Sistemas Digitales

(66.17/86.41)

Trabajo práctico final

Diseño de un motor de rotación gráfico 3D basado en el algoritmo CORDIC

- 2017 -

1. Objetivo

En el presente Trabajo Práctico el alumno desarrollará una arquitectura de rotación de objetos 3D basada en el algoritmo CORDIC. El objetivo principal es desarrollar tanto la unidad aritmética de cálculo como así también el controlador de video asociado.

Para la realización completa del Trabajo Práctico se cargarán en memoria externa las coordenadas correspondientes a un objeto tridimensional predefinido mediante una interfaz serie UART. A partir de los valores de las componentes, se rotará el objeto alrededor de cada uno de los ejes de coordenadas según el valor que adquieran las entradas del sistema, y por último las componentes rotadas serán presentadas en un monitor VGA mediante la aplicación de una proyección plana. En la Figura 1 puede observarse un diagrama en bloques del sistema completo. Deberá determinarse la cantidad mínima de bits de ancho de palabra (bits de precisión) para alcanzar las especificaciones requeridas.

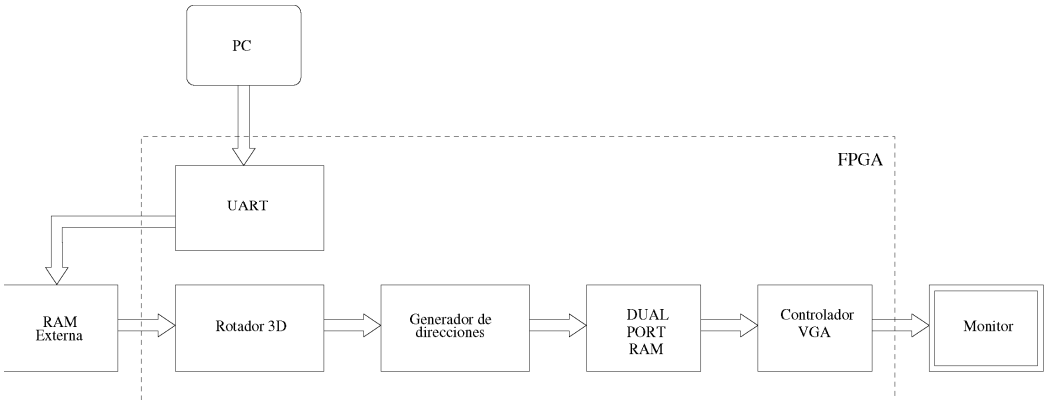

Figura 1. Esquema completo del sistema

2. Introducción teórica

2.1. Proyección plana

Sobre un monitor, un objeto tridimensional debe ser siempre representado utilizando una proyección plana, de forma tal de dar sensación tridimensional. Distintas proyecciones planas pueden encontrarse, pero en general puede decirse que una proyección plana es una función que asocia a un vector a = (x, y, z) el vector b = (v, w), tal como se observa en la Figura 2.

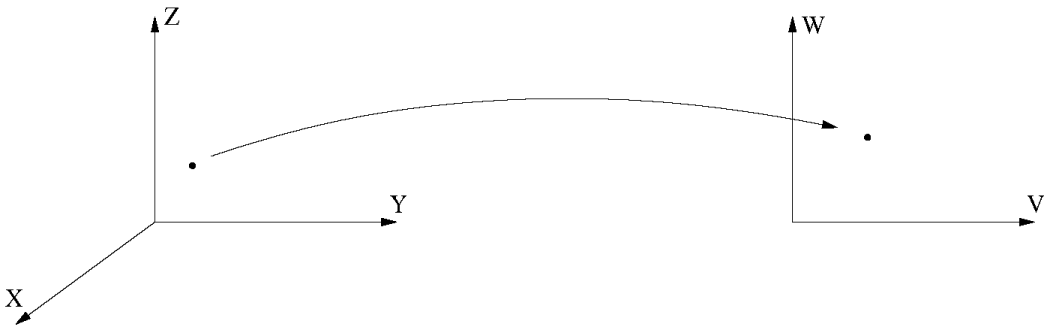

Figura 2. Proyección plana

Esta transformación no siempre resulta lineal y por lo tanto, en general, no puede ser representada por un producto matricial.

