PROTOTIPO PARA EL GRABADO MECÁNICO DE PLACAS DE ALUMINIO, ASITIDO POR COMPUTADORA.

Mc Apolonio Ata Pérez
Facultad de Cs de la Computación, BUAP
Ciudad Universitaria, Puebla Pue.
apolonio@cs.buap.mx

RESUMEN

Se presenta el desarrollo a nivel de prototipo de un sistema para el grabado mecánico de placas de aluminio, estos sistemas son muy usados en la industria para grabar en piezas de distintos materiales logotipos, números de serie, letras, códigos, o cualquier otro símbolo.

El prototipo consiste básicamente de una mesa de coordenadas xyz, controlada por una computadora, sus movimientos son generados por medio de motores de pasos. Sobre el eje z tiene montado un taladro de alta velocidad, el cual emplea como herramienta de grabado una pequeña *fresa* de un milímetro de diámetros, para la comunicación entre la PC y la mesa, se utilizaron los puertos estándares de la PC.

El dibujo que se quiere grabar se edita primero en la computadora utilizando un programa de edición gráfica, que tenga la capacidad de generar archivos de salida con el formato HPGL (*Hewlett Packard Graphics Language*), por ejemplo: *Harvard Graphics, Corel Draw, Auto Cad* ,etc.

El programa de control toma el archivo HPGL, interpreta los comandos que lo forman, y los convierte en pulsos de control para motores que mueven la fresa de grabado, la cual al moverse reproduce fielmente el dibujo generado originalmente. .

Debido a que los movimientos de la mesa son controlados por la computadora, son confiables, rápidos y precisos, además se obtiene repetitividad en las operaciones de grabado.Al final de este trabajo se obtuvo un sistema capaz de grabar dibujos placas de aluminio con un área de trabajo de 210 x 280 mm^2 y se esta usando actualmente para el grabado de placas conmemorativas.

1. INTRODUCCION

El trabajo fue motivado al tratar de solucionar un problema real, el cual consistía en grabar placas conmemorativas hechas de aluminio, normalmente el grabado se hace por *fotograbado* que hace uso de productos químicos que son muy contaminantes, motivo por el cual se decidió utilizar un proceso mecánico de grabado.

Se diseño y construyo una mesa de coordenadas XYZ controlada por una computadora, y para la programación se trabajó con los algoritmos de Bresenham que sirven para la graficación de círculos y rectas en un ambiente discreto.

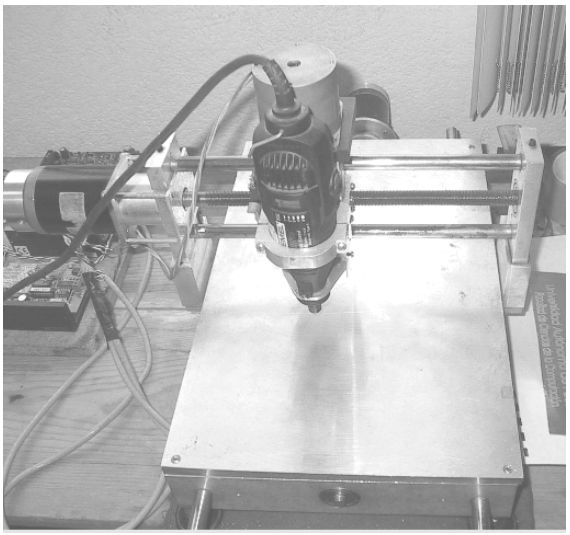
La contribución del trabajo es, además de resolver el problema para lo cual fue construido permite conocer y manejar los conceptos básicos de un sistema de CAD-CAM.

2. SISTEMA MECÁNICO

2.1 MESA DE COORDENADAS.

Permite el movimiento de la pieza de trabajo a lo largo de los ejes x-y, utiliza un husillo que transforma el movimiento circular de los motores en un movimiento lineal.

La mesa tiene 25cm de ancho por 40 cm. de largo y fue construida en aluminio, el husillo fue hecho en acero con paso de 20 hilos por pulgada (1.27mm).



