Real-time FPGA Based 3D Camera Resectioning

Daniel M. Ho, Feng-ming Zhu and Wai-Chi Fang, *IEEE Fellow*
Department of Electronics Engineering, National Chiao Tung University, 1001 University Road, Hsinchu, Taiwan
wfang@mail.nctu.edu.tw

***Abstract*—This paper describes an embedded system based framework for processing depth and RGB images. This work provides a FPGA accelerated real-time solution for the problem of resectoning image data from a 3D camera to the coordinates of an RGB camera.**

I. Introduction

Over the last years, gesture-based input devices, such as Nintendo Wii, Playstation Move and Microsoft Kinect, have greatly enhanced gaming experience and transformed the way we play videogames.

One of these devices that call special attention is Microsoft Kinect, which relies only on image data and does not require a game controller for gesture input. One of the main features of this device is that it is capable to obtain a high-quality depth image from a 3D camera. This depth image can be used to track hand gestures, face expressions, body pose, and even for augmented reality using whole scene depth information [1].

One should note that this technology is not limited to Microsoft‟s console XBOX360. Recently, Microsoft released A Kinect SDK for developing Windows applications. Moreover, PrimeSense, the same manufacturer of the Kinect hardware, also released a similar sensor for ASUS (ASUS Xtion PRO).

Therefore, it is natural to expect that this technology will eventually be available to embedded mobile devices, such as portable videogame consoles, smart phones and tablets. Fig.1 shows a design sketch an example of such device.

On the software side, one big obstacle for using this technology in an embedded platform is that the processing power required for the image processing algorithms is very high.

One solution for executing these algorithms in real-time is to use dedicated hardware for performing calculations. The usage FPGA as a hardware accelerator is a possible approach.

A common requirement for many of these image processing algorithms is to combine the image from the depth camera to the image from the RGB camera. This requires the depth image to be transformed to the coordinates of the RGB image, since both cameras have different viewpoints and parameters.

On this paper we present a real-time FPGA based dedicated hardware solution for this resectioning problem.

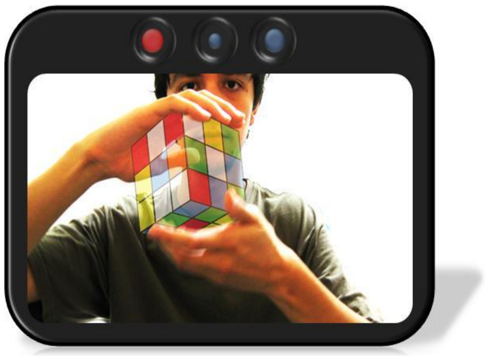
Fig. 1. Design sketch of the expected finished product: a portable device with depth sensors and an RGB camera. One design idea is to use an articulated joint to allow the depth sensors and RGB camera to be rotated to the back of the device.

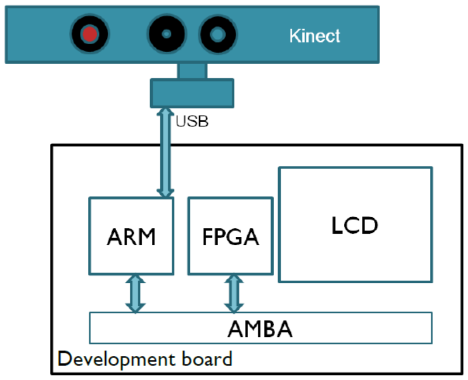

Fig. 2. Current prototype setup. A development board connected to Microsoft Kinect.

II. Algorithm

Camera resectioning is a well-known algorithm in image processing. Consider first that both cameras can be modeled by the pinhole model

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} [\mathbf{R} \quad \mathbf{T}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{1}$$

Where s is an arbitrary scale factor, *(u,v)* are the coordinates in the image, *fx* and *fy* are the camera focal lengths, *[R T]* is a matrix of extrinsic parameters, *(X,Y,Z)* are the real world coordinates.

The coordinates the resectioning equation from the depth camera to the RGB camera are given by

$$\begin{bmatrix} u^i \\ v^i \\ 1 \end{bmatrix} = \frac{1}{s^i}\begin{bmatrix} f_x^i & 0 & c_x^d \\ 0 & f_y^i & c_y^d \\ 0 & 0 & 1 \end{bmatrix} \cdot \left(Z \cdot \mathbf{R} \cdot \begin{bmatrix} x^d \\ y^d \\ 1 \end{bmatrix} + \mathbf{T} \right)$$

$$s^i = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \left(\mathbf{R} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathbf{T} \right) \quad . \qquad (2)$$

$$x^d = \left(u^d - c_x^d\right) / f_x^d$$
$$y^d = \left(v^d - c_y^d\right) / f_y^d$$

Where the *i* and *d* superscript indicate the values related to the RGB camera and depth camera respectively.

