Vision Sensor with an Active Digital Fovea

Donald G. Bailey[1] and Christos-Savvas Bouganis[2]

[1] School of Engineering and Advanced Technology
Massey University, Palmerston North, New Zealand
[2] Department of Electrical and Electronic Engineering
Imperial College London, London, United Kingdom

Abstract. Foveated image sensors have a variable spatial resolution enabling a significant reduction in image size and data volume. In this work, the requirements for a foveated sensor within an active vision system are analysed. Based on these requirements, the constraints on the resolution mapping function are determined, and a range of Cartesian based mapping schemes investigated. The results demonstrate that separable mappings, which independently map two orthogonal dimensions are efficient to implement using an FPGA. The computational requirements for an L_2 or Euclidean radial mapping are significantly higher, although this yields more natural looking low-resolution images. A compromise is based on using an L_∞ radial mapping. The requirement to process data as it is streamed from the camera necessitates implementing the warping using a forward mapping rather than the more common reverse mapping. The implications of this for FPGA implementation are described.

1 Introduction

In recent years image sensors have reached very high capacity making 5 to 10 megapixels in one sensor a commodity. This allows the acquisition of high-resolution images, which usually has a positive impact on the overall performance of many computer vision algorithms. However, it also creates a computational burden in processing systems when real-time constraints have to be met and the whole information from the sensor has to be processed. Moreover, in certain applications, such as tracking and pattern recognition, it is not so important to maintain the same resolution across the image sensor as to have a wide field of view and have high resolution only on specific regions of the sensor. Thus, a trade-off between high resolution and processing power has been created.

While it is possible to process such data in real time using dedicated hardware or a DSP microprocessor, any algorithm that requires multiple frames will require significant off-chip memory access, with its consequent bandwidth bottleneck. To enable on-chip storage, the volume of data, hence image size, must be reduced considerably. This chapter expands on earlier work by the authors (Bailey and Bouganis 2008).

1.1 Foveal Vision

The simplest way to reduce the data volume is to reduce the image size, and hence resolution. The consequence of this is a loss of information that may be critical in

S.C. Mukhopadhyay et al. (Eds.): Recent Adv. in Sensing Technology, LNEE 49, pp. 91–111.
springerlink.com

many applications. Often, high resolution is only critical in small regions of the image. A foveated window, inspired by the human visual system, provides a balance between high resolution, and large data volume. Within the window, the high resolution is maintained in the centre, with the resolution decreasing towards the periphery. Compared to a uniform resolution image, the increase in resolution in the centre comes at the expense of a decrease in resolution at the periphery.

Multi-resolution techniques are usually employed to address the above problem, where recently space-variant or foveating image sensors have been introduced that address the problem in its origin. These sensor architectures have variable spatial resolution across the surface of the sensor targeting data reduction without a severe impact on the final performance of the application. In (Martinez and Altamirano 2006) Martinez and Altamirano have demonstrated that a data reduction by a factor of 22 can be achieved without significant degradation in the performance of their tracking algorithm. Similarly, (Bailey and Bouganis 2009) show that tracking performance is not significantly affected after reducing the data volume by a factor of 64. An important part of such systems is the use of active vision techniques to ensure that the high resolution part of the sensor corresponds to the region of the scene where it can be most effective.

1.2 Potential Applications

Pattern recognition and tracking are two of the domains where a foveating image sensor has been successfully employed. Wilson and Hodgson (Wilson and Hodgson 1992a; b) addressed the problem of pattern recognition using a space-variant sensor. They exploited the rotation and scale invariance properties of a log-polar mapping combined with the reduction in data volume to efficiently represent patterns as templates, and to recognize images through a relatively simple template matching process for quite complex patterns. This approach requires an accurate and dynamic positioning of the fovea, for which the centre of gravity of the input Cartesian image was used. Scale invariance may be achieved with other mappings by dynamically matching the resolution (and mapping function) to the size of the object being recognized.

A space-variant sensor system was employed in (Cui et al. 1998; Xue and Morrell 2002) for tracking. Tracking requires positioning the fovea region on the object of interest. As the object moves from the centre of the fovea, this may be detected, and the position of the fovea dynamically adjusted in the next frame to enable the object to be tracked. By predicting the motion, a foveated sensor is able to maintain the target within the high-resolution region, and effectively track at this high resolution (Bailey and Bouganis 2009) while efficiently processing much smaller images.

A further application is in image compression (Wang and Bovik 2001). A foveated image is able to maintain resolution where detail is important, while reducing the resolution, or increasing the compression, in regions where detail is not required.

For many of these applications, a saliency detection algorithm (Itti et al. 1998; Liu et al. 2006) is required to identify regions of interest within the image. These algorithms model the human attention model and estimate the regions of the image that

attract the human attention and thus should be given more resolution than the other regions. These algorithms are typically used on the original high-resolution images.

Different applications may require a variety of mapping functions to be incorporated in the system so that the mapping function may be dynamically selected at run-time. It is also necessary to adapt the range of standard image processing techniques to operate on the low-resolution images.

1.3 Requirements Analysis

Many methods have been proposed for acquiring variable resolution images. In this section, a set of key requirements of a low power active foveal vision system is proposed and discussed. FPGAs are becoming increasingly used for low power embedded vision systems because they are able to exploit parallelism to enable a lower clock frequency to be used than a serial processor. For this reason, this chapter focuses primarily on an FPGA based implementation. The requirements are:

1) The system must be able to dynamically control the position of the fovea. In most applications, the vision system has to handle dynamic scenes. Thus, the regions of the image that require high resolution are not known in advance, leading to the requirement of repositioning of the fovea from one frame to the next. This may be accomplished either by mechanically scanning the fovea, or by electronically repositioning the fovea region over the region of interest.

2) The system must be able to achieve a significant reduction in the volume of data. Even though modern systems can readily access large off-chip memories, the communication between the memory and the computational unit is often the bottleneck for real-time performance. A significant reduction in the volume of the data will reduce the transfer times, and will even allow multiple images to be stored on-chip. For example, a 64×64 output image requires only 4 kbytes. Therefore, in an FPGA implementation, a monochrome image could be held in a single Virtex 5 Block RAM (Xilinx 2008), and a colour image could be held in a single Stratix IV M144K block (Altera 2008).

