In the proposed application a frame contains up to five BLOBs. Each BLOB is represented by an X-/Y-coordinate and an index number encoded in 29 bits. For the proposed task a single frame will contain up to five BLOBS. With 145 bits of BLOB results per frame the serial interface would be able to support a processing speed of up to 847 frames per second. This is much more than any of the available image sources for the project can obtain.

The serial communication module operates on a 40 MHz input clock and processes one byte at a time. For decoupling the processing speed of the serial interface module from the center-point computation module a first-in-first-out (fifo) memory module with asynchronous write and read clocks has been used. The fifo module is Intellectual Property (IP) of Altera and available with the development environment [4].

Figure 2 shows the schematic architecture for the serial communication module in the system design. The serial module reads the information about the BLOB result from the fifo and transmits it to the RS232 controller. A result has to be split into four one byte blocks to be processed by the RS232 controller. Missing bits in the last byte of the 29 bit result are filled with zero values on the most-significant-bit (MSB) side of the byte. The serial interface module operates independent

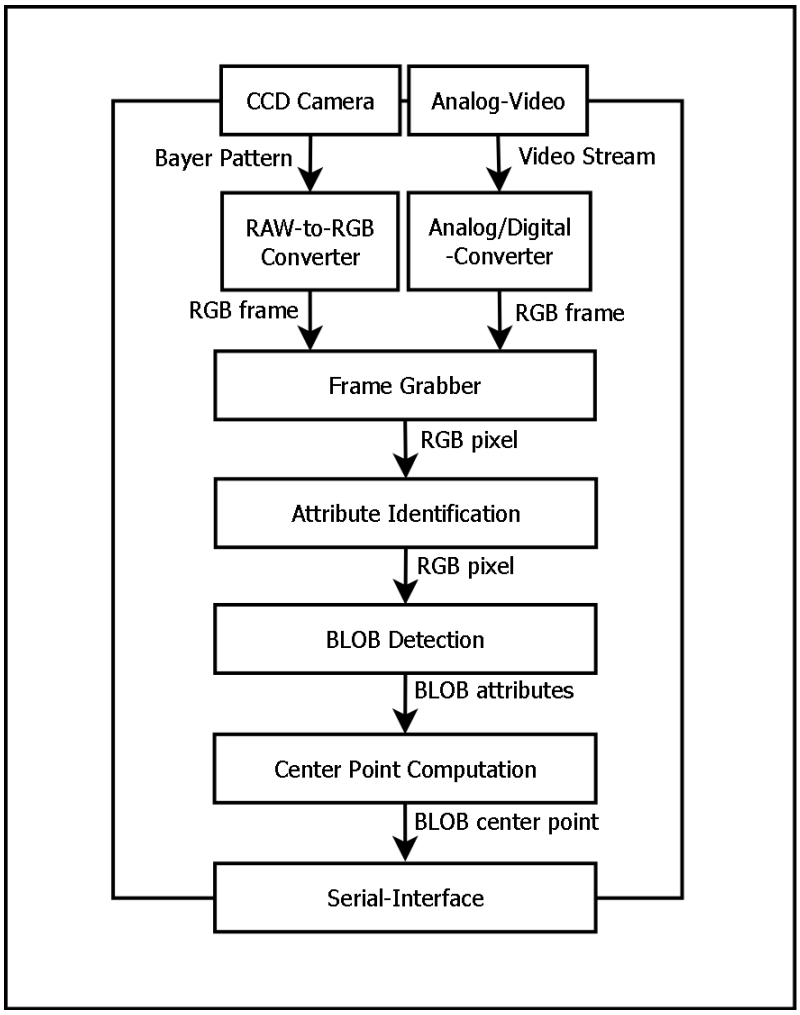


Figure 1. Schematic design of the system architecture.

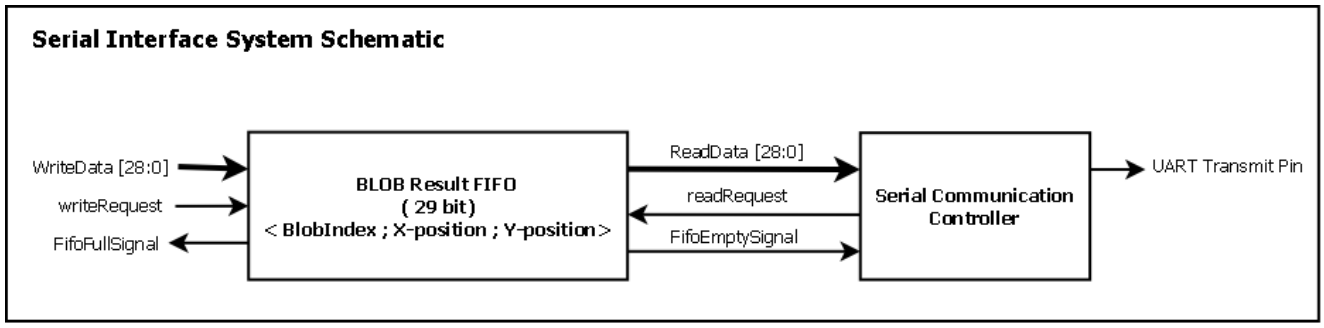


Figure 2. Schematic for serial communication interface module in system design.

from the other system modules and sends results as long as the fifo with BLOB results is not empty.

2.2 CCD Camera

The performance evaluation of the first BLOB detection approach in previous work [5] was restricted to the performance of the Analog-/Digital-Converter on the DE2 board. The video input signal could not be converted as fast as the BLOB detection module would have been able to process it. In the current project for providing faster image acquisition the CCD camera D5M (Figure 3) from Terasic has been attached to the DE2 development board [11]. The camera sensor has a maximum resolution of five Megapixels arranged in a Bayer pattern format. The camera can be configured over a dedicated I2C bus and contains an internal phase-locked-loop (PLL) unit if available clocks from the target board are too slow. In accordance with the technical documentation, a maximum read-out speed of 150 frames per second is possible with this camera. The read-out speed depends on several features such as resolution, exposure time and the activation of skipping or binning. Skipping and binning is used to combine lines and columns of the CCD sensors. This functionality allows to read out a larger area of the sensor in the form of a smaller image resolution. The consecutive lines in the sensor are merged and handled as a single image line. The controller on the D5M camera can read out one pixel per clock pulse and processes each sensor line in sequential order.

Figure 3. Five mega pixel digital camera from Terasic.

For the project requirements a transformation of the image data into RGB format was necessary. The intention was to prepare the image data from the camera as similar as possible to the expected image material in the application area later on. The image acquisition of the Immersion Square will happen with infrared cameras to eliminate irrelevant image data immediately. The image material in the proposed application will be in greyscale format. With the CCD camera used in this project, the greyscale representation of the image data can be computed after transforming it into

the RGB color model. To compute a pixel in RGB format out of a pixel in Bayer pattern format, two adjacent pixels in two consecutive rows (Figure 4) need to be merged. The combination of two green, one red, and one blue pixel results in one RGB pixel. The red and the blue sensor data can be used right away. Only the sensor data from the two green pixels need to be merged to one.

