USER DIRECTED VIEW SYNTHESIS ON OMAP PROCESSORS

Mursel Yildiz, Gozde Bozdagi Akar

Middle East Technical University
Electrical and Electronics Engineering Department
Ankara, Turkey

ABSTRACT

In this paper, we propose a system for user directed real time view synthesis for hand-held devices. Stored image frames with corresponding depth maps are used as input to the system. Users view point choice is captured using a GYRO based system. OMAP3530 microprocessor is used as the main processor which processes suggested view synthesis algorithm with occlusion handling and frame enhancement techniques. Proposed algorithms are implemented on DSP core and ARM core of OMAP3530 separately and their performances are evaluated through experiments. In addition, two daughter cards are designed for user view point determination. First daughter card handles communication process with EVM board and calculates view point according to the input from the second daughter card with single axis response GYRO sensor (ADIS16060).

Index Terms — Real time view synthesis, intermediate view reconstruction, OMAP3530 processor

1. INTRODUCTION

In recent years, there has been growing interest in applications for hand-held devices, especially for the ones with graphical accelerators and video display capabilities. With the merge of these two technologies, there exists a magnificent opportunity in the market for generating novel 3D applications on hand-held devices. One of the applications can be given as real-time view generation on hand-held devices. Most of the current stereoscopic/autostereoscopic displays rely on sending two different views to left and right eyes. Once the view direction of the user changes, different views have to be presented to him/her. Capturing and storing such number of views is not feasible. Instead of this, intermediate view reconstruction algorithms can be used. Many researchers have been developing 3D based intermediate view reconstruction or image-based rendering techniques for changing view points.

Ince et al. [2] proposed an image-based view interpolation algorithm based on usage of 4 input images taken from different viewpoints. Proposed algorithm consists of three steps. Firstly, all pixels in the input image are classified in terms of their visibility for the new view point. Using different image inputs, disparity for each pixel is calculated in the second phase, depending on the first phase output. Finally, each pixel color is calculated adaptively at the final step from image pairs that are selected by their visibility labels. Experimental results for this algorithm shows improvement on occlusion region handling. However, this algorithm should be implemented on systems with parallel processing capabilities because many middle step images are constructed for algorithm not only from a single image but from 4 images. Moreover, high speed memory usage is required for this system.

Debevec et al. [3] suggested an image-based intermediate view reconstruction technique with occlusion handling. They basically filled occlusion hole-filling problem using polygon view maps, which are basically depending on connectivity between each pixels at vertex positions. Firstly, they set up a linked list showing connectivity between vertexes, secondly they computed average color according to linked list and finally they assigned remaining invisible pixels with the closest visible pixel. Their method gives successful results for images having almost smooth depth maps, i.e. images in which objects are far away from camera. However, occlusion regions reveal themselves clearly for scenes in which objects are closer to cameras.

In [1], view point is detected by tracking the position of single observer and virtual camera view is rendered according to eye position of observer. The system is implemented on a home PC and tracking operation is done by a web-cam connected to the PC. However, this system is proposed for a single user, and system is suggested with an additional web-cam for eye tracking. This system would not show high performance for hand-held devices because users head and eyes are almost stable while watching video from this kind of devices.

In [9], a fast view synthesis method that generates multiple intermediate views in real time for a 3D display system, using predetermined camera geometry and depth map of each image frame which is very similar to theoretical approach for intermediate view reconstruction method discussed in this paper. Very fast view synthesis method is suggested by processing each frame in blocks entirely in parallel by many multiprocessors simultaneously, under the control of a CPU. They suggested specialized processor architecture of which experimental environment consists of a dual core CPU with 1.86 GHz speed, 2048 MB RAM, 1.62 GHz GPU of 16 multiprocessors having 128 cores in total. The system consists of many high speed processors, which in turn brings about high costs for the device, too much PCB space for production and very high power requirement not only for hand-held devices but also for any other display system.

