Smart Camera System-on-Chip Architecture for Real-Time Brush Based Interactive Painting Systems

Luc Claesen, Peter Vandoren, Tom Laerhoven, Andy Motten, Fabian Fiore, Frank Reeth, Jing Liao, Jinhui Yu

Smart Camera System-on-Chip Architecture for Real-Time Brush Based Interactive Painting Systems

Luc Claesen[*], Peter Vandoren[*], Tom Van Laerhoven[*], Andy Motten[*], Fabian Di Fiore[*], Frank Van Reeth[*], Jing Liao[$], Jinhui Yu[$]

([*])[1] EDM Expertise Center Digital Media, University Hasselt, tUL, IBBT
Wetenschapspark 2, B3590 Diepenbeek Belgium
{firstname.lastname}@uhasselt.be
([$]) Dept. Digital Media and Internet Technology, CAD & CG State Key Lab, Zhejiang
University, Hangzhou, 310027, P.R. China
{liaojing,jhyu}@cad.zju.edu.cn

Abstract. Interactive virtual paint systems are very useful in editing all kinds of graphics artwork. Because of the digital tracking of strokes, interactive editing operations such as save, redo, resize etc. are possible. The structure of artwork generated can be used for animation in artwork cartoons. A novel System-on-Chip Smart Camera architecture is presented that can be used for tracking infrared fiber based brushes as well as real brushes in real-time. A dedicated SoC hardware implementation avoids unnecessary latency delays caused by PC based architectures, that require communication-, PC and GPU frame-buffer delays, thereby considerably enhancing the interactivity experience. The system is prototyped on an FPGA.

Keywords: System-on-Chip, SoC, FPGA, digital painting, active canvas, animation, cartoons, artwork, editing, Video Processing, HCI, human-computer interface, real-time, Smart Camera, embedded video, FTIR, video pipeline.

1   Introduction

During the centuries several kinds of art and graphics representations have been in use up till today. Most of the artwork is still produced by means of traditional tools such as pencil and paint-brushes. Since the introduction of graphic computer displays, software painting applications have been developed [1]. Most current computers provide software where simple paintings can be made by means of a computer mouse, a touch screen or a stylus/tablet input device. Usually a color bit-map file is generated

---

by the input device by adding and combining one or more bit-map layers of the canvas and the virtual brush at the mouse pointer or pen-stylus cursor.

The most widely available paint systems make use of the "rubber stamp principle". This means that a specific two-dimensional image "a stamp", with texture, size and color is attached to the mouse or stylus cursor. When moving the mouse or drawing pen over the canvas, the image of the "rubber stamp" is added to the background bitmap image.

The use of computers in art- and graphics image creation has several advantages. The artwork can be saved for later retrieval. It is also possible to edit the artwork to adapt it better to the ideas of the artist or graphics designer. Mistakes can be coped with in an easier way by means of "undo" operations. Parts of designs can be selected, copied and pasted to generate other images. When individual strokes are recorded and represented as parametric curves, they can be manipulated afterwards by the user, to adapt the shape and size. An even further step is the composition of artwork for use in the generation of moving images and cartoons [18,19].

Besides simple "rubber-stamping" based paint systems more complex models have been developed over the years as discussed next.

1.1   Active-canvas digital paint methods.

More detailed models for digital painting have recently been developed which attempt to mimic the real painting process using detailed simulation of the complex interaction between brushes and the paint canvas. These are so-called "active-canvas" methods. They model the paint as a solvent fluid that can flow and evaporate and that contains color pigments and a fixation binder glue. The pigments are small 10..150nm particles having their own dispersion and color characteristics. The canvas is characterized by its non-uniform texture and solvent/paint absorption characteristics.

Curtis e.a. [2] introduce an empirically based watercolor paint system that is based on an ordered set of translucent glazes. Each wash is simulated by three layers: a shallow-water layer, a pigment deposition layer and a capillary layer where the water is absorbed by and diffused in the paper. Hereby all kinds of water-color effects such as wet-on-wet and wet-on-dry, dry-brush effects, edge darkening, backruns, granulation and separation of pigments, flow patterns etc. can be simulated. This process is based on the finite element simulation of the Navier-Stokes fluid-dynamics equations, Poisson diffusion equations and Kubelka-Munk color model. The IMPaSTo system [3] specifically models paint methods such as oils, acrylics or gouache, based on a conservative advection scheme that simulates the basic dynamics of paint. These physical based painting models are computationally very intensive, posing a problem for real-time execution. Van Laerhoven e.a. [4] realized new algorithms suited for the real-time execution on GPUs (Graphics Processing Units). The above methods [2-4] are all based on differential equations like Navier-Stokes and Poisson, describing the macroscopic effects of the paint dynamics. Numerical integration methods of the differential equations based on forward Euler can result in unstable behavior in case of fast changes (fast brush strokes). Backward Euler based methods on the other hand,

although stable, require a number of iterations, adding to the computational complexity. To cope with these problems, Chu e.a. [5] introduced the use of the Lattice-Boltzmann equations for paint-simulation, modeling the physics of the movement of paint particles at a mesoscopic level. This method is implemented on GPU's and can model the physics of ink flow in absorbent paper.

