VLSI Implementation of High-Performance CORDIC-Based Vector Interpolator in Power-Aware 3-D Graphic Systems

TZE-YUN SUNG
Department of Microelectronics Engineering
Chung Hua University
707, Sec. 2, Wufu Road
Hsinchu, 30012, TAIWAN

Abstract: - High performance architectures for the data intensive and latency restrained applications can be achieved by maximizing both parallelism and pipelining. In this paper, the CORDIC based hardware primitives of 3-D rotation with high throughput 3-D vector interpolation are presented. The proposed architecture for 3-D vector interpolator, which is based on the redundant CORDIC arithmetic, has been implemented by VLSI and achieve up to energy saving without image quality degradation. The graphic system is power-aware.

Key-Words: - Redundant arithmetic, CORDIC, 3-D, vector interpolation, power-aware, high-throughput, parallelism and pipelining, VLSI.

1 Introduction

Flexible hardware along with precision control is very desirable for the power-aware 3-D graphics rendering applications. In [1], 3-D vector interpolation is required. The 3-D vector interpolator of Euh et al. provides multiple precisions for the design of power-aware systems [2].

The well known CORDIC algorithm, which has been applied with a great success to the hardware implementations of many signal processing tasks, e.g. sine and cosine generation, vector rotation, coordinate transformation, and linear system solving, is suitable for the implementation of 3-D vector interpolation [3]-[4]. In CORDIC, only simple shifters and adders are needed, which can be realized by the use of reconfigurable hardware platforms, especially by FPGA [5]. Thus, the CORDIC-based 3-D vector interpolator is more flexible for the interpolation task.

In this paper, the architecture of 3-D vector interpolator based on the CORDIC algorithm is proposed. It is suitable for VLSI implementation in terms of the computational complexity. The remainder of the paper is organized as follows. In section 2, the conventional CORDIC algorithm is reviewed. In section 3, the 3-D CORDIC algorithm is given. The proposed VLSI architecture of 3-D vector interpolator based on the CORDIC rotation algorithm is presented in section 4. Its analysis is given in section 5, and the conclusion can be found in section 6.

2 The CORDIC Algorithm

CORDIC (COordinate Rotation DIgital Computer) is an algorithm performing a sequence of iteration computations by the use of coordinate rotation [3] [4]. It can be used to generate important elementary functions by using only simple adders and shifters. The basic CORDIC iteration equations are given by

$$x_{i+1} = x_i - m\sigma_i 2^{-s(m,i)} y_i \quad (1)$$

$$y_{i+1} = y_i + \sigma_i 2^{-s(m,i)} x_i \quad (2)$$

$$z_{i+1} = z_i - \sigma_i \alpha_{m,i} \quad (3)$$

where m denotes the circular (m=1), linear (m=0) or hyperbolic (m=-1) coordinate system, i=0, 1,2,…., n-1,

$$s(m,i) = \begin{cases} 0,1,2,3,4,5,...., & m = 1 \\ 1,2,3,4,5,6,...., & m = 0 \\ 1,2,3,4,4,5,...., & m = -1 \end{cases}, \text{ and}$$

$$\alpha_{m,i} = m^{-1/2} \tan^{-1}[\sqrt{m}\, 2^{-s(m,i)}] \quad (4)$$

The rotation $\sigma_i = sign(z_i)$ for the rotation mode $(z_n \to 0)$; $\sigma_i = -sign(x_i) \cdot sign(y_i)$ for the vectoring mode $(y_n \to 0)$. The scale factor $k_{m,i} = \sqrt{1 + m\sigma_i^2 2^{-2s(m,i)}}$ in the i-th iteration. After n iterations, the product of all the scale factors is as follows.

$$K_m = \prod_{i=0}^{n} k_{m,i} = \prod_{i=0}^{n} \sqrt{1 + m\sigma_i^2 2^{-2s(m,i)}} = \prod_{i=0}^{n} \sqrt{1 + m 2^{-2s(m,i)}} \quad (5)$$

where the rotation direction is defined by $\sigma_i = \{-1, +1\}$.