Sin embargo, en el presente Trabajo Práctico se considerará la función de proyección plana más simple que puede utilizarse, la cual consiste en la eliminación de una de las coordenadas, tal como se observa el la Figura 3.

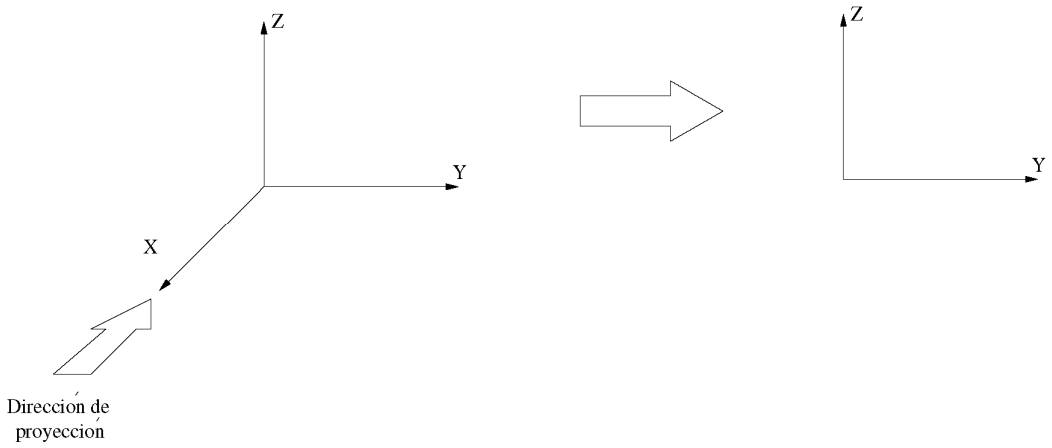

Figura 3. Eliminación de una de las coordenadas

Por lo tanto, una vez calculadas las coordenadas tridimensionales del objeto rotado, puede directamente quitársele una de las componentes obteniendo la proyección plana del mismo. Por ejemplo, todo punto a = (x, y, z) que forma la superficie de un objeto tridimensional, puede ser proyectado en dos dimensiones haciéndole corresponder el punto b = (y, z).

2.2. Rotaciones 3D

A continuación, $\mathbf{R}_x(\alpha)$, $\mathbf{R}_y(\beta)$ y $\mathbf{R}_z(\gamma)$ representan la rotación de un objeto en un ángulo α, β y γ alrededor del eje X, Y y Z, respectivamente. En las Figuras 4, 5 y 6 se observan dichas rotaciones.

$$\mathbf{R}_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix}$$

$$\mathbf{R}_y(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix}$$

$$\mathbf{R}_z(\gamma) = \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Ecuaciones (1), (2) y (3)

