UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ACATLÁN

LIBRERÍAS GRÁFICAS HVAC EN AUTOLISP

T E S I S

QUE PARA OBTENER EL TÍTULO DE

LICENCIADO EN MATEMÁTICAS APLICADAS Y COMPUTACIÓN

P R E S E N T A

JESÚS ANTONIO SOSA HERRERA

ASESORA: M. EN C. MARIA DEL CARMEN VILLAR PATIÑO

ABRIL 2005

DEDICATORIA

DEDICADO A MIS PADRES:

Gladys Noemí Herrera Cruz
Jesús Antonio Sosa Martínez

POR LA ENORME PACIENCIA QUE ME HAN TENIDO TODOS ESTOS AÑOS.

A MI ESPOSA:

Elizabeth Ávila Reyes

MI MÁS PRECIADO TESORO
POR TODO LO QUE HEMOS RECORRIDO Y POR LO QUE NOS FALTA POR RECORRER ↑.

Y A MIS HERMANOS:

Ma. Guadalupe y Víctor Hugo

AGRADECIMIENTOS

A nuestra máxima casa de estudios
Universidad Nacional Autónoma de México

AGRADEZCO DE MANERA ESPECIAL EL APOYO DE :

M. en C. Ma. Del Carmen Villar Patiño.

POR ASESORARME EN LA ELABORACIÓN DE ESTE TRABAJO Y POR SUS VALIOSAS SUGERENCIAS.

ASI COMO A LOS PROFESORES:

Ing. Rubén Romero Ruíz.
Ing. Juan José Cortés Buenrostro.
Lic. Oscar Gabriel Caballero Martínez
Lic. Ernesto Baltazar Martínez

QUE SE SIRVIERON A REVISAR ESTE TRABAJO DE INVESTIGACIÓN.

AGRADEZCO DE MANERA ESPECIAL A:

Ing. Ricardo Ruíz García

POR LA INFORMACIÓN TÉCNICA Y POR FOMENTAR EL PROYECTO DE LAS LIBRERÍAS GRÁFICAS HVAC.

INDICE.

	Pág.
Índice de tablas	
Índice de figuras	
INTRODUCCIÓN	1
CAPITULO I DISEÑO ASISTIDO POR COMPUTADORA.	2
I.1 Concepción asistida por computadora. CAE/CAD/CAM.	2
I.2 El entorno de diseño asistido por computadora AutoCAD.	6
I.2.1 Comandos básicos de AutoCAD.	8
CAPITULO II EL LENGUAJE DE PROGRAMACIÓN LISP.	13
II.1 El lenguaje de programación LISP.	13
II.2 AutoLISP.	15
II.2.1 Necesidad de añadir programación sobre los paquetes de diseño asistido por computadora.	15
II.2.2 Opciones de programación para AutoCAD.	16
II.2.3 AutoLISP como lenguaje de programación.	18
CAPITULO III LIBRERIAS GRAFICAS HVAC.	25
III.1 Tareas de diseño HVAC que requieren automatización.	25
III.2 Descripción de las funciones HVAC.	27
III.3 Medidas de eficiencia de las librerias HVAC.	50
CONCLUSIONES	58
BIBLIOGRAFÍA	59

INDICE DE TABLAS

	Pág.
I.1.1 Software CAD comercial.	3
III.3.1 Operaciones sustituidas por la función *Ducto*	51
III.3.2 Operaciones sustituidas por la función *Codo*	52
III.3.3 Operaciones sustituidas por la función *YeCDCRect*	52
III.3.4 Operaciones sustituidas por la función *VerObjetos*	53
III.3.5 Operaciones sustituidas por la función *CambiarMedidas*	53
III.3.6 Operaciones sustituidas por la función *DetectarOrigen*	54
III.3.7 Operaciones sustituidas por la función *DibujarMedidas*	54
III.3.8 Operaciones sustituidas por la función *GenerarInforme*	55
III.3.9 Operaciones sustituidas por la función *DibujarNumeración*	55
III.3.10 Operaciones sustituidas por la función *GirarObjeto*	56
III.3.11 Operaciones sustituidas por la función *LlegarA*	56

INDICE DE FIGURAS

	Pág.
I.1.1 Ejemplo de dibujo en un sistema CAD bajo diferentes tipos de render.	5
I.2.1 Entorno de diseño AutoCAD.	7
I.2.2 Comandos básicos de dibujo.	9
I.2.3 Comandos de edición de objetos.	9
I.2.4 Comandos de acotación y consulta.	10
I.2.5 Comandos de ajuste y alineación.	10
I.2.6 Comandos de sombreado.	10
I.2.7 Comandos de inserción.	11
I.2.8 Comandos de formatos de presentación.	11
I.2.9 Comandos de control de coordenadas.	11
I.2.10 Comandos de control de vistas.	12
II.1.1 Definición de una función en LISP.	14
II.2.3.1 El entorno de programación Visual LISP.	19
III.2.1 Dibujo generado por la función *Ducto*	28
III.2.2 Dibujo generado por la función *DuctoRect*	28
III.2.3 Dibujo generado por la función *DuctoF*	28
III.2.4 Dibujo generado por la función *Codo*	29
III.2.5 Dibujo generado por la función *CodoF*	29
III.2.6 Dibujo generado por la función *CodoRectRed*	30
III.2.7 Dibujo generado por la función *CodoRectRE*	30
III.2.8 Dibujo generado por la función *CodoRect*	31
III.2.9 Dibujo generado por la función *CodoEscRect*	31
III.2.10 Dibujo generado por la función *Desviacion*	32
III.2.11 Dibujo generado por la función *DesviacionRect*	32
III.2.12 Dibujo generado por la función *TapaRed*	32
III.2.13 Dibujo generado por la función *TapaRect*	33
III.2.14 Dibujo generado por la función *Reduccion*	33
III.2.15 Dibujo generado por la función *ReduccionEX*	33
III.2.16 Dibujo generado por la función *ReduccionRect*	34
III.2.17 Dibujo generado por la función *ReduccionEXRect*	34
III.2.18 Dibujo generado por la función *ReduccionEXEXRect*	34
III.2.19 Dibujo generado por la función *CTS*	35
III.2.20 Dibujo generado por la función *CTR*	35
III.2.21 Dibujo generado por la función *CTSRect*	35
III.2.22 Dibujo generado por la función *CTRRectRed*	36
III.2.23 Dibujo generado por la función *CIE*	36
III.2.24 Dibujo generado por la función *CIER*	36
III.2.25 Dibujo generado por la función *TIE*	37

Pág

III.2.26 Dibujo generado por la función *TIER* 37
III.2.27 Dibujo generado por la función *TIR* 37
III.2.28 Dibujo generado por la función *TIRR* 38
III.2.29 Dibujo generado por la función *TCR* 38
III.2.30 Dibujo generado por la función *SALM* 38
III.2.31 Dibujo generado por la función *YIC* 39
III.2.32 Dibujo generado por la función *YICR* 39
III.2.33 Dibujo generado por la función *YeCCRect* 40
III.2.34 Dibujo generado por la función *YeCDRect* 40
III.2.35 Dibujo generado por la función *YeCDCRect* 41
III.2.36 Dibujo generado por la función *DifusorCuad* 41
III.2.37 Dibujo generado por la función *Difusor* 42
III.2.38 Dibujo generado por la función *RejillaRect* 42
III.2.39 Dibujo generado por la función *UMA* 42
III.2.40 Resultado de aplicar la función *LlegarA* 44
III.2.41 Resultado de aplicar la función *DibujarMedidas* 44
III.2.42 Resultado de aplicar la función *GenerarInforme* 45
III.2.43 Resultado de aplicar la función *DibujarNumeración* 45
III.2.44 Menú principal de las librerías HVAC 46
III.2.45 Barras de herramientas de las librerías HVAC 47
III.2.46 Cuadro de diálogo *VerObjetos* 48
III.2.47 Cuadro de diálogo *GenerarInforme* 48
III.2.48 Cuadro de diálogo *DibujarMedida* 48
III.2.49 Cuadro de diálogo *DibujarNumeración* 49
III.2.50 Cuadro de diálogo *DimensionesAdicionales* 49
III.3.1 Ejemplo de aplicación de las librerías HVAC 57

INTRODUCCIÓN

El presente trabajo consistió en la creación de una serie de funciones en AutoLisp agrupadas en librerías. El motivo principal que llevó a realizar éstas librerías surgió a partir de la necesidad de agregar rutinas de automatización a dibujos realizados en AutoCAD, pues a partir de la transición que se ha adoptado últimamente del paso de representación de dibujos técnicos en dos dimensiones a tres dimensiones, se requiere contar con herramientas adaptadas a cada necesidad de trabajo. Se observa que en algunos campos, el software comercial no presenta siempre las características apropiadas para cada situación específica, por lo cual es necesario aplicar técnicas de programación.

En este caso, se implementaron librerías gráficas que automatizan tareas de dibujo para el área de ingeniería conocida por las siglas HVAC (*Heating and Ventilating Air Conditioning*), de tal forma que los dibujos que corresponden a elementos comúnmente utilizados en esta área se realizan de manera automatizada, dejando al usuario solamente la tarea de proporcionar los parámetros necesarios para los dibujos, incluyendo la indicación de puntos de inserción del dibujo y sus medidas correspondientes. Se crean también funciones auxiliares para la obtención de datos complementarios al dibujo, como son lista de materiales, descripción e identificación de objetos gráficos y cambio de características; así como también funciones para la comunicación del programa con el usuario como son cuadros de diálogo, menús de opciones e introducción de datos por medio del teclado.

Se utilizó como base el entorno CAD comercial de AutoCAD ya que éste ha representado un estándar para la industria durante muchos años, además de contar con una interfase de programación abierta. De otra manera, se hubiera tenido que crear un motor gráfico adicional a las tareas que se realizaron, y hacerlo compatible con AutoCAD, de forma que pudiera ser utilizado en proyectos de ingeniería, lo cual consumiría bastante tiempo. Es por esto que se prefirió mejor extender las capacidades de un software ya existente y con gran aceptación.

Para poder ejecutar las tareas anteriormente mencionadas se recurrió a la programación con AutoLISP, aplicando técnicas de la geometría analítica tridimensional; y se requirió el conocimiento de los conceptos y métodos utilizados en el campo de la graficación por computadora.

También se realiza una evaluación de las librerías para determinar las mejoras que representan con respecto a las herramientas de dibujo originales de AutoCAD.

CAPITULO I. DISEÑO ASISTIDO POR COMPUTADORA

I.1 Concepción asistida por computadora. CAE/CAD/CAM.