A massively parallel hardware implementation of the Lattice-Boltzmann model for active canvas simulation has recently been prototyped making use of FPGA's [20].

1.2   Artist-Computer Interfaces for Digital Paint systems.

To create digital paintings, a computer mouse can be used. Although computer mice have proven their usefulness in a lot of daily computer tasks, the expressiveness for digital painting is rather limited. Computer mice only record relative movements and have little expressiveness for pressure input. Therefore professional artists currently prefer tablets and stylus pens. These systems have a good absolute accuracy with respect to the drawing tablet and also provide a measurement of the drawing force along the axis of the pen stylus shaft. This allows paint programs to model the force exercised on the pen tip while drawing and consequently generating thinner or thicker pen strokes depending on the force employed by the artist.

Mueller [9] describes a real-time painting system based on frustration of internal reflected light in a prism. The light in the prism is generated, via an optical setup, by the scan signal of a CRT (Cathode Ray Tube) image. The frustrated light generated by a drawing utensil can be detected by a (photo multiplier) light sensor in a synchronous way with the CRT scan signal. This enables a quasi real-time brush detection. The rendering of the painting result on the screen is not co-located with the drawing surface.

Greene [6] introduced the drawing prism, commercialized under the name OptiPaint [7]. This is a drawing system built around a prism. One side of the prism is used as a drawing surface. Wet brushes and other drawing tools can be detected by means of frustration of a light source which is normally reflected on the drawing surface side of the prism by means of total internal reflection. The frustrated light caused by the contact of drawing tools is captured by means of a video camera located at an other side of the drawing prism. This system requires a setup with a bulky optically transparent prism of which one side is at least the size of the drawing surface and the other side is at least large enough for enabling a good imaging by a video camera. The aspect ratio of the camera side deviates from the standard aspect ratios of video cameras, thus requiring a special optical setup or camera design. The rendering of the painting result on the screen is separate from the drawing surface [6,7] just like the method of Mueller [9]. This means that a user is drawing or painting on a tablet or surface where he/she does not directly see anything of his/her artwork. The coordination of drawing on one surface and viewing the result on a separate screen requires special training and concentration. It is not intuitive to users.

Carver Mead et al. [8] proposed a paintbrush stylus sensed by a capacitive sensor array. Because of the resolution of the capacitive sensors the main input parameters are the coordinates. Because of the capacitive sensing mechanism only electrically

4   **Luc Claesen*, Peter Vandoren*, Tom Van Laerhoven*, Andy** Motten*, Fabian Di
Fiore*, Frank Van Reeth*, Jing Liao$, Jinhui Yu$

conductive brushes can be used in this system. The capacitive sensor technology is what is used in current tablet computers and smart phones such as the Apple iPad and iPhone.

Electro-magnetic tablets are probably the most widely used input devices for paint-systems. In Wacom tablets, in one phase, two orthogonal grids of wires generate alternating and localized electromagnetic fields on the tablet. This transfers some energy to an LC tank inside the stylus pen. This energy is used in an other phase to generate an electromagnetic field from the pen that can than be sensed by the same orthogonal grid of wires in the tablet. These tablets can generate an accurate stylus position and can also often sense the one dimensional force in the direction of the stylus axis. By using a layer of optically transparent wires on top of an LCD display, the Wacom Cintic system [10] integrates the input tablet with the drawing screen. This provides direct feedback of the drawing result under the pen tip. Because tablet based systems use stiff styluses, painting with a stiff stylus is different than painting with real brushes with flexible tufts made of camel, hog, squirrel hair or other natural or synthetic fibers. In Western painting and Chinese calligraphy, the specific movement and deformation of the brush tuft is crucial for achieving special effects. Although some tablet systems [10] provide co-located drawing input/painting display, they suffer from the distance between the drawing plane and the display plane which causes a parallax effect. Depending on the relative position of the artists, the pen tip and its drawing result on the screen will be different.
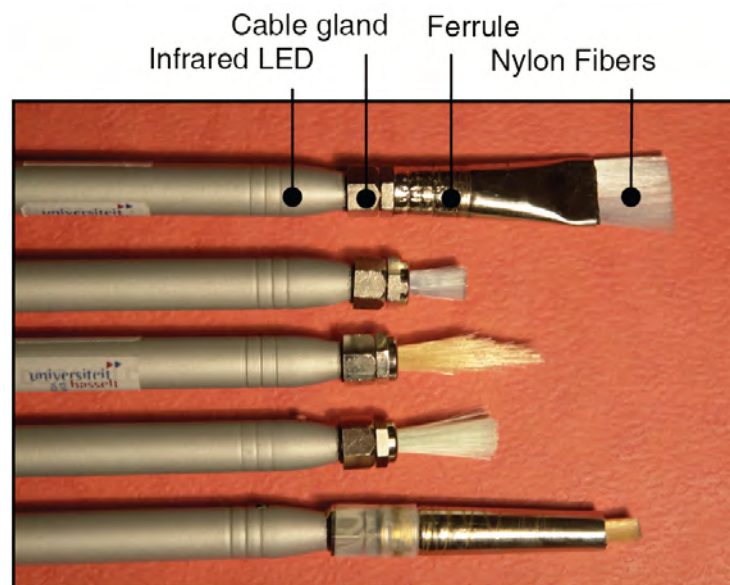
Fig. 1. Infrared brushes used in the IntuPaint system [12].