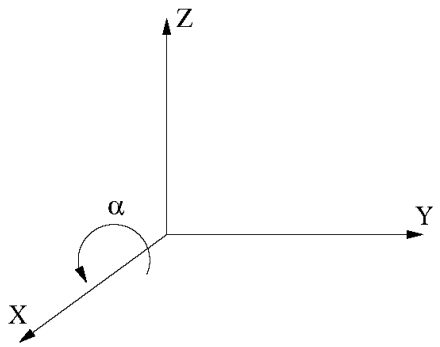

Figura 4. Rotación alrededor del eje X

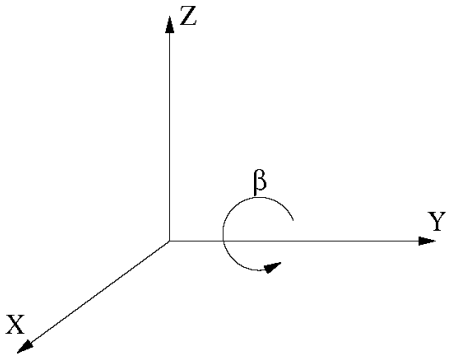

Figura 5. Rotación alrededor del eje Y

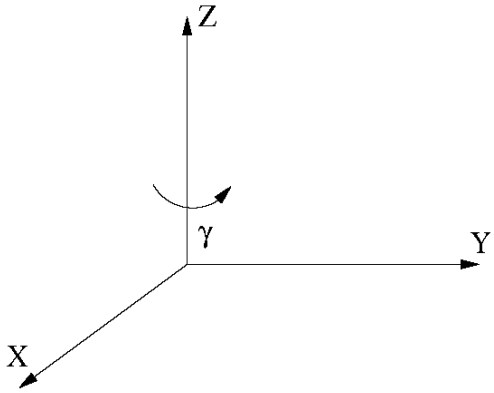

Figura 6. Rotación alrededor del eje Z

Al ser la transformación de rotación representable por una transformación lineal, una rotación arbitraria puede ser descompuesta siempre en las tres rotaciones básicas mencionadas anteriormente, y por lo tanto, esta es representable como el producto de las matrices $\mathbf{R}x(\alpha)$, $\mathbf{R}y(\beta)$ y $\mathbf{R}z(\gamma)$ [1]. Debe notarse que, en general, el producto matricial no resulta conmutativo, pudiéndose definir seis rotaciones compuestas:

$$y = \mathbf{R}_{xyz}(\alpha, \beta, \gamma)x = \mathbf{R}_x(\alpha)\,\mathbf{R}_y(\beta)\,\mathbf{R}_z(\gamma) \quad (4)$$

$$y = \mathbf{R}_{xzy}(\alpha, \beta, \gamma)x = \mathbf{R}_x(\alpha)\,\mathbf{R}_z(\gamma)\,\mathbf{R}_y(\beta) \quad (5)$$

$$y = \mathbf{R}_{yxz}(\alpha, \beta, \gamma)x = \mathbf{R}_y(\beta)\,\mathbf{R}_x(\alpha)\,\mathbf{R}_z(\gamma) \quad (6)$$

$$y = \mathbf{R}_{yzx}(\alpha, \beta, \gamma)x = \mathbf{R}_y(\beta)\,\mathbf{R}_z(\gamma)\,\mathbf{R}_x(\alpha) \quad (7)$$

$$y = \mathbf{R}_{zxy}(\alpha, \beta, \gamma)x = \mathbf{R}_z(\gamma)\,\mathbf{R}_x(\alpha)\,\mathbf{R}_y(\beta) \quad (8)$$

$$y = \mathbf{R}_{zyx}(\alpha, \beta, \gamma)x = \mathbf{R}_z(\gamma)\,\mathbf{R}_y(\beta)\,\mathbf{R}_x(\alpha) \quad (9)$$

En este trabajo se considerará la rotación $\mathbf{R}_{xyz}(\alpha, \beta, \gamma)$

[1] En general, se utiliza el concepto de cuaterniones cuando se trabaja con transformaciones 3D en imágenes. Sin embargo, el objetivo del presente trabajo práctico consiste en utilizar la arquitectura CORDIC como motor de cálculo, por lo que la introducción del concepto de cuaternión carece de sentido.

3. Desarrollo

La Figura 7 muestra una imagen 2D del objeto a representar. Las coordenadas 3D de los puntos que lo forman se encuentran especificados en el archivo coordenadas.txt, disponible en el campus de la materia. Los mismos están normalizados a un valor máximo de 1 y cuentan con 10 decimales de precisión.