Desde las primeras generaciones de computadoras, éstas empezaron a utilizarse en tareas de concepción de modelos matemáticos y en proyectos de ingeniería de manera auxiliar, poco a poco fue posible asignarles una gran variedad de tareas al respecto, como es el caso de los programas CAE, CAD y CAM, los cuales pueden consistir en rutinas de programación sencillas, hasta complicados sistemas que abarcan varios aspectos de un área tecnológica. Estos programas generalmente pueden interactuar o compartir datos con entre sí, o con bases de datos para aumentar más sus capacidades. Las tecnologías CAE, CAD, CAM , se describen brevemente a continuación.

Tecnología CAD.

Las siglas CAD (*Computer Aided Design*) Diseño Asistido por Computadora, se usan para denotar a aquellos programas de computadora que facilitan o realizan una tarea de diseño de cualquier especialidad tecnológica. Estos programas permiten obtener un modelo bastante detallado de un proyecto de momento intangible, el cual aún no está construido físicamente, pero que una vez definido por el programa, se puede someter a pruebas y criterios de evaluación antes de que empiece siquiera a construirse, también tienen aplicaciones a la realidad virtual, publicidad y entretenimiento. Al ser de interés en este trabajo, se incluye una lista (tabla I.1.1) de los programas CAD más utilizados actualmente, junto con los nombres de las compañías que los han creado.

Tecnología CAE.

La tecnología CAE (Computer Aided Engineering) consiste en software dedicado a las tareas de cálculo, simulación y análisis de procesos correspondientes a problemas de ingeniería o ciertos fenómenos que puedan representarse mediante un modelo matemático; de modo que el software CAE proporciona datos para determinar el comportamiento de un sistema para crear o mejorar su diseño. Generalmente este tipo de software utiliza una gran variedad de algoritmos numéricos y suele tener interfaces con programas CAD.

Tecnología CAM

Los sistemas CAM (Computer Aided Manufacturing) consisten en la interacción de software, hardware y diversos dispositivos mecánicos, de forma que la manufactura de un producto se realiza automáticamente a partir de su diseño en un sistema CAD. Con esta tecnología se logra que mediante tornos, fresadoras, máquinas dobladoras de hojas de metal, cortadoras, entre muchas otras,

conectadas a computadoras se manufacturen productos de la manera más automatizada posible.

Compañía	**Producto(s) CAD**
Autodesk	AUTOCAD, 3DS MAX, MECHANICAL DESKTOP, ARCHITECHTURAL DESKTOP, INVENTOR **[I-4]**
Silicon Graphics / Alias System Corp	ALIAS WAVEFRONT, ALIAS MAYA **[I-2]**
Dassault Systèmes	CATIA **[I-7]**
Robert McNeel & Associates	RHINOCEROS, ACCURENDER **[I-16]**
Advent	ART.LANTIS, SKETCHUP, ZOOM **[I-1]**
Graphisoft	ARCHICAD **[I-10]**
Formz	FORMZ 4.1 **[I-9]**
SoftImage	SOFTIMAGE XSI **[I-23]**
Avid Technology	AVID 3D **[I-5]**
VX Corporation	VX BASIC VX DESIGNER **[I-25]**
Capvidia	SOLIDWORKS **[I-8]**
Kubotek USA	KEYCREATOR **[I-13]**
ImsiSoft	TURBOCAD **[I-11]**
Caligari Corporation	TRUESPACE **[I-6]**
Nemetschek North America	VECTORWORKS **[I-17]**
Intellicad Technology Consortium	INTELLICAD **[I-12]**
Newtek	LIGHTWAVE3D, VT3 **[18]**
Opend Mind	HYPERCAD **[I-19]**
Parametric Technologies	PROENGINEERING WILDFIRE **[I-20]**
Unigraphics Solutions	SOLIDEDGE **[I-24]**
Arris	ARRISCAD **[I-3]**
Side Effects Software	HOUDINI **[I-22]**

Tabla I.1.1. Software CAD comercial.

Este trabajo de tesis se relaciona de manera estrecha con los programas CAD, pues se toma uno de ellos como plataforma con el objeto de extender sus prestaciones hacia un área específica. Las aplicaciones CAD utilizan para su funcionamiento técnicas de graficación por computadora con el fin de lograr una representación del objeto que se está diseñando en la pantalla de la computadora o como salida a una impresora. Dentro de la computadora, los objetos se representan verdaderamente en tres dimensiones, y de esta forma se manejan internamente vértices, aristas, curvas y la caras de un objeto, así como información correspondiente a materiales, texturas y fuentes de luz. Sin embargo, la pantalla de un monitor convencional sólo permite la presentación de dibujos en dos dimensiones, por lo que se tiene que hacer una proyección o mapeo de la figura tridimensional sobre un plano, que corresponde a la vista mostrada en un monitor. La presentación de un objeto no se limita a proyectar los vértices y aristas del mismo, sino que se aplican algoritmos especiales para determinar las características de la vista que se debe mostrar; a este proceso se le conoce como *render*, y según la cantidad y el tipo de cálculos involucrados, obtenemos varias categorías de *render*. Mencionaremos las siguientes:

- *Wireframe.*(Estructura de alambre). Con este tipo de *render* se muestran solamente los vértices, aristas y curvas que componen un objeto tridimensional (o bidimensional) sin aplicar otro algoritmo adicional. Es la representación más rápida de generar, por lo que es muy utilizada durante la fase de creación de un diseño. Con esta representación se pueden apreciar los elementos que conforman la estructura de un objeto gráfico (figura I.1.1a).

- *Hidden.* (Líneas ocultas). Esta representación es similar a la anterior con la diferencia de en este caso se ocultan, o bien se marcan de alguna forma los detalles de un objeto que serían cubiertos por otros desde la vista presentada en el monitor. Con esto se consigue apreciar un diseño sin la interferencia de los detalles de la estructura de los objetos que no se verían si se tratara de objetos opacos (figura I.1.1b).

- *Shaded.* (Sombreado). Para estas técnicas se agregan algoritmos que determinan qué vértices forman las caras de un objeto, y por lo tanto, tienen la propiedad de reflejar la luz con una cierta intensidad y a un determinado ángulo. Existen principalmente dos tipos de sombreado: *Flat* (plano) que considera solamente la reflexión de la luz por cada cara del objeto, y *Gouraud*, que proporciona un color interpolado entre las aristas un objeto, con el fin de suavizar el contorno de éste (figuras I.1.1c y I.1.1d).

- *Raytrace* .(Trazado de rayos). Es un tipo de *render* más elaborado, en el cual no sólo se calcula la reflexión de los rayos de luz provenientes de una fuente sobre los objetos, sino que se va calculando el recorrido de los rayos de luz al rebotar entre los diferentes objetos que se encuentran dentro de lo que se conoce como una escena. A cada objeto se le asigna un material específico, lo que determina sus propiedades de reflejar absorber o desviar la luz; además

puede asociarse una determinada textura al objeto, que estará repartida en sus caras visibles. Según la calidad del *render*, este proceso puede tardar desde unos cuantos segundos hasta varias horas, o si lo requiere una aplicación y si se cuenta con el tiempo y equipo necesario, se puede continuar con el proceso hasta varias semanas. Este tipo de *render* se utiliza para crear la presentación final de un diseño y permite generar dibujos con un cierto grado de realismo (figura I.1.1e).

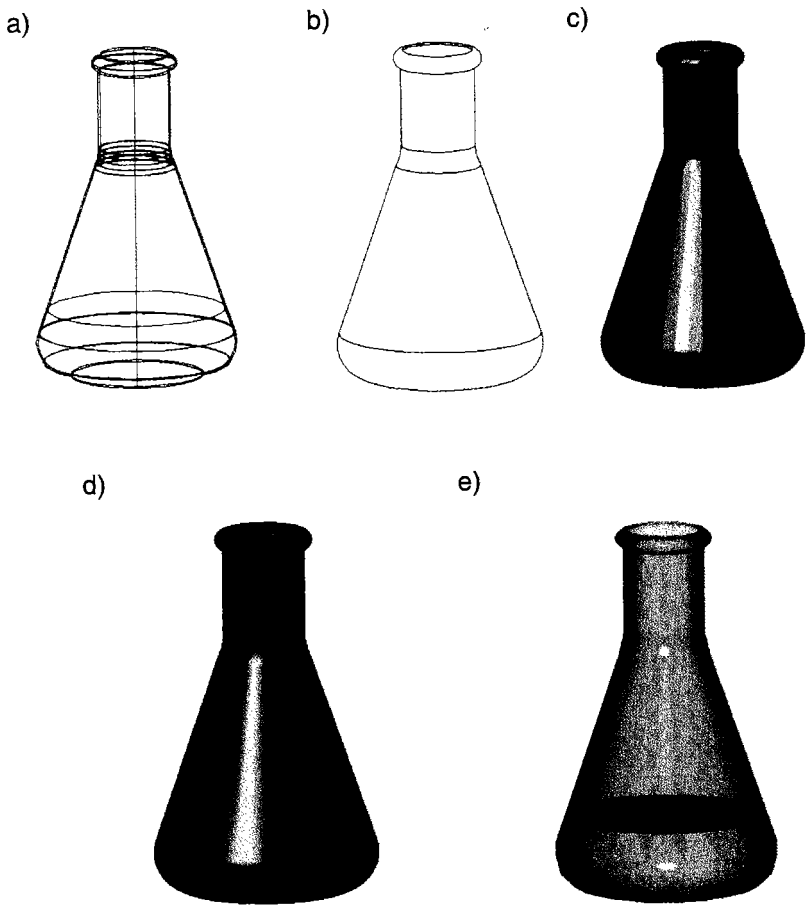


Figura I.1.1. Ejemplo de dibujo en un sistema CAD bajo diferentes tipos de *render*. a) *Wireframe*, b) *Hidden*, c) *Flat Shaded* d) *Gouraud Shaded*, e) *Raytrace*.

I.2 El entorno de diseño asistido por computadora AutoCAD.

Se dice que AutoCAD es el programa de Diseño Asistido por Computadora más difundido en el mundo **[B-8]**. Se trata de un programa de dibujo de precisión asistido por computadora, el cual cuenta con una interfaz de programación de "arquitectura abierta", razón por la cual se utilizó como base en esta investigación.

Las sucesivas versiones de este programa han ido apareciendo de la siguiente forma:

Versión 1.0 (*Release* 1): Noviembre de 1982.
Versión 1.2 (*Release* 2): Abril de 1983.
Versión 1.3 (*Release* 3): Septiembre 1983.
Versión 1.4 (*Release* 4): Noviembre 1983.
Versión 2.0 (*Release* 5): Octubre de 1984.
Versión 2.1 (*Release* 6): Mayo de 1985.
Versión 2.5 (*Release* 7): Junio de 1986.
Versión 2.6 (*Release* 8): Abril de 1987.
Release 9: Septiembre de 1987.
Release 10: Octubre de 1988.
Release 11: Octubre de 1990.
Release 12: Junio de 1992.
Release 13: Noviembre de 1994.
Release 14: Febrero de 1997.
AutoCAD 2000. Año 1999.
AutoCAD 2002. Año 2001.
AutoCAD 2004. Año 2003.
AutoCAD 2005. Abril 2004.