3 3-D CORDIC Algorithm

Figure 1 shows a vector R in the 3-D space. Its respective Cartesian and spherical coordinates are (X_i, Y_i, Z_i) and (R_i, θ_i, ϕ_i). R can be rotated and then becomes a new vector denoted by S with Cartesian coordinates $(X_{i+1}, Y_{i+1}, Z_{i+1})$ and spherical coordinates $(R_i, \theta_i + \alpha_i, \phi_i + \beta_i)$. The relationship between the Cartesian coordinates and spherical coordinates of R and S are given by

$$X_i = R_i \cos\theta_i \sin\phi_i \quad (6)$$

$$Y_i = R_i \sin\theta_i \sin\phi_i \quad (7)$$

$$Z_i = R_i \cos\phi_i \quad (8)$$

$$X_{i+1} = R_i \cos(\theta_i + \alpha_i)\sin(\phi_i + \beta_i) \quad (9)$$

$$Y_{i+1} = R_i \sin(\theta_i + \alpha_i)\sin(\phi_i + \beta_i) \quad (10)$$

$$Z_{i+1} = R_i \cos(\phi_i + \beta_i) \quad (11)$$

Equations (9), (10) and (11) can be rewritten by

$$\begin{aligned} X_{i+1} &= R_i(\cos\theta_i\cos\alpha_i - \sin\theta_i\sin\alpha_i)(\sin\phi_i\cos\beta_i + \cos\phi_i\sin\beta_i) \\ &= R_i\cos\theta_i\sin\phi_i\cos\alpha_i\cos\beta_i + R_i\cos\theta_i\cos\phi_i\cos\alpha_i\sin\beta_i \\ &\quad - R_i\sin\theta_i\sin\phi_i\sin\alpha_i\cos\beta_i - R_i\sin\theta_i\cos\phi_i\sin\alpha_i\sin\beta_i \\ &= X_i\cos\alpha_i\cos\beta_i + U_i\cos\alpha_i\sin\beta_i \\ &\quad - Y_i\sin\alpha_i\cos\beta_i - V_i\sin\alpha_i\sin\beta_i \end{aligned} \quad (12)$$

$$\begin{aligned} Y_{i+1} &= Y_i\cos\alpha_i\cos\beta_i + V_i\cos\alpha_i\sin\beta_i \\ &\quad + X_i\sin\alpha_i\cos\beta_i + U_i\sin\alpha_i\sin\beta_i \end{aligned} \quad (13)$$

$$Z_{i+1} = Z_i\cos\beta_i - W_i\sin\beta_i \quad (14)$$

where U_i, V_i and W_i are defined as follows.

$$U_i = R_i\cos\theta_i\cos\phi_i \quad (15)$$

$$V_i = R_i\sin\theta_i\cos\phi_i \quad (16)$$

$$W_i = R_i\sin\phi_i \quad (17)$$

It is noted that U_{i+1}, V_{i+1} and W_{i+1} can be written by

$$\begin{aligned} U_{i+1} &= U_i\cos\alpha_i\cos\beta_i - X_i\cos\alpha_i\sin\beta_i \\ &\quad - V_i\sin\alpha_i\cos\beta_i + Y_i\sin\alpha_i\sin\beta_i \end{aligned} \quad (18)$$

$$\begin{aligned} V_{i+1} &= V_i\cos\alpha_i\cos\beta_i - Y_i\cos\alpha_i\sin\beta_i \\ &\quad + U_i\sin\alpha_i\cos\beta_i - X_i\sin\alpha_i\sin\beta_i \end{aligned} \quad (19)$$

$$W_{i+1} = W_i\cos\beta_i + Z_i\sin\beta_i \quad (20)$$

Based on equations (6), (7) and (8), equations (12), (13), (14), (18), (19) and (20) can be computed by using the following set of CORDIC rotations.