3) The system has to be able to create the foveated image as fast as possible for high frame rate processing to be achieved. This sets an upper limit to the computational complexity of the system that performs the conversion to a foveated image. Ideally, to minimize the latency the image should be converted as the data is streamed from the sensor onto the FPGA.

4) The system must be able to successfully perform the image processing operations on the low-resolution images, despite the large targeted reduction in the volume of the data. Successful means that existing algorithms must be able to be adapted (or new algorithms devised) that give similar results to processing the original high-resolution image but with significant reduction in processing time. In general, what is considered successful will be application dependent, however, any small reduction in the quality of the results must come with the benefit of significant reductions in processing time.

5) If multiple foveal mappings are available, there must be some way of dynamically selecting the best mapping for application. This may be as simple as tuning the mapping function based on the size of the object of interest, to dynamically optimizing the mapping function as part of the active vision process.

2 Related Work

Many configuration topologies have been introduced in the literature for spatially variant imaging. Most are inspired by the human vision, having a resolution that decreases with the distance from the centre of the sensor.

2.1 Methods of Acquiring Variable Resolution Images

2.1.1 Optical Approaches

In (Kuniyoshi et al. 1995), Kuniyoshi et al. achieved a variable spatial resolution by using a regular CCD sensor combined with a specifically designed lens that mimics the acuity of the human visual system. The authors focused on achieving a projection curve that would mimic the human eye projection characteristics and at the same time the current vision applications could be applied to the acquired image data. The authors consider three types of projection curves. In the fovea region, the authors employed a standard projection, since this region would be mainly used to extract local image features. For the periphery curve, a spherical projection was selected, since this area focuses on motion analysis and detection and for the extraction of global features from the environment in general. The chosen projection type for the periphery is supported by the fact that it simplifies the optical flow analysis for separating the target motion from ego-motion and also can help with navigation. In order to connect the two distinct areas, the authors propose the use of a log-polar projection. They concluded that the design of the optical system with the above characteristics was challenging and some digital processing was a necessity.

More recently, Hua and Liu (Hua and Liu 2008) proposed an approach that uses two off-the-shelf image sensors combined with a beam splitter to create a foveating imaging system. In this work, the authors' aim was to utilize the spatially variant resolution characteristic of the human visual system along with the space variant characteristics of contrast and color sensitivity in order to reduce the data bandwidth and maximize the information throughput. An imaging system based on a dual-sensor approach was presented which employed two detectors that shared the same entrance pupil. Moreover, the two detectors were coupled using a two-axis MEMS scanner to enable the high-resolution sensor to be mapped anywhere within the field of view of the low-resolution sensor. The authors reported a 90.4% bandwidth reduction, compared to a uniform sampling system.

2.1.2 VLSI Approaches

Targeting a small form factor and reduced power consumption, researchers have focused on manufacturing image sensors that have variable spatial resolutions. In (Etienne-Cummings 2000) the authors present a 2D foveated silicon retina which is realized in a single chip. The presented system exhibits two static areas with different resolutions. The fovea is in the center of the sensor and contains a 9×9 dense array, where the surrounding area implements the peripheral region and contains a sparse array of 19×17 larger photoreceptors. Moreover, the proposed system implements some

functions of the primate retina such as direction-of-motion detection in the retina and the localization of spatiotemporal edges in the periphery.

The main drawback of the VLSI approach is the fixed topology. In order for the system to perform tracking, a pair of motors is required. This problem is overcome by Vogelstein et al. (Vogelstein et al. 2004) by dynamically combining adjacent sensing elements to achieve varying spatial resolution. Their 80×60 image sensor has an address-event representation communication protocol which is interfaced with a digital microprocessor and up to four aVLSI chips that implement the integrate-and-fire neurons. By suitable configuration of the aVLSI chips, a fovea region can be realized across the image sensor, and by pooling events, a lower resolution may be achieved in the periphery.

2.1.3 Software Emulation

Emulating a variable resolution sensor using software has been adopted by many researchers due to the low cost and high flexibility that it offers. However, for real-time applications, the required processing time limits its applicability.

2.1.4 Hardware Emulation

A widely used approach is the use of a standard uniform resolution CCD sensor combined with either a VLSI chip or FPGA that maps the image to a topology that emulates a variable resolution sensor. Camacho et al. (Camacho et al. 1998) interface a progressive scan CCD camera with an FPGA to perform an adaptive mapping of the fovea. The FPGA is responsible for processing the data and constructing the multi-fovea image. It should be noted that the authors propose the use of many types of fovea that exhibit different spatial resolution capabilities under a log-Cartesian topology. Moreover, all the fovea regions exhibit a rectangular shape.

In (Ovod et al. 2005), the VASI (variable acuity super-pixel imager) system is described. It uses a CCD sensor and an FPGA for real-time image processing. The system supports multiple fovea, but only two levels of resolution are possible.

Arribas and Macia (Arribas and Maciá 1999) proposed an implementation of log-polar mapping using an FPGA that was able to achieve real-time performance. More recently, Martinez and Altamirano (Martinez and Altamirano 2006) proposed an FPGA pipelined architecture that transforms Cartesian images to a foveated image. The authors approximate a log-polar mapping by multiple square regions of different resolutions. They argue that the non-linearities that introduced in the image representation when a log-polar transformation is applied makes current computer vision algorithms such as correlation-based detection or recognition hard to apply.

2.2 Types of Mapping

The various types of foveal mappings that have been proposed in the literature can be categorized into four main types.

2.2.1 Dual Resolution

This type of mapping consists of two Cartesian regions, usually rectangular, of different spatial resolution. This may be achieved by employing two different

types of lens (Hua and Liu, 2008) or by explicitly constructing the image sensor to have two sampling regions of different densities (Etienne-Cummings 2000).