En estas versiones sucesivas, la forma básica de funcionamiento, ha cambiado muy poco, al demostrar que se maneja una manera efectiva de trabajo, aunque las prestaciones que ofrece cada versión se han ido incrementando considerablemente, de forma que las facilidades presentadas para tareas de diseño, organización y programación se encuentran en un estado técnicamente muy avanzado.

Con el fin de mostrar la forma en la que trabaja AutoCAD, se describirán las funciones más importantes que tiene este programa, se considera que esto ayudará a ejemplificar las prestaciones que se tienen, y el patrón general de trabajo, el cual se pretendió que fuera el mismo para el programa creado en este proyecto, de esta misma manera se mostrarán los comandos añadidos en el capítulo III de esta tesis, estos comandos adicionales, al estar orientados a una aplicación específica, no están incluidos en AutoCAD, cuyos comandos están dirigidos a tareas de propósito general.

En la ilustración de abajo (figura I.2.1) podemos observar sus componentes principales: La barra de menús, donde se puede escoger todos los comandos disponibles, la barra de herramientas, que consta de una serie de botones de acceso rápido a los comandos de mayor uso, esta barra es completamente personalizable, el área de dibujo llamada *"model"*, presenta un fondo negro para utilizarlo durante largas sesiones de dibujo, para apreciar el dibujo sobre un fondo blanco existen los espacios de dibujo *"layout"*, en los que se puede especificar las características del papel en el cual se va a imprimir físicamente . Autocad también nos presenta por omisión una línea de texto editable llamada línea de comandos, aquí se puede introducir cualquier comando de AutoCAD, también nos muestra información sobre el comando en curso. Un aspecto muy interesante de esta línea de comandos, es que también sirve como interfaz de un intérprete de AutoLISP, el dialecto de LISP que reconoce AutoCAD. De esta manera, se puede especificar cualquier instrucción de AutoLISP de la misma manera como se pide un comando normal de dibujo. AutoCAD también nos presenta en la parte inferior una barra de estado, donde muestra las coordenadas del punto actualmente seleccionado e informa y permite modificar el estado de las variables del sistema más comúnmente utilizadas en una sesión de dibujo.

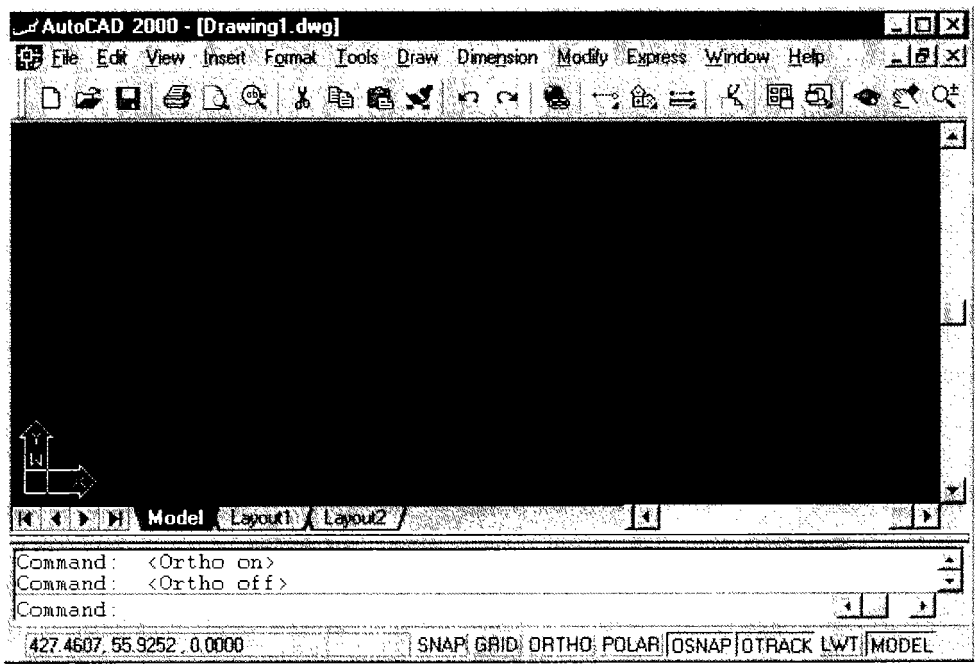


Figura I.2.1 Entorno de diseño AutoCAD.

A primera vista, AutoCAD nos presenta un entorno de dibujo muy simplificado, fácil de usar, y siguiendo los estándares de los entornos GUI (*Graphic User Interface*). Basta con hacer uso del sentido común para empezar a realizar figuras elementales con él. AutoCAD, permite realizar la mayoría de las tareas del tipo de dibujo conocido como dibujo técnico, dibujo de precisión, dibujo de ingeniería o dibujo industrial, según el área de aplicación. Se pueden realizar dibujos detallados en dos dimensiones, tres dimensiones, o en un modo de perspectiva falso **[B-2]**, que ayuda a representar objeto tridimensional en dos dimensiones, llamado modo isométrico.

I.2.1 Comandos básicos de AutoCAD.

A continuación se enumeran las principales funciones de dibujo suministradas por AutoCAD junto con las barras de herramientas correspondientes:

Comandos básicos de dibujo (*Draw, Solids, Surfaces*).

Las funciones básicas de dibujo con las que cuenta AutoCAD se pueden dividir en

a) Figuras de dos dimensiones como son líneas, rayos (líneas de longitud indefinida), multilíneas (líneas paralelas múltiples), polilíneas (líneas y arcos adyacentes agrupados como un mismo objeto de dibujo), polígonos, cajas (rectángulos), arcos, círculos, *splines* (curvas NURBS[1] cuadráticas o cúbicas), elipses, inserción de bloques (conjunto de objetos de dibujo identificados con un nombre), puntos, achurado[2], regiones (conjunto de objetos que delimitan un área) y cajas de texto (figura I.2.2 a). Aunque estas figuras aceptan coordenadas tridimensionales, siempre están contenidas en un solo plano.

b) Superficies en tres dimensiones para delimitación de áreas, caras, mallas *3Dmesh*, superficies de revolución, superficies determinadas por curvas y contornos y figuras de superficies predeterminadas (figura I.2.2 b).

c) Sólidos en tres dimensiones para creación de figuras sólidas de revolución, figuras sólidas por extrusión, rebanar un sólido (*slice*), sección de sólido, chequeo de interferencias, vistas y figuras de sólidos predeterminadas (figura I.2.2 c).

1. Non-Uniform Rational Bézier Spline. Representación matemática de geometrías que pueden crear exactamente cualquier curva en base a líneas bidimensionales con ciertos parámetros.
2. achurado (*hatch*) patrón de relleno de un área delimitada.

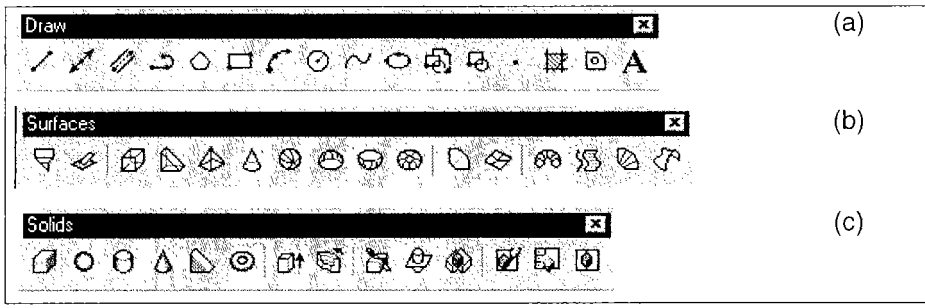


Figura I.2.2. Comandos básicos de dibujo.

Comandos de edición de objetos de dibujo (*Modify, Modify II, Solids editing*).

A través de estas funciones se permite realizar modificaciones sobre figuras que ya se encuentran dentro de un dibujo, entre ella tenemos borrar, copiar, crear imagen en espejo, contorno paralelo (*offset*), arreglos de objetos, mover, girar, escalar, estirar, longitud, recortar, extender, romper, chaflán, explotar (deshacer bloque), orden se superposición de objetos y edición de: achurado, polilínea, *spline*, multilínea, atributo de texto de bloque, caja de texto; unión de sólidos, substraer un sólido de otro, intersección de sólidos, separar partes de un sólido, envolver, limpiar definición de sólido, checar definición de sólido, imprimir sobre una cara, copiar vértices, colorear vértices; edición de caras del un sólido: extrudir, mover, borrar, girar, inclinar, copiar, colorear (figura I.2.3).

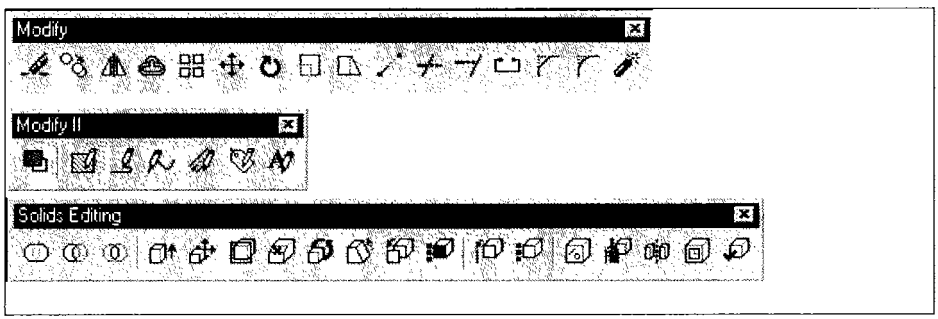


Figura I.2.3. Comandos de edición de objetos

Comandos de acotación de dimensiones y consulta (*Dimension, Inquiry*).

Esta clase de funciones sirve para conocer y mostrar en el dibujo las dimensiones y propiedades de los objetos dentro de un dibujo, los comandos mas utilizados son: acotación lineal, alineada, coordenada, acotación de radio, acotación de diámetro, acotación angular, automática, consecutiva, con respecto a una línea, indicación de flecha (*leader*), marcas de centro, edición de texto y tolerancias; consultar longitud, área, propiedades de masa, listado de propiedades, localizar punto (figura I.2.4).

