Stitching of Streaming Video from Multiple Fisheye Cameras using Soft core processors

Authors:
Omveer Sihag, 4th Year Electrical and Electronics Engineering, MIT, Manipal
UG Intern, Manipal Dot Net Pvt. Ltd.
sihag.omveer@gmail.com, omveer.sihag@manipal.net

Rama Shankar, 4th Year Electrical and Electronics Engineering, MIT, Manipal
UG Intern, Manipal Dot Net Pvt. Ltd.
rama.shk@gmail.com, rama.shankar@manipal.net

Dr. Narasimha B. Bhat, CEO, Manipal Dot Net Pvt. Ltd.
narasim@manipal.net

Dr. Vinay Kumar, Member of Technical Staff, Manipal Dot Net Pvt. Ltd.
vinay.kumar@manipal.net

Address: Manipal Dot Net Pvt. Ltd. (EHTP Unit)
#37 (2nd Floor), Ananth Nagar,
Manipal, Karnataka.
Phone: 09741736027/09844543937

Abstract: Fisheye lenses *[1]* have very large wide-angle views, so fewer cameras are needed to generate a panoramic view by stitching multiple images. However, the stitching of fisheye images is a non-trivial task as fisheye images suffer from severe distortion and correction can involve intensive computations and image processing. This paper discusses an innovative architecture for the real-time generation of panoramic views by stitching multiple fisheye images on an FPGA when basic camera parameters (field of view and focal length) are known. The method involves the one time computation of a look-up table to map a fisheye image onto a cylindrical compositing surface followed by an automatic similarity-based registration of two such images leading to a seamless stitch. The method is implemented on a Nios® II soft-core embedded processor and the result of processing video feeds from two OmniVision fisheye cameras is presented.

Introduction

Fisheye lenses achieve extremely wide fields of view (FOVs) by foregoing the perspective (rectilinear) mapping common to non-fisheye lenses and opting instead for certain special mappings. In our earlier work *[2]*, we have described different fisheye mappings (e.g., the linear scaled projection mapping) and developed a flexible architecture for correcting fisheye images to perspective versions. As a result of "Barrel Distortion" fisheye images do not preserve the most important feature of rectilinear images, that of mapping straight lines in the scene onto straight lines in the image.

Figure 1: Fisheye Image

Panoramic views are characterized by very large horizontal FOVs and they typically are used to generate 360° views of scenes such as city skylines. Since fisheye images have large FOVs they make ideal candidates for generating panoramic views. However, generating panoramic views from fisheye images is not a simple matter of correcting the images to generate corresponding perspective images.

Indeed perspective images cannot be panoramic, because objects at the edges of such images are stretched as a result of the perspective mapping. This stretching, known as perspective distortion, becomes more severe as one gets further away from the principal axis. Generating panoramic

views from perspective images therefore requires that several perspective images with small FOVs (to minimize perspective distortion) be stitched together along the horizontal direction. So one way to generate panoramic views is to correct the fisheye images to perspective versions and then stitch them. However, since the perspective images themselves need to be of small FOV, this solution cannot take advantage of the wide-angle properties of fisheye cameras.
An alternate solution is to directly stitch fisheye images and generate a panoramic view. However this requires a special correction, different from the one that corrects a fisheye image to a perspective image. This paper describes this special correction which is motivated by the desirable properties of panoramic views. First note that while a panoramic view has a large horizontal FOV, its vertical FOV need not be large. This implies that in order to generate a panoramic view, fisheye images are to be corrected in such a way that they show little or no perspective distortion in the horizontal direction, while admitting perspective distortion in the vertical direction.

Algorithm Description

A fisheye image is formed when rays entering a pinhole camera are incident on a spherical surface of radius equal to the focal length of the camera. On the other hand, a perspective image is formed when the rays are incident on a plane whose distance from the pinhole is equal to the focal length of the camera. In order to correct the fisheye image such that it shows no perspective distortion in the horizontal direction, the incident surface must be circular in the horizontal direction. Since perspective distortion in the vertical direction is admissible, the incident surface can be planar in the vertical direction. Assuming that the incident surface is a vertical cylindrical with a radius equal to the focal length would satisfy these requirements. The panoramic view is obtained by "unrolling" the image formed on the cylindrical surface (known as a compositing surface) and stitching it with similar images obtained from other fisheye cameras. The application assumes that the objects in the scene to be stitched are sufficiently far away from the camera, so that stereographic disparity is negligible.