The IntuPaint system [12] uses electronic brushes with bristles made of optically transparent fibers (Fig. 1.). An infrared light source inside the brush propagates light through the transparent fibers by means of total internal reflection. The light exits at the bristle tips. When IntuPaint brushes are in contact with a diffuser screen, the tuft footprint and position can be imaged by an infrared camera behind the screen. In

IntuPaint, the diffuser screen is used to display the result of painting, thereby providing co-located input/display. By using a brush with bristles, an artist can exploit the deformation of the brush tuft during drawing by brush movements, inclination and pressure on the canvas. Fig. 2. shows the system in use. Because of its infrared light emission operating principle, the IntuPaint system requires specially built brushes and drawing tools.
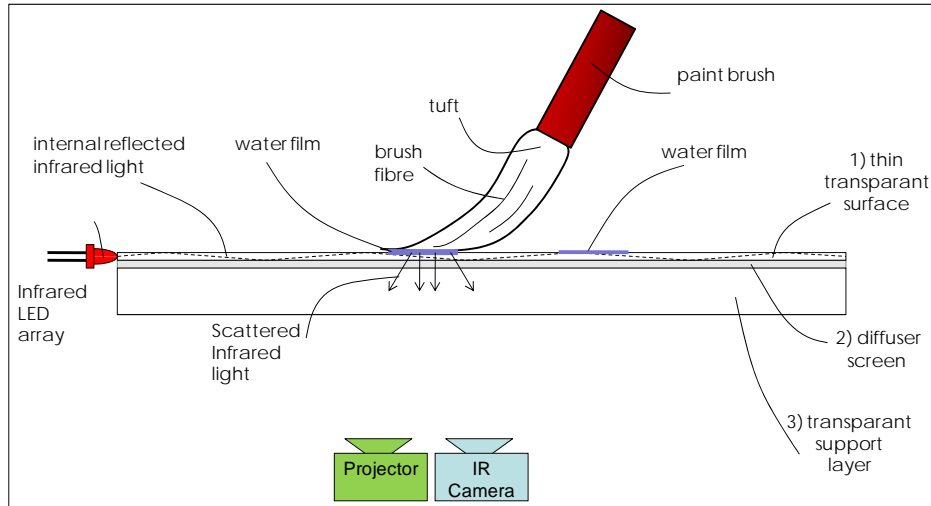

Fig. 2. IntuPaint system [12] in use.

All previous methods still limit artists in their expressiveness in comparison to traditional painting with brushes and paint. To solve this problem, the authors have introduced the FluidPaint system [13]. Both the IntuPaint and the FluidPaint system are built on top of physical based painting simulation software [4] running on a high-end GPU powered PC. The novelty of FluidPaint is that it uses *real brushes* on a *co-located painting input/display canvas* surface. To enable the real-time and low-latency virtual painting, a dedicated Smart Camera based SoC architecture has been developed and is presented in this paper.

In Section II, the system setup of the FluidPaint virtual painting system with real brushes is presented. In Section III the usage of the Smart Camera is introduced. The Smart Camera hardware architecture and prototype is presented in Section IV. Section V formulates conclusions and further work.

2   Virtual Painting With Real-Brushes.

FluidPaint is a novel digital painting system that operates with real brushes. In this section the operation principle of FluidPaint is briefly described. The reader is referred to [13] for a more in-depth presentation and user tests.

6   **Luc Claesen*, Peter Vandoren*, Tom Van Laerhoven*, Andy** Motten*, Fabian Di
        Fiore*, Frank Van Reeth*, Jing Liao$, Jinhui Yu$

Fig. 3. Operation principle of the FluidPaint digital painting system.

The FluidPaint paint canvas constitutes the key component of the system. It is a 3-layered system as shown in Fig. 3.

Layer 1 consists of a 0.6mm thick transparent surface layer. On the four sides of the transparent layer there is an array of 950nm infrared (IR) LEDs, introducing IR light inside the transparent layer, which acts as an optical waveguide. This IR light is propagated in the layer by means of total internal reflection and normally exits the layer at the other side.