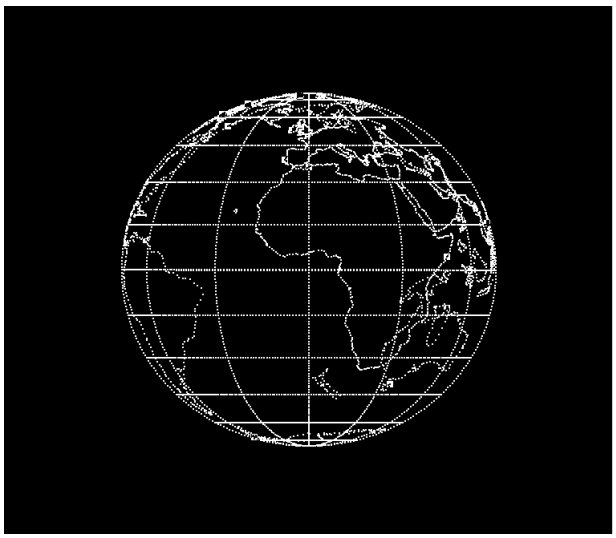

Figura 7. Imagen 2D del objeto a representar

Una vez cargada la imagen en una memoria externa (incluida en el kit de desarrollo), el sistema debe leer el valor lógico de 6 pulsadores de entrada, los cuales indican la rotación alrededor de cada uno de los ejes. Se utiliza un par de pulsadores para cada eje, uno de los cuales indica si la rotación se realiza en sentido positivo y el otro en sentido negativo en el caso en que los mismos se encuentren presionados. Caso contrario, no se produce rotación alguna sobre el eje correspondiente.

Para determinar el ángulo de rotación, cada pulsador es utilizado como entrada de un Up/Down counter, el cual incrementa o decrementa el ángulo de rotación por un paso fijo de ángulo $\Delta\phi$. En la Figura 8, se muestra el esquema correspondiente.

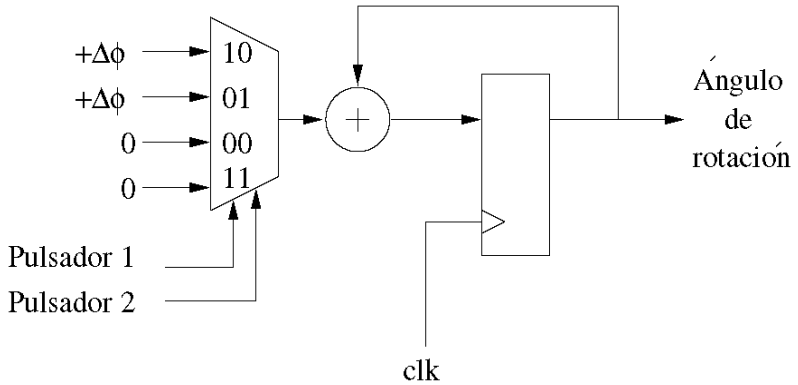

Figura 8. Esquema de un Up/Down Counter

Una vez que se poseen los valores de los ángulos de rotación, debe procederse a realizar la rotación propiamente dicha. Debe observarse a partir de la ecuación (4) que la forma de realizar esto implica el uso de sumadores, multiplicadores, cálculo de senos y cosenos. Sin embargo, el algoritmo CORDIC realiza la rotación de un vector (x_0, y_0) en un ángulo z_0. Por

lo tanto, el producto de un vector de entrada (punto en coordenadas 3D del objeto a representar) por la sucesión de las matrices de rotación presentadas en las ecuaciones (1), (2) y (3), puede ser llevado a cabo mediante el uso del algoritmo CORDIC. El esquema de la Figura 9 muestra una forma de realizar dicho producto matricial.

Debe notarse que si la ganancia de procesamiento de un CORDIC de n etapas corresponde a A_n, entonces la ganancia de la arquitectura anterior corresponde a $(A_n)^3$.

A continuación, debe ser realizada la proyección plana. En este caso, se elimina directamente la componente x de las coordenadas rotadas, obteniéndose las componentes (y, z) a ser representadas en pantalla.