2.2.2 Log-Polar Mapping

The log-polar mapping has been one of the most widely used foveated mapping due to its close characteristics to the human retina. In (Wilson and Hodgson 1992a; b) Wilson and Hodgson use a log-polar mapping inspired by the sampling structure of the human retina, for a range of pattern recognition tasks. The main advantage of the log-polar mapping is that the logarithmic radial resolution dependence and polar coordinate systems enable scale and rotation invariance through translation within the low-resolution image if the fovea has been positioned consistently. This invariance can be exploited to simplify many pattern recognition tasks. Traver and Pla (Traver and Pla 2003), consider the trade-offs in the sensor topology when a log-polar mapping is targeted.

Although the log-polar mapping is a widely used mapping, one of its drawbacks is the non-linear effects that the polar coordinate system introduces to the regions of the sensor that do not align with the optical axis. Another problem is the singularity and consequent blind-spot in the centre of the fovea.

2.2.3 Ring Structures

A compromise is to base the mapping on a ring structure like the log-polar mapping, without using polar coordinates. Bandera and Scott (Bandera and Scott 1989), investigate the use of various imager topologies for a foveal vision system with respect to the data structure size and computations. In their work, they propose the use of rectangular and hexagonal lattices with varying resolutions, where the effective resolution of the system decreases with the distance from the center.

2.2.4 Others

Recently, research has focused on sensors that can provide many fovea regions rather than having a fixed topology. Camacho et. al. (Camacho et al. 1996) proposed a multi-resolution topology where the fovea region can be shifted across the sensor emulating the saccadic eye movements. In (Camacho et al. 1998) they generalised this idea to multiple foveal regions and moving object detection.

3 System Architecture

The approach taken in this chapter utilizes a continuously variable spatial resolution from the fovea to the periphery. It uses a standard uniform resolution image sensor and emulates a variable resolution sensor by mapping the uniformly sampled input image to a low resolution foveated image using an FPGA.

To meet the requirements listed in section 1.3, the architecture shown in Fig. 1 is proposed. It makes use of the high-resolution (3 to 10 megapixel) CMOS sensors that are now readily available. The high pixel count enables a wide-angle lens to be used without a loss of resolution compared to that of a standard camera. However, the high pixel count also has the disadvantage of limiting the frame rate.

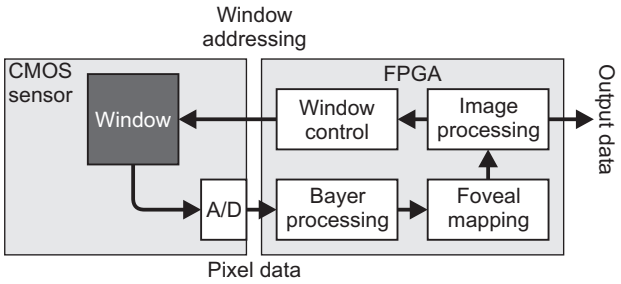

Fig. 1. System architecture

This may be overcome with CMOS sensors, because such sensors allow any arbitrary rectangular region or window of the sensor to be read out. Since the readout window is programmable, it may be positioned anywhere within the field of view of the camera.

Such a sensor enables variable spatial resolution in at least two different ways. The first is the dual resolution approach, which can be provided with a single high-resolution sensor. First the full image may be read out, using binning within the readout circuitry to combine groups of adjacent rows and columns of pixels. This reduces the resolution of the full image, reducing its size enabling it to be read out quickly. This provides the low-resolution "background" image for the dual resolution system. Then a full resolution image may be read out using the window, providing the high-resolution "detail" image. This approach effectively multiplexes the single sensor between it use for the high resolution and low resolution required by a dual resolution system.

The second approach, which will be explored in this chapter, is to read the high resolution data from the readout window of the sensor, and resample the uniform resolution image to provide a variable resolution foveal image for processing.

Digital CMOS sensors have integrated analogue to digital conversion, enabling direct connection to an FPGA. Single chip colour sensors use a Bayer pattern (Bayer 1976) or similar array of coloured filters over each pixel. Each pixel therefore only provides a single colour channel, so adjacent pixels must be interpolated to obtain a full colour image. The image read from the camera is then resampled using the foveal mapping to give an image that is significantly reduced in size. The small foveal image is then processed to give the required output from the complete camera. This processing will depend significantly on the application. The foveal images are also processed to determine the next location for the fovea, which is used to control the position of the readout window in the sensor, closing the feedback loop shown in Fig. 1.

3.1 Fovea Positioning

The ability to read out pixel data from within a predefined region of the sensor is essential for high-speed foveal vision. It enables the position of the high-resolution region to be positioned from one frame to the next under program control.

Without such an ability (for example using an optical approach, or defining the variable sensor geometry at the silicon level) it is necessary to mechanically pan and tilt the camera to position the fovea. Repositioning the window within a frame provides the equivalent of a very fast electronic pan and tilt. As the camera does not move, both the latency and motion blur associated with physically panning or tilting the camera are avoided. The limited angle of view, even of a wide-angle lens, means that some applications may still require a mechanical pan-tilt head. However, in applications where only a limited field of view is required, the need for a pan-tilt head may be eliminated.

3.2 Bayer Processing

There is a wide range of algorithms for interpolating the missing colour values in an image captured using a colour filter array. The demosaicing algorithms are effectively filters, which require data from adjacent rows to provide an output value. Data from previous rows must be cached using row buffers. From an FPGA processing perspective, it is desirable to minimize the required hardware resources.

Nearest neighbour and bilinear interpolation are relatively low cost, requiring one and two row buffers respectively. However, the problem with such simple interpolation is that it is prone to artifacts, particularly around edges and in regions of fine detail.

Many of the more complex algorithms are designed to reduce the artifacts introduced by the simpler methods. Unfortunately, while such algorithms can give good results, both the computational complexity, and the number of row buffers required can be quite large.

One compromise described in (Hsia 2004) uses a small window requiring only two row buffers. The improvement in quality is obtained by detecting edge directionality and weighting accordingly, and using a local gain to account for fine details. In addition to the row buffers, the technique requires two small dividers and three small multipliers. Both the multiplication and division may be implemented efficiently using logic gates on the FPGA (Bailey 2006).