Layer 2 consists of a diffuser screen. A projector positioned below the paint canvas can project an image of a painting on the diffuser screen (Fig. 2).

Layer 3 is a transparent support layer giving mechanical strength to the drawing surface.
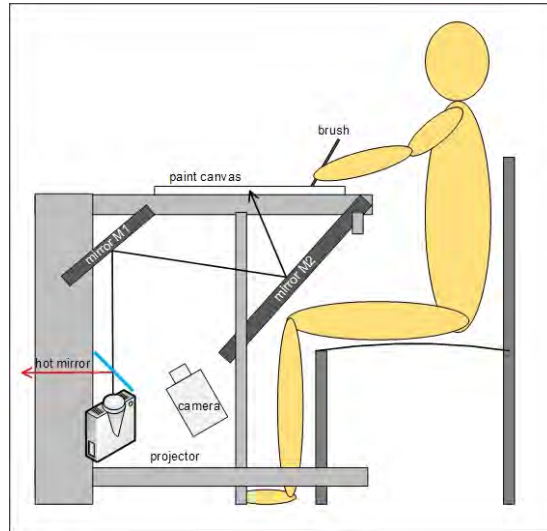
Fig. 4. FluidPaint global system setup

Fig. 4. illustrates how the paint canvas is set up as a painting system. Two mirrors (M1 and M2) are used for folding the optical projection path of the projector. A hot mirror placed before the projector lens blocks the infrared light emitted by the projector. Fig. 5. is a photograph of the side view of the actual FluidPaint prototype. The bottom side of the paint canvas in use and mirror M2 is clearly visible.

Fig. 5. Photograph of the FluidPaint System Setup

8   **Luc Claesen*, Peter Vandoren*, Tom Van Laerhoven*, Andy** Motten*, Fabian Di Fiore*, Frank Van Reeth*, Jing Liao$, Jinhui Yu$

Fig. 6. Left: dry brush A (12mm) and wet brush B (10mm). Right: the infrared footprint of the two brushes. Notice the clear footprint image for the wet brush B caused by frustrated total internal reflection. Dry brush A does not generate a footprint image.

When a wet brush makes contact with the top layer, as illustrated in Fig. 3, the IR light inside the top layer is not internally reflected anymore and can propagate outside the layer and propagate inside the water in the wet brush until it arrives at the brush bristles. Here the IR will be scattered in different directions according to the bristle structure. An IR camera placed below the screen can capture this IR image. It is in fact a footprint of the brush contact surface as illustrated by brush B in Fig. 6.. When a dry brush is put into contact with the surface layer, there is nearly no optical contact and consequently the light inside Layer 1 remains internally reflected and is not frustrated. Consequently no image is visible by the IR camera as is illustrated by brush A in Fig. 6.. When using wet brushes, wet traces are left on the drawing surface. As shown in Fig. 3 these waterfilms do not frustrate the internal IR light reflection. At the interface of the Layer 1 surface and the waterfilm the IR light leaves Layer 1 and propagates further inside the waterfilm under a similar angle. When it reaches the top of the waterfilm it is internally reflected again and propagates back into the transparent Layer 1.

This input method of painting with real and wet brushes results in a feeling and expressiveness like in real-world painting. The feeling and brush-hand feedback to the artist is similar as when painting with real paint. The IR camera only images the brush contact surface. During real painting it is also only the contact surface that really matters. The image of the contact surface images the real brush and bristle structure in the contact zone. Such a brush footprint can be very well used in physical model based painting systems [4]. This enables an artist to express very small nuances due to the specific brush movements and complex tuft deformation during the act of painting. The painting input and rendered painting canvas display is co-located. The artist directly sees the result of the painting under the brush as illustrated in Fig. 7. In Fig. 8 the bristle structure dependent brush stroke output and paint result is visible.

Although also based on the principle of total internal reflection, multi-touch systems as introduced by Han [17] are not directly usable for painting like the 3-Layer structure of FluidPaint. These multi-touch systems usually consist of a ~1cm thick transparent acrylic layer with IR leds on the side. On top of this acrylic layer there is a "compliant layer", usually made of silicone. On top of the compliant layer is a diffuser screen. Touch is detected by the IR light frustration of the contact points of the middle silicone with the bottom acrylic layer. Using a brush on such a system requires an