$$U_{i+1} = \frac{1}{k_i^2}(U_i - X_i\rho_i 2^{-i} - V_i\delta_i 2^{-i} + Y_i\delta_i\rho_i 2^{-2i}) \quad (21)$$

$$V_{i+1} = \frac{1}{k_i^2}(V_i - Y_i\rho_i 2^{-i} + U_i\delta_i 2^{-i} - X_i\delta_i\rho_i 2^{-2i}) \quad (22)$$

$$W_{i+1} = \frac{1}{k_i}(W_i + Z_i\rho_i 2^{-i}) \quad (23)$$

$$X_{i+1} = \frac{1}{k_i^2}(X_i + U_i\rho_i 2^{-i} - Y_i\delta_i 2^{-i} - V_i\delta_i\rho_i 2^{-2i}) \quad (24)$$

$$Y_{i+1} = \frac{1}{k_i^2}(Y_i + V_i\rho_i 2^{-i} + X_i\delta_i 2^{-i} + U_i\delta_i\rho_i 2^{-2i}) \quad (25)$$

$$Z_{i+1} = \frac{1}{k_i}(Z_i - W_i\rho_i 2^{-i}) \quad (26)$$

where

$$\cos\alpha_i = \frac{1}{\sqrt{1+2^{-2i}}} \quad (27)$$

$$\sin\alpha_i = \frac{\delta_i 2^{-i}}{\sqrt{1+2^{-2i}}} \quad (28)$$

$$\cos\beta_i = \frac{1}{\sqrt{1+2^{-2i}}} \quad (29)$$

$$\sin\beta_i = \frac{\rho_i 2^{-i}}{\sqrt{1+2^{-2i}}} \quad (30)$$

$$k_i = \sqrt{1+2^{-2i}} \quad (31)$$

In the 2-D CORDIC rotation, $\alpha_i = \delta_i \tan^{-1} 2^{-i}$, $\beta_i = \rho_i \tan^{-1} 2^{-i}$, and δ_i and ρ_i are $\in \{-1, 1\}$.

Equations (21) and (22) can be expressed in the matrix form, which is given by

$$\begin{bmatrix} U_{i+1} \\ V_{i+1} \end{bmatrix} = \frac{1}{k_i^2}\begin{bmatrix} 1 & -\delta_i 2^{-i} \\ \delta_i 2^{-i} & 1 \end{bmatrix}\begin{bmatrix} U_i \\ V_i \end{bmatrix} - \frac{1}{k_i^2}\rho_i 2^{-i}\cdot\begin{bmatrix} 1 & -\delta_i 2^{-i} \\ \delta_i 2^{-i} & 1 \end{bmatrix}\begin{bmatrix} X_i \\ Y_i \end{bmatrix} \quad (32)$$

Similarly, equations (24) and (25) can be rewritten by

$$\begin{bmatrix} X_{i+1} \\ Y_{i+1} \end{bmatrix} = \frac{1}{k_i^2}\begin{bmatrix} 1 & -\delta_i 2^{-i} \\ \delta_i 2^{-i} & 1 \end{bmatrix}\begin{bmatrix} X_i \\ Y_i \end{bmatrix} + \frac{1}{k_i^2}\rho_i 2^{-i}\cdot\begin{bmatrix} 1 & -\delta_i 2^{-i} \\ \delta_i 2^{-i} & 1 \end{bmatrix}\begin{bmatrix} U_i \\ V_i \end{bmatrix} \quad (33)$$