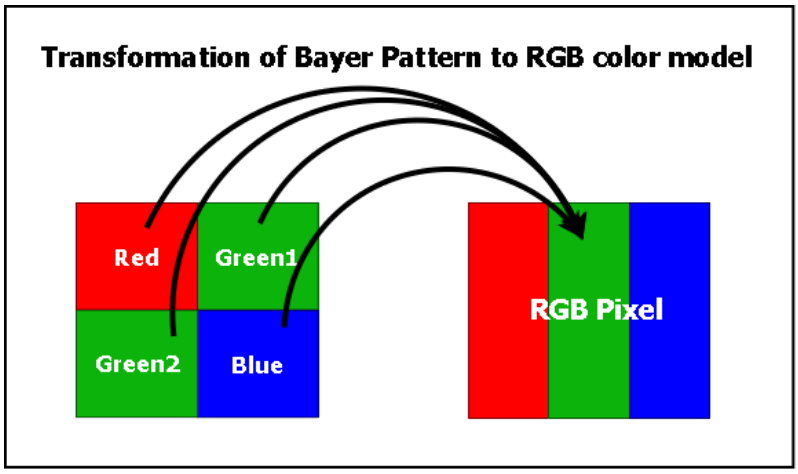


Figure 4. Applied transformation for sensor data from D5M camera.

The system design uses the input data from the camera for two different processing tasks. Based on the demonstration design, that comes with the camera, the acquired image data is displayed on the VGA output of the DE2 board. The schematic in Figure 5 shows the module design for that.

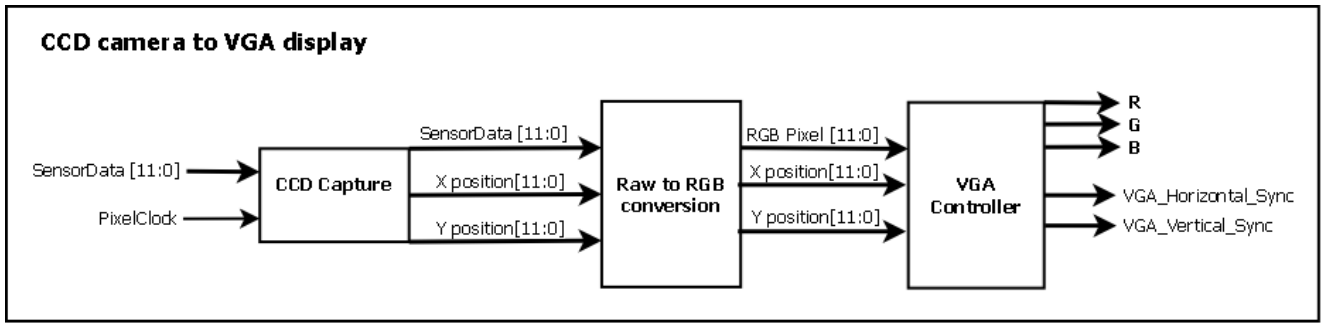


Figure 5. Module design for dataflow CCD camera to VGA output.

To allow the additional use of the image data in the BLOB detection module the RGB pixel values have been stored into a dual-clocked fifo. The BLOB detection module reads the fifo and searches for BLOBs in the image data. The result of the BLOB detection is the computed center-point, which is stored in another dual-clocked fifo. From there the serial communication module gets its data as described in the previous Section (Figure 6).

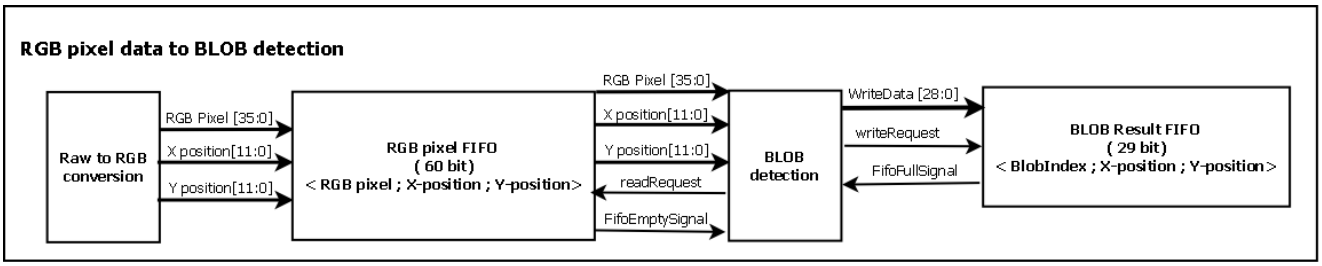


Figure 6. Module design for BLOB detection.

The modular architecture allows one to switch between the two designed BLOB detection approaches, Bounding Box and Center-Of-Mass, with ease.

2.3 BLOB Detection

In Computer Vision a binary large object is considered as a set of pixels, which describe a common attribute and are adjacent. This common attribute in most cases is defined by color or brightness of the pixel. The representation of the pixel depends on the applied color model in the image material. In the RGB color model a pixel consists of three values for red, green and blue also known as the color channel. The required memory size in the RGB model is linearly dependent on the color depth. An eight bit RGB color model for example has a value range from 0 to 255 for each channel, which allow to represent approximately 16.7 million colors. If the image material is provided in a greyscale model the pixel information is encoded in only one channel.

For the detection of the BLOBs the first problem to be solved is the identification of relevant pixels. A pixel is considered to be relevant if its color or brightness value exceeds a specified threshold value. This threshold value can be a static parameter or a computed average value, based upon the pixel values of previous frames.

The adjacency condition is the second important step in BLOB detection. The two most common definitions for adjacency are known as four pixel neighbourhood and an eight pixel neighbourhood. Figure 7 is showing the two ways of labelling pixels to describe adjacency. On the left image the four pixel neighbourhood is applied and four BLOBs can be detected. The detection applies the adjacency check only on the horizontal and vertical axis. On the right image it can be seen that the same pixels are labelled as only two BLOBs. For the eight pixel neighbourhood the diagonal axis is taken into account as well [5].

The check of the adjacency condition has to be performed on all pixels which match the criteria of the common attribute. Based on those two actions all BLOBs in a frame can be found. Based on the quality of the proposed image material and the specific application case the definition of further attributes for the BLOB can be helpful to get rid of false positive results. For example

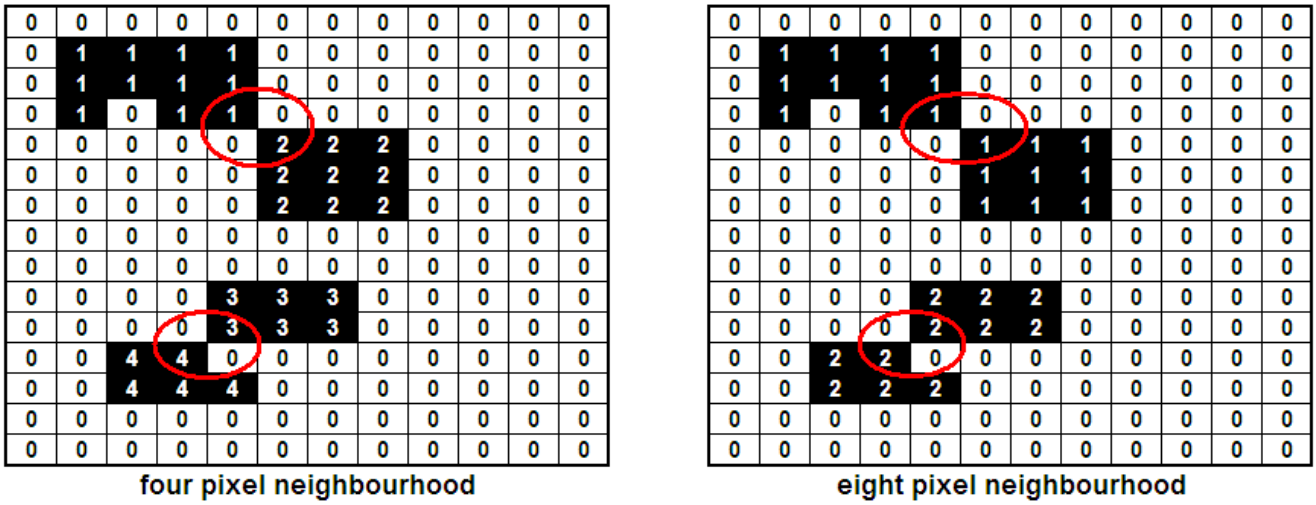


Figure 7. Four and eight pixel neighbourhood.

the size of the BLOB is a very dominant indicator if information about the object to detect in the image frame is available. But it requires a precise calibration of the system, since in the target application the size of the BLOB changes with the movement of the light emitting device. Also information about shape and orientation of the BLOB might be applied to increase the detection accuracy.