In this paper, a system for intermediate view reconstruction algorithm for hand-held devices with gyroscope sensor and touchscreen capabilities is proposed. Our aim is to provide a free view point video application for which view point changes with the rotation of the device from left to right or vice versa. In this paper, section 2 gives detailed information on proposed system and section 3 concludes the paper.

978-1-4244-6378-7/10/$26.00

2. PROPOSED SYSTEM

The block diagram of the proposed system and its hardware set up, i.e., EVM board and daughter boards are given in figure 1 and 2, respectively.

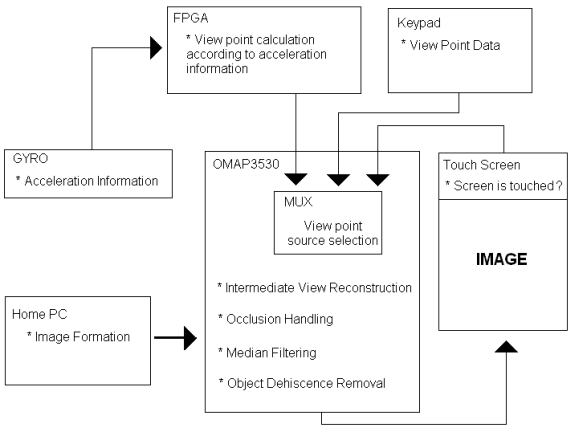

Figure 1. Block diagram of the proposed system

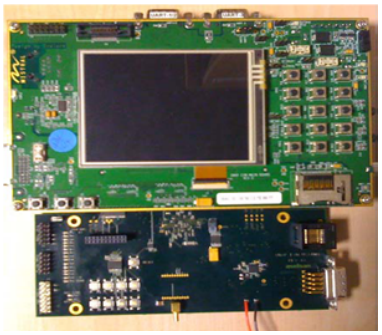

Figure 2. Basic system hardware setup

Each of these units together with the involved hardware will be described in details in the hardware and software architecture sections.

2.1. Hardware Architecture

EVM board is used for realization of algorithm and used as main processing unit. First daughter card is designed for view point calculation in order to interrupt main board and transmit view point once user changes it either by touch-screen, keyboard or gyro sensor. Second daughter card consists of gyro sensor independent from first daughter card. In this study, single Z-axis (yaw rate) response gyro sensor is used. In order to sense angular movement of system in correct direction, this card is designed independently from first daughter card and kept perpendicular to the EVM board and first daughter card.

Daughter boards are developed and designed for exploring GYRO sensors, FPGA (SPARTAN family with DSP slices) and developing applications together with OMAP3530 processor. In figure 3, functional block diagram of these cards are given.

GYRO sensor provides user with the ability to change free view point by only rotating the hardware either to the left or right. The exact position information is calculated and updated by FPGA

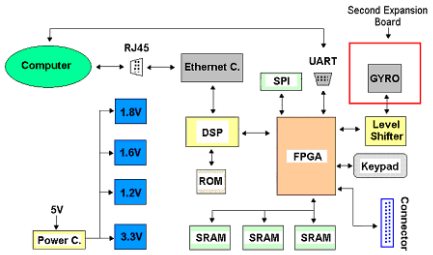

Figure 3. Functional block diagram of the daughter boards

including hardly embedded DSP slides in the system, which communicates by i2c protocol with OMAP processor. Once camera position changes, FPGA provides new position information for main rendering algorithm.

The digital data available at the SPI port of GYRO sensor is proportional to the angular rate about the axis that is normal to the top surface of the package. There are two poly-silicon sensing structures containing a dither frame that is electro statically driven to resonance. This generates the necessary velocity element to produce a coriolis force while rotating. At both of the outer extremes of each frame, orthogonal to the dither motion, are movable fingers that are placed between fixed pickoff fingers to form a capacitive pickoff structure that senses coriolis motion [8]. Figure 4 shows rate sensitive axis.