It is noted that there are four 2-D CORDIC rotations involved in the 3-D rotation of a vector. In addition, the scale factor of Z_{i+1} and W_{i+1} is different from that of $U_{i+1}, V_{i+1}, X_{i+1}$ and Y_{i+1}. They can be compensated via the pre-scale of inputs or post-scale of outputs with their respective constants K and K^2, which are given by

$$K = \prod_{i=0}^{n-1} k_i \quad (34)$$

$$K^2 = \prod_{i=0}^{n-1} k_i^2 \quad (35)$$

4 VLSI Architecture for 3-D Vector Interpolator

Vector interpolation can be obtained by using algorithms based on spherical interpolation, linear interpolation or CORDIC interpolation. The spherical interpolation involves complex computations. The linear interpolation requires post normalization, which is also complex. The proposed CORDIC-based 3-D interpolator, which is performed on polar components, is efficient and very flexible in terms of the hardware implementations. Figure 2 shows the architecture of the proposed 3-D vector interpolator by using the circular CORDIC algorithm with rotation mode. In which, the generators of (U_{i+1}, V_{i+1}) and (X_{i+1}, Y_{i+1}) consist of two 2-D CORDIC processors, two hardwire shifters, and two adders/sub-tractors. The generators of W_{i+1} and Z_{i+1} consist of half 2-D CORDIC Processor.

The initial coordinates (U_0, V_0, W_0) are obtained by using the auxiliary generator (U_0, V_0, W_0) [6], which is shown in Figure 3. Thus, the proposed architecture is composed of the auxiliary generator (U_0, V_0, W_0), the redundant CORDIC arithmetic (for the computation of 3-D vector interpolation), and dual-memory banks (for storing the coordinates (X_i, Y_i, Z_i) and (U_i, V_i, W_i), respectively).

The hardware code of the proposed system is written in Verilog-hardware description Language (HDL) [7]. The system diagram is shown in Figure 4. The control unit is designed by the finite state machine (FSM), the state diagram of FSM is shown in Figure 5, and the hardware code of FSM is shown in Appendix.

The chip is synthesized by TSMC 0.18 μm 1P6M CMOS cell libraries [8]. The gate count is reported by the Synopsys® design analyzer [8]. The power consumption is reported by PrimPower®[8]. The layout view of the 32-bit 3-D vector interpolator is shown in Figure 6. The core size is $5320\mu m \times 5320\mu m$, and the power dissipation is 49.35mW with the clock rate of 20 MHz at 1.8V. The critical path is 14.27 ns. All control signals are generated internally on-chip. This chip offers a high throughput with low gate counts by using a parallel-pipelined architecture.

5 Advantages of New Architectures and Algorithms

The Euler angle method takes a sequence of three rotations [2], [9], each of which rotates with respect to one of the three orthogonal axes. This method can be represented by the Euler angles corresponding to the sequence of rotations with respect to the coordinate axes. In [2], the 3-D rotation is implemented by cascading two 2-D CORDIC processors. Lang and Antelo developed a method to replace the two 2-D CORDIC processors by one 3-D CORDIC processor [7]. The sequence of rotations is composed of one 2-D CORDIC rotation followed by one 3-D CORDIC rotation. Both of the aforementioned methods require more than two 2-D CORDIC computations. In the proposed 3-D rotation algorithm, the architecture based on the conventional CORDIC processor requires one 2-D CORDIC computation in parallel with five CORDIC processors. The auxiliary generator of coordinate (U_0 , V_0 , W_0) and the redundant arithmetic CORDIC for 3-D rotation can perform in parallel.

Four intermediate vectors between V1 and V2 are shown in Figure 7. Vector interpolator implemented by CORDIC consists of two steps. First step interpolates the polar components of the two given vectors linearly according to the position of intermediate vector. Instead of vector normalization, 3-D CORDIC vector rotation is performed to produce normalized vector in the second step.

We make the function of 3-D geometry rotation and graphic rendering by using the proposed algorithm and architecture is shown in Figure 8, Figure 8(a) shows the original image, and Figure 8(b) shows the rotated and rendered image. 3-D vector interpolation could achieve more than 70% energy saving without image quality degradation. We have a power-aware graphic system.