Las características de la mesa son:

Desplazamiento total en X	200 mm.
Desplazamiento total en Y	300 mm.
Desplazamiento total en Z	50 mm.
Paso por vuelta en el husillo X	1.27 mm
Paso por vuelta en el husillo Y	1.27 mm
Paso por vuelta en el husillo Z	2.116 mm

Hilos por pulgada del husillo X,: 20
Hilos por pulgada del husillo Y: 20

2.2 MOTORES DE PASO

El Motor de Pasos es un dispositivo electromecánico que convierte pulsos digitales de entrada, en un movimiento angular, se usa frecuentemente para rotar un eje con incrementos precisos, cuando se le da la secuencia apropiada de pulsos de entrada, es capaz de reproducir movimientos muy precisos. Los motores de paso tienen la característica de no tener corrimiento en su posición , su rotor tiene poca inercia y no genera error acumulativo de posición.

Los motores utilizados en la mesa de coordenadas para los movimientos x y z, son de la compañía *Microkinetics* y tienen las siguientes características:

Motor Z

Modelo	23M89
Torque	(83oz/in) 0.581Nm
Corriente por fase	1.2 Amp
Voltaje	6.0 volts
Pulsos por vuelta	200

Motor X,Y

Modelo	23M110
Torque	(112oz/in) 0.784Nm
Corriente por fase	4.2 Amp
Voltaje	4.7 volts
Pulsos por vuelta	200

Los motores de paso acoplados a un husillo, permiten transformar un código digital de entrada en un desplazamiento lineal. El desplazamiento mínimo de la mesa en los ejes x-y, utilizando un husillo con un avance de 20 hilos por pulgada por vuelta será de: (25.41/(200*20)) =0.00635 de mm, El desplazamiento mínimo del taladro en el eje Z, utilizando un husillo de 12 hilos por pulgada sera de 25.4/(200*12)=0.01058 mm

2.3.- CONTROLADORES DE LOS MOTORES.

El controlador del motor es el circuito electrónico de potencia que controla el funcionamiento de los motores. Transforman una señal digital de entrada de baja potencia (5v, 20ma), en un señal digital con la potencia necesaria para energizar las bobinas de los motores (4.2v , 4.7A por bobina).

Para el manejo de los motores se emplearon controladores de la compañía Microkinetics, el DR3535 de 3.5 amps para los motores de X y Z, el DM4050 de 5 amps para el motor en Y.

Estos funcionan con las señales de dirección y pulso, una señal recibe el sentido y la otra el pulso para mover e motor.

2.3.- COMUNICACIÓN CON LA PC.

Para el envío de los pulsos de control a los controladores y le recepción de los estados de los interruptores de limite se utilizó el puerto paralelo de la PC[1]

El programa de control genera y envía a los controladores de los motores los pulsos digitales necesarios para generar el movimiento de los motores, la comunicación se realiza por medio del puerto paralelo de la PC; Las señales presentes en el puerto paralelo y su utilización en el controlador son:

Conector Paralelo Terminal/ bit	**Conector del controlador Señal**
2 b0	Pulso del motor X
3 b1	Dirección del motor X
4 b2	Pulso del motor Y
5 b3	Dirección del motorY
6 b4	Pulso del motor Z
7 b5	Dirección del Motor Z
8 b6	No usada
9 b7	No usada
15 b3	Limite en X (ent)
13 b4	Limite en Y (ent)
12 b5	Limite en Z (ent)

Para generar un movimiento es necesario enviar un pulso e indicar la dirección.

Con los bits de b0 a b6 se crea la palabra de control. **MOTXYZ = b7 b6 b5 b4 b3 b2 b1 b0** donde
b7 es No Usado,
b6 es No Usado,
b5 es Dir Mot Z
b4 es Pulso Mot Z
b3 es Dir Mot y
b2 es Pulso Mot y
b1 es Dir Mot X
b0 es Pulso Mot X

Al programar, se le asignó a una variable el nombre de MOTXYZ. La palabra de control, se envía al puerto paralelo por medio de la instrucción en lenguaje "C", ***outport (MOTXYZ, 0x378);*** , donde 278H es la dirección del puerto paralelo seleccionado