The aim of the BLOB detection for our requirements is to determine the center point of the BLOBs in the current frame. With respect to the application area this project describes BLOBs as a set of white and light gray pixels while the background pixels are black. This circumstance follows from the setup for the image acquisition where infrared cameras will be applied to track laser dots on a plain projection surface. The application of infrared cameras will eliminate unnecessary image scenes of the immersive visualization environment. The expected image material will be similar to the samples in Figure 8.

The characteristics of the blurring effect will depend on the acceleration of the light source by the user. This effect causes some issues for the estimation of the BLOBs center points, which have been addressed in this project.

For the computation of the BLOB's center-point two different concepts have been applied. The Bounding Box method and the Center-Of-Mass method. The Bounding Box based computation estimates the center-point of the BLOB by searching for the minimal and maximal XY-coordinates of the BLOB. The computation can be implemented very efficiently and will not cause big performance issues.

$$\text{BLOB's } X\_center\_position = \frac{maxX\_position + minX\_position}{2}$$

$$\text{BLOB's } Y\_center\_position = \frac{maxY\_position + minY\_position}{2}$$

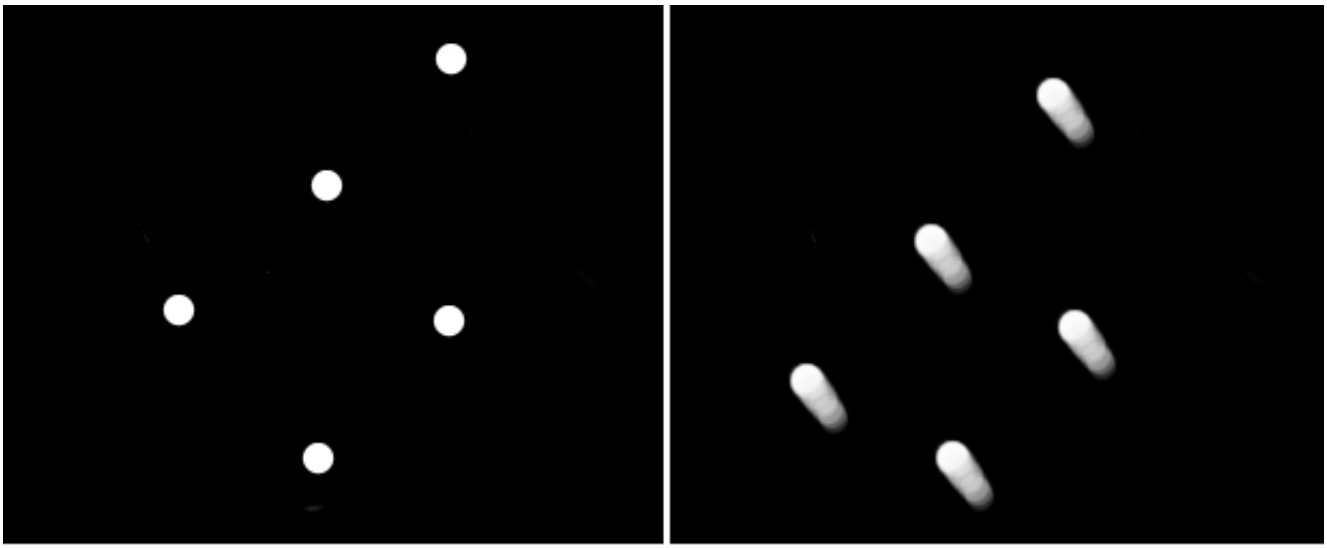


Figure 8. Example of BLOBs perfect shape and with blur.

Division by two can be realized by a bit shift and mathematic operations, such as addition, are not computationally expensive either. For the computation of the center-point the Bounding Box offers not enough information about the BLOB to extract very precise results. The center coordinates are strongly affected by the pixels at the BLOB's border. This effect becomes even stronger for BLOBs in motion. With reference to the light emitting device for the Immersion Square environment, the angle between the light beams and the projection surface changes the shape of the BLOBs and increases the range of pixels with less intensity. In addition the movement of the device by the user will cause motion blur. These effects will increase the flickering of pixels at the BLOB's border and will cause a flickering in the computed center-point. In Figure 9 some examples are given for inaccurate results based on the Bounding Box approach.

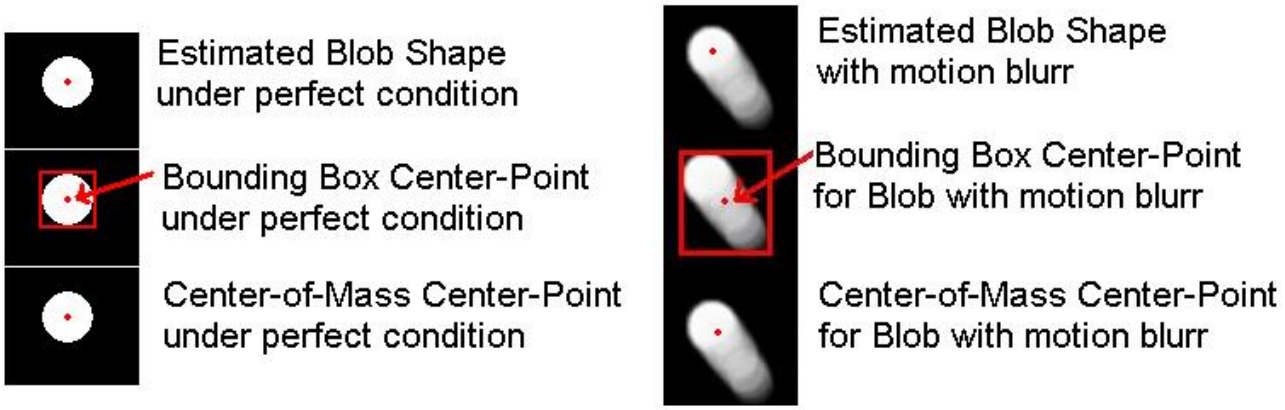


Figure 9. Example of center point inaccuracy.

The estimation of the BLOB's center-point is very accurate if the BLOB does not show a blurring effect. But as shown on the right hand side of the image, the Bounding Box computation has a higher error than Center-Of-Mass for BLOBs in motion.

With Center-Of-Mass the coordinates of all pixels of the detected BLOB are taken into account for the computation of the center point. The algorithm combines the coordinate values of the detected pixels as a weighted sum and calculates an averaged center coordinate [5].

$$\text{BLOB's } X\_center\_position = \frac{\sum X\_position \text{ of all BLOB pixels}}{\text{number of pixels in BLOB}}$$

$$\text{BLOB's } Y\_center\_position = \frac{\sum Y\_position \text{ of all BLOB pixels}}{\text{number of pixels in BLOB}}$$

To achieve even better results the brightness values of the pixels have been applied as weights as well. This increases the precision of the estimated center-point with respect to the BLOB's intensity.

$$\text{center position} = = \frac{\sum \text{ pixel position} * \text{pixel brightness}}{\sum \text{pixel brightness}}$$

As shown in Figure 9 with the Center-Of-Mass computation the accuracy increases slightly for BLOBs with motion blur. The number of BLOBs in the proposed application task can vary between zero and five, which complicates the conditions for the detection approach. This problem is caused by the construction of the application environment. With the three-sided cubicle as a projection surface, the user can change his orientation to any of the screens. In the situation the user faces a point at the corner of the cubicle one of the light dots might be split on two different projection screens [5].