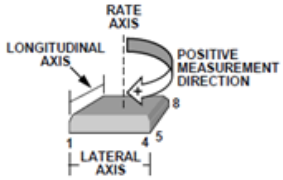

Figure 4. Rate sensitive axis of gyro sensor

A computer is used for monitoring, controlling embedded operating system and starting applications through hyper terminal. Linux (Kernel-2.6.22) is used as the embedded operating system and applications are developed by OMAP35x SDK 1.0.2 together with DVSDK 3 00 00 29 provided by Texas Instruments.

2.2. Software Architecture

Software architecture of the system is mainly consisted of 5 algorithm blocks as show in figure 1. For OMAP3530, supported color formats on OSD (Graphics pipeline) are RGB565, RGB444, and RGB888. And on Video pipelines; YUV422 interleaved, RGB565, RGB888, RGB565X [5]. Although increment in the number of bits per pixel increases the color depth and quality, true color can also be represented after blending primary colors from each pixel [6]. Therefore in the **image formation step**, 24 bit bitmap files are converted to RGB565 file format. The aim is to save memory and consider each pixel by only one ”short” variable, which is advantageous for median filtering stage for colored pictures.

3 different methods are used for **view point calculation**. Variation of view point determination algorithm makes this application program integrate to various hand-held devices, having gyro sensors, touchpad LCD screens or keypad. **Gyro sensor algorithm** is based on calculation of view point relative to an initially assigned position by considering angular rate once user flips-flops the sys-

tem. Algorithm is performed in FPGA according to the acceleration input provided by GYRO sensor. High speed multiplications, divisions, additions and subtractions are performed by DSP slices that are hardly implemented inside FPGA. Availability of 84 DSP slices became the reason of Spartan XC3SD1800A FPGA family choice [7].

Secondly,using **keypad**, once user presses left shift key, view point is changed to the left smoothly until user releases the button and once user insists on pressing left shift button, view point is changed until left end is reached. Similarly, pressing right shift key changes view point to the right smoothly until right end is reached. EVM module supports **touch-screen** with 4 wire touch-screen controller of TSC2046, which is interfaced with main processor OMAP through McBSP port (multi-channel buffered serial port) [4]. Using touch-screen drivers provided by Linux-2.26 embedded operating system, it is possible to read touch-screen information. In this study, this feature is used for final option in view point updating. Once user touches the screen, view point is changed to the left end firstly and than backward until the right end is reached and finally comes to the initial point. Changes in view point occur smoothly. This process is resembled in figure 5.

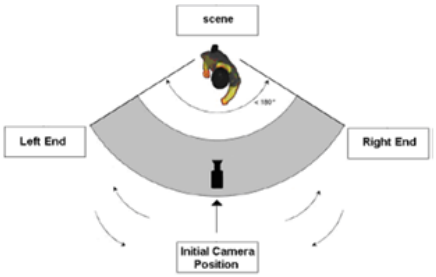

Figure 5. Touch-screen controlled virtual camera orbit

Once view point is calculated, next step is to generate correct views that will be presented to the user. **Intermediate view reconstruction** part of the software constitutes the main step in realization of user-defined view point video display algorithm. This algorithm is implemented on OMAP. Intermediate views according to a user defined view points are constructed using base camera input frames and their corresponding depth maps. Input frame sequences are obtained from Microsoft research group Sing Bing Kang, which provides 100 frames and their corresponding depth maps for a ballet sequence [10]. In figure 6, original frame for camera 5 is given together with constructed intermediate views from camera 4 and 6 for the virtual camera position at camera 5 [10].