La PC recibe los estados de *límite* de la mesa XYZ, por medio del puerto paralelo. Para conocer el estado de los interruptores de límite, se lee el puerto usando la instrucción en lenguaje "C" ***limite_xyz = inport (0x379) ,***,donde 379H es la dirección del puerto seleccionado y **"limite_xyz"** es la palabra de control leída. La palabra leída se interpreta de la siguiente forma:

b0,b1,b2	No usado
b3	Limite en X
b4	Limite en Y
b5	Limite en Z
b6,b7	No Usado

Por lo cual, si se desea saber si se ha llegado al limite sobre el eje X, hay que determinar el valor del bit3.

3.- TRAZADO DE LINEAS Y CÏRCULOS

La mesa de coordenadas x-y, se desplaza por medio de incrementos, en **x** y en **y**, los cuales son producidos por los motores de pasos conectados a sus ejes. Si estos desplazamientos se utilizan para hacer figuras complicadas, es necesario plantear primero la forma de mover la mesa para obtener figuras más simples, por esta razón hay que trazar primero líneas o círculos con base a desplazamientos discretos en X y en Y.

A partir de estos algoritmos se pudo programar los comandos más usados por el lenguaje HPGL, que es el formato en el que vienen los archivos usados por el programa de control desarrollado.

3.1-TRAZADO DE LÍNEAS.

La programación de una función para dibujar líneas a partir de conocer el origen y fin de la línea, es una rutina fundamental en cualquier sistema x-y discreto.

Aunque realmente es fácil dibujar líneas horizontales y verticales , es más difícil crear una función que dibuje líneas en diagonal . Por ejemplo, para dibujar una línea desde la posición (0,0) a la (80,120), hay que determinar los puntos de la recta entre estas dos coordenadas.

Lo primero que hay que hacer para diseñar una función que dibuje líneas es usar la relación entre el cambio de las dimensiones de **X** y de **Y**. Para esto, por ejemplo si se considera una recta que va de (0,0) a (5,10); el cambio en **X** es de 5 y en **Y** de 10. La relación en este caso es de ½, y se usa para determinar la relación de cambio entre las coordenadas **X** e **Y** a lo largo de la línea. En este caso la coordenada X se incrementa la mitad de veces que la coordenada Y. Aunque este método matemáticamente es fácil de entender, en el momento de trabajar con esta propiedad se presentan situaciones, con problemas serios de redondeo Por esta y otras razones un mejor método para dibujar líneas o círculos es emplear el algoritmo de Bresenham. [2]

El mejor método desarrollado para dibujar una línea es por medio del algoritmo de Bresenham.

Aunque está basado conceptualmente en las relaciones entre las distancias X y Y, no se requiere división alguna o cálculo en punto flotante. En su lugar, la relación entre el cambio en los valores de **X** e **Y** se maneja implícitamente a través de la suma y resta de series .

La idea básica sobre la que se apoya este algoritmo es la de registrar en cada punto el error entre la posición real y la posición ideal.

El error entre la posición real y la ideal es debido a las limitaciones mecánicas. Debido al hecho de que la mesa de coordenadas x-y no tiene una resolución infinita entonces, la posición actual de cada punto en la línea debe ser la aproximación mas cercana.

En cada iteración del ciclo hecho para dibujar, las dos variables llamadas error-x y error-y se incrementan con los cambios en la magnitud de las coordenadas **X** e **Y**, respectivamente.

Cuando el error llega al límite predeterminado, se le resta una constante al error y el contador de la coordenada es incrementado. Este proceso continúa hasta que la línea queda dibujada en su totalidad.

La figura 3.1 muestra los puntos calculados con este algoritmo para una recta con inicio en (1,2) y final en (10,6).

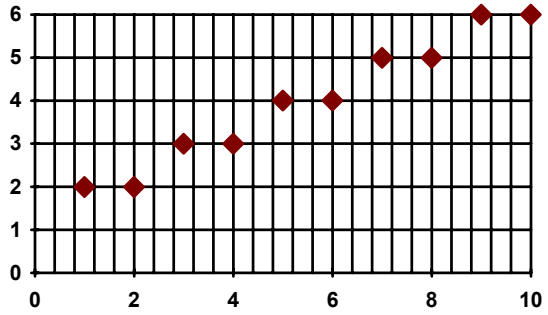


Figura 3.1 Puntos calculados con el algoritmo de Bresenham para una recta con inicio en (1,2) y final en (10,6).