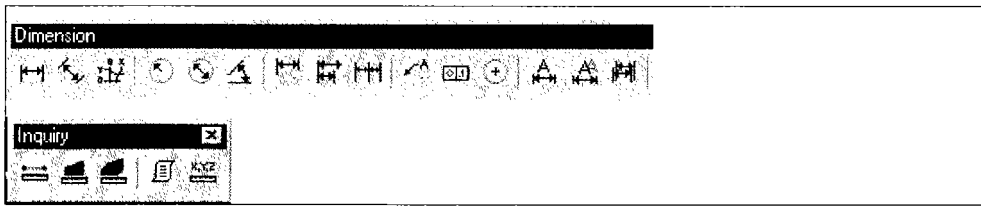


Figura I.2.4 Comandos de acotación y consulta

Comandos de ajuste y alineación (*Object snap*).

Este tipo de comandos proporcionan ayuda para definir puntos y orientar objetos dentro de un espacio tridimensional, con esto se pueden ubicar puntos con las funciones de rastreo temporal, alinear con: punto final, punto medio, intersección, intersección aparente, extensión, punto central, cuadrante, tangente, punto perpendicular, línea paralela, punto de inserción, nodo, punto más cercano (figura I.2.5).

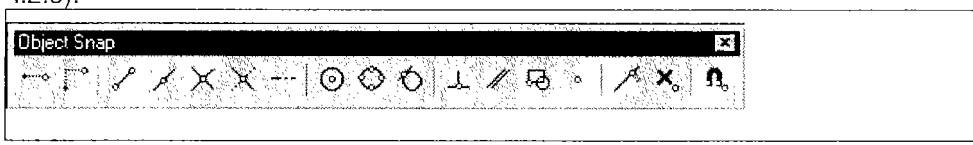


Figura I.2.5 Comandos de ajuste y alineación

Comandos de sombreado de superficies y sólidos (*Shade, Render*).

Mediante estas funciones de AutoCAD se pueden visualizar los objetos tridimensionales que definen superficies y sólidos de diferentes formas para obtener un cierto grado de realismo, o bien un desempeño eficiente en las operaciones de visualización. Los comandos más destacados son: Estructura de alambre 2D y 3D (2D, 3D *wireframe*), líneas ocultas (*Hidden*), sombreado plano (*Flat shaded*), sombreado "*Guoraud*" (*Guoraud shaded*), sombreado plano mostrando contornos, sombreado "*Guoraud*" mostrando contornos; ocultar, *render*, escenas, materiales, librería de materiales, mapear imagen sobre objeto, fondo, niebla, paisaje, edición de paisaje, librería de paisajes, opciones de *render*, estadísticas (figura I.2.6).

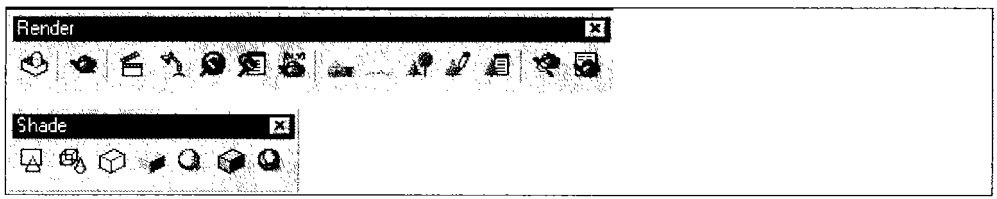


Figura I.2.6 Comandos de sombreado

Comandos de inserción de objetos(*Insert*).

Estos comandos permiten insertar objetos desde otras aplicaciones o bien desde otros dibujos mediante las funciones insertar bloque, insertar referencia externa, insertar imagen, importar objeto, importar objeto OLE (figura I.2.7).

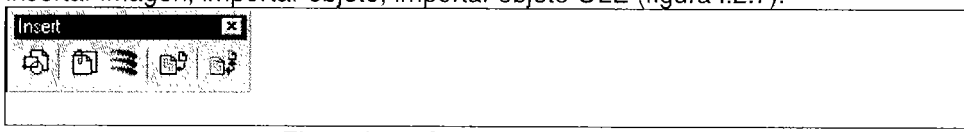


Figura I.2.7 Comandos de inserción

Comandos de formatos de presentación (*Layouts*).

Como se había mencionado, existen funciones de AutoCAD que permiten configurar un dibujo en una vista equivalente al formato en papel impreso, entre ellas tenemos a la creación de un nuevo formato, formato desde una plantilla, configuración de página, mostrar dialogo de creación de formatos (figura I.3.8).

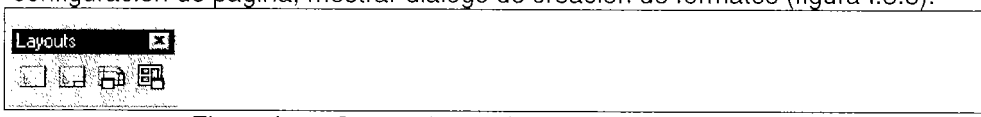


Figura I.2.8 Comandos de formatos de presentación

Comandos de control de coordenadas (*UCS, UCS II*).

Estos comandos permiten al usuario determinar diferentes sistemas de coordenadas dentro del espacio de dibujo. Para esto se cuenta con comandos como: mostrar ejes de coordenadas, restablecer sistema de coordenadas anterior, sistema de coordenadas universales, coordenadas con respecto a un objeto, coordenadas con respecto a una cara, coordenadas con respecto a la vista actual, definir origen, definir vector normal, definir coordenadas por tres puntos, girar el sistema de coordenadas con respecto a un eje (figura I.2.8).

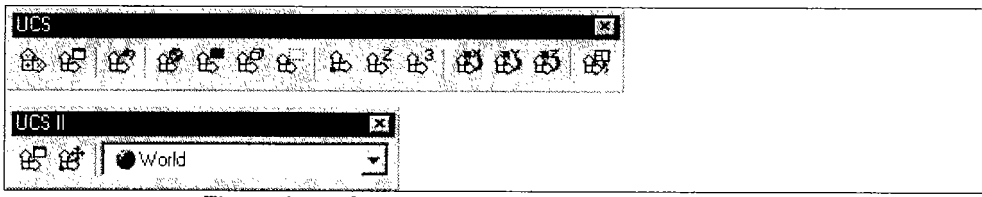


Figura I.2.9 Comandos de control de coordenadas

Comandos de control de vistas (*Views, 3D Orbit, Viewports, Zoom*).

AutoCAD también cuenta con una amplia gama de funciones de visualización, entre ellas tenemos funciones como aplicar vistas con nombre asignado; vistas ortográficas: superior, inferior, izquierda, derecha, frontal, trasera; vistas isométricas: noreste, noroeste, sureste, suroeste; posición de cámara; mover vista

del área de dibujo(*pan)*, girar vista, ajustar cámara, ajustar planos de delimitación de vista (*clip*); ajustar enfoque en primer plano (*zoom*), enfoque dinámico, factor de escala de enfoque, enfoque central, acercar, alejar, enfoque al máximo, enfoque a extensión de área de dibujo; ventanas de vista (*viewports*), ventana de vista poligonal, convertir objeto a ventana de vista, recortar ventana de vista (figura I.2.9).

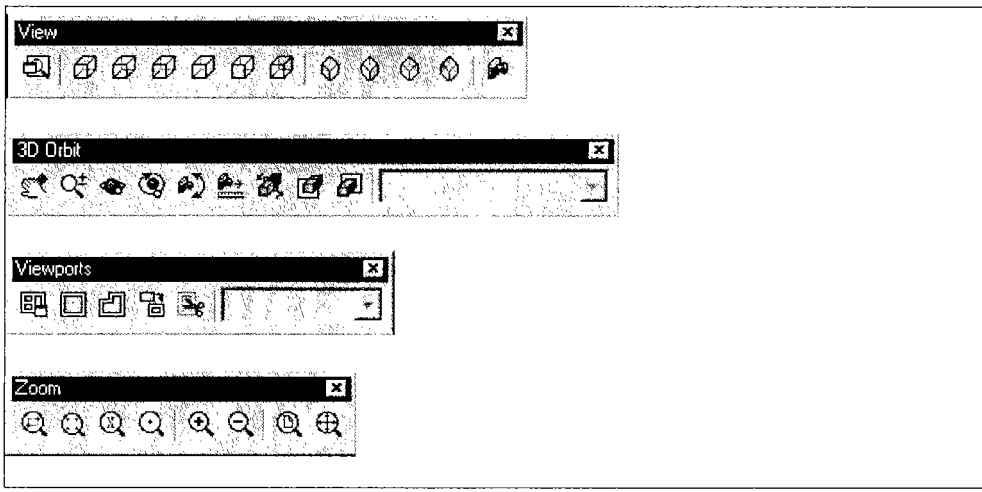


Figura I.2.10 Comandos de control de vistas.

Las descripciones detalladas de los comandos de AutoCAD se pueden encontrar en las referencias bibliográficas citadas en este trabajo **[B-2][B-3][B-8]**; existe además una gran cantidad de literatura con respecto a este entorno de diseño debido a su gran popularidad, incluso en la *Internet* se puede encontrar una gran cantidad de información al respecto, y por supuesto, la referencia más completa, aunque probablemente no sea la más didáctica, es la referencia incluida en el archivo electrónico de ayuda del mismo AutoCAD **[E-1]**.

CAPITULO II. AUTOLISP

II.1 El lenguaje de programación LISP.

La palabra LISP es una simplificación de la expresión de *LISt Processor*. El lenguaje de programación LISP sigue el paradigma de "programación funcional", es decir, no es como los "lenguajes declarativos" que dependen en su sintaxis de la arquitectura de la máquina sobre la cual están trabajando, y en los que cada comando tiene una equivalencia fácil de interpretar en el lenguaje de máquina. En un lenguaje orientado a la programación funcional, se sustituyen a las sentencias de control por recursividad, que es un concepto más apegado a las expresiones matemáticas, y el uso de las localidades de memoria de la máquina en forma de variables tiende a desaparecer, de modo que todo se expresa a manera de funciones.

Los origenes de LISP se remontan a 1954, cuando John McCarthy crea el compilador FLPL (*FORTRAN-compiled List Processing Language*), diseñado para la computadora IBM 704. Posteriormente mejoró este compilador para que contara con recursividad. La primera versión que apareció en el mercado fue LISP 1.5, diseñada para manejar listas algebraicas para implementar rutinas de inteligencia artificial.

Desde la invención de LISP se han estado desarrollando varios dialectos **[B-5]**, como son: Lisp 1.5, MacLISP , FranzLisp, InterLisp, NIL, ZetaLisp, Lispvm, IqLisp, MuLisp, Xlisp, Scheme, AutoLISP. Existe también un conjunto de dialectos con un núcleo en común, pero con ciertas extensiones orientadas hacia diferentes tareas, se dice que estos dialectos pertenecen al grupo CommonLisp. A partir de este conjunto de dialectos se crea un estándar para el lenguaje: ANSI (X3.226 1994) **[I-14]** e incluso, se implementan características de la programación orientada a objetos conocidas en conjunto como CLOS (*CommonLisp Object System*).