3.3 Foveal Mappings

The main idea of the foveal mapping is to deliberately introduce distortion, so that part of the image (usually the centre) has a high resolution, while the periphery has a lower resolution. There are two alternatives in terms of the distortion mapping function.

One is to base the output on a polar coordinate system, such as that used by the log-polar mapping (Wilson and Hodgson 1992b; Traver and Pla 2003). The foveal mapping in this case performs a rectangular to polar conversion. In such a mapping, the angular resolution is constant, which naturally gives higher spatial resolution at the origin and has radially descreasing spatial resolution. The radial mapping function is designed to maintain a uniform radial to angular aspect ratio.

The alternative is to maintain Cartesian coordinates in both the input and output images. The foveal mapping in this case introduces a distortion, a little like an exaggerated lens distortion or fisheye lens distortion. These distortions result from a

non-uniform magnification in the lens. Inspired by this, the input image can be deliberately warped to both maintain the field of view by using a lower magnification in the periphery, while keeping the resolution in the fovea.

Let u be the distance from the centre of the input window and f be the distance from the centre of the foveated image. The foveal mapping can then be defined either in terms of the forward mapping, which gives the position of a pixel in the foveated image in terms of the undistorted input

$$f = \mathrm{map}_f(u), \tag{1}$$

or the reverse mapping which gives the location in the input image in terms of the output

$$u = \mathrm{map}_r(f). \tag{2}$$

The magnification, M, at any point is defined as the ratio between the output and input pixel distances. It may be expressed either in terms of the input pixels,

$$M_f(u) = \frac{f}{u} = \frac{map_f(u)}{u}, \tag{3}$$

or the output pixels

$$M_r(f) = \frac{f}{u} = \frac{f}{map_r(f)}. \tag{4}$$

The magnification relates the overall sizes of the input and foveated images, and therefore gives the average size of each output pixel in terms of input pixels. However, since the size of the output pixels is variable, a more useful relationship is the acuity, which is the effective resolution of an output pixel in terms of the uniform input resolution. The acuity, A, is therefore given from the slope of the mapping.

$$A(f) = \frac{df}{du} = 1 \Big/ \frac{d\,\mathrm{map}_r(f)}{df}. \tag{5}$$

When expressed in terms of input pixels, it gives the size of the specified input point in the output image:

$$A(u) = \frac{df}{du} = \frac{d\,\mathrm{map}_f(u)}{du}. \tag{6}$$

There are relatively few constraints on the mapping function. To ensure that an input point maps to only a single point in the foveated image, the mapping must be monotonic. There is little reason for having a higher resolution in the fovea than is in the original input image. (An exception perhaps is to introduce magnification into a dynamic mapping to maintain scale invariance in an object recognition application.) These imply

$$0 < A \le 1 \tag{7}$$

To map an $N \times N$ input window to a $W \times W$ output image also requires that both the centre and edges of the input map to the centre and edges of the output respectively:

$$\text{map}_f(0) = \text{map}_r(0) = 0 \tag{8}$$

And

$$\begin{aligned} \text{map}_f(N/2) &= W/2 \\ \text{map}_r(W/2) &= N/2 \end{aligned}. \tag{9}$$

If the fovea is in the centre of the image, and the resolution of the fovea corresponds to the input resolution then

$$A(0) = 1.0\,. \tag{10}$$

Note, this is not actually a constraint; the resolution in the fovea does not have to correspond to the input resolution, and the best resolution does not have to be in the centre of the output image.

Consider a typical input image resolution (sensor readout window size) of 512×512. To give a significant level of data reduction (as defined in section 1.3), a suitable output image size would be 64×64. Three example mappings that match the constraints of Eqs. (7) to (10) are

$$\text{map}_r(f) = f + \tfrac{7}{32} f^2, \tag{11}$$

$$\text{map}_r(f) = f + \tfrac{7}{1024} f^3, \tag{12}$$

and

$$\text{map}_r(f) = f + \tfrac{7}{32768} f^4\,. \tag{13}$$

The corresponding acuity functions, expressed in terms of the output pixels are shown in Fig. 2.

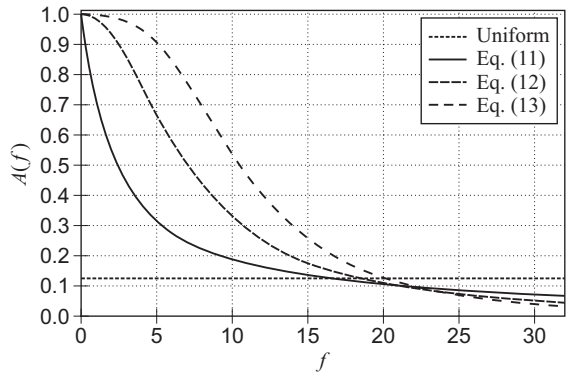

Fig. 2. Acuity for three different fovea mappings compared with uniform downsampling

A couple of observations may be made regarding the mappings. There is a gradation in the width of the fovea from Eq. (11) through to Eq. (13). The resolution of Eq. (11) drops very quickly away from the centre, whereas the mapping of Eq. (13) has several high resolution pixels in the fovea. There is a tradeoff with the size of the fovea, because a larger fovea means the periphery must be fitted into fewer pixels, resulting in a significantly lower resolution at the edges of the image. For example a pixel on the edge of the output image corresponds to an 8×8 block of input pixels for uniform magnification, whereas it corresponds to 15×15, 22×22 and 29×29 blocks respectively for Eqs. (11) to (13).