A way to increase the BLOB detection precision is based upon the concept of foreground-background segmentation. The combination of consecutive frames would allow to identify the size of the area which is caused by motion blur. In addition the estimation of the BLOB's flow direction can be used to eliminate the pixels in the motion blur area. By averaging the motion of the center points, the effect of flickering center-point results can be smoothed. This would work for single BLOBs and for multiple BLOBs as well. The difficulty for single BLOBs is the indexing of the BLOBs to continuously track the BLOB's motion. This becomes easier with entangled multiple BLOBs, which move towards the same direction.

The elimination of false-positive results and further improvement of precision can be done at different parts and in different ways in the detection and processing flow. Checking the size of the BLOBs based on the number of pixels in it, can be a first point to reduce the error rate. The reference value for the size of the BLOB can be a fixed parameter or an averaged value based on the information about the BLOB's size in previous frames. In the proposed application task

the size of the BLOB depends on the angle between the light source and the reflection surface. Therefore an averaged size value based on previous frames should be preferred. This elimination of false-positives could take place during the detection phase. For example while processing a frame the BLOB's attributes can be checked continuously for how many pixels have been added lately. If the BLOB stops "growing" the logic of the sequential processing of the frame tells us that all adjacent pixels of the BLOB have been identified. If the size of the BLOB at that point is still under the reference value, it can be dropped from the results for the current frame. Again with respect to the application environment, special cases such as the split of a BLOB on two surfaces might cause further issues. In such cases the reference value for the BLOB's size needs to be adjusted. Aside from the Center-Of-Mass based computation in combination with the brightness value of the BLOB's pixel, other additional improvements have not been realized yet but might be part of future work.

As described in [5] the sequential processing of a frame requires an additional check of adjacency for the BLOBs itself. Dependent on the BLOB's shape or orientation the detection might separate pixels of one BLOB into two different BLOBs. As can be seen in Figure 10 the pixels of the same BLOB have been identified as two individual BLOBs in the first two examples. Or like shown in the third example pixels are used twice.

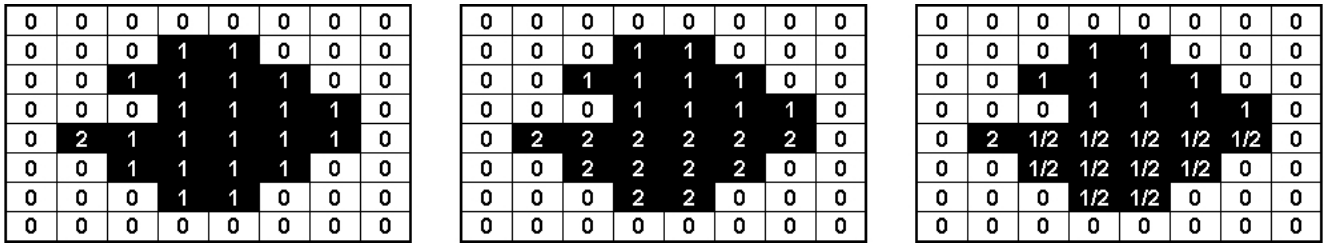


Figure 10. Example of BLOB shape require additional adjacency check.

The adjacency check for BLOBs is based on the same concept as has been explained for pixels earlier.

3 Implementation

The implementation of the systems design was done in the Altera development environment Quartus II (ver. 9.0). Further information about this tool and the proposed target hardware are given in [2, 1].

The overall processing of BLOB detection is separated into several sub-processes. The identification of pixels, which might belong to a BLOB, and the collection of BLOB attributes is very similar for both detection solutions. Only the computation of the center point for the detected BLOBs is different based on the algorithm explained in the previous chapter.

3.1 Attribute Identification

In the first stage the check of the common attribute on the RGB pixel data will identify relevant data. All pixels which do not match the criteria are dropped from further processing. The chosen identification criteria is the brightness value of the pixel. For pixel data in RGB format the brightness information can be computed by:

$$pixel\_brightness = \frac{(pixel\_Red + pixel\_Green + pixel\_Blue)}{3}$$

Every pixel which has a higher value than the specified threshold value, will be saved for the adjacency check. Those pixels are saved in a dual clock fifo from where the adjacency check gets its input data. The threshold value can be configured as a constant value by the user. The concept of an average threshold has not been realized in this approach. The processing flow of the attribute identification is visualized in Figure 11.

The state "UPDATE_THRESHOLD" updates the configured threshold during runtime. This allows the user to adjust the threshold while the BLOB detection is in progress. In addition this state appends an artificial pixel value ("FFF" hex) in the sorted pixel FIFO to mark the end of the previous frame.

3.2 BLOB Detection

The BLOB Detection module implements the adjacency check for the pixel data and sorts them into data structures, referred to as a container. During the detection process the containers are continuously updated until a new frame starts. At that point the BLOB attributes are written into a fifo and the container contents are cleared. In addition the detected BLOBs are checked for adjacency to eliminate the effect which is visualized in Figure 10. The state machine in Figure 12 shows the processing of this module.

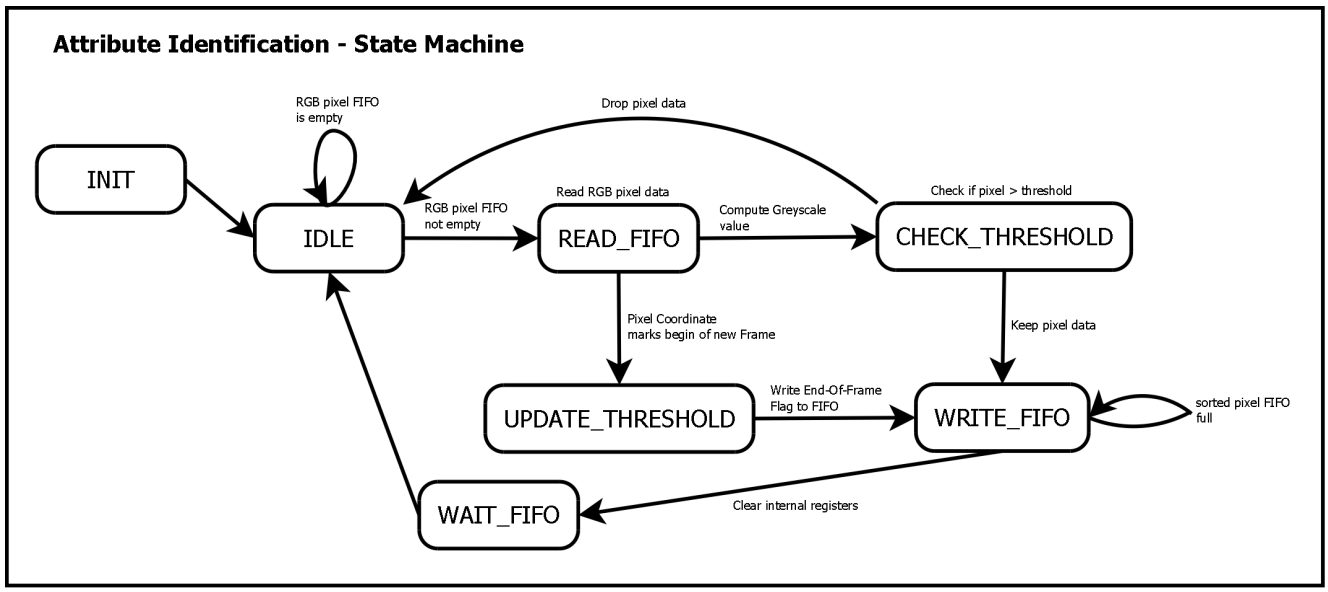


Figure 11. State machine for attribute identification.