Assume that the principal axis of the camera is along the z-axis, the horizontal and vertical directions correspond to the x and y-axes respectively. Let the ray entering the camera make an angle θ with the principal axis and its projection on the x-y plane make an angle φ with the x-axis, as shown in Figure 2. On passing through the pinhole the ray is incident on a cylindrical surface whose radius is equal to the focal length f and whose axis lies along the y-axis. Let (xc, yc, zc) be the coordinates of the point at which the incident ray impinges on the cylinder. It follows that the point (xc, yc, zc) depends on (f, θ, φ) in the following manner.

$$x_c = \frac{f \tan\theta \cos\varphi}{\sqrt{1 + \tan^2\theta}} \tag{1}$$

$$y_c = \frac{f \tan\theta \sin\varphi}{\sqrt{1 + \tan^2\theta \cos^2\varphi}} \tag{2}$$

$$z_c = \frac{f}{\sqrt{1 + \tan^2\theta \cos^2\varphi}} \tag{3}$$

The coordinates of the panoramic (unrolled) image (x_q, y_q) corresponding to (x_c, y_c, z_c) are given by

$$x_q = f \tan^{-1}(x_c/z_c) = f \tan^{-1}(\tan\theta \cos\varphi) \tag{4}$$

$$y_q = y_c = \frac{f \tan\theta \sin\varphi}{\sqrt{1 + \tan^2\theta \cos^2\varphi}} \tag{5}$$

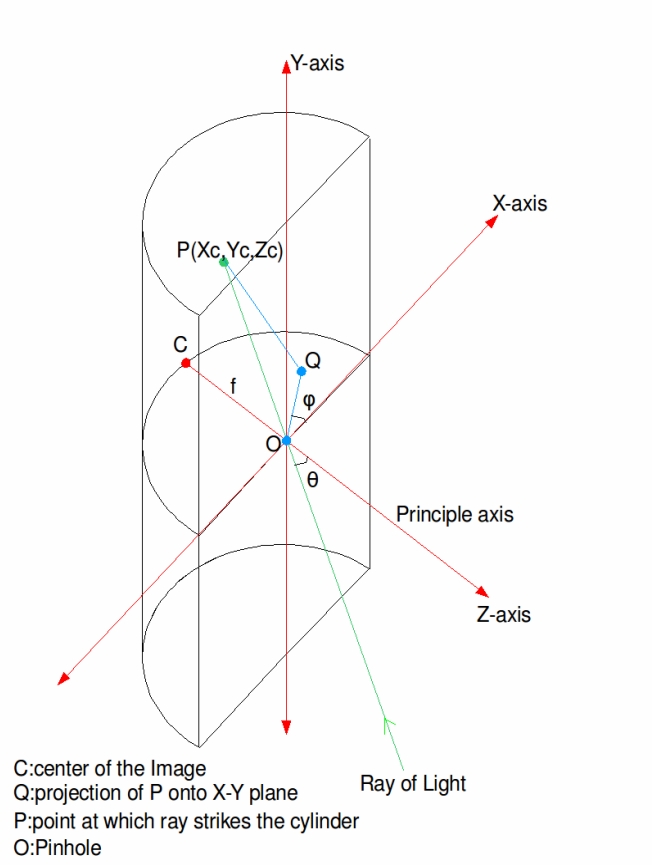

Figure 2 Projecting onto cylindrical surface to produce panormic image

Assuming that the camera uses the linear scaled projection for its mapping function, the coordinates of the fisheye image (x_f, y_f) corresponding to (x_c, y_c, z_c) are given by

$$x_f = f\,\theta \cos\varphi \tag{6}$$

$$y_f = f\,\theta \sin\varphi \tag{7}$$

The correction that we are seeking follows by eliminating θ and φ from the above equations (*4 - 7*). This gives us the following relationship between the coordinates of the panoramic view and the fisheye image

$$x_f = f \left[\tan^{-1} \left\{ \frac{\sqrt{(y_q/f)^2 + \sin^2(x_q/f)}}{\cos(x_q/f)} \right\} \right] \frac{\sin(x_q/f)}{\sqrt{(y_q/f)^2 + \sin^2(x_q/f)}} \quad (8)$$

$$y_f = f \left[\tan^{-1} \left\{ \frac{\sqrt{(y_q/f)^2 + \sin^2(x_q/f)}}{\cos(x_q/f)} \right\} \right] \frac{y_q/f}{\sqrt{(y_q/f)^2 + \sin^2(x_q/f)}} \quad (9)$$

Stitching Fisheye Images using LUT

Equations *(8)* and *(9)* allow us to map every pixel in the corrected image to a unique pixel in the input image. The corrected fisheye image shows no perspective distortion in the horizontal direction. If we have a circular fisheye image these equations are enough to give us a panoramic view with a 180° FOV. However, if we don't have circular fisheye images or if we want panoramic views with an FOV greater than 180°, then we need to stitch two or more such images.