Algunas de las características más notables de la sintaxis de LISP se describen a continuación:

- Se puede identificar dos tipos de expresiones básicas, átomos y listas.

- Los átomos pueden ser, números enteros: 0, 1, 2, 3, etc.; números reales 7.6, 9.5; cadenas de caracteres: "algo", "algo más".

- Las listas son estructuras de datos que consisten de agrupaciones de átomos o de listas, su sintaxis se representa encerrando entre paréntesis a los elementos de la lista separados por espacios en blanco: (u v w x), ((u v) w x), (1 2). Una lista se puede construir por ejemplo con la instrucción *list*: (*list* 1 3 4 6) da como resultado la lista (1 3 4 6); un caso especial de lista es la lista vacía: nil.

- La estructura de un programa en LISP se basa en la evaluación de funciones, la misma ejecución de un programa completo es el resultado de la evaluación de una función. El intérprete de LISP siempre tratará de aplicar la regla de evaluación de funciones. La cual consiste en evaluar una función (programación funcional) de la obedeciendo a la sintaxis mostrada a continuación (figura II.1.1).

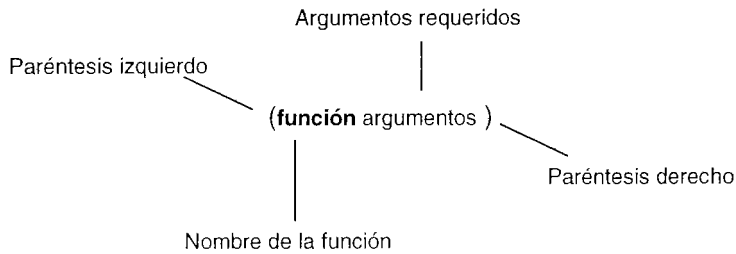


Figura II.1.1 Definición de una función en LISP.

Donde *función* es una función definida previamente definida, *argumentos* son los argumentos que se le pasarán a la función, estos pueden ser átomos o listas separados por espacios en blanco, los paréntesis forman parte de la sintaxis.

- Para definir una función se usa la instrucción *defun*, cuya sintaxis se puede apreciar en el siguiente ejemplo: (*defun* cuadrado (x) (* x x)), esta función calcula el cuadrado de su argumento si se llama mediante la regla de evaluación principal, por ejemplo: (cuadrado 3) da como resultado 9, un átomo.

- Todas las instrucciones en LISP son procesadas de la forma anteriormente descrita, incluyendo las operaciones aritméticas y lógicas, por lo que, si se quiere realizar la operación 5 + 4, esta deberá estar en notación prefija: (+ 5 4).

- Cuando se quiere especificar al intérprete que detenga la aplicación de la regla de evaluación principal, para especificar un elemento atómico o por alguna otra circunstancia, se debe usar la instrucción *quote*, con lo que tendríamos que escribir (*quote* abc) para indicar al intérprete que no pretendemos que el nombre abc sea evaluado como una función, una forma abreviada de *quote* se puede indicar con el símbolo: ', por ejemplo 'abc.

- Con el fin de implementar la recursividad, para variar una lista de parámetros pasados a una función definida recursivamente, se tienen las funciones *car* que devuelve el primer elemento de una lista, y *cdr* que

devuelve una lista sin el primer elemento, por ejemplo, (*car* '(1 2 3 4)) devuelve el átomo 1, (*cdr* '(1 2 3 4)) devuelve la lista (2 3 4).

Existe una gran cantidad de otras instrucciones de LISP así como formas de combinarlas, aquí se mostraron unas pocas con el fin de hacer notar algunas de las diferencias que hay con respecto a los lenguajes declarativos. Una referencia detallada de la sintaxis y comandos más usuales en los diferentes dialectos de LISP, se puede encontrar en la referencia bibliográfica **[B-5][I-14][I-15].**

En la actualidad los diferentes dialectos de LISP son ampliamente utilizados en el campo de la inteligencia artificial, en la investigación matemática y como interfase a sistemas CAD, sin embargo no es un lenguaje muy popular en otro tipo de aplicaciones si lo comparamos con otros lenguajes tradicionales, como son C, FORTRAN, BASIC, PASCAL, todos estos son lenguajes declarativos. Es de notar el hecho de que las implementaciones en LISP presentan una relativa lentitud para efectuar cálculos numéricos, su sintaxis se basa por completo en paréntesis, y que junto con su característica notación prefija, confunde a muchos programadores principiantes. Considerando sus ventajas y desventajas, para realizar el presente trabajo, se encontró que el dialecto de LISP, AutoLISP fue la mejor opción como lenguaje de programación para la tarea que se realizó, debido a las características descritas en el siguiente apartado.

II.2 AutoLISP.

II.2.1 Necesidad de añadir programación sobre los paquetes de diseño asistido por computadora.

Como es de esperarse, las herramientas CAD deben de cumplir con una amplia gama de funciones, lo que las coloca en cierto grado, como un tipo de herramientas de propósito general, de forma tal, que casi siempre se necesita la ayuda de la programación para realizar o agilizar tareas muy especializadas.

En la actualidad, para resolver estos problemas, existen diversos programas que se "incrustan" dentro de los paquetes de software comerciales de diseño, tales programas posibilitan y agilizan diversas tareas especificas para varias ramas de ingeniería o investigación; sin embargo, nos encontramos con nuevos problemas relativos a dichos programas, entre los que cabe destacar los siguientes:

- La mayoría de estos programas son elaborados en el extranjero, por lo que los comandos y manuales de estos programas se encuentran en algún idioma diferente al español (inglés, en la mayoría de los casos).

- Las especificaciones técnicas a las que responden dichos programas son generalmente aplicables en otros países (EE.UU. Inglaterra, Francia), que aunque a veces siguen ciertos estándares o métodos de trabajo, muchos de

estos estándares no se utilizan en México, y los métodos de trabajo pueden variar de una empresa a otra.

- Por tratarse de software de aplicación especializada, este no se distribuye masivamente, por lo que su precio en el mercado tiene que ser relativamente más elevado, comparado con un software de mayor comercialización.

- El proveedor del software sólo entrega cierto número de licencias para utilizar una copia del software compilado, y el precio aumenta según el número de licencias, aunque se trate del mismo sistema física y lógicamente.

- El cliente, al no ser poseedor del código fuente, no tiene oportunidad de realizar correcciones, actualizaciones, o modificaciones del software, además de que muchas veces incluso, de acuerdo a su contrato con el proveedor, el cliente esta imposibilitado legalmente para realizar tales tareas, por lo que tiene que esperar la respuesta (si es que hay alguna) y atención del proveedor cuando se le presentan problemas concernientes a estas actividades.

II.2.2 Opciones de programación para AutoCAD.

AutoCAD, desde sus primeras versiones ha optado por una arquitectura de programación abierta, que permite a terceras personas incrementar las capacidades del entorno CAD mediante el uso de varios lenguajes de programación. Inclusive, muchos comandos internos de AutoCAD son aplicaciones en ARX ó AutoLISP. A continuación tenemos las opciones de programación que acepta AutoCAD junto con sus características más representativas.

Programación mediante C y C++.

Para este tipo de programación, Autodesk proporciona una serie APIs (*Application Programming Interface*) y clases que permiten comunicar un programa en C ó C++ con AutoCAD. Este método resulta ser el más rápido y con más posibilidades de programación que cualquier otro, pues se trata de una interfase de bajo nivel. Este conjunto de prestaciones es conocido como ARX (*AutoCAD Runtime Extension*), el sistema de interfase para versiones anteriores se llamaba ADS (*AutoCAD Develpment System*). Para implementar programas con ARX se necesita un compilador externo para lenguaje C y C++; actualmente se requiere de Microsoft Visual C++ 6.0 para tener acceso a las características de programación de las últimas versiones de AutoCAD. Por tratarse de rutinas de bajo nivel, queda como responsabilidad del programador, el manejo de errores y excepciones, así como de revisar la consistencia de datos u objetos y funciones o métodos; esto hace que el desarrollo de una aplicación con estas técnicas sea más lento que con otras opciones, aunque la velocidad de ejecución y prestaciones aumentan.

Programación mediante AutoLISP.

AutoLISP es la opción de programación compatible con más versiones de AutoCAD. El intérprete está incluido dentro de la línea de comandos de dibujo de forma que los usuarios más experimentados se encuentran familiarizados con el uso de rutinas de AutoLISP para auxiliarse en la creación de sus dibujos. Este dialecto de LISP cuenta con una gran cantidad de extensiones que permiten manejar entidades gráficas y no gráficas dentro de AutoCAD y tener acceso a bases de datos. Todo lo necesario para la creación de programas en AutoLISP como editor, intérprete, compilador, gestor de proyectos y herramientas de depuración viene incluido dentro del mismo AutoCAD, por lo que no se necesita de otro tipo de software externo para crear aplicaciones de este tipo. Esta opción de programación es muy confiable debido a que ha tenido ya varios años de depuración y desarrollo. La documentación completa acerca del lenguaje AutoLISP se encuentra incluida dentro AutoCAD **[E-1]**.

Programación mediante VBA y ActiveX.

AutoCAD implementa una versión de VBA (*Visual Basic for Aplications)*, esta versión es similar a la que se emplea en paquetes de software como Microsoft Word, Excel, Access y presenta las prestaciones básicas del lenguaje de programación Microsoft Visual Basic. Para lograr una interfase con AutoCAD se cuenta con un conjunto de clases denominadas ActiveX, éstas pueden ser accesadas por cualquier programa compatible con esta tecnología, incluso desde AutoLISP, sin embargo, por la forma de manejar las clases se obtiene un código más sencillo desde VBA que desde AutoLISP. Esta tecnología se presenta desde la versión R14 de AutoCAD y muchos objetos comunes de Visual Basic aún no son del todo compatibles para AutoCAD.

Lenguaje DCL.

El lenguaje DCL (*Dialog Control Languaje*) es proporcionado por Autodesk para la creación de cajas de dialogo. Estas se definen en archivos con formato ASCII y su comportamiento puede ser controlado mediante rutinas de AutoLISP. Con esta característica es posible presentar cuadros de dialogo, ventanas de información y objetos comunes en pantalla como botones, texto editable, barras de desplazamiento, listas de selección, entre otros, desde AutoLISP.

Lenguaje DIESEL.