3.2. TRAZADO DE CÍRCULOS.

Una forma estándar de la ecuación de la circunferencia es el teorema de pitágoras:

$$(x-xc)^2 + (y-yc)^2 = r^2$$

Esta ecuación puede usarse para trazar una circunferencia recorriendo el eje x en pasos unitarios de xc-r a xc+r y calculando los valores y correspondientes en cada posición como

$$y = yc + \sqrt{r^2 - (x-xc)^2}$$

La desventaja de este método es que la determinación de las posiciones de los puntos a lo largo de una circunferencia, mediante el uso de la ecuación cartesiana o polar del círculo; requiere una cantidad considerable de tiempo de computo. El enfoque del teorema de Pitágoras implica multiplicaciones y extracción de raíces cuadradas, mientras que las ecuaciones paramétricas contienen cálculos trigonométricos y multiplicaciones.

La forma más rápida y mejor para trazar círculos es usar el algoritmo de Bresenham para trazado de círculos, en forma similar al algoritmo propuesto para trazar líneas. De igual manera no se requiere de cálculos en punto flotante, excepto para ciertos cálculos de razones, por lo que es bastante rápido.

4.- COMANDOS HPGL

Los comandos HPGL son un conjunto estándar de instrucciones seleccionadas que forman un lenguaje de graficación, y es actualmente el más utilizado por los fabricantes de graficadores. Este lenguaje fue creado por Hewlett-Packard. [3]

Como se menciono anteriormente el sistema trabaja con archivos gráficos compuesto de comandos HPGL. Fue necesario conocer la sintaxis de estos comandos, para efectuar la programación de su decodificador y a partir de esto generar los movimientos discretos en las direcciones X e Y que permiten la reproducción del dibujo en la mesa X-Y.

El conjunto de instrucciones del lenguaje HPGL está formado con aproximadamente 60 clases de comandos de dibujo. Los comandos consisten en mnemónicos de dos letras mayúsculas o minúsculas seguidas por parámetros y un símbolo de terminación. Los parámetros tienen que ser separados por al menos una coma o espacio, o por signo + o -. Un comando termina con un punto y coma, o por el código de alimentación de línea, o por el siguiente mnemónico.

La sintaxis del comando y sus ejemplos se muestran a continuación:

Sintaxis del comando

XX(sepop)parámetro1(sep)parámetro2(sepop)terminación

donde :

XX	Es el mnemónico del comando.
(sepop)	Es un separador opcional que puede ser coma o espacio
(sep)	Es un separador que puede ser coma o espacio
Parámetro	Son los parámetros del comando.
Terminación	Es generalmente un punto y coma.

Ejemplo:

PA1000 1000;	(sep = espacio)
PA1000.5,-100.5;	(sep = coma)
PA 32767,-32768 (LF);	(sep = LF:alimento de línea)

Para finalizar un archivo o conjunto de comandos HPGL, se emplea la etiqueta del carácter de terminación. Este carácter por default es el código ETX (fin-de-texto ASCII, equivalente al decimal 3, control C o CHR$(3)).

Si bien el lenguaje HPGL consta de aproximadamente 60 comandos, en la practica los editores gráficos que utilizan este lenguaje, solo utilizan un pequeño subconjunto de comandos, utilizando los comandos básicos para generar las funciones de los comandos complejos. Por esta razón solo se programó el decodificado y ejecución de los comandos básicos: AA, AR, CI, EW, RR, PU, PU, PD, EA, ER, RA, PA, OA.

5.- PROGRAMA DE CONTROL

El programa recibe como entrada un archivo de comandos HPGL, con este archivo genera los menús y presentaciones, interpreta los comandos, prueba la sintaxis, ejecuta los comandos, permite un manejo manual de la mesa XY y genera un dibujo previo en la pantalla; también envía los pulsos de control a los motores de paso, y de esta forma se puede grabar el dibujo.