unnatural high force from the brush to make contact through the diffuser screen and through the silicone layer with the acrylic layer. No detailed footprint as in FluidPaint is possible with such a system. Multi-touch systems without compliant layer have also been realized. They consist of a ~1cm thick acrylic layer with a diffuser screen under it. Although here wet brushes could be used like in FluidPaint, the distance (~1cm) between the brush contact surface and the diffuser screen is too high, resulting in a very unclear and blurred image below the diffuser screen and would also reintroduce an undesired parallax.
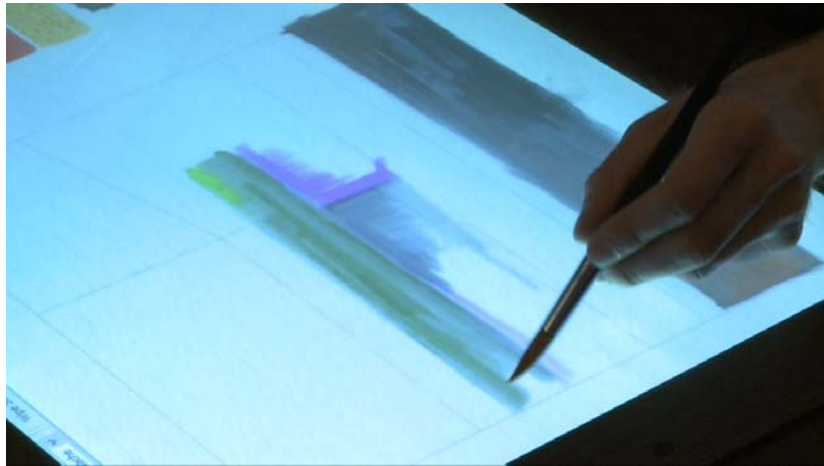

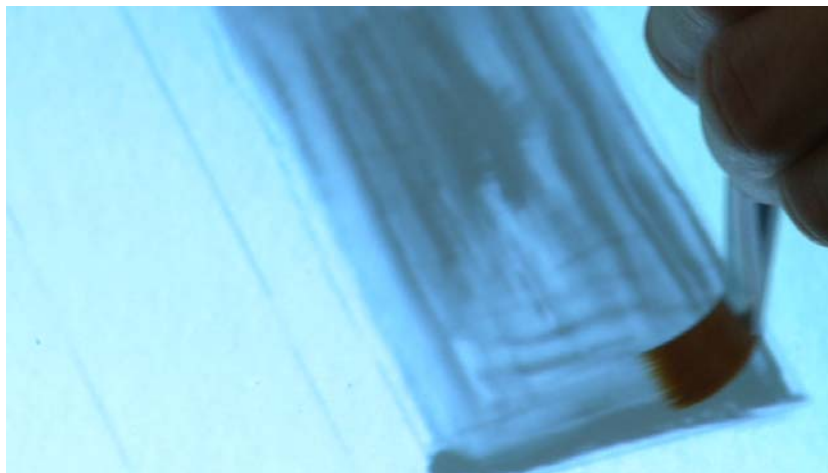**Fig. 7.** Interactive Painting in FluidPaint with real wet brushes.

Fig. 8. FluidPaint screenshot showing how the bristle structure of the brush influences the rendered painting result on the virtual canvas.

10   **Luc Claesen*, Peter Vandoren*, Tom Van Laerhoven*, Andy** Motten*, Fabian
Di Fiore*, Frank Van Reeth*, Jing Liao$, Jinhui Yu$

2.1   Smart Camera Usage.

The constant evolution in VLSI technology expressed by Moore's Law is an enabler for the complete integration of Systems-on-Chip (SoC). Because of the use of compatible silicon technologies, several applications already integrate CMOS cameras and image processing in Smart Cameras [16] on the same SoCs [11]. A widely used Smart Camera SoC is the PixArt infrared blob-position detector in the Nintendo Wii remote controller. In this section the usage of a Smart Camera in the FluidPaint digital painting system is presented.

The first prototype of the FluidPaint system [13] made use of a standard machine vision camera. It was a PointGrey GRAS-20S4C camera with an IEEE 1394b FireWire interface to the host PC. Using standard cameras has the advantage of fast prototyping. The disadvantage is however that in applications such as digital painting, there are very stringent real-time requirements, both on the overall processing time as well as on the latency between brush input and processed display reaction. A standard machine vision camera sends full images to the PC, where further image processing is to be done to detect the brush footprint images and positions. It is well known that streaming video data and real-time image processing are very computation intensive. Delays in a traditional video pipeline occur in the camera, the transmission via Cameralink, Firewire or Ethernet, the capture in the receiving PC, the preprocessing, the transfer from the PC memory to the GPU memory and the displaying. In addition, standard cameras add delays between the capture of the image in the camera sensor and the delivery of the image processed results to the painting application. This delay which usually consists of several frame periods, causes a latency between painting with a brush and displaying the result on the screen. This is noticed by the fact that the paint on the canvas screen does not immediately follow the brush movements.

As the application PC is already very occupied with the paint simulation software, the combination with the camera image streaming communication and processing limits the real-time simulation effects.