Occlusion handling algorithm is the step applied after intermediate view reconstruction algorithm on second camera input frames. Although some regions are visible for base camera view point, they can be invisible for virtual camera. This phenomenon can be easily understood from figure 6. Once a region on the background is invisible for base camera but visible for virtual camera position, occlusion handling algorithm is applied on second camera input frames in order to grab additional information of such kind of regions on image. This algorithm detects candidates of occlusion region filling pixels. This is for speeding up software blocks in order to construct a real time operating display system. It is easier and faster to determine starting pixel location by camera 4 depth map and finishing pixel location by camera 6 depth map. Considering about abruptly drawn top view of 3D scene in

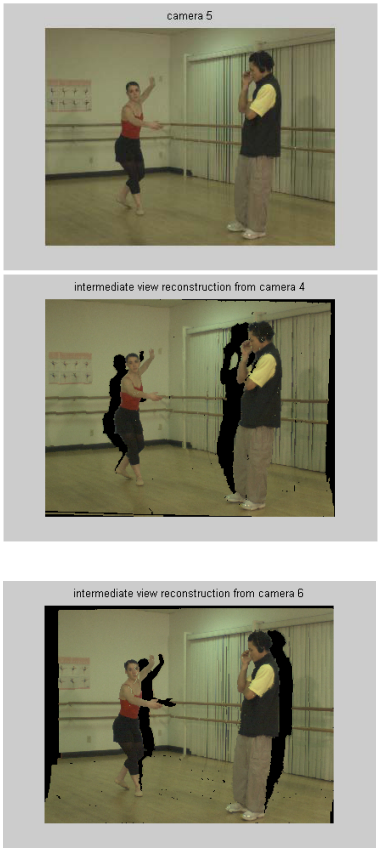

Figure 6. Application of intermediate view reconstruction

figure 7, it is obvious that R1 and R2 regions can not be viewed by camera 6. Although R1 and R2 regions are seen by camera 4, it will be unnecessary to determine where pixels in R1 regions are projected according to camera 5, because R1 region is not in the sight of camera 5 and these pixels will be unseen at the end of rendering algorithm.

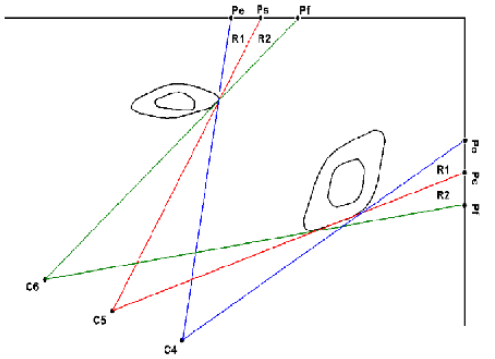

Figure 7. Abruptly drawn real scene top view

R2 resembles the occlusion regions according to camera 6. In this algorithm, Ps (starting pixel of occlusion region filling candidates) and Pf (finishing pixel of occlusion region filling candidates) are determined for each row on image frame from camera 4 and corresponding pixels are rendered using camera 4 input in order to fill holes that will occur at the end of rendering algorithm for camera 6. Considering camera 4 frame 0 depth map, scanning

column by column from left to right, it is possible to understand starting and finishing pixel positions for occlusion region as shown in figure 8.

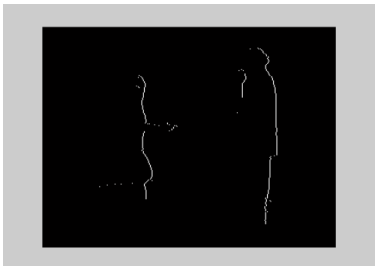

Figure 8. Edge image of depth map in vertical direction

Frame enhancement algorithms are based on mainly two stages. Median filtering is proposed in the first stage in order to deal with impulse noise. Moreover, due to lack of pixel amounts on objects that are near to the base camera, after view reconstruction, separation and spread of pixels belonging same column and having same depth values is observed. This resulted in a mixed form of objects that are near to base camera with background pixels. In figure 9, these kinds of conditions are represented. For this kind of problems, considering the edge images of depth maps in vertical direction, nearer objects to the camera are detected and possible background pixels behind these objects are deleted for preventing such situations.