It is also important to consider the relative radial distortion between the cameras. This distortion can be corrected using Brown's distortion model [2]. Considering the distortion independent from Z, a distortion map was calculated offline and used to correct the distortion online.

III. System Design

A diagram of the prototype setup is shown in Fig. 2. The development kit used for the prototype is the FPGA Socle CDK. The processor is an ARM9 and the FPGA is a Spartan-3A, which includes 1728kb of block ram, large enough to store one frame of image. The ARM processor runs an Embedded Linux Kernel and the FPGA is connected to the processor by AMBA bus.

The hardware architecture designed in the FPGA is similar to the one proposed in [3] for distortion correction.

Each corrected image is sent back to the system before the second image is feed in. A block diagram of the system architecture is shown in Fig. 3.

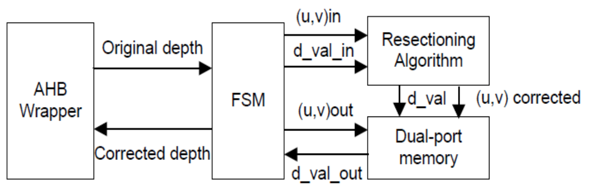

Fig. 3. Block design of the hardware architecture in the FPGA.

The resectioning algorithm is the one described in the previous section. It maps the depth pixels to color pixels, the first step is to undistort the depth pixel location using a undistort map, and then transform the depth pixel location to the color pixel location. The depth value for each pixel is fed into the design serially. The correct location of the depth pixel is calculated and then the corrected depth value is stored in the corresponding memory address.

IV. Results

For comparison, the algorithm was also completely implemented in software. In a desktop computer, the algorithm can be run in the GPU, which can make these calculations in real-time. However, in the ARM processor, running this algorithm is not practical.

By using the FPGA as a dedicated hardware solution, the algorithm can run in real-time and the ARM processor is relieved to execute other tasks. Fig. 4 shows a snapshot of the resectioning.

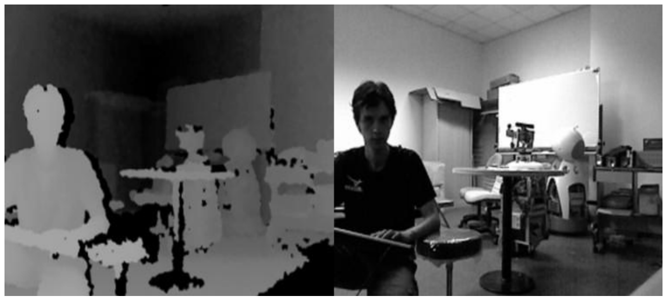

Fig. 4. Result of the resectioning (left) and the RGB image (right).

V. Future Work

With this same platform, we also intend to implement hardware accelerated tracking and gesture recognition algorithms.

If can successfully make these algorithms in real-time and prove that depth sensors can significantly improve portable gaming experience, we intend to seek support for manufacturing like the one in Fig. 1.

Acknowledgment

This work was partially supported in part by the National Science Council of Taiwan, R.O.C., under grants NSC-99-2220-E-009-028, NSC-99-2220-E-009-030, and NSC-99-2220-E-009-071.

References

[1] Kinect + AR Marker (June 9, 2011) http://kinecthacks.net/kinect-ar-marker/.

[2] Brown, D., "Close range camera calibration," *Photogrammetric Engineering* 8, 855–855 (1971).

[3] Hernández, A. and Gardel, A. and Pérez, L. and Bravo, I. and Mateos, R. and Sánchez, E., "Real-Time Image Distortion Correction using FPGA-based System", *IECON 2006-32nd Annual Conference on*, IEEE Industrial Electronics, nil7--nil11, IEEE.

[4] Y. Wang, T. Yu, L. Shi, and Z. Li, "Using human body gestures as inputs for gaming via depth analysis," in *Multimedia and Expo*, 2008 IEEE International Conference on, Hannover, Jun./Apr. 2008, pp. 993– 996.

[5] Zhengyou Zhang, "A Flexible New Technique for Camera Calibration", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v.22 n.11, p.1330-1334, November 2000.

[6] http://nicolas.burrus.name/index.php/Research/KinectCalibration

[7] http://opencv.willowgarage.com/documentation/camera_calibration_and_3d_reconstruction.html

[8] G. Barrow, J.M. Tanenbaum, R.C. Both and H. C. Wolf, „„,Parametric correspondence and chamfer matching: Two new techniques for image matching," In Proceedings of 5lh International Joint Conference on Artijkial Intelligence, Cambridge, MA, pp. 659-653,1977