A Smart Camera SoC architecture can 1) perform the required image processing in hardware and 2) reduce the delay time from image capture to processing. Implementing the image processing in dedicated hardware relieves the host PC from a compute intensive part, but also allows to reduce the required communication to the brush position and footprint only. The direct processing of the image data in hardware can avoid the use of unnecessary frame buffers in the camera and the PC. In a SoC, frame buffers can be reduced to the absolute minimum and can directly be employed for the required image processing at hand. In case of a controlled environment lighting, frame buffers could even be avoided.

An SoC architecture also allows for a direct per-frame camera control without lost frames. Hereby the SoC can directly change the camera field of view, shutter times, gains, black level calibration etc.

3   Smart Camera SoC Architecture.

3.1   Image Processing Pipeline

An infrared camera captures the image of the contact of the wet brush with the canvas as shown in Fig. 6. Image processing [14,15] and segmentation enable the accurate determination of the brush location on the canvas and the determination of the detailed brush footprint image. The segmented brush footprint image is the input for the physical model-based paint simulation software [4].

The image processing consists of the following actions:

- Preprocessing of the sensor data.

- 5x5 Low pass Gaussian filter.

- Background subtraction.

- 5x5 Population thresholding.

- Contrast enhancement.

- Segmented Footprint Center of Gravity Determination.

- Brush footprint image identification.

- Camera/Screen image rectification.

- Transmission to PC application.

The first steps in the image processing isolate and enhance the image of the footprint. The location of the footprint could be determined by its edges or its horizontal or vertical histograms. In order to obtain a more accurate and stable position determination, the center of gravity of the footprint is determined. The footprint image around this center of gravity is transmitted to the painting application on the PC. Painting applications can make use of this property.

The co-location of the brush input/canvas screen requires a transformation of camera coordinates to screen coordinates of the projector. Careful placement of the mirrors M1 and M2 (see Figure 2.) can generate a rectangular projector image corresponding to the canvas screen. It is however very difficult to position the camera so that its canvas field of view corresponds to the projector canvas field of view. Distortions due to the other placement of the camera, due to lens distortions (cushion effect), due to different pixel densities etc. need to be compensated. This camera image rectification is done by a grid of calibrated control points in which camera coordinates are transformed to projector coordinates by means of bilinear transformation [15].

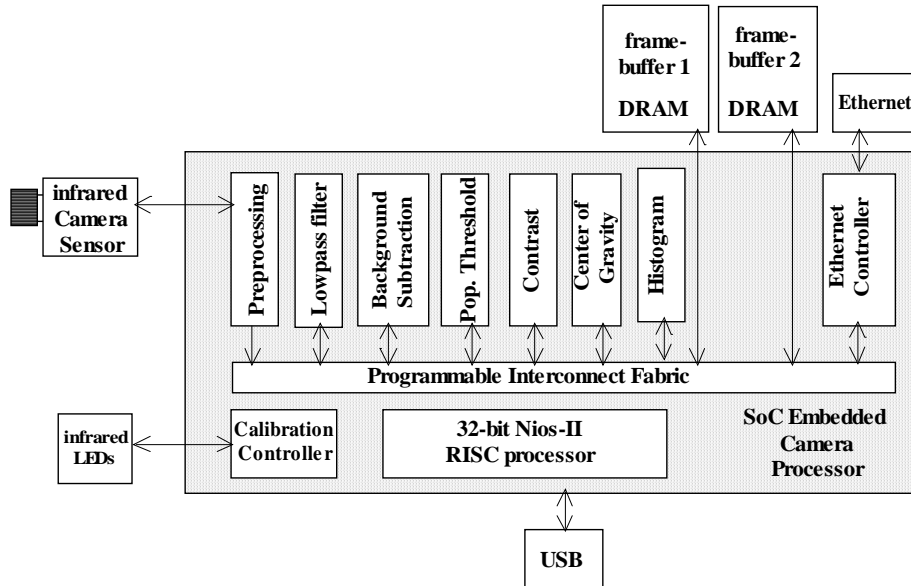## 3.2   **Smart Camera SoC Processor Architecture**

Fig. 9. Smart Camera SoC Architecture

Fig. 9 illustrates the Smart Camera architecture. The architecture consists of a programmable interconnect fabric that allows the flexible arrangement of image processing operations in a pipeline. This architecture supports the operators required by the Image Processing Pipeline: camera preprocessing, lowpass Gaussian filtering, Background subtraction, Population based Thresholding, Contrast enhancement, Center of Gravity Calculation, Histogram calculation. The 5x5 lowpass filter and the 5x5 population thresholding operators use on-chip line buffer memories. Two independent DRAM frame buffer based memories can be used. A first frame buffer stores the background image for background subtraction. This background image is used for an adaptive background determination by means of a first order IIR (Infinite Impulse Response) temporal image filter. This can be interesting in environments with (slowly) changing infrared background lighting. A second frame buffer is used to store the incoming image. After the location of the brush has been determined, the brush position and footprint image are sent to the host PC via a direct Ethernet link.