6 Conclusion

High-throughput architecture for the 3-D vector interpolation task based on the CORDIC algorithm is presented. It takes only one conventional CORDIC computation time.

The proposed architecture by the use of CORDIC processor is simple, regular and therefore suitable for VLSI implementation. In power-aware 3-D graphics rendering, the performance of 3-D vector interpolation can be improved by using the proposed algorithm and architecture. Table 1 shows the comparison of this work with Eberly [10] , Lang and Antelo [9] and Euh [2].

References:

[1] B. Phong, Illumination for Computer Generated Pictures, *Communications of the ACM*, 1975, pp.311-317.

[2] J. Euh, J. Chittamuru, W. Burson, CORDIC Based Interporator for 3-D Graphics, *IEEE Workshop on Signal Processing Systems*, 2002, pp.240-245.

[3] J. E. Volder, The CORDIC Trigonometric Computing Technique, *IRE Transactions on Electronic Computers*, Vol. EC-8, 1959, pp.330-334.

[4] J. S. Walther, A Unified Algorithm for Elementary Functions, *Spring Joint Computer Conference Proceedings*, Vol.38, 1971, pp.379-385.

[5] O. Mencer, L. Semeria, M. Morf, J. Delosme, Application of Reconfigurable CORDIC Architecture, *The Journal of VLSI Signal Processing, Special Issue on Reconfigurable Computing*, 2000.

[6] T.-Y. Sung, Y.-H Hu, H.-J. Yu, Doubly Pipelined CORDIC Array for Digital Signal Processing, *Int'l Conf. on Acoustic, Speech and Signal Processing*, Tokyo, Japan , 1986, pp.1169-1172.

[7] D. E. Thomas, P. H. Moorby, *The Verilog Hardware Description Language*, Fifth Edition, Kluwer Academic Pub., 2002.

[8] Synopsys, *http://www. synopsys. com/products.*

[9] T. Lang, E. Antelo, High-Throughput CORDIC-Based Geometry Operations for 3D Computer Graphics, *IEEE Transactions on Computers*, Vol. 54. No. 3, 2005, pp.347-361.

[10] D. H. Eberly, *3-D Game Engine Design-A Practical Approach to Real-Time Computer Graphics*, Morgan Kaufmann Pub., 2001.