Another consideration is how to define the distances u and f. Different definitions will affect the nature of the distortion introduced. Let the coordinates relative to the centre of the input window or output image respectively be defined as (x_u,y_u) and (x_f,y_f). A radial Euclidean mapping, using the L_2 distance metric would have

$$u = \sqrt{x_u^2 + y_u^2} \tag{14}$$

And

$$f = \sqrt{x_f^2 + y_f^2} = \mathrm{map}_f\left(u\right) = uM_f\left(u\right) \tag{15}$$

With a radial mapping, the angle of the point relative to the centre of the image is unchanged. This means that the x and y coordinates are both scaled by the magnification for the given radius:

$$\begin{aligned} x_f &= x_u M_f\left(u\right) \\ y_f &= y_u M_f\left(u\right) \end{aligned} \tag{16}$$

or equivalently, using the reverse mapping

$$\begin{aligned} x_u &= x_f M_r\left(u\right) \\ y_u &= y_f M_r\left(u\right) \end{aligned} \tag{17}$$

A computationally simpler transform may be obtained by using the L_∞ or chessboard distance metric,

$$u = \max\left(\left|x_u\right|,\left|y_u\right|\right), \tag{18}$$

again with radial scaling using Eq. (16) or (17).

An even simpler transform may be obtained by considering the mapping separable (rather than radial) and independently mapping x and y:

$$\begin{aligned} x_f &= \mathrm{map}_f\left(x_u\right) \\ y_f &= \mathrm{map}_f\left(y_u\right) \end{aligned}. \tag{19}$$

Note that with the separable mapping there are different magnifications in each of the coordinate directions.

Fig. 3 compares the effects of these three mappings using the mapping function of Eq. (11). The size and shape of the pixels depends on both the mapping, and their position within the output image. The pixels are only square in the centre of

Fig. 3. Comparison of the different mapping methods: the small images are the foveated images, and the large images have remapped them back onto the original to show the acuity. **(a)** original Lena (512×512); **(b)** radial L_2 mapping; **(c)** radial L_∞ mapping; **(d)** separable mapping.

the fovea, and along the diagonals of the separable mapping. The separable transform does have rectangular pixels because of the separability, but the radial mappings result in pixels which are in general diagonal. For the radial mappings, the longest dimension of the pixel (which is radial) is given by the acuity, but the shortest dimension (tangential) is given by the magnification. This means that the pixels in the foveal image generally have better resolution tangentially than

Fig. 4. Comparison of the different mappings using the L_∞ mapping method: the small images are the foveated images, and the large images have remapped them back onto the original. **(a)** uniform downsampling to (64×64); **(b)** mapping of Eq. (11); **(c)** mapping of Eq. (12); **(d)** mapping of Eq. (13).

radially. The change of aspect ratio with position for the radial mappings is more uniform than for the separable mapping, so that standard image processing algorithms are more likely to work correctly on these foveal images. This is less so for the separable mapping.

In the foveal images, objects appear least distorted with the L_2 mapping although horizontal and vertical lines become curved. However, the overall resolution with this mapping is lower because of the regions in the corners without data. The other two mappings show distinct distortion along the diagonals, especially with the separable mapping where straight lines have a distinct corner along the diagonals of the foveal image.

It is also instructive to compare the different size foveas. Fig. 4 compares the mappings of Eqs. (11) to (13) with uniform downsampling. With the larger foveas, the periphery becomes noticeably more compressed, and the consequent increase in aspect ratio at the periphery makes the pixels appear to radiate from the centre in the reconstruction, especially along the diagonals. The pinching distortion along the diagonals also becomes more noticeable.

4 FPGA Implementation

Modern FPGAs have sufficient resources to enable whole applications to be implemented on a single FPGA, making them an ideal choice for embedded real-time vision systems. This section will discuss the issues associated with implementing the foveal mapping on an FPGA.

4.1 Reverse Mapping

Most commonly, the reverse mapping is used to perform image warping. The reverse mapping, defined by Eq. (2), determines the corresponding location in the input image for each output pixel , using some form of interpolation to handle fractions of pixels (Wolberg 1990). This is an advantage when the output pixels must be produced in a particular order, for example when streaming the output for a display. However, there are several problems with using a reverse mapping for implementing the foveal mapping.

First, producing the reverse mapping to implement a warping requires random access to the input image to select the corresponding input pixels for each output pixel. Implementing the mapping in this way would require significant memory for frame buffering, which was contrary to the original design goals.

Storing the image in an intermediate frame buffer also introduces latency into the transformation. With active vision, the outputs are used to control the capture, including the positioning of the fovea. Additional delays will reduce the ability to control the system.

A further problem is that the foveal mapping results in a reduction in resolution. Simply sampling the input image can result in significant aliasing, particularly in the periphery. This must be overcome by prefiltering with an anti-aliasing filter. With the spatially variant resolution, this requires a spatially variant filter, with little or no filtering required in the fovea, and a large window in the periphery.

4.2 Forward Mapping

These problems may be overcome by using a forward mapping, defined by Eq. (1), which determines where in the output image each input pixel maps to. A

forward mapping processes the input pixels as they are streamed from the camera. This minimizes the need for frame buffering giving the maximum benefit from the reduction in data volume from the foveal mapping. Since each pixel is processed as it arrives, it can minimize the latency, at least of the mapping stage.

A common problem with using the forward mapping is holes in the output, where there are output pixels with no corresponding inputs (Wolberg 1990). This is not an issue with a foveal mapping because the acuity is always less than or equal to one, resulting in a many to one mapping.

As outlined earlier, associating a single input pixel with each output pixel can result in aliasing. Simulating a low resolution sensor, with larger pixels on the image sensor, requires averaging all of the input pixels associated each output pixel. This averaging is effectively performing the spatially variant filtering required to reduce aliasing.

Such averaging requires maintaining two accumulators for each pixel within the output image. Each pixel is mapped to an output accumulator as determined by the mapping function. If the input pixel spans the boundary of multiple output pixels, then that pixel must be split among the associated output accumulators. The second accumulator counts the number of input pixels accumulated for each output pixel. Dividing one by the other will enable the corresponding output pixel value to be calculated.

4.3 Separable Mapping Implementation

The simplest mapping to implement is the separable mapping. It enables the X and Y directions to be mapped separately, which reduces the logic. This implementation is shown in Fig. 5. The left part of the figure shows the mapping in the X direction, with the circuit repeated on the right for the Y direction.