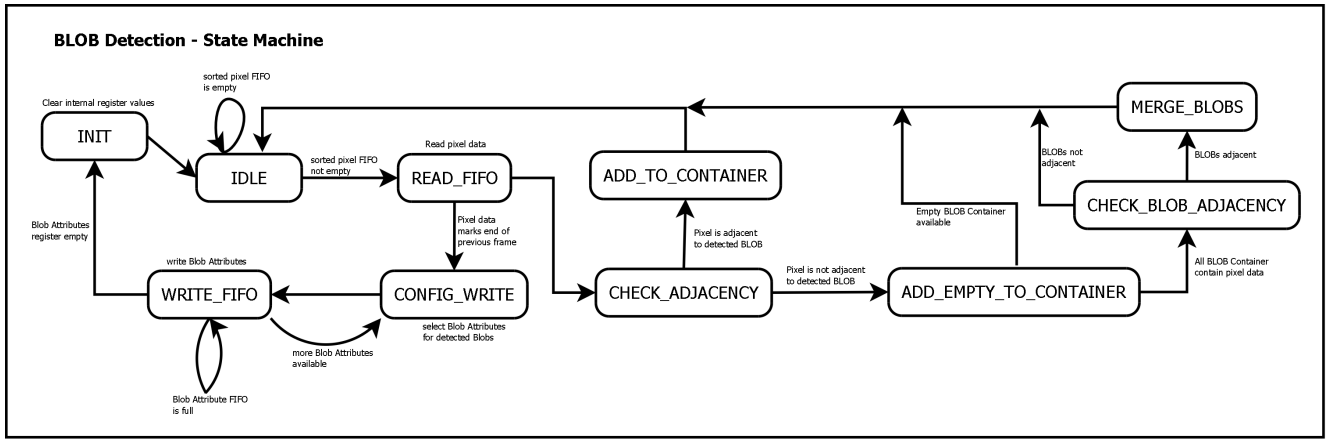


Figure 12. State machine for BLOB detection.

The amount of attributes stored for each BLOB depends on the selected method for computing the center point. For the Bounding Box based approach, only the minimum and maximum X-/Y-coordinates are required to compute the center point of each BLOB. This requires four 11 bit registers for each BLOB. In case of the Center-Of-Mass based computation the summation of the weighted coordinates and the brightness values need to be performed during the detection phase. This allows to reduce the amount of data to be stored down to three 36 bit registers for each BLOB detected.

3.3 Center Point Computation

The last task to be processed in the BLOB detection is the computation of the center point. For the reading of the fifo containing the BLOB attributes and the writing of the BLOB center points into the result fifo, the state machine for Bounding Box and for Center-Of-Mass is the same. As can be seen in Figure 13 the system computes the center points for all BLOBs, that are available in the fifo.

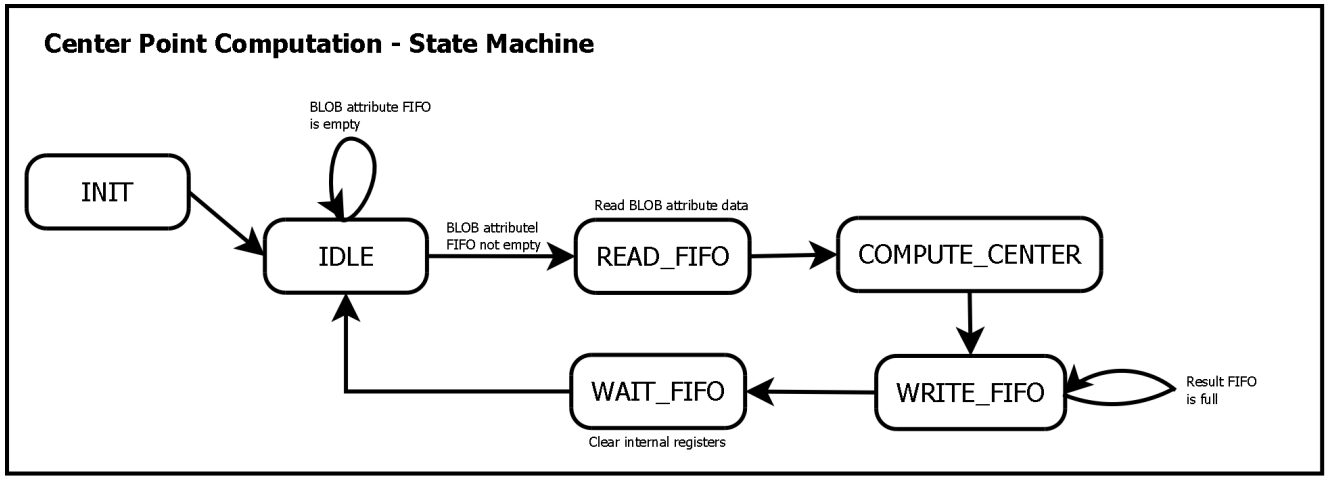


Figure 13. State machine for reading BLOB attributes and writing center points.

The difference is inside the state "COMPUTE_CENTER". Here the algorithm from Section 2.3, BLOB detection, is implemented. For the Bounding Box computation only two additional internal states are required. In the first state the difference between the minimum and maximum coordinate values is computed and divided by two. In the second state the center point can be computed with the minimum coordinate and the result from the previous state.

For the Center-Of-Mass based computation some more complex operations are required. The implementation of the division operation has been pipelined. This was necessary since the divisor is not necessarily a power of two value. The block diagram in Figure 14 shows the application of another IP core from Altera for division of large integer numbers [3].

The pins inSumPosX, inSumPosY and inSumPixels are bus connections and contain the BLOB attributes as input values. On the output pins outBlobXcenter and outBlobYcenter the division results will be available after the pipeline delay of one clock tick. Those results are written into the result fifo from where the serial communication module get its input data.

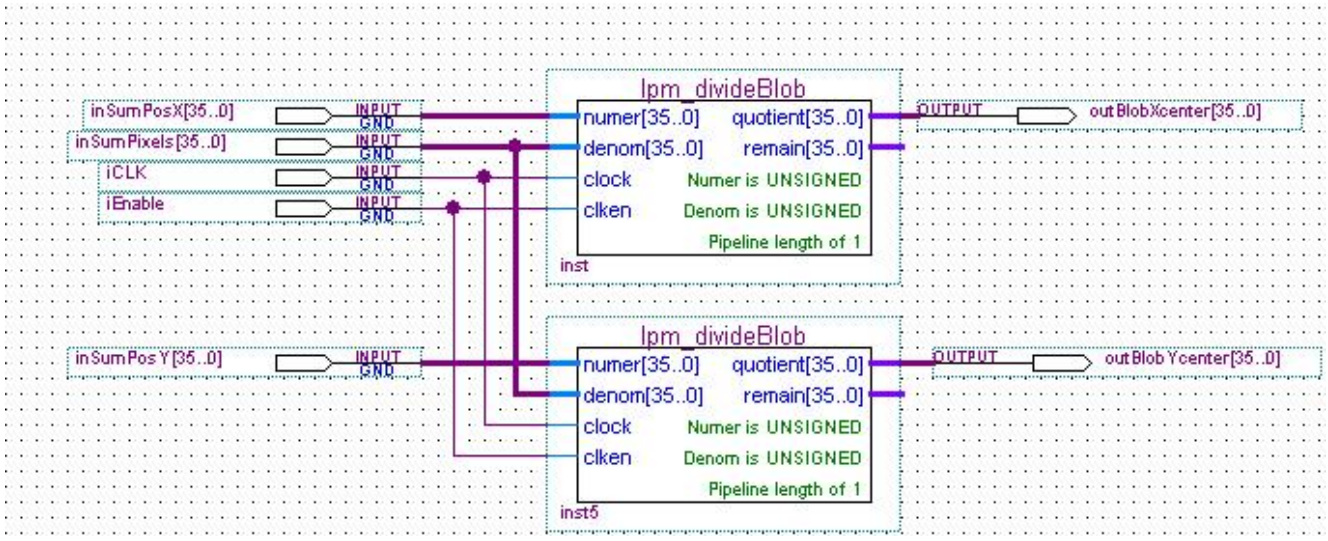


Figure 14. Block diagram for division part in Center-Of-Mass computation.