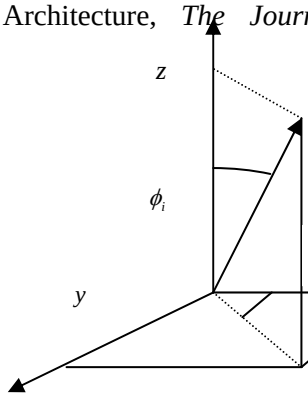

Fig.1. Vector *R* in the 3-D space

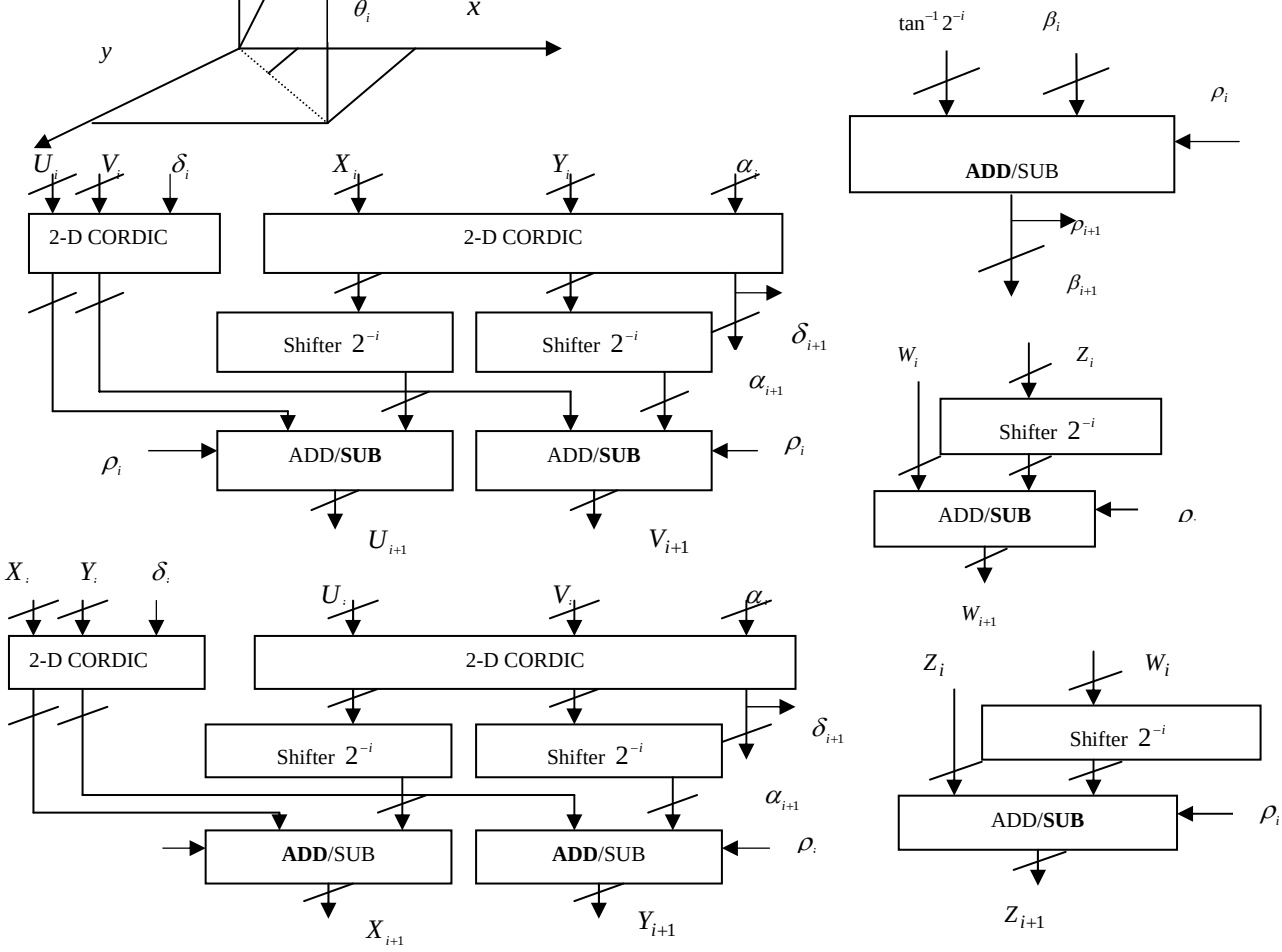

Fig.2. The architecture of