The mapping is effectively a two-pass algorithm similar to that defined by Catmull and Smith (Catmull and Smith 1980). The first mapping, or pass, assigns the input pixels into the correct output column, and the Y mapping (the second pass) then operates on the column to place each pixel in the correct row. To maintain a low latency, the Y pass is performed on all of the columns in parallel, to

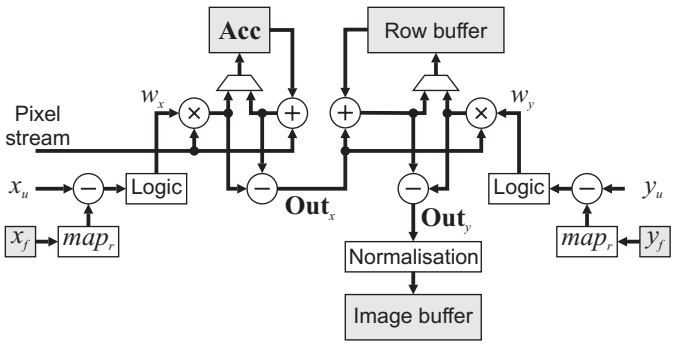

Fig. 5. Separable mapping implementation

enable the pixels to be processed in the order that they are streamed out from the first pass.

Consider first the X mapping. Rather than calculate the mapping algebraically (for example by directly using one of Eqs. (11) to (13)), it is more convenient to store the mapping directly in a lookup table. The simplest would be to use the forward mapping of Eq. (1) to indicate which pixels the input address maps to. However, since there are significantly fewer output pixels than input pixels, the reverse map is considerably smaller. The map_r block of Fig. 5 stores the boundaries in the input image associated with each output position. The incoming address is compared with the mapped address to determine whether or not the incoming pixel straddles two output pixels. If the incoming pixel is completely within the output pixel, the value is simply added to the accumulator. Otherwise, if the incoming pixel overlaps two output pixels, the weight for the second pixel is determined. This becomes the new accumulator and the accumulator plus the remainder of the pixel is passed on to the Y mapping. Note that the accumulator has two parts to it: the total accumulated pixel value, and the number of pixels contributed, to enable the value to be normalized.

$$\mathbf{Acc}[Total, Area] = \begin{cases} \mathbf{Acc} + [Pixel, 1] & \text{if } x_u - map_r(x_f) < 1 \\ w_x \times [Pixel, 1] & \text{otherwise} \end{cases} \tag{20}$$

where the weight is

$$w_x = x_u + 1 - map_r(x_f)\,. \tag{21}$$

The completed output is

$$\begin{aligned} \mathbf{Out}_x &= \mathbf{Acc} + (1 - w_x) \times [Pixel, 1] \\ &= \mathbf{Acc} + [Pixel, 1] - w_x \times [Pixel, 1] \end{aligned} \tag{22}$$

As a pixel is output, x_f is also incremented, beginning accumulation for the next pixel. Therefore, the previous value of x_f is used to index the row buffer an image in Eqs. (23) to (26).

A similar process is used for the mapping in the Y direction, with the output from the X mapping added to the accumulator for the column. Since all of the columns are processed in parallel, a row buffer is used to hold the accumulators for each of the output columns until sufficient rows have accumulated to complete an output pixel. The accumulation therefore is

$$\mathbf{RowBuffer}(x_f - 1) = \begin{cases} \mathbf{RowBuffer}(x_f - 1) + \mathbf{Out}_x & y_u - map_r(y_f) < 1 \\ w_y \times \mathbf{Out}_x & \text{otherwise} \end{cases} \tag{23}$$

where the weight

$$w_y = y_u + 1 - map_r(y_f) \tag{24}$$

is calculated once at the start of the row. The completed output is

$$\begin{aligned}\mathbf{Out}_y &= \mathbf{RowBuffer}(x_f - 1) + (1 - w_x) \times \mathbf{Out}_x \\ &= \mathbf{RowBuffer}(x_f - 1) + \mathbf{Out}_x - w_x \times \mathbf{Out}_x\end{aligned}. \tag{25}$$

Before being saved to the image memory, the output is normalized to give the average pixel value:

$$\mathbf{Image}(y_f, x_f - 1) = \frac{\mathbf{Out}_y[Total]}{\mathbf{Out}_y[Area]}. \tag{26}$$

The division required for this may be implemented efficiently (Bailey 2006) using about the same logic as that require for a multiplication.

The most expensive operation is the multiplication for the weighting. For grayscale images, three multiplications are required, one for Eq. (20) and two for Eq. (23); a colour image requires seven multiplications. One (or three for colour) division operations are required for normalization. Note that the lookup table for performing the mapping and the logic for calculating the weight could be multiplexed between the rows and columns because the vertical mapping only needs to be accessed once at the start of each row.

4.4 Radial Mapping Implementation

The radial mappings are more complex than the separable mapping because of the dependency between the X and Y components. For the L_∞ mapping, Eq. (18) implies that the maximum dimension can be determined from the mapping as in Fig. 5. This can be clearly seen in the larger images in Fig. 4. The diagonal partitions of the mapping from the input to the foveal image mean that the reverse mapping used for the separable map provides insufficient information to correctly map the pixels, and the larger forward mapping must be used.

The two-pass mapping also cannot be used directly, because it requires all input pixels that map to a given column to be treated the same. The diagonal partitions imply that some of these map to two separate rows. This becomes worse close to the diagonals, especially when the acuity is low (in the periphery when a large fovea is used) giving the input region for a pixel a large aspect ratio. As a result, several output rows may need to be accumulated simultaneously for a given low resolution column. This case was not considered adequately in the original proposal (Bailey and Bouganis 2008) which only allowed two simultaneous output rows to be accumulated per input column.

A compromise for the L_∞ mapping is to make all of the pixels rectangular rather than trapezoidal, as shown in Fig. 6. As these are in the periphery, where the resolution is lower, the modified geometry is less significant than in the fovea. The pixels along the diagonals in the foveal image, however, can have significantly lower resolution than other pixels.