4 Verification and Validation

For the verification and validation of the BLOB detection results two different input sources have been applied. The S-Video input for the verification of detection accuracy and precision of the different center point computation methods. The CCD camera for performance benchmarking of the system design. Both alternatives allow the observation of the image material while performing the BLOB detection. But only up to a certain speed. For the performance measuring, the visualization of the captured image data from the CCD camera on the VGA display had to be disabled. This was necessary because the VGA controller module did not support higher frame rates.

Based on the specified application area, the shape of the BLOBs to be detected has been estimated as perfect circular and circular shapes with blur effect. The perfect circular shape is given in the proposed application task if the light source, which will be tracked on the projection surface of the Immersion Square, is kept still or the motion is slower than the frame rate of the applied camera device. With faster movement of the light source a blur effect will occur. The angle between light source and the projection surface is another factor which can cause a distortion of the BLOB shape. This distortion might look similar to a blur effect. If the angle is small enough the BLOB's shape will deform into an elliptical form. But in difference to the blur effect the brightest spot of the BLOB will still be in the center. For both problems the solution is the Center-Of-Mass based computation of the BLOB's center points, which has been applied in the system design. Samples of the applied image material for evaluation of the precision is given in the Appendix.

For applying the BLOB detection system it needs to be configured before using the BLOB detection results for further processing. For the applied image material the computed results are shifted by a constant factor on the X and the Y axis. A calibration of the system for each new application environment is expected by the user, to confirm the proper results.

4.1 Precision

For the computation of the BLOBs' center point the Bounding Box and the Center-Of-Mass based method showed the exact same results for the clear BLOBs with perfect circular shape. If a BLOB is not showing any blur effect the, applied method for computing the center point has no influence on the precision. This holds for all applied threshold values. A summary is given in Table 1 and results are visualized in Figure 15.

The table shows what was expected already. If a BLOB is not showing any blur effect the applied method for computing the center point has no big effect on the precision. In the given

	Bounding Box		Center-Of-Mass	
Threshold	**Y**	**X**	**Y**	**X**
0x190	234	311	234	311
0x20B	234	311	234	311
0x286	234	311	234	311
0x301	234	311	234	311
0x37C	234	311	234	311
0x3E0	234	311	234	311

Table 1. Results for center point computation with clear shape BLOBs.

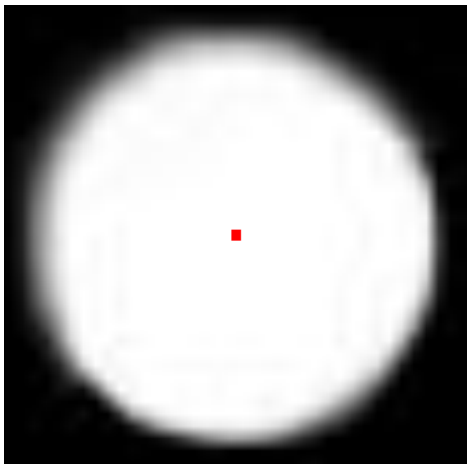


Figure 15. Clear shape BLOB center point computation.

results it can be observed that the estimated center point for both methods is at the exact same position.

The Bounding Box and Center-Of-Mass computation showed different results for the image material showing BLOBs with blur effect. The Center-Of-Mass results turned out to be closer to the BLOB's center point. Results for one particular example are shown in Table 2 and are visualized in Figure 16. Again a proper calibration of the system is required to guarantee correct results.

Center point computation with Center-Of-Mass shows higher precision for BLOBs with blur effect, compared to Bounding Box. With the applied visualization on the VGA output the frame rate of the blob detection was restricted to 12 frames per second. The threshold values which have been used for evaluation of precision are in-between the value range of the BLOBs in the applied image material. This value range is dependent on the applied image material and can not be simply reused for any image source or material. The estimation of the value range is a configuration requirement before using the BLOB detection system. For threshold values above

	Bounding Box		Center-Of-Mass	
Threshold	**Y**	**X**	**Y**	**X**
0x190	226	308	229	307
0x20B	227	308	229	307
0x286	228	307	230	306
0x301	231	305	233	304
0x37C	236	300	238	300
0x3E0	241	299	241	298

Table 2. Results for center point computation with blur shape BLOBs.

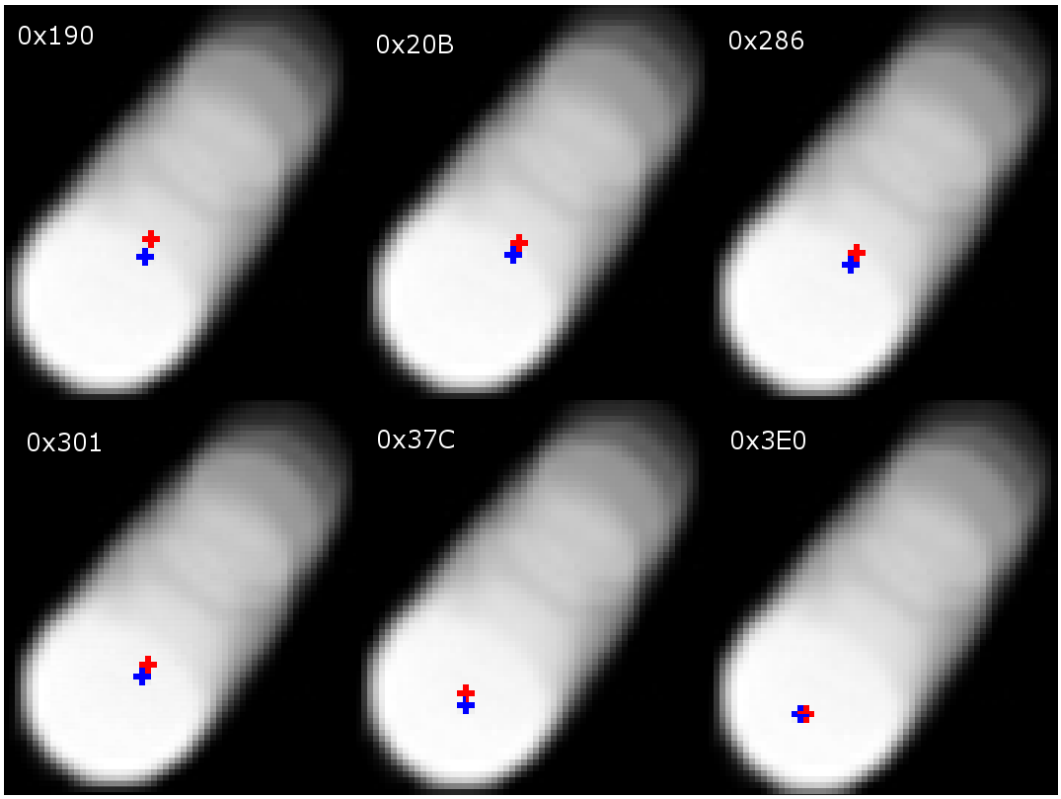


Figure 16. Blur shape BLOB center point computation.

or below the value range the system was not able to detect all BLOBs in the image material accurately.

4.2 Performance

The system performance has been evaluated on a fixed environment setup. Reported values about performance and resource allocation are given in Table 3.