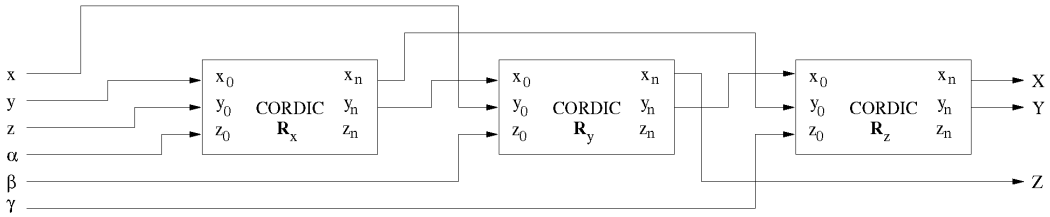

Figura 9. Esquema de rotador 3D

Los datos de salida del rotador deberán ser almacenados en una memoria (memoria de video) que deberá ser leída por un controlador (controlador de video) para mostrar la imagen en pantalla. La calidad de video será VGA monocromático de 1 bit de 640 x 480 pixeles. De esta forma, las coordenadas planas (y, z) de cada punto de la imagen corresponderán a una dirección de la memoria de video, donde se escribirá un '1' lógico en el caso en que en dicha posición de memoria se encuentre un punto de la imagen. Por lo tanto, cada coordenada deberá ser mapeada a una dirección de memoria tal como se detalla en el esquema de la Figura 10. Dicha memoria constará de 307200 posiciones de 1 bit cada una.

Debido a que el controlador VGA y el rotador escriben y leen a la vez direcciones de memoria de video que son independientes, la memoria de video deberá ser implementada por medio de una DUAL PORT RAM, utilizando para ello las Block RAMs disponibles en la FPGA.

Por último, cabe destacar que la memoria externa cuyo contenido corresponde a las coordenas 3D de la imagen a representar será cargada a través de una interfaz serie.

Notar que luego de producir una rotación del objeto deberá limpiarse la memoria de video. Caso contrario, aparecerán en pantalla la imagen actual y precedente a la rotación. Por lo tanto, entre dos rotaciones sucesivas debe realizarse un ciclo de limpieza de la memoria de video. Asimismo, no resulta necesario utilizar una memoria de 307200 direcciones, debido a que según la Figura 10 la imagen ocupará una superficie total mucho menor a la de la pantalla completa.