Such a change would allow a two-pass circuit similar to Fig. 5 to be used. A reverse mapping could still be used, but would also need to include the width of the rectangles as well to enable the output pixel to be determined. Extra logic and an additional buffer would be required to determine the output pixel, and hence the

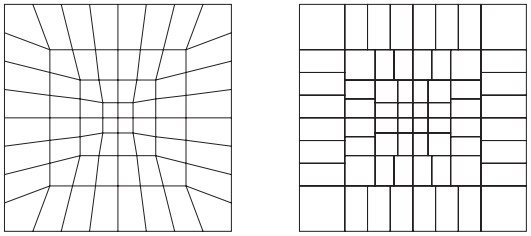

Fig. 6. Rectangularising the partitions for the L_∞ mapping. The original radial mapping is on the left, and the partitions are made rectangular in the mapping on the right.

weights. The buffer would enable incremental calculations to be used to accumulate the rectangle widths to avoid the need for an additional multiplication.

These techniques cannot easily be adapted for the L_2 mapping. Fig. 7 shows a scheme for the direct mapping of each input pixel using a radial mapping. When an input pixel spans multiple output pixels, it is necessary to determine the proportion of the input pixel that maps to each output pixel in order to apportion the pixel value accordingly. The geometric calculations can be quite complex, so a relatively simple approximation is to map the midpoints of the top, bottom, left, and right edges of each pixel. The right edge of one pixel becomes the left edge of the next pixel, and the bottom edge can be cached in a buffer to give the top edge in the next row. Therefore, only two points need to be transformed. A radial mapping requires that the input coordinates be scaled radially, so rather than store the mapping, the forward magnification function of Eq. (3) is used to scale the input coordinates to give the output coordinates. If (x_u,y_u) is the top left corner of an input pixel then the edges are transformed as

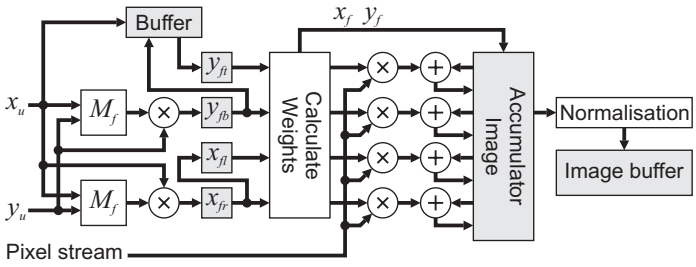

Fig. 7. Structure for radial foveal mappings

$$\begin{aligned} x_{fr} &= (x_u+1)M_f\left(\sqrt{(x_u+1)^2+(y_u+\tfrac{1}{2})^2}\right) \\ y_{fb} &= (y_u+1)M_f\left(\sqrt{(x_u+\tfrac{1}{2})^2+(y_u+1)^2}\right) \end{aligned} \quad . \tag{27}$$

The squaring in Eq. (27) may be performed without multiplication since x_u increments from pixel to pixel and exploiting the fact that

$$(a+1)^2 = a^2 + 2a + 1 . \tag{28}$$

Also, the square root is not necessary if the magnification function is stored in a lookup table and is indexed by u^2 rather than u, i.e. $M_f(u^2)$ (Gribbon et al. 2003).

If the integer parts of the pixel edges are the same then the weight is 1, otherwise the weight for apportioning the pixel horizontally is

$$w_x = \frac{\text{fract}(x_{fr})}{x_{fr} - x_{fl}}, \tag{29}$$

and similarly for the vertical weight. The input pixel value is then multiplied by the weights, and both the total and area accumulated for each output pixel. This will require up to four simultaneous accesses to the accumulator image (if the input is spread over four output pixels). This may be achieved by partitioning the accumulator image over four memory blocks, using separate blocks for the odd and even row and column addresses.

After all of the input pixels have been accumulated for an output pixel, it can then be normalized by dividing the total by the area. This gives the output pixel value for the foveal image.

The hardware resource requirements may again be estimated in terms of the number of multiplications and divisions. The magnification lookup tables will need to be interpolated tables, although a multiplication may be avoided with the interpolation by using a bipartite table (Schulte and Stine 1997). If implemented with dual-port memory, only a single magnification table is required. Two multiplications are required to calculate the pixel edges in the foveal image. Two divisions and a multiplication are required to calculate the weights w_x, w_y, and $w_x w_y$. To apportion the input pixel value, only three multiplications are required because the sum of the weights is one. Finally one division is required to normalize the output value. Thus a total of six (twelve for colour) multiplications and three (five for colour) divisions are required by this circuit. The intermediate memory requirements for this mapping are also larger. The buffer for y_{fb} must be the width of the input image, and an accumulator image is required.

5 Discussion and Conclusions

The mapping logic is the smallest for the separable mapping, although only slightly more is needed for a two-pass L_∞ mapping. The L_2 mapping is the most complex. Of the mappings explored, the two-pass L_∞ mapping provides the best compromise in terms of resource requirements and uniformity of pixel size.

With all of the mapping structures presented here, the mapping may be specified in a lookup table. This gives significant flexibility, as changing the mapping requires only changing the contents of the lookup table. Multiple foveal maps may be predefined and selected depending on the application. If the separable mapping is used, the mapping could even be adjusted dynamically between frames, given a suitable mechanism

for defining a new mapping. This is more complex for the radial mappings, as their efficient implementation require more details to be precalculated.

The low storage of the warped images and modest resource requirements for implementing the mapping also mean that multiple warp engines could be implemented in parallel if the processing requires different resolutions for different steps.

The design presented in this chapter meets the design requirements for active foveal vision systems identified in the introduction. The position of the fovea may be repositioned dynamically from frame to frame by using a windowed access from a CMOS sensor. An embedded system, consisting of a sensor and FPGA is possible, enabling high speed, active vision to be implemented without a mechanical pan-tilt platform. It is shown that a foveal mapping can reduce the image size significantly, reducing the storage and potential processing time of the output image. In the examples presented here, a 512×512 is reduced to 64×64 – a data reduction factor of 64. The mapping is performed on the data as it is streamed from the sensor, reducing the latency and enabling real-time operation.