Todas estas acciones fueron programadas en forma de módulos los principales son:

Simulador o Vista Previa:
Permite visualizar los movimientos de la mesa XY en forma real o simulada, Al simular el movimiento de la herramienta nos da una vista previa del dibujo, y permite comprobar su forma y tamaño antes de ser grabado.

Módulo de Grabado:
Realiza el grabado de un dibujo o letrero en forma manual o automática

Movimiento Manual:
Los movimientos del taladro se controlan en forma manual por medio del teclado.

Movimiento Paso a paso:
Graba un dibujo ejecutando un comando a la vez.

Movimiento Continuo:
Graba un dibujo en forma continua

Ejecución de comandosHPGL.
La ejecución de los comandos se realiza por medio de los siguientes submódulos.

a.-Obtiene una tecla.
Lee una tecla desde el teclado

b.-Lee una cadena
Lee una cadena de comandos

c.-Es carácter,número o letra
Determina el tipo del carácter leído.

d- Es un comando
Determina si la cadena de caracteres es un comando

e.-Rutina de análisis de sintaxis
Determina si hay errores de sintaxis

f.-Identifica Comandos HPGL
Determina el tipo de comando hpgl.

g.-Ejecuta un comando
Ejecuta la subrutina adecuada.

h.-Ejecuta una cadena
Ejecuta una cadena de comandos

i.-Mueve los motores de paso
Envía los pulsos de control a los motores de pasos, de acuerdo al comando HPGL ha ser ejecutado.

PROGRAMACION

Para el diseño del programa se empleó el enfoque TOP-DOWN, o el de **refinamiento por pasos**, que consiste en visualizar el problema en su forma general, y descomponer después el proceso en un cierto número de pasos más simples de manera que éstos puedan ser manejados más fácilmente. Luego, cada subproblema se

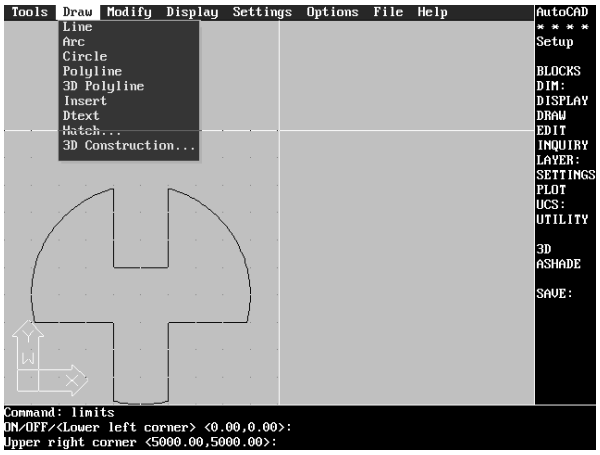
divide en problemas aún más pequeños, hasta que los pasos a seguir sean los **suficientemente detallados**.

Se utilizo el lenguaje "C" porque permite realizar programación estructurada, la cual intenta minimizar la posibilidad de error en el proceso de programación. La programación estructurada es una técnica que intenta utilizar al máximo los recursos del lenguaje, limita el conjunto de estructuras necesarias para programar y, utiliza una serie de reglas para coordinar adecuadamente el desarrollo de las diferentes fases de programación.

6.-PRUEBAS

6.1 Simulador (Vista Previa)

El simulador permite revisar dimensiones y aspecto del dibujo antes de mandarlo a grabar. Para probar el simulador se utilizaron tres editores gráficos Autocad, Harvard Graphics, y Corel DRAW. La figura 6.1a muestra un dibujo hecho en **Autocad,** después se salva como archivo de comandos hpgl. (archivo.plt). La figura 6.1b muestra la prueba del módulo de simulación (vista previa) utilizando el archivo generado por autocad, el dibujo presentado es similar en aspecto y dimensiones al generado inicialmente, por lo cual se hizo la prueba con otro editor.