El lenguaje DIESEL (*Direct Interpretively Evaluated String Expression Language*) es otra funcionalidad proporcionada para ejecutar secuencias de comandos (conocidas como macros) de manera automática, en este tipo de lenguaje todas las expresiones son cadenas de texto, así como sus resultados**[B-8]**; éstas secuencias se guardan en archivos de texto y pueden ser llamadas desde la línea de comandos o a través de instrucciones de AutoLISP, esta opción de

automatización tiene muchas limitaciones comparándola con otras opciones, como por ejemplo, sólo cuenta con un pequeño conjunto de instrucciones y estas están limitadas a sólo 10 parámetros como máximo de entrada. Esta opción fue creada para obtener una forma rápida y sencilla de automatización por ello es que no tiene muchas prestaciones.

Como se puede notar, AutoCAD cuenta con varias opciones de programación. En este caso se escogió AutoLISP como lenguaje de desarrollo debido principalmente a las siguientes características:

- LISP, como su nombre lo indica, es un lenguaje especializado en el manejo de listas complejas de manera dinámica. Si consideramos a un dibujo como una lista de puntos, líneas, arcos, círculos, etc, y además tomamos en cuenta que las coordenadas cartesianas pueden tomarse como listas de tres elementos, vemos que LISP es un buen lenguaje para manipular dibujos.

- La implementación de programas se realiza de una manera muy rápida, pues se cuenta con una gran cantidad de extensiones del lenguaje creadas especialmente para la creación de gráficos en AutoCAD.

- Por ser una de las primeras interfaces creadas, está bastante depurada y resulta ser una plataforma altamente confiable.

- La sintaxis de LISP se enseña en la carrera de Matemáticas Aplicadas y Computación, en primer semestre en forma del dialecto de LISP llamado SCHEME, el cual se considera el segundo dialecto de LISP más difundido en la actualidad **[I-21]** y se estudia LISP más detalladamente en séptimo semestre en la materia de Programación Lógica y Funcional; por lo que a los egresados de esta carrera nos resulta un lenguaje muy práctico.

II.2.3 AutoLISP como lenguaje de programación.

Como se ha mencionado, AutoLISP es un dialecto de LISP que viene incorporado con AutoCAD a partir de la versión 2.1. **[B-3]**. Este lenguaje incluye varias extensiones que permiten interactuar con AutoCAD e incorporar programas sofisticados para facilitar el trabajo.

Las instrucciones de AutoLISP pueden introducirse directamente en la línea de comandos del dibujo, es decir, el intérprete de AutoLISP está activo todo el tiempo durante una sesión de dibujo, por lo que las instrucciones sencillas pueden ejecutarse de manera inmediata. Si se quiere cargar un programa que conste de varias líneas, se puede recurrir a las herramientas de AutoCAD, y solicitar que cargue un archivo de texto con extensión .LSP, el cual contenga el programa que se quiera cargar en el intérprete.

Con la finalidad de facilitar la programación en AutoLISP, también se incluye en las versiones más recientes de AutoCAD, un entorno de programación llamado "*Visual LISP for AutoCAD*" (figura II.2.3.1), el cual incluye ciertas herramientas de edición como corrector de sintaxis, identificación de instrucciones por color, tabulación automática, compilación de proyectos por módulo y utilidades de depuración. Un aspecto muy importante de este entorno, es que permite compilar instrucciones de AutoLISP de forma tal que se tenga un archivo con extensiones .VLX o .FAS **[E-1]**, que puede ser reconocido por AutoCAD y cuya ejecución es más rápida que si se introdujera mediante el intérprete en la línea de comandos.

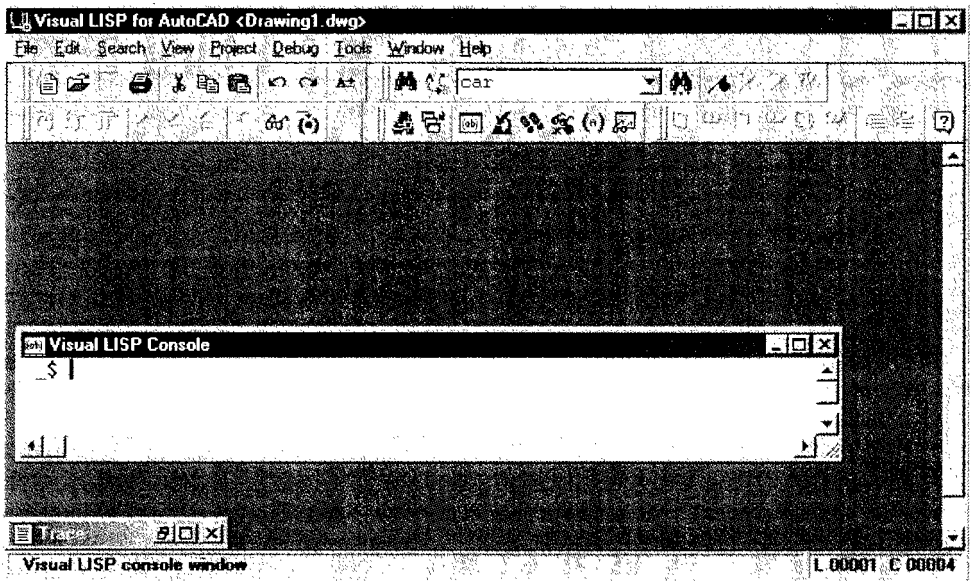


Figura II.2.3.1 El entorno de programación Visual LISP.

Debido a la gran importancia que tienen las extensiones de AutoLISP en este proyecto, se mencionarán las que más se ocuparon y se dirá la forma en la que fueron empleadas. En las descripciones de las sintaxis mostradas se muestran entre corchetes [ ] los parámetros opcionales de las funciones. Las extensiones citadas a continuación no forman parte del estándar CommonLISP **[I-15]**, pero son indispensables para interactuar con AutoCAD.

Extensión *command.*
Ejecuta un comando de AutoCAD.
Sintaxis: *(command [argumentos] ...)*
Donde *argumentos* corresponde al comando de AutoCAD y sus respectivas opciones. Con esta instrucción se pueden realizar los comandos de dibujo (líneas, superficies, sólidos, etc.) a través de AutoLISP, como si el usuario los hubiera escrito manualmente desde la línea de comandos de tal forma que se proporciona un método para realizar casi todas las instrucciones gráficas disponibles en AutoCAD.

Extensión *entmake.*
Esta extensión proporciona otra manera de realizar instrucciones de dibujo mediante AutoLISP, a diferencia de la extensión *command*, esta no es evaluada por la línea de comandos, sino que envía datos directamente a la base de datos interna de AutoCAD, por lo que tiene una ejecución mucho más rápida. A cambio de la velocidad de ejecución ganada con esta instrucción, tenemos que a cargo del programador quedan los cálculos de rotación y traslación, además de que los parámetros pasados a esta función deben estar de acuerdo a las claves especificadas por el formato DXF **[E-1]**, que maneja internamente AutoCAD.
Sintaxis: (*entmake* [*lista*])
Donde *lista* es una lista de asociación que contiene las claves DXF de la entidad de dibujo que se quiere crear con el formato de pares punto: (clave_dxf . valor).

En el programa desarrollado para esta tesis, se utilizaron ambas formas de creación de entidades gráficas, para la mayoría de los dibujos automatizados se utilizó la extensión *entmake*, de forma que la mayor parte de las operaciones realizadas por el programa se ejecutan a la mayor velocidad posible. La extensión *command* se utilizó en las rutinas de programación que implicaban cierto grado de complejidad, o que no tenían un equivalente directo mediante otras instrucciones.

Otras extensiones auxiliares suministradas para el uso eficaz de *entmake* son:

Extensión *entdel.*
Para borrar objetos gráficos o restaurar objetos previamente borrados.
Sintaxis: (*entdel nombre*)
Donde *nombre* es el nombre de entidad gráfica asociado al objeto en la base de datos interna de AutoCAD.

Extensión *entget.*
Recupera los datos de definición de un objeto en forma de una lista de asociación.
Sintaxis: (*entget nombre* [*aplicación*])
Donde *nombre* es el nombre de la entidad gráfica y el parámetro opcional *aplicación* indica que se requieren también los datos extendidos agregados por determinada aplicación.

Extensión *entmod.*
Modifica los datos de definición de un objeto gráfico.
Sintaxis: (*entmod lista*)
Donde *lista* es una lista de asociación con los nuevos datos de la entidad gráfica.

Extensión *entnext.*
Regresa el nombre de entidad del siguiente objeto en la base de datos de dibujo.
Sintaxis: (*entnext* [*nombre*])
Donde el parámetro opcional *nombre* es el nombre del objeto a partir del cual se quiere la siguiente entidad.

Extensión *entlast.*
Devuelve el nombre de la última entidad no borrada que se añadió a la base de datos del dibujo.
Sintaxis: (*entlast*)

Se cuenta también con una serie de extensiones que se sirven para que un programa pueda interactuar con el usuario, siguiendo el mismo formato que sigue AutoCAD en sus comandos originales, es decir, permiten la entrada de datos a través del ratón en el área de dibujo o por medio de la línea de comandos. Las extensiones de este tipo usadas en el presente trabajo son:

Extensión *getpoint.*
Permite al usuario especificar un punto con el ratón o escribiendo sus coordenadas desde el teclado.
Sintaxis: (*getpoint* [*pt*] [*msg*])
Donde *pt* es un punto de referencia con respecto al cual se va a pedir el siguiente punto, *msg* es la cadena de caracteres que se mostrará al usuario al momento de pedir un punto.

Extensión *getint.*
Pregunta al usuario por un número entero y devuelve ese entero.
Sintaxis: (getint [*msg*])
Donde *msg* es una cadena de caracteres opcional, en la que se le puede dar información al usuario acerca del valor requerido.

Extensión *getreal.*
Pregunta al usuario por un número real y devuelve ese número.
Sintaxis: (*getreal* [*msg*])

Extensión *getkword.*
Pregunta al usuario por una palabra clave, usada para pedir que el usuario escriba una opción correspondiente a un determinado comando.
Sintaxis: (*getkword* [*msg*])

Extensión *initget.*
Establece valores previos para ser usados por la siguiente llamada función que requiera entrada por parte del usuario.
Sintaxis: (*initget* [*bits*] [*cadena*])
Donde *bits* son los bits que definen una opción determinada, *cadena* corresponde a las cadenas de caracteres de las palabras clave que se preguntarán al usuario.

Se utilizaron otras extensiones del lenguaje que sirven para interactuar con el entorno de diseño, manejar las variables del sistema, y manipular cuadros de diálogo. Las extensiones de este tipo a las que se recurrió son:

Extensión *getvar.*
Obtiene el nombre de una variable de sistema de AutoCAD.
Sintaxis: (*getvar variable*)
Donde variable es una cadena de caracteres que corresponde a una opción dentro del entorno de dibujo.