The image sensor camera and all of the image processing operators and communication are controlled by a 32 bit RISC processor. The processor can also communicate via a USB link to the host PC. In this way the application PC can indirectly control all of the functions in the Smart Camera system.

Due to transportation, the camera/projector setup could become misaligned. Therefore four 950nm infrared LEDs have been placed on the corners of the lower side of the canvas screen. The RISC processor can control these LEDs and their location

determination. In this way the camera coordinates can automatically be aligned with the projector screen coordinates.
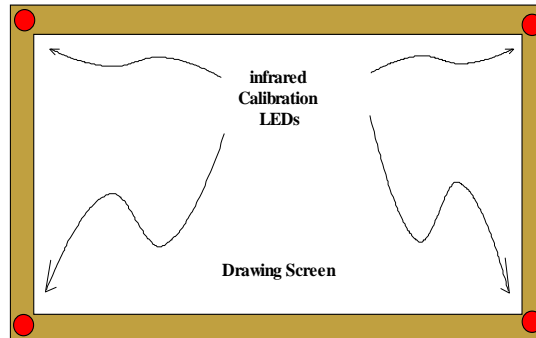

Fig. 10. Calibration LEDs for direct camera/projector alignment.

3.3   Prototype Implementation

The Smart Camera SoC architecture has been designed using Verilog and implemented on an Altera Cyclone II EP2C70 FPGA.

A 5 mega pixel (2592x1944) digital camera is used with an infrared sensitive lens and 950nm infrared bandpass filter. The camera can be programmed in resolution and field of view. The camera has on-chip 12-bit ADC and is used in our application at its maximum parallel pixel output rate of 96 MHz. The frame rate is determined by this maximum output speed, by the resolution chosen and by the shutter width. The full resolution frame rate is 15 frames/sec. At 640x480 VGA resolution frame rates of 150 frames/sec are possible. In our application we use a camera resolution of 1024x768 pixels for the brush image capture. This resolution and the field of view matches the resolution and the field of the projector. With this resolution a frame rate of 60 frames/second is used. There is a tradeoff between footprint resolution and frame rate. Higher resolutions and higher frame rates result in lower shutter times and weak lighting of the image sensor.

Using the image processing pipeline, described in the previous section, the brush position is determined by a real-time center-of-gravity calculation of the segmented footprint image. This is calculated immediately after the last pixel of a frame has been received. During the vertical blanking period of the camera, the footprint image around the center-of-gravity is retrieved from memory and sent to the application PC as UDP packets over the Ethernet connection.

The synthesis results with Quartus II 9.0 are shown in the Table I:

14   **Luc Claesen*, Peter Vandoren*, Tom Van Laerhoven*, Andy** Motten*, Fabian Di Fiore*, Frank Van Reeth*, Jing Liao$, Jinhui Yu$

TABLE I.   SYNTHESIS RESULTS OVERVIEW

Description	
Total logic elements	12,867
Total combinational functions	11,547
Dedicated logic registers	4,891
Total registers	5,012
Embedded Multiplier 9-bit elements	7
Total memory bits	724,548

A 50 MHz NiosII/e processor is used as a 32 bit RISC processor for the overall control. For the Gaussian lowpass filtering power-of-two coefficients are used to economize on multipliers. Completely programmable coefficients would also be possible as in the current prototype architecture only 2% of the available multipliers are used (7 / 300).

The frame buffers for the storage of the current image and background image are implemented in ISSI DRAM memories. In case of controlled lighting environments both frame buffers can be left away.

3.4   User Experience

The user interface to the FluidPaint real-brush based painting system is very obvious and self explaining. As even young children are familiar to the water paint paradigm, they can use the FluidPaint system immediately (Fig. 11).

Fig. 11. Without any previous training, 4-year old Ann and 6-year old Ryan can use the water and real-brush based intuitive FluidPaint system in seconds.

Fig. 12. FluidPaint painting artwork made in 30 minutes.

A number of test persons, that never used the FluidPaint system before have been invited to make paintings. For each test 30 minutes were provided. Fig. 12 shows a painting made by artist Karel Robert. All test persons found the system to be very intuitive in its use.

4   Conclusions and further work

The Smart Camera SoC removes several frame period delays in the image transmission and processing chain for determining the brush footprint in the real-time real-brush FluidPaint system to a maximum of a single frame delay. It frees up the processing power on the host PC thereby drastically enhancing the response time of the overall system.

User interaction tests are planned in the future to evaluate the Smart Camera enhanced FluidPaint system and to get feedback on possible future improvements.

As the architecture has been designed in the Verilog HDL, it can be integrated together with an image sensor on the same CMOS chip as a Smart Camera System-on-Chip. Further research focuses on setting up a generic architecture for Smart Cameras in such a way that vision processors can easily be configured for other applications as well.