the 3-D vector interpolator with the CORDIC algorithm

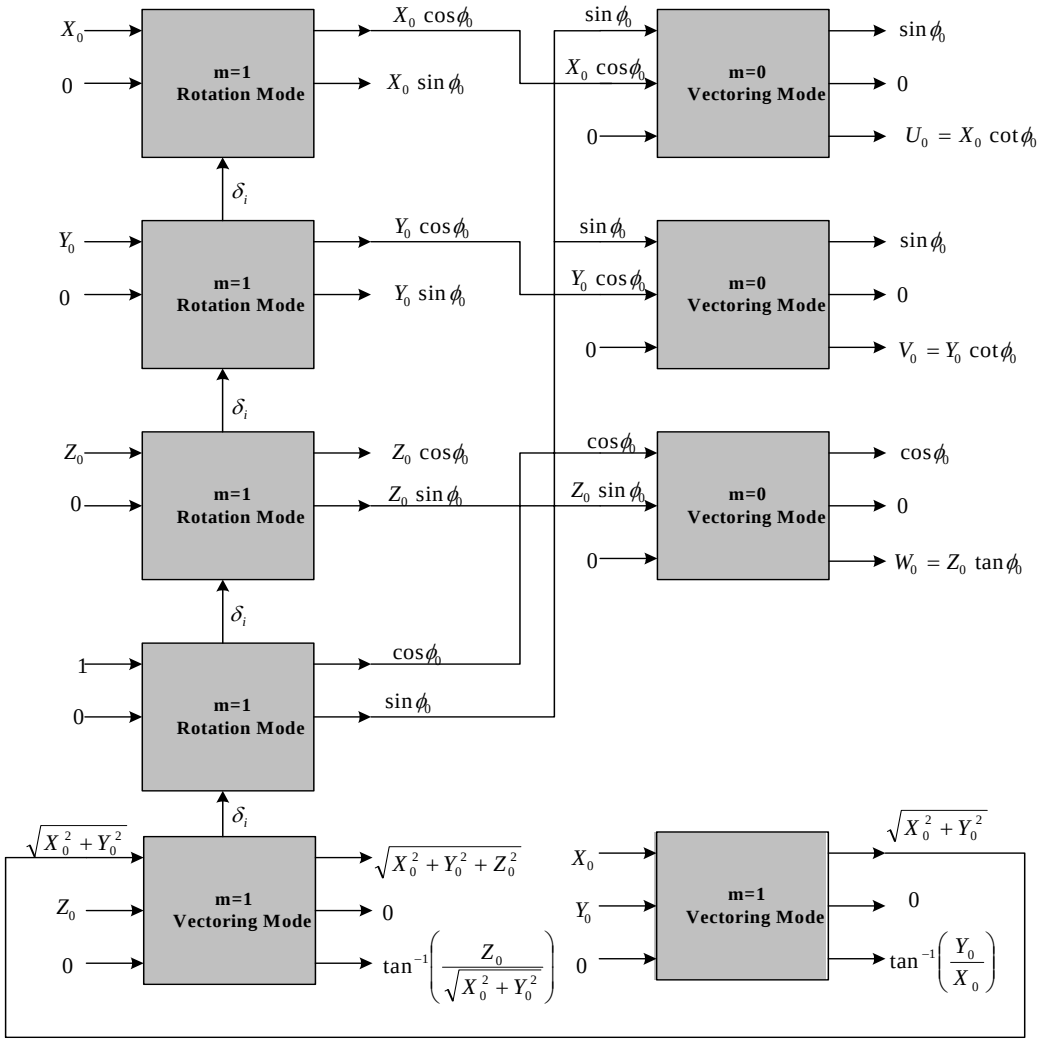

Fig.3. The auxiliary coordinate (U_0, V_0, W_0)