Extensión *setvar.*
Establece el nombre de una variable de sistema de AutoCAD que no sea de sólo lectura.
Sintaxis: (*setvar variable*)

Extensión *alert.*
Muestra una caja de diálogo conteniendo un mensaje de advertencia.
Sintaxis: (*alert mensaje*)

Extensión *action_tile.*
Asigna una acción a evaluar cuando el usuario selecciona el elemento especificado de una caja de diálogo definida en lenguaje DCL.
Sintaxis: (*action_tile clave expresión*)
Donde *clave* es el nombre del elemento que dispara la acción y *expresión* es la función que se evalúa cuando sucede el evento.

Extensión *done_dialog.*
Termina una caja de dialogo.
Sintaxis: (*done_dialog* [*estado*])
Donde *estado* es un entero con el significado de aceptar =1 y cancelar = 0.

Extensión *get_tile.*
Obtiene el valor en tiempo de ejecución de un elemento de una caja de diálogo.
Sintaxis: (*get_tile clave*)

Extensión *load_dialog.*
Carga en memoria una caja de diálogo definida en un archivo con formato DCL.
Sintaxis: (*load_dialog* archivo)

Extensión *mode_tile.*
Establece el modo en el que un elemento de un cuadro de diálogo es mostrado.
(*mode_tile clave modo*)
Donde *clave* es el identificador del elemento del cuadro de diálogo actualmente cargado y *modo* es un entero que determina si el elemento está habilitado, deshabilitado, seleccionado o resaltado.

Extensión *start_dialog.*
Muestra una caja de diálogo y comienza a aceptar entrada de datos por parte del usuario.
Sintaxis: (start_dialog)

Extensión *term_dialog.*
Cierra todas las cajas de diálogo mostradas actualmente, como si el usuario hubiera cancelado cada una de ellas.
Sintaxis: (*term_dialog*)

Extensión *ssget.*
Crea un conjunto que consta de los objetos seleccionados por el usuario o por el programa.
Sintaxis: (*ssget* [*modo_de_selección*] [*punto1* [*punto2*]] [*lista_puntos*] [*lista_filtro*])
Donde *método_de_selección* indica la forma en que se escogen los objetos, ya sea por el programa o por el usuario, *lista_de_filtro* indica que tipo de objetos se seleccionarán, *punto1*, *punto2* y *lista_de_puntos*, son puntos relativos a la selección que especifican los puntos que el usuario marcó con el ratón para seleccionar a los objetos.

También se cuenta en AutoLISP con instrucciones que permiten que AutoCAD cargue automáticamente aplicaciones y que reconozca los datos generados por éstas, las instrucciones utilizadas fueron:

Extensión *regapp.*
Registra el nombre de una aplicación dentro del dibujo actual de forma que AutoCAD esté preparado para usar datos extendidos en los objetos.
Sintaxis: (*regapp* aplicación)
Donde aplicación es el nombre de la aplicación que usará datos extendidos sobre los objetos gráficos.

Extensión *load.*
Evalúa una expresión de AutoLISP contenida en un archivo.
Sintaxis: (*load nombre_de_archivo* [*en_falla*])
Donde *nombre_de_archivo* incluye la ruta y nombre del archivo que contiene la expresión, si no se incluye la extensión, se intentarán buscar las extensiones .vlx, .fas y .lsp; si no se incluye el directorio, se buscará en el directorio de librerías de AutoCAD. El parámetro *en_falla* es un átomo que corresponde al valor devuelto por la función en caso de error.

Extensión *tblsearch.*
Busca un nombre dentro de una tabla de símbolos.
Sintaxis: (*tblsearch nombre_de_tabla* símbolo [*ajustar_siguiente*])
Donde *nombre_de_tabla* es el nombre de la tabla de símbolos donde se buscará, símbolo es el símbolo a buscar y *ajustar_siguiente* indica que se ajusta un contador interno en la tabla de símbolos para que en la siguiente llamada a la función se empiece a buscar desde ahí.
Dentro de este trabajo, esta función se utilizó para buscar definiciones de objetos o partes de objetos gráficos que ya hayan sido dibujados, de forma que si ya se tiene cierta entidad gráfica, se pueda reutilizar, simplemente insertando una referencia a ésta. Lo anterior da como resultado una mejora en la velocidad de ejecución y un requerimiento menor de espacio en disco.

Las extensiones de lenguaje presentadas, fueron en resumen, las instrucciones utilizadas, junto con las instrucciones comunes del lenguaje LISP, en la creación del programa cuyo desarrollo se mostrará en un capítulo posterior. Una referencia completa sobre dichas instrucciones puede encontrarse en la bibliografía **[E-1]**.

CAPITULO III. LIBRERIAS GRAFICAS HVAC

III.1 Tareas de diseño HVAC que requieren automatización.

El área de atención de este trabajo es la del dibujo técnico relacionado con la ingeniería de aire acondicionado, mejor identificada por las siglas HVAC (*Heating and Ventilating Air Conditioning*) de forma que usaremos estas siglas para identificar a las librerías gráficas implementadas.

Para plantear cuales son las tareas que se pretenden mejorar con la introducción de la librerías analizaremos las actividades que se realizan con programas CAD o con otro tipo de software, como sería una hoja de cálculo.

Tenemos entonces que se plantea un diseño de aclimatación de aire dentro de un recinto determinado, en el cual se determinan, entre otros, los ductos a través de los cuales se transportará el aire de un lugar a otro y las dimensiones de éstos, el flujo y velocidad del aire que pasará por ellos, la temperatura y presión del aire, y la forma en que se difundirá éste sobre un lugar específico, así como los niveles de ruido; además de que se calculan los mismos datos para el retorno o la extracción del aire; a esto se añade también el diseño o la especificación del sistema de máquinas e instalaciones que servirán para impulsar, calentar, enfriar, extraer y filtrar el aire. A esto se agrega el hecho de que se debe determinar los materiales y detalles necesarios para su construcción física **[B-1]**.

Una vez especificado un diseño, se procede a realizar su representación gráfica sobre un plano, con lo cual se conocerá su ubicación, las piezas utilizadas, las medidas y el número en que se usan; este plano, generalmente estará de acuerdo con otros planos del tipo arquitectónico, mecánico, hidráulico, eléctrico, y algunos más según el tipo de instalación que se esté diseñando. Basándose en este plano se procederá a la construcción física del diseño mencionado.

Normalmente la representación gráfica consiste en un dibujo bidimensional del diseño proyectado sobre una vista superior, conocida como vista en planta. Al tratarse de una proyección a una escala fija, muchos detalles quedan ocultos, por lo que se elaboran planos adicionales conocidos como planos de detalles en los que se representan las características que no se pueden apreciar en la vista en planta. Si es necesario tener una representación de otra perspectiva del diseño, se crean dibujos de diferentes vistas del diseño conocidas como cortes o secciones, según se trate de una representación total o parcial de la vista citada.

Para contar con un registro de las medidas exactas de los elementos y su disposición, se elaboran planos adicionales en perspectiva isométrica, estos planos no están detallados en cuanto al aspecto físico de una instalación, su propósito es suministrar los datos en cuanto a material utilizado y dimensiones de los componentes de un diseño.

En fechas recientes, debido al gran desarrollo tecnológico que se ha visto en los últimos años, se ha optado por realizar estas representaciones gráficas mediante un modelo tridimensional, con esto se puede apreciar, desde un mismo modelo, cualquier detalle de un diseño desde las perspectivas que se quiera, sin necesidad de crear un nuevo dibujo, por lo que las vistas isométricas, cortes y detalles se obtiene de manera inmediata sin tener que realizar nuevas representaciones. Con esto se consigue, que al ocurrir un cambio en el diseño, como constantemente pasa dentro de un proyecto de ingeniería, sólo se tiene que modificar el modelo tridimensional, y los cambios se reflejan en las vistas del modelo de manera instantánea.

Al utilizar modelos tridimensionales en lugar de dibujos en dos dimensiones, se requieren herramientas más avanzadas tanto de software como de hardware, actualmente se cuenta con hardware especializado para este tipo de trabajo, como son estaciones de trabajo, o computadoras personales con tarjetas gráficas muy potentes, así como *plotters* e impresoras veloces y eficientes. Sin embargo, en cuanto al software, la situación es más complicada aún: dado que existe una gran cantidad de aplicaciones que se le puede dar al hardware, y que cada una de estas requiere de tareas específicas, tenemos que difícilmente se puede encontrar un software que realice todas las tareas requeridas en ciertas áreas muy especializadas.

Las grandes compañías de software, han puesto a la disposición de sus usuarios una gran cantidad de productos CAD, tanto de propósito general como de aplicación especializada, aunque como es de esperarse, estas resultan frecuentemente insuficientes o inadecuadas para ciertas áreas. Sabiendo esto, las empresas dedicadas a la programación de herramientas CAD, presentan productos como AutoCAD, que además de su flexibilidad, incorporan una característica más: sus comandos pueden ser accedidos no sólo manualmente, sino también a través de lenguajes de programación, además puede de esta forma agregarse más comandos al entorno de diseño ó redefinir los comandos ya existentes.

Esto permite, que las empresas que lo requieran, puedan desarrollar el software basado en el producto CAD, cuando necesiten realizar tareas específicas no incluidas originalmente, o bien adquirirlo por parte de una tercera empresa, que se dedique a esta actividad.

Como principal objetivo de este trabajo, se plantea la creación de una librería de funciones que agilicen la elaboración de dibujos tridimensionales para el área de ingeniería HVAC. Con este propósito se crean funciones en AutoLISP para la generación automática de dibujos tridimensionales que sean consistentes con los estándares actuales de representación simbólica para esta rama **[B-7]** y al mismo tiempo proporcionen una representación realista del diseño a dibujar.

III.2 Descripción de las funciones HVAC.

Las librerías creadas para ayudar en las tareas descritas en la sección anterior se pueden dividir en:

a) Funciones que realizan un dibujo específico. Estas funciones generan una serie de comandos o modificaciones a la base de datos interna de AutoCAD, de tal forma que al terminar de ejecutarse dejan un dibujo en la pantalla como si hubiera sido creado por el usuario con los comandos originales de AutoCAD, con la diferencia de que sólo se solicita al usuario la introducción de unos parámetros básicos, y de que el dibujo se realiza en unos cuantos segundos.

b) Funciones de ayuda al diseño. Estas funciones no realizan un dibujo en pantalla, pero calculan valores que se necesitaran para poder realizar un diseño de manera rápida, para esto se crean funciones para determinar la orientación de un objeto con respecto a otro, cambiar característica de un objetos, extender o recortar objetos hasta un punto determinado, crear informes sobre el numero de objetos, el tipo de estos y determinar su área.

c) Funciones de interacción con el usuario. Este tipo de funciones se encargan de tareas como cargar y controlar cuadros de diálogo, mostrar datos concernientes a un objeto, cargar y controlar menús y de la configuración de botones en la barra de herramientas de AutoCAD, además de solicitar la entrada de datos desde la línea de comandos.