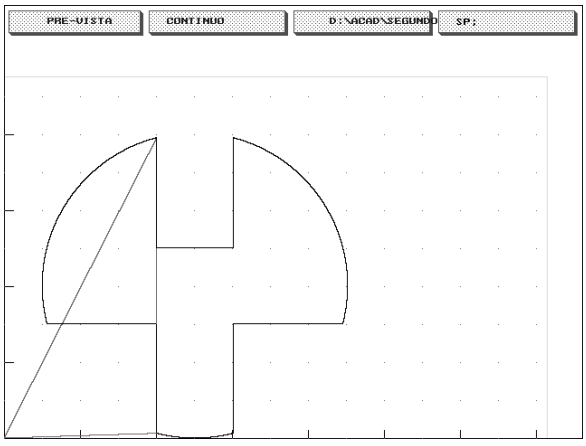


Figura 6.1 a) Diseño en AUTOCAD 10, b) Simulación del diseño.

Para la siguiente prueba se digitalizó una imagen y se vectorizo empleando CorelTRACE posteriormente se guarda el archivo con formato .PLT, el resultado de la simulación se muestra en la figura 6.1, el dibujo es similar en aspecto y dimensiones a la imagen digitalizada En el simulado aparecen los movimientos de la herramienta cuando esta cortando (en color blanco) y cuando no lo hace (color verde).

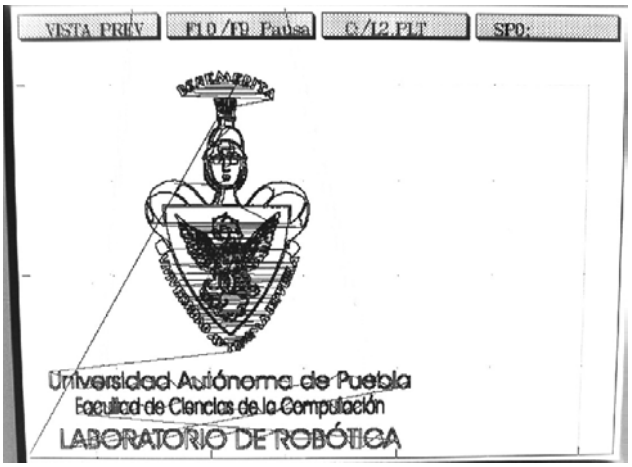


Fig 6.2.- Simulación del diseño.

6.2 Grabado

Para la prueba de grabado se utilizo una placa de aluminio de calibre 18, y una herramienta de corte en forma esférica de 1mm de diámetro.

El resultado de la prueba se muestra en la figura 6.3 , el grabado tiene las misma dimensiones que el dibujo original, el acabado es satisfactorio y depende del estado de la herramienta de corte. El tiempo de grabado fue de aproximadamente de 3 horas, este tiempo se debe principalmente al tipo de la herramienta usada que no acepta mayores velocidades y a los movimientos no optimizados de la herramienta.

Figura 6.3 Prueba de grabado.

3. CONCLUSION

El sistema trabajó satisfactoriamente, con él podemos grabar placas de aluminio y con la herramienta adecuada puede usarse para grabar en placas de diferentes tipos de material.

Una ventaja de haber construido y programado el sistema es que pude ser fácilmente modificado y adaptado para realizar otro tipo de tareas, por ejemplo, con algunos ajustes puede usarse como taladro automático, o para el grabado mecánico de circuitos impresos.

Una desventaja es que no tiene optimizado los movimientos lo que hace que se consuma mas tiempo de lo debido.

A futuro se pretende programar los siguientes módulos:

a.-Interface de usuario, para hacerla más amigable

b.-Optimización de movimientos de la herramienta, con el fin de disminuir el tiempo de grabado.

c.-Calculo del movimiento de la herramienta en metros lineales, con el fin de predecir su desgaste..

d.-Programar las funciones de taladro automático

REFERENCIAS

[1].- Lewis C. Eggebrecht *Interfacing To TheIBM PC* (Howard W Sams and Co. Inc.)

[2] - Herbert Schildt, *lenguaje c, programación avanzada*(Mc graw hill. 1988)

[4].- HP 7475A Graphics Plotter Interfacing and programing manual of Hewlett Packard