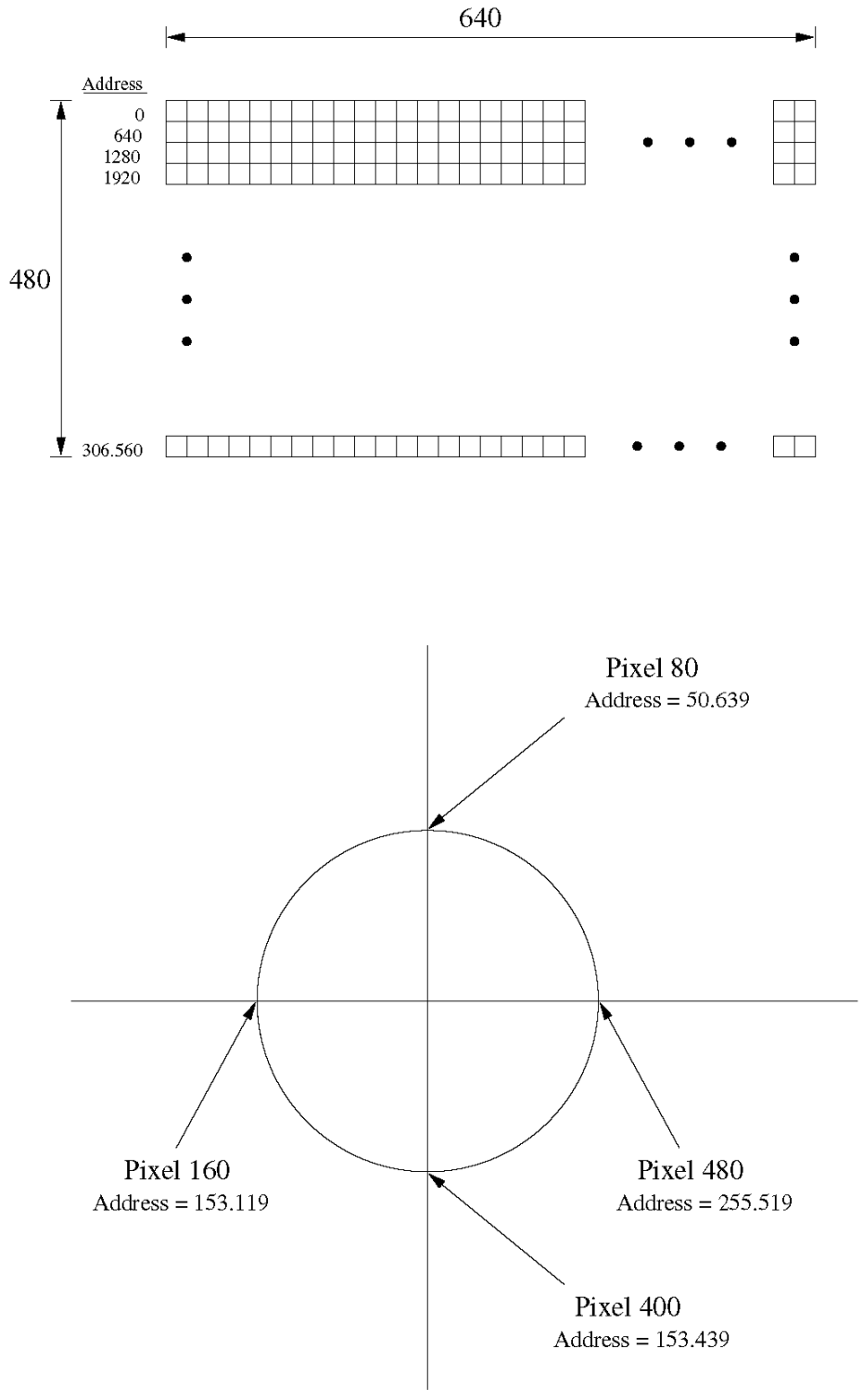

Figura 10. Mapeo de las direcciones en la pantalla

4. Especificaciones

- Resolución de video: 640 x 480, 1 bit monocromo, 50 Hz.
- Velocidad angular de rotación mínima en c/u de los ejes: 35.15625 grados/seg.

- Paso angular $\Delta\phi$: 0.703125 grados.
- Dispositivo: Spartan 3E-500 (Kit Nexys 2 o Starter Kit Board).

5. Ejercicios teóricos

Los ejercicios teóricos siguientes son voluntarios. La aprobación del trabajo práctico no depende de ellos.

1. Calcule la cantidad de operaciones aritméticas (sumas y multiplicaciones) equivalentes por segundo que el sistema deberá realizar en el caso de trabajarse con rotaciones basadas en matrices
2. Suponiendo que se dispone de un procesador RISC de 5 etapas (arquitectura Load-Store estricta, tipo [2,1] - dos operandos fuente, un destino implícito), con 16 registros internos de 32 bits cada uno, sin multiplicador interno, un tiempo de acceso a memoria de un ciclo de reloj y sin lógica de forwarding; calcule la frecuencia de reloj necesaria para soportar la cantidad de operaciones aritméticas por segundo calculadas en el ejercicio 1. (Nota: recuerde los ciclos de stall que se producen en los saltos). Suponga que para el cálculo del seno y del coseno se cuenta con una tabla en memoria de tamaño apropiado según las especificaciones de la sección 4. Suponga también una implementación del algoritmo CORDIC por software.
3. Idem 2, suponiendo un multiplicador interno con latencia de un ciclo de reloj.
4. Idem 3, suponiendo lógica de forwarding (unidad MAC - Multiply And Accumulate).