Stitching the images captured by two or more cameras requires that the images be registered to determine the region of overlap. In our case, we are concerned with images from two or more identical cameras whose principal axes all lie in the same horizontal plane. Furthermore, the cameras are separated by a small distance and rotated with respect to each other. Assuming that all the objects in the scene are sufficiently far away, this situation can be modeled as one where the different images are captured by the rotation along a vertical axis of a single camera. Under these assumptions, the correction represented by equations *(8)* and *(9)* transforms the images (by mapping them onto a cylindrical surface) into horizontal translations of each other. Thus the problem of image registration is reduced to merely figuring out the horizontal shift that aligns one image with another, after correction.

However, in practice due to errors in camera alignment, it is likely that mere horizontal alignment is insufficient for exact image registration. Therefore in addition to a horizontal shift we also determine a vertical shift (if any) that might be necessary for exact alignment. A simple similarity-based method (see [4] for details) to determine the registration parameters (the horizontal and vertical shifts) that align the images with each other.

Once the parameters for registering the corrected fisheye images are known, equations *(8)* and *(9)* (by suitably translating (x_q, y_q) by the horizontal and vertical shifts) are used to directly map every pixel of the output stitched image to one of the multiple input images. Clearly this mapping is independent of the contents of the images, but depends only on the characteristics of the cameras (including the FOV and the resolution of the input image), the display (including the display resolution) and the registration parameters. Therefore the mapping can be determined through a one-time computation at the start and stored as a LUT. In our system for stitching fisheye images, the computation of the LUT proceeds in a two-step fashion: In the first step, the LUT for correcting a single image is computed off-line (since this does not require the knowledge of the registration parameters) using equations *(8)* and *(9)* and stored in the memory. In the second step, at the start of the system, images are captured by all the cameras in the system and corrected using the stored LUT. These corrected images are then registered and the

registration parameters computed. Using the registration parameters and the stored LUT, a final LUT for directly stitching fisheye images is generated.

Note that although equations *(8)* and *(9)* can produce real numbers for the input pixel locations, the LUT is not required to store floating point values, since we use the 9-point interpolation method which allows for efficient pixel interpolation and fixed point representations needed for an FPGA based implementation. The reader is referred to our earlier work [2], where these aspects have been described in considerable detail.

Design Implementation

This section discusses the implementation of fisheye image stitching using devices from the Altera® Cyclone® FPGA series and the Nios® II soft-core embedded processors.