Further research is required on the imaging algorithms for processing the foveated images. In particular, saliency detection algorithms that operate on the foveated images require further study. A range of both tracking and recognition algorithms need to be explored to determine the benefits and limitations of the mappings proposed here. Schemes for dynamically adapting and optimizing the mapping function for an application require investigation.

References

Altera, Stratix IV Device Handbook, vol. SIV5V1-2.0. Altera Corporation (2008)

Arribas, P.C., Maciá, F.M.H.: FPGA implementation of a log-polar algorithm for real time applications. In: Conference on Design of Circuits and Integrated Systems, Mallorca, Spain, pp. 63–68 (1999)

Bailey, D.G.: Space efficient division on FPGAs. In: Electronics New Zealand Conference (EnzCon 2006), Christchurch, New Zealand, pp. 206–211 (2006)

Bailey, D.G., Bouganis, C.S.: Reconfigurable foveated active vision system. In: International Conference on Sensing Technology, Tainan, Taiwan, pp. 162–169 (2008)

Bailey, D.G., Bouganis, C.S.: Tracking performance of a foveated vision system. In: International Conference on Autonomous Robots and Agents (ICARA 2009), Wellington, New Zealand, pp. 414–419 (2009)

Bandera, C., Scott, P.D.: Foveal machine vision systems. In: IEEE International Conference on Systems, Man and Cybernetics, Cambridge, Massachusetts, USA, vol. 2, pp. 596–599 (1989)

Bayer, B.E.: Colour filter array. United States of America patent 3971065 (1976)

Camacho, P., Arrebola, F., Sadoval, F.: Shifted fovea multiresolution geometries. In: International Conference on Image Processing, Lausanne, Switzerland, vol. 1, pp. 307–310 (1996)

Camacho, P., Arrebola, F., Sadoval, F.: Multiresolution sensors with adaptive structure. In: 24th Annual Conference of the IEEE Industrial Electronics Society (IECON 1998), Aachen, Germany, vol. 2, pp. 1230–1235 (1998)

Catmull, E., Smith, A.R.: 3-D transformations of images in scaline order. ACM SIGGRAPH Computer Graphics 14(3), 279–285 (1980)

Cui, Y., Samarasekera, S., Huang, Q., Greiffenhagen, M.: Indoor monitoring via the collaboration between a peripheral sensor and a foveal sensor. In: 1998 IEEE Workshop on Visual Surveillance, Bombay, India, pp. 2–9 (1998)

Gribbon, K.T., Johnston, C.T., Bailey, D.G.: A real-time FPGA implementation of a barrel distortion correction algorithm with bilinear interpolation. In: Image and Vision Computing New Zealand 2003, Palmerston North, New Zealand, pp. 408–413 (2003)

Hsia, S.C.: Fast high-quality color-filter-array interpolation method for digital camera systems. Journal of Electronic Imaging 13(1), 244–247 (2004)

Hua, H., Liu, S.: Dual-sensor foveated imaging system. Applied Optics 47(3), 317–327 (2008)

Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(11), 1254–1259 (1998)

Kuniyoshi, Y., Kita, N., Sugimoto, K., Nakamura, S., Suehiro, T.: A foveated wide angle lens for active vision. In: IEEE International Conference on Robotics and Automation, Nagoya, Japan, vol. 3, pp. 2982–2988 (1995)

Liu, Y., Bouganis, C.S., Cheung, P.Y.K.: A spatio-temporal saliency framework. In: IEEE International Conference on Image Processing, Atlanta, Georgia, USA, pp. 437–440 (2006)

Martinez, J., Altamirano, L.: FPGA-based pipeline architecture to transform cartesian images into foveal images by using a new foveation approach. In: IEEE International Conference on Reconfigurable Computing and FPGA's, San Luis Potosi, Mexico, pp. 1–10 (2006)

Ovod, V.I., Baxter, C.R., Massie, M.A., McCarley, P.L.: Advanced image processing package for FPGA-based re-programmable miniature electronics. In: Infrared Technology and Applications XXXI, Orlando, Florida, USA. SPIE, vol. 5783, pp. 304–315 (2005)

Schulte, M.J., Stine, J.E.: Symmetric bipartite tables for accurate function approximation. In: 13th IEEE Symposium on Computer Arithmetic, Asilomar, California, USA, pp. 175–183 (1997)

Traver, V.J., Pla, F.: Designing the lattice for log-polar images. In: Nyström, I., Sanniti di Baja, G., Svensson, S. (eds.) DGCI 2003. LNCS, vol. 2886, pp. 164–173. Springer, Heidelberg (2003)

Vogelstein, R.J., Mallik, U., Culurciello, E., Etienne-Cummings, R., Cauwenberghs, G.: Spatial acuity modulation of an address-event imager. In: 11th IEEE International Conference on Electronics, Circuits and Systems (ICECS 2004), Tel-Aviv, Israel, pp. 207–210 (2004)

Wang, Z., Bovik, A.C.: Embedded foveation image coding. IEEE Transactions on Image Processing 10(10), 1397–1410 (2001)

Wilson, J.C., Hodgson, R.M.: A pattern recognition system based on models of aspects of the human visual system. In: International Conference on Image Processing and its Applications, Maastricht, Netherlands, pp. 258–261 (1992a)

Wilson, J.C., Hodgson, R.M.: Log-polar mapping applied to pattern representation and recognition. In: Computer Vision and Image Processing, pp. 245–277. Academic Press, London (1992b)

Wolberg, G.: Digital image warping. IEEE Computer Society Press, Los Alamitos (1990)

Xilinx, Virtex-5 FPGA User Guide, vol. UG190 (v4.4). Xilinx Inc. (2008)

Xue, Y., Morrell, D.: Adaptive foveal sensor for target tracking. In: Thirty-Sixth Asilomar Conference on Signals, Systems and Computers, Pacific Grove, California, USA, vol. 1, pp. 848–852 (2002)