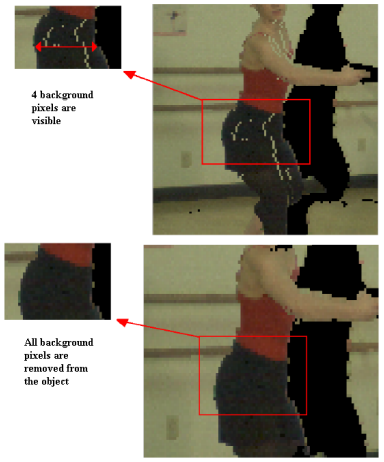

Figure 9. Background pixel observation through dehiscence region and its removal

3. PERFORMANCE RESULTS AND CONCLUSION

In this paper, view rendering system for hand-held devices with gyro sensors and touch-screen peripheral, is proposed and implemented. 256x340 image frames are used for robust intermediate view reconstruction implementation. Up to 12 frames/second are reached which is a very satisfactory result for hand-held devices. Proposed algorithms were applied on OMAP3530 processor using both cores ARM and DSP separately. Both of the cores were not used simultaneously due to memory collision problem. Both cores perform memory read and write on same bus which causes a need of additional application in kernel space that controls memory reaches from cores. Such applications are directly related to transferring one or more whole blocks into cache memories by block drivers and devices [11]. Algorithm performances for occlusion handling and intermediate view reconstruction are given in table 1.

Core	Algotihm	Fps	Occlusion?
Arm	FLp	1	Yes
DSP	FLp	0.33	Yes
DSP	FIp	13	No
Arm	FIp	19	No
Arm	FIp, FLp	7.5	Yes
Arm	FIp, FLp, OcH, FrE	12	Yes

OcH: Occlusion Handling Algorithm
FrE: Frame Enhancement Algorithm
FIp: Fixed point Algorithm
FLp: Floating point Algorithm

Table 1. Core performance table

4. ACKNOWLEDGEMENTS

The research leading to these results has received funding from MOBILE3DTV project which has received funding from the European Community's ICT programme in the context of the Seventh Framework Programme (FP7/2007-2011) under grant agreement n 216503. The text reflects only the authors views and the European Community or other project partners are not liable for any use that may be made of the information contained herein.

5. REFERENCES

[1] Y. Bediz, "Automatic Eye Tracking and Intermediate View Reconstruction for 3D Imaging Systems", MSc Thesis, Middle East Technical University, Ankara, Sept. 2006.

[2] S. Ince and J. Konrad, "Occlusion-aware view interpola-tion", EURASIP J. Image and Video Process., vol. 2008, Article ID 803231, 15 pages, 2008, doi:10.1155/2008/803231.

[3] Paul E. Debevec, George Borshukov, and Yizhou Yu, "Efficient View-Dependent Image-Based Rendering with Projective Texture-Mapping", 9th Eurographics Rendering Workshop, Vienna, Austria, June 1998.

[4] "OMAP35x Evaluation Module Hardware User Guide", Mistral Solutions Pvt. Ltd., Rev. 1.2 May 2008

[5] "OMAP35x EVM Linux PSP User Guide 1.0.2", Texas Instruments Incorporated, Published 06 OCT 2008

[6] Juan Gonzales, Neal Frager, Ryan Link, "Digital Video Using DaVinci SoC", Texas Instruments, SPRAAN0 - June 2007

[7] "Spartan-3A DSP FPGA Family: Data Sheet", XILINX, DS610 - June2, 2008

[8] "Wide Bandwidth Yaw Rate Gyroscope with SPI (ADIS16060)", ANALOG DEVICES, 2008

[9] Hong-Chang Shin, Yong-Jin Kim, Hanhoon Park, and Jong-Il Park, "Fast View Synthesis using GPU for 3D Display", IEEE Transactions on Consumer Electronics, Vol. 54, No. 4, NOVEMBER 2008.

[10] Microsoft Research, http://research.microsoft.com/vision/ InteractiveVisualMediaGroup/3Dvideo Download/, visited 2 September 2008

[11] Jonathan Corbet, Alessandro Rubini, and Greg Kroah-Hartman, "LINUX DEVICE DRIVERS", O'Reilly Me-dia, Inc., Third Edition, pp. 6-7, USA, February 2005.