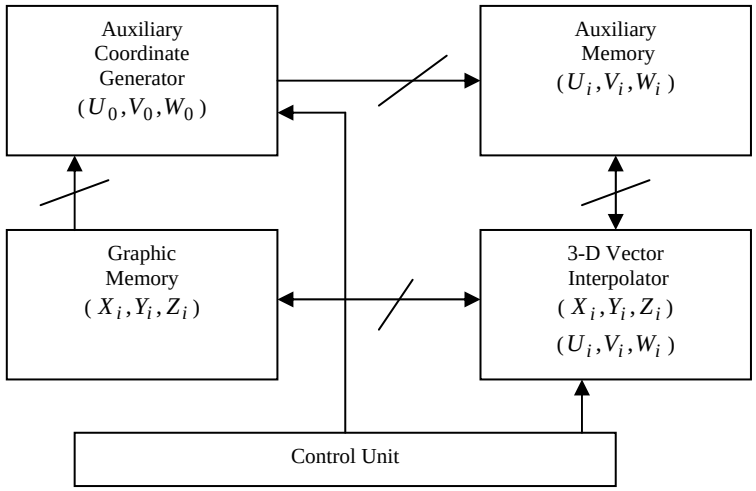

Fig.4. The system diagram of 3-D vector interpolator

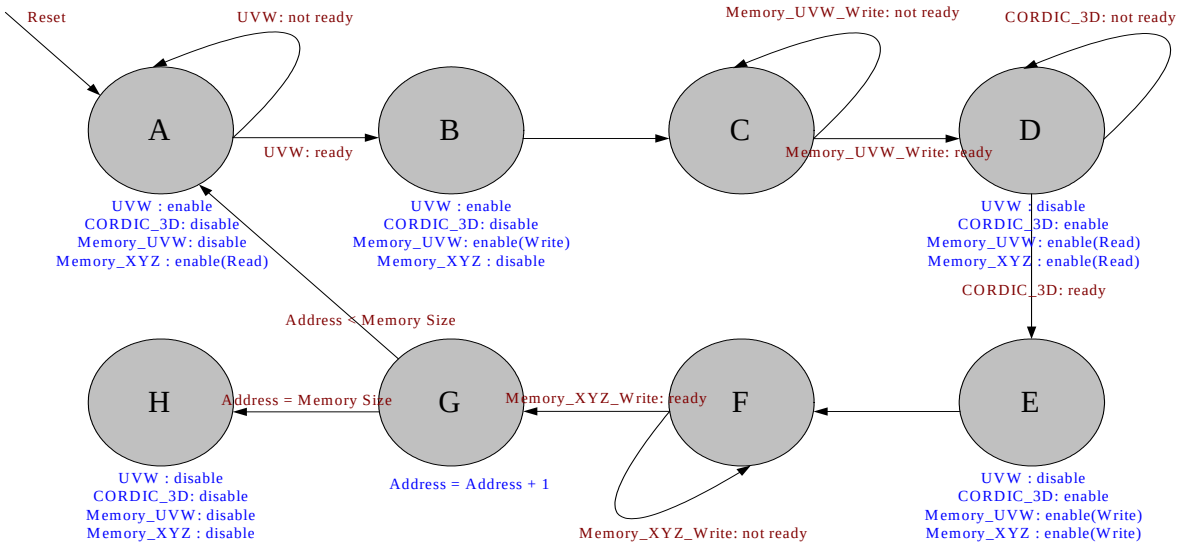

Fig.5. The state diagram of FSM of control unit

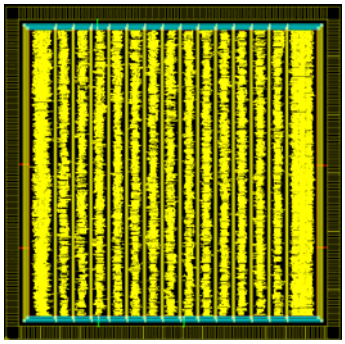

Fig.6. The layout view of 32-bit 3-D vector interpolator

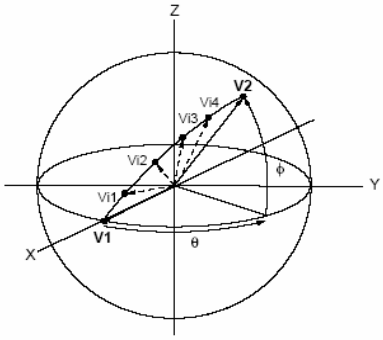

Fig.7. Vector interpolation

(a) The original image

(b) The rotated and rendered image

Fig.8. The image rotation and rendering in 3-D space

Table 1 The Comparison of 3-D rotation with Eberly, Lang & Antelo and this work

3-D Graphics Rendering	Eberly [10]	Lang & Antelo [9]	Euh [2]	This work
Coordinate system	Cartesian	Cartesian	Polar	Polar
No. of 2-D CORDIC processor	1	3	2	5
No. of memory bank	1	1	1	2
CORDIC computation(s)	3	2	2	1
Auxiliary coordinate generator	No.	No.	No.	Yes
Throughput	Low	Medium	Medium	High