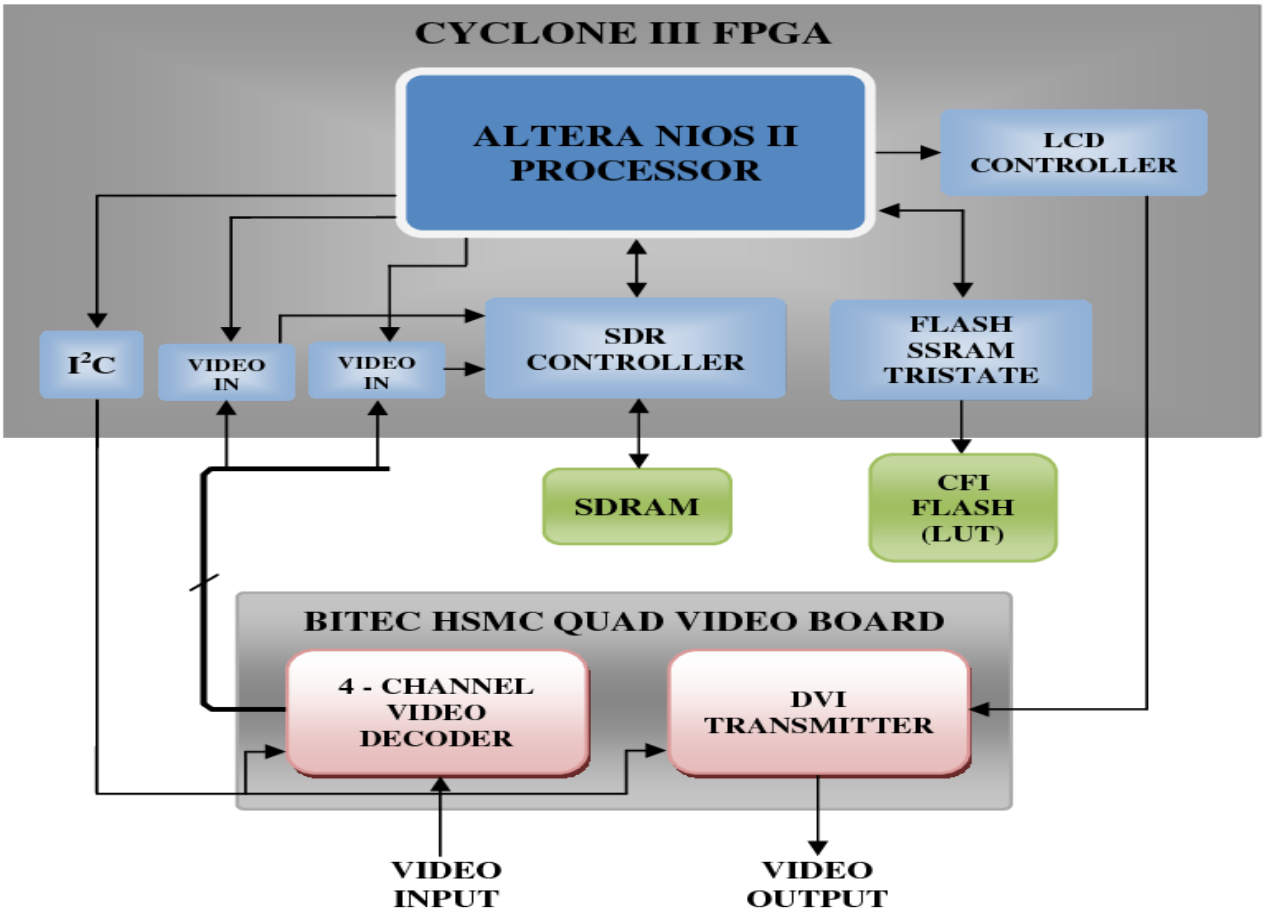

Figure 3 System block diagram

The Nios II architecture is a RISC soft-core architecture, which is implemented entirely in the programmable logic and memory blocks of Altera FPGAs, and is capable of handling a wide range of embedded computing applications, from DSP to system control. The soft-core nature of the Nios II processor lets the system designer specify and generate a custom Nios II core, tailored for his specific application requirements. Altera's Nios II Embedded Evaluation Kit (NEEK) is used as the development platform. The architecture is based on the following: a Nios II soft-core embedded processor, a Bitec Quad Video Input module, an I^2C configuration

module, a DDR-SDRAM controller, and a LCD controller. In Figure 3, the different modules present external to the Cyclone III FPGA are shown.

Results and Conclusion

Currently the system described above is used to capture the video feeds coming from two fisheye cameras. After a one time registration procedure to determine the registration parameters, the resultant image sequences are stitched using the LUT and the stitched image is displayed on a LCD screen at a resolution of 1024 × 768. Currently using Nearest Neighbor interpolation and Nine point Interpolation the system runs at around 7 and 4 frames per second respectively. An example stitched image of an outdoor scene produced by the above system is shown in Figure 5 with the input frames shown in Figure 4.

Using FPGAs and soft-core embedded processor technology, this paper presented a novel architecture for real time stitching of fisheye images and generating panoramic views. This architecture is flexible, scalable, and makes efficient use of the FPGA's resources. Because the architecture's Nios II processor is versatile and powerful enough to take on additional embedded processor functions, this technology is ideally suited for use in applications where panoramic views are used, such as automotive rear-view cameras and others.

Figure 4 Input Images

Figure 5 Stitched Image

Acknowledgement

The authors thank Manipal Dot Net Pvt. Ltd. and Altera Corporation for providing the facilities and infrastructure for research and implementation of the algorithm.

References

[1] “Fisheye lens,” Wikipedia: http://en.wikipedia.org/wiki/Fisheye_lens
[2] A Flexible Architecture for Fisheye Correction in Automotive Rear-View Cameras: http://www.altera.com/literature/wp/wp-01073-flexible-architecture-fisheye-correction-automotive-rear-view-cameras.pdf
[3] David Salomon, *“Transformations and Projections in Computer Graphics”*, Springer, 2006
[4] Image Stitching - Comparisons and New Techniques, Technical Report, University of Auckland, Oct 1998: http://citr.auckland.ac.nz/techreports/1998/CITR-TR-30.pdf.