16   **Luc Claesen*, Peter Vandoren*, Tom Van Laerhoven*, Andy** Motten*, Fabian
Di Fiore*, Frank Van Reeth*, Jing Liao$, Jinhui Yu$

5   References

[1]   Smith, A.R. . Digital Paint Systems: An Anecdotal and Historical Overview, IEEE Annals of the History of Computing, 23, 2(2001), pp. 4-30.

[2]   C.J. Curtis, S.E. Anderson, J.E. Seims, K.W. Fleischer, D.H. Salesin, "Computer-generated watercolor", ACM Siggraph, Proc. 24th ann. conf. on Computer Graphics and Interactive Techniques, pp. 421-430, 1997.

[3]   William V. Baxter, Jeremy Wendt, and Ming C. Lin, "IMPaSTo: A realistic, interactive model for paint." In Stephen N. Spencer (ed.), Proceedings of the 3rd International Symposium on Non-Photorealistic Animation and Rendering, Annecy, France, June 5-7, 2004.

[4]   T. Van Laerhoven, F. Van Reeth, "Real-time simulation of watery paint", CASA 2005, Volume 16, Issue 3-4, July 2005, pp. 429-439.

[5]   Nelson S.-H. Chu and C.-L. Tai, MoXi: Real-Time Ink Dispersion in Absorbent Paper, ACM Transactions on Graphics (SIGGRAPH 2005 issue), Vol. 24, No. 3, August 2005.

[6]   R. Greene, "The Drawing Prism: A Versatile Graphic Input Device", ACM SIGGRAPH Computer Graphics, Proceedings of ACM SIGGRAPH 85, San Francisco, CA, July 22--26, 1985, pp. 103-110.

[7]   http://www.optipaint.com/

[8]   Carver Mead, Ralph Wolf, Timothy P. Allen, "Paintbrush stylus for capacitive sensor pad", US Patent 5,488,204 Jan. 30, 1996.

[9]   Robert E. Mueller, "Direct Television Drawing and Image Manipulation System", US Patent 3,846,826, Nov. 5, 1974.

[10]   Wacom Cintic. http://www.wacom.com/cintiq

[11]   P.F. Ruedi, P. Heim, S. Gyger, F. Kaess, C. Arm, R. Caseiro, J.L. Nagel, S. Todeschini, "An SoC combining a 132db QVGA pixel array and a 32b DSP/MCU processor for vision applications", Proc. IEEE Solid-State Circuits Conference - Digest of Techincal Papers, ISSCC 2009, 8-12 Feb, 2009, pp. 46-47, 47a.

[12]   Peter Vandoren, Tom Van Laerhoven, Luc Claesen, Johannes Taelman, Chris Raymaekers, Frank Van Reeth, "IntuPaint: Bridging the Gap Between Physical and Digital Painting" Proc. of TABLETOP 2008, pp. 71 - 78, IEEE, ISBN 978-1-4244-2897-7, Oct. 2008.

[13]   Peter Vandoren, Luc Claesen, Tom Van Laerhoven, Johannes Taelman, Chris Raymaekers, Eddy Flerackers, Frank Van Reeth, "FluidPaint: an Interactive Digital Painting System using Real Wet Brushes" Proc. of ITS2009, pp. 53 - 56, ACM, ISBN 978-1-60558-733-2, Nov. 2009.

[14]   Rafael. C. Gonzalez, Richard E. Woods, "Digital Image Processing", Prentice Hall, 2002, ISBN 0-201-18075-8.

[15]   Kenneth. R. Castleman, "Digital Image Processing", Prentice Hall, 1996, ISBN 0-13-211467-4.

[16]   Wayne Wolf, Burak Ozer, Tiehan Lü, "Smart Cameras as Embedded Systems", IEEE Computer, Vol. 35, Issue 9, Sept. 2002, pp. 48-53.

[17]   J. Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In Proc. UIST 2005, ACM Press, 115-118.

[18]   Jinhui Yu, Jing Liao, John Patterson, "Modeling the interaction between objects and cartoon water", The Journal of Computer Animation and Social Agent, Vol. 19(3-4), 2008, pp. 375-385.

[19]   Fabian Di Fiore, Frank Van Reeth, J. Patterson, P. Willis, "Highly Stylized Animation", Computer Graphics International 2006, June 2006, Lecture Notes in Computer Science LNCS series, 4035, Springer, ISSN 32-0302-9743 pp. 36-53.

[20]   Domien Nowicki, Luc Claesen, "SoC Architecture for Real-Time Interactive Painting based on Lattice-Boltzmann", Proceedings of the 17th IEEE ICECS 2010, Athens, GR, dec. 12-15 2010, IEEE, ISBN ISBN 978-1-4244-8156-9, Dec. 2010, pp. 239 - 242.