The performance result "fps." refers to the obtained frame rate during the benchmarking. The

	Monitor Output		No Monitor Output	
	Bounding Box	Center-Of-Mass	Bounding Box	Center-Of-Mass
Speed (fps.)	12	12	46	50
Camera Speed (MHz)	25	25	96	96
System Speed (MHz)	40	50	125	125
Max. System Speed (MHz)	72	65	140	189
Allocated Resources on the FPGA				
Logic Elements	7,850	14,430	5,884	13,311
Memory Bits	147,664	273,616	113,364	239,316
Registers	2,260	2,871	1,510	2,078
Verified	*	*	*	*

Table 3. Resource allocation and benchmark results.

"camera speed" is the particular clock rate that is used to read out the CCD sensor. "System Speed" is the clock rate for the BLOB detection module during the performance test and "Max. System Speed" describes the maximum possible clock rate for the implemented design.

For the evaluation with monitor output a faster frame rate would have been possible in theory. But it would have required time consuming configuration of the camera settings and the VGA controller module. For monitoring the image data while performing the BLOB detection the VGA module has to run synchronized with the image capturing module. The applied CCD camera has been used for the evaluation of faster frame rates. It will not be used in the target application for the active tracking device of the MI6 project, because of its missing ability to record infrared light. For this reason it was reasonable to not put more effort into the integration of the CCD camera than was required for the performance measuring.

The CCD camera itself has an average speed of 48 frames per second for the applied configuration parameter. While both BLOB detection approaches would have been able to perform on faster frame rates, the estimation of the maximum performance was again restricted by the input source. The same problem about slow input sources existed in [5] as well and did not allow to test the system for maximum performance. The resource allocation shows that Center-Of-Mass requires about twice as many logic elements and memory bits than Bounding Box.

5 Future Work

The project results offer several opportunities for future work. The additional pre-processing of the image data could increase the center-point precision. With foreground-background segmentation of consecutive frames, the blur effect of moving BLOBs could be reduced. In addition with the calibration of the average BLOB-size, the detection result could be checked for missing or false-positive pixels.

It has been shown in Section 4.2 that the BLOB detection could not be tested for its maximum performance. Both available input sources turned out to be a bottleneck for the acquisition of image material. A recommended step would be the integration of the BLOB detection approach into a camera with an onboard FPGA. With direct access to the sensor of a high performance camera, the BLOB detection can be evaluated for its maximum processing speed.

The detection of BLOBs in its current implementation is working for up to five BLOBs. The number is based on the required points for estimation of position and orientation of the user in a CAVE environment [13]. The extension for detecting more BLOBs is possible. For the detection of BLOBs the system does not track the movement of the BLOBs. The index for the detected BLOBs is based on the order of how the frame is processed. This works usually from the top-left corner to the bottom-right corner of the frame. The continuous tracking of BLOBs would be another possible functionality to improve the system. This would allow to estimate the speed and direction of the BLOBs movement as well.

For the continuation of the MI6 project the BLOB detection system needs to be tested with the Immersion Square [8]. This requires the extension of the FPGA system with a specialized infrared camera as the input source or the integration of the BLOB detection into an infrared camera. At this time decisions about the future of the project are pending.

6 Conclusion

With Bounding Box and Center-Of-Mass two common methods for the estimation of a BLOB center point have been implemented. The evaluation has shown reliable precision results with respect to the given application area. As it has been estimated in [5] the Center-Of-Mass based approach shows higher precision and is the recommended solution for the given application task.

The BLOB detection approach is working for threshold based and specialized for BLOBs which consist of white and light grey pixels on a black image background. The BLOB detection has turned out to be non-critical with respect to performance and timing requirements for the proposed application task. Both available input sources have been identified as a bottleneck for the system's processing speed. This has proven the FPGA design to be well qualified for the proposed computer vision problem, if faster input sources can be provided.

For the evaluation and further processing a module for transmitting results on the serial interface has been designed, tested and applied. The maximum performance of the serial interface is high enough for even faster frame processing rates, as described in Section 2.1. The output format of the computation results can be easily changed in the system design. Or if further information about the BLOBs is computed on the FPGA system and needs to be transmitted as well, such as direction of movement or speed of the BLOBs.

This report has shown that a BLOB detection system can be successfully implemented and evaluated on a FPGA platform. Validation and verification showed reliable results and the advantages of image processing tasks designed in hardware. The work required a higher time effort compared to the implementation of a similar system in a high-level language, such as C++ or Java. But for customized solutions with high demand on performance and precision specialized hardware outperforms high-level software implementations. FPGAs are proven to be a good way for short prototyping development cycles to decrease time-to-market.

Acknowledgments

Thanks to my advisors Prof. Rainer Herpers (Bonn-Rhein-Sieg University of Applied Sciences, Germany) and Prof. Kenneth B. Kent (University of New Brunswick, Canada) for their help and stimulating suggestions. This work is supported in part by Matrix Vision, CMC Microsystems and the Natural Sciences and Engineering Research Council of Canada.

References

[1] Altera, 101 Innovation Drive, San Jose, CA 95134. *DE2 Development and Education Board - User Manual*, 1.41 edition, 2007.

[2] Altera, 101 Innovation Drive, San Jose, CA 95134. *Quartus II Handbook Version 9.1*, 1 edition, 2009.

[3] Altera Cooperation, 101 Innovation Drive, San Jose, CA 95134. *Integer Arithmetic Megafunctions User Guide*.

[4] Altera Cooperation, 101 Innovation Drive, San Jose, CA 95134. *SCFIFO and SCFIFO Megafunctions User Guide*, January 2010.

[5] A. Bochem, R. Herpers, and K. B. Kent. Acceleration of blob detection within images in hardware. Technical Report TR 09-197, University of New Brunswick, Faculty of Computer Science, Fredericton, NB, E3B 5A3, December 2009.

[6] C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti. Surround-screen projection-based virtual reality: the design and implementation of the cave. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 135–142, New York, NY, USA, 1993. ACM.

[7] J. Hammes, A. P. W. Bőhm, C. Ross, M. Chawathe, B. Draper, and W. Najjar. High performance image processing on fpgas. In *In Proceedings of the Los Alamos Computer Science Institute Symposium. Santa Fe, NM*, 2000.

[8] R. Herpers, F. Hetmann, A. Hau, and W. Heiden. Immersion square - a mobile platform for immersive visualisations. University of Applied Sciences Bonn-Rhein-Sieg, 2005.

[9] M. E. Latoschik and E. Bomberg. Augmenting a laser pointer with a diffraction grating for monoscopic 6dof detection. *Journal of Virtual Reality and Broadcasting*, 4(14):–, jan 2007. `urn:nbn:de:0009-6-12754,`, ISSN 1860-2037.

[10] P. Milgram and F. Kishino. A taxonomy of mixed reality visual displays. *IEICE Transactions on Information Systems*, E77-D(12), December 1994.

[11] Terasic. *Terasic TRDB-D5M Hardware Specification*, June 2008.

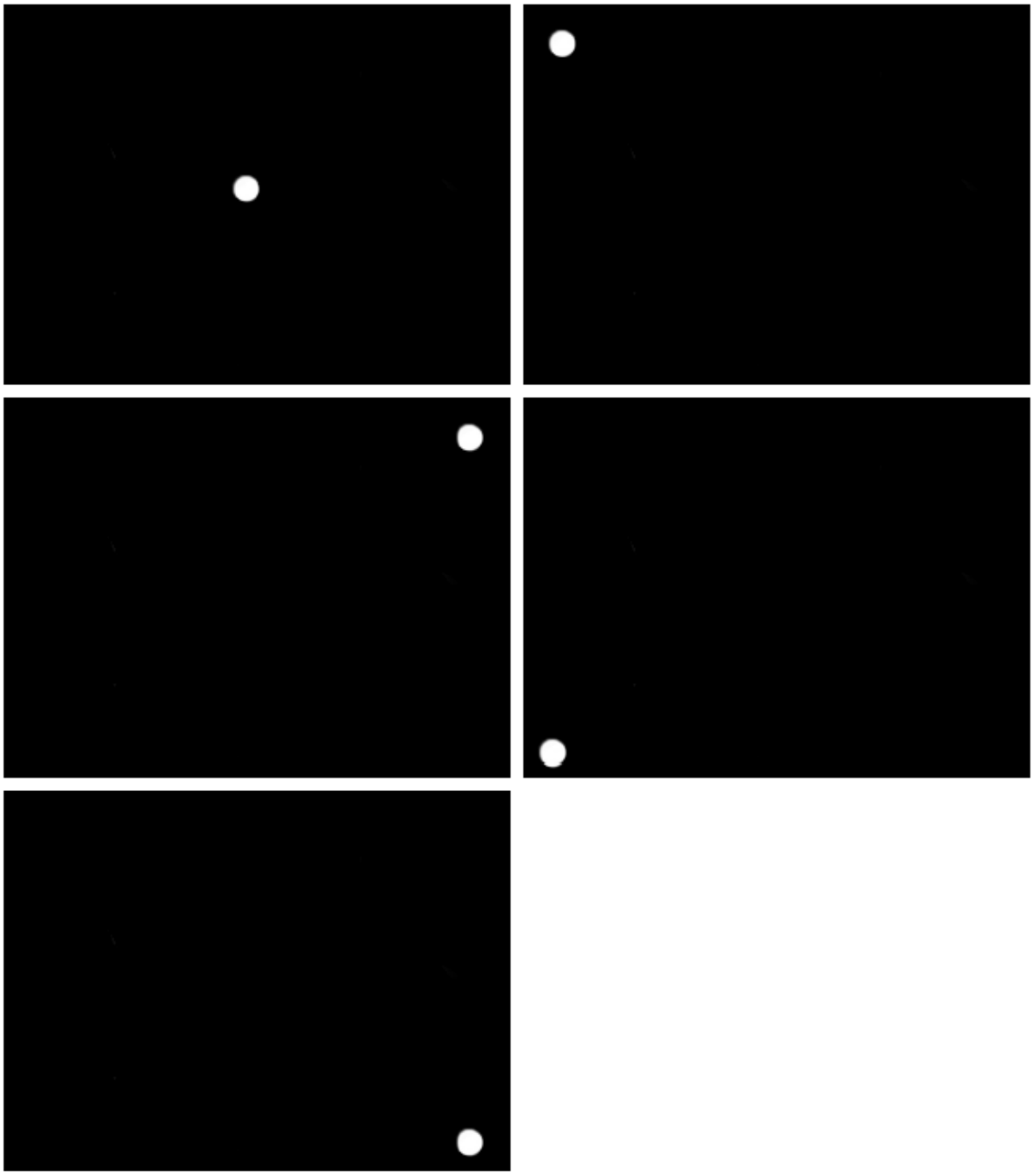
[12] Texas Instruments, Post Office Box 655303, Dallas, Texas 75265. *MAX232, MAX232I DUAL EIA-232 DRIVERS/RECEIVERS*, March 2004.

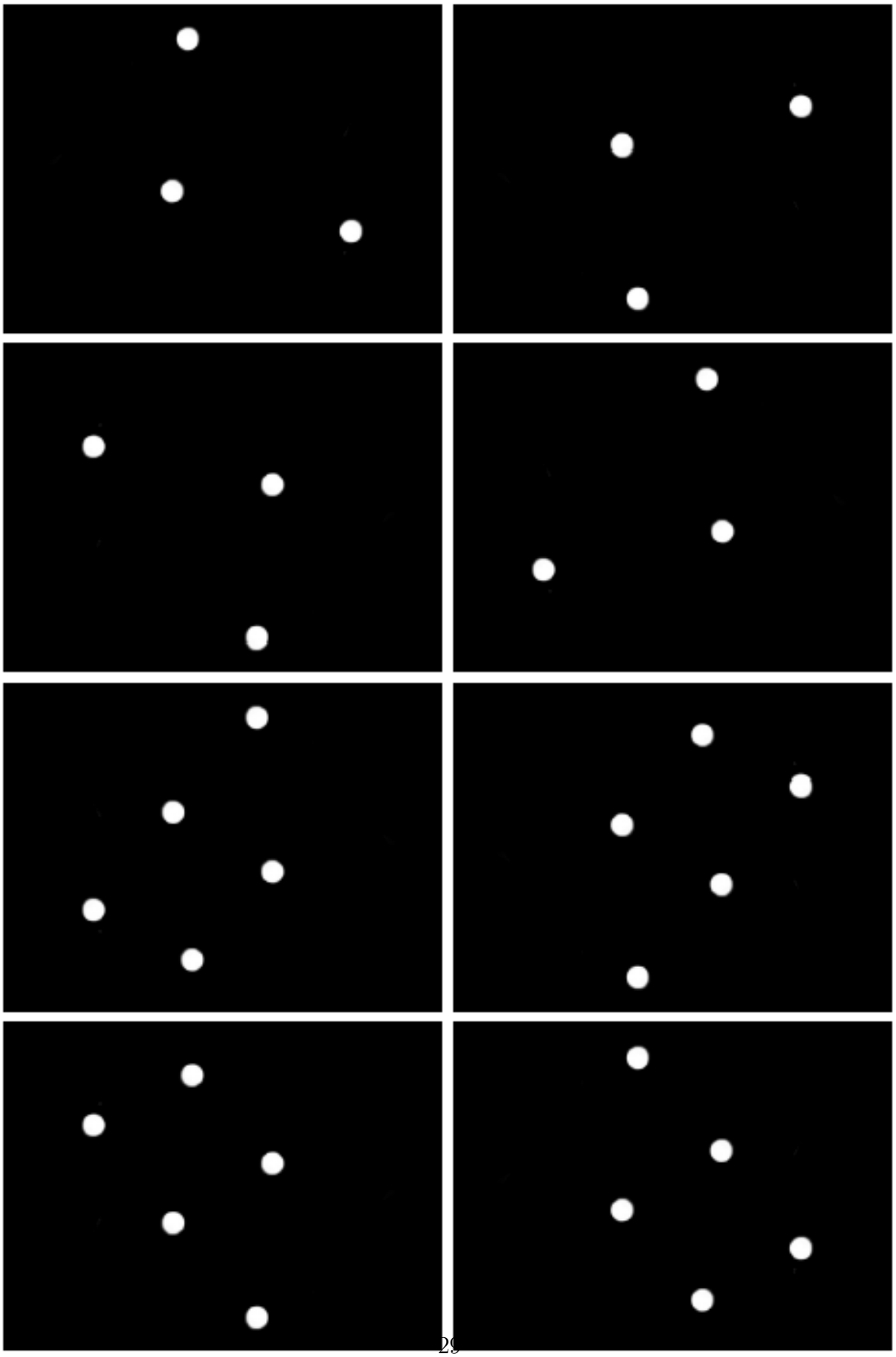
[13] C. Wienss, I. Nikitin, G. Goebbels, K. Troche, M. Gőbel, L. Nikitina, and S. Műller. Sceptre - an infrared laser tracking system for virtual environments. In -, volume isbn 1-59593-321-2. Proceedings of the ACM symposium on Virtual Reality software and technology VRST 2006, 2006.

Appendix

Sample images for perfect circular BLOBs

The following images have been used for the evaluation of precision.





Sample images for BLOBs containing blur effect

The following images have been used for the evaluation of precision.