En las páginas siguientes se dará la descripción de las funciones creadas para agilizar las tareas de dibujo que forman parte de un proyecto de ingeniería HVAC.

Para dar suficientes detalles de las funciones mencionadas se seguirá un mismo formato para todas aquellas que realizan un dibujo específico, en este formato se tiene primero el nombre de la función subrayado, el nombre de la pieza que dibujan entre paréntesis, sus parámetros de entrada y el tipo de datos al que corresponde cada parámetro, las variables locales de la función que representan coordenadas y medidas para realizar el dibujo; y una figura que muestra vistas del dibujo creado después de ejecutar la función. Se mostrarán las ventajas que dan al usuario las funciones de ayuda al diseño, así como de las barras de herramientas y menús creados para el acceso a la librería gráfica, de forma muy similar a la forma en que se mostraron los comandos originales de AutoCAD, esto es con el propósito de marcar las diferencias que existen en cuanto a comandos de usuario al utilizar las librerías y los comandos originales.

Nombre de la Función: *Ducto* (Ducto Redondo)
Parámetros de entrada: p(lista), q(lista), diam(entero).
Variables de coordenadas y medidas: p, q, diam.

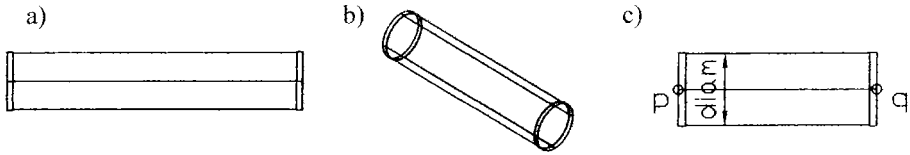


Figura III.2.1 Dibujo generado por la función *Ducto*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *DuctoRect* (Ducto Rectangular)
Parámetros de entrada: p(lista), q(lista), ancho(entero), alto(entero).
Variables de coordenadas y medidas: p, q, ancho, alto.

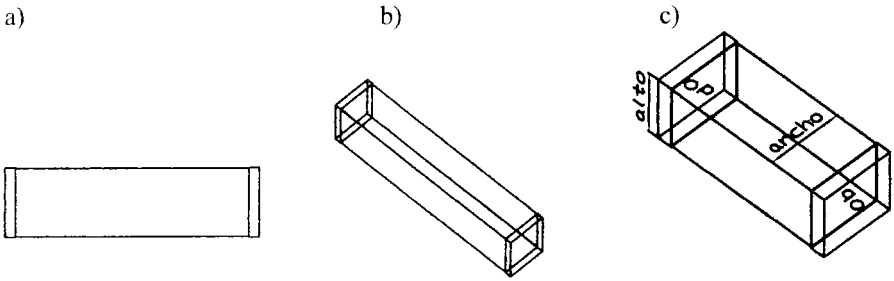


Figura III.2.2 Dibujo generado por la función *DuctoRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *DuctoF* (Ducto Flexible)
Parámetros de entrada: p(lista), q(lista), diam(entero).
Variables de coordenadas y medidas: p, q, diam, h, s, ds, n, n1, n2, dist.

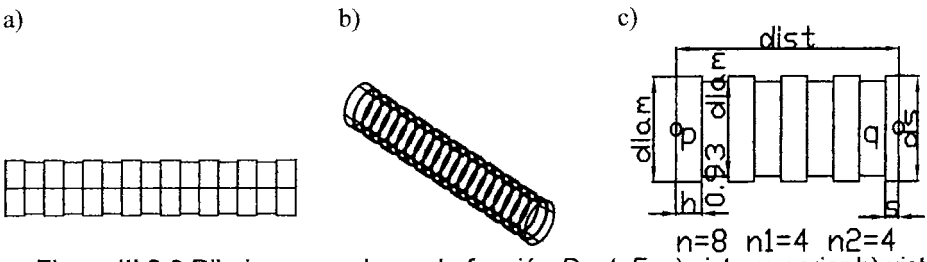


Figura III.2.3 Dibujo generado por la función *DuctoF*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *Codo* (Codo Redondo)
Parámetros de entrada: p(lista), q(lista), dir (cadena) diam(entero), ang(real), relrd(real).
Variables de coordenadas y medidas: p, q, diam, p0, p1, p3, p3, pa, pb, pc,ang.

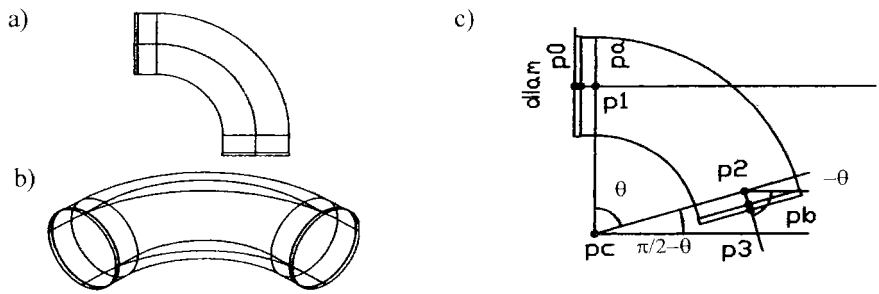


Figura III.2.4 Dibujo generado por la función *Codo*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *CodoF* (Codo Flexible)
Parámetros de entrada: p(lista), q(lista), dir(cadena), diam(entero), ang(real), relrd(real).
Variables de coordenadas y medidas: p, q, diam, long, lp, lq, n, i, p2, pr1, ang, rc.

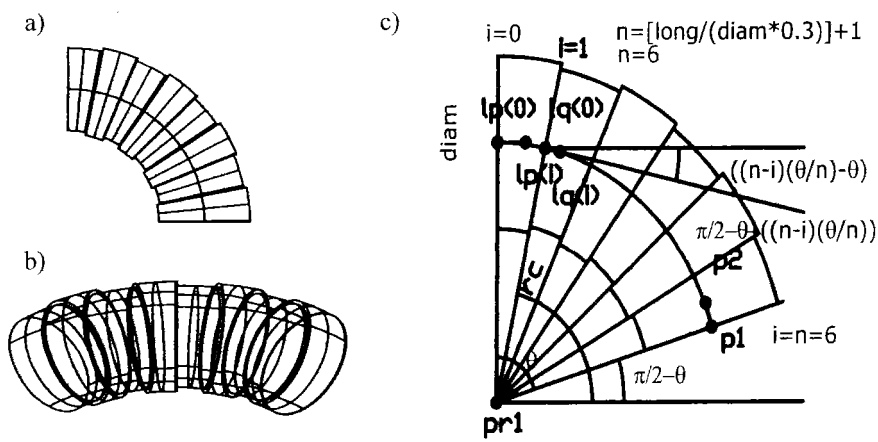


Figura III.2.5 Dibujo generado por la función *CodoF*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *CodoRectRed* (Codo Rectangular en Reducción)
Parámetros de entrada: p(lista), q(lista), dir (cadena) ancho1(entero), ancho2(entero), alto(entero), ang(real), relrd(real).
Variables de coordenadas y medidas: p, q, ancho1, ancho2, dir, pc1, p0, p1, pd1, p2, p3.

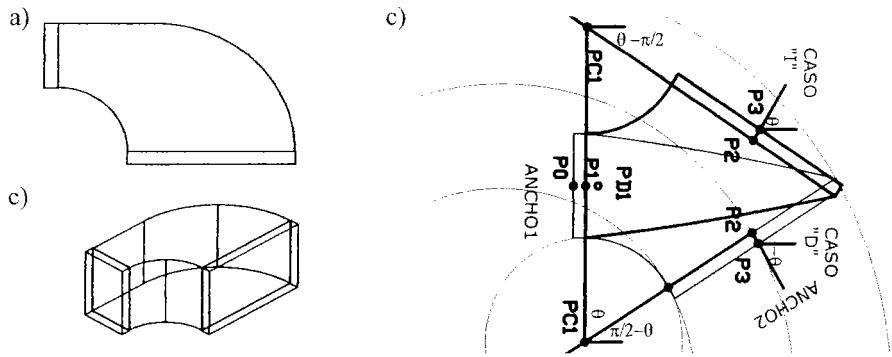


Figura III.2.6 Dibujo generado por la función *CodoRectRed*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *CodoRectRE* (Codo Rectangular con Radio en Escuadra)
Parámetros de entrada: p(lista), q(lista), dir (cadena) ancho(entero), alto(entero), ang(real), cuello (real).
Variables de coordenadas y medidas: ancho, alto, cuello, a, b, pr, p1, p2, p3, p4, p5, p6.

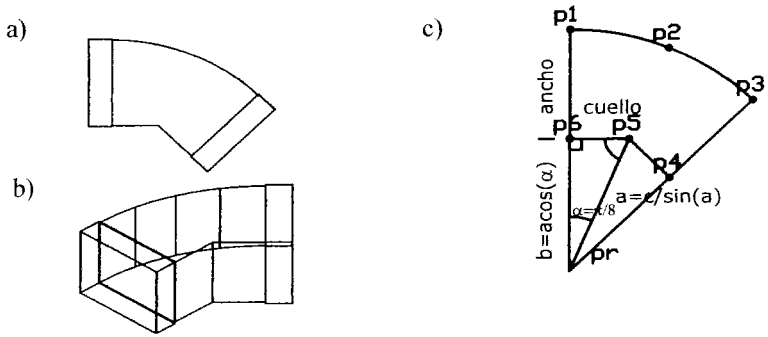


Figura III.2.7 Dibujo generado por la función *CodoRectRE*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *CodoRect* (Codo Rectangular)
Parámetros de entrada: p(lista), q(lista), dir (cadena) ancho(entero), alto(entero), ang(real), relrd(real)
Variables de coordenadas y medidas: Las mismas que para *Codo*.

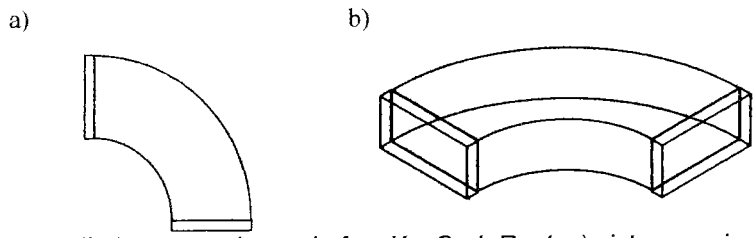


Figura III.2.8 Dibujo generado por la función *CodoRect*. a) vista superior b) vista isométrica.

Nombre de la Función: *CodoEscRect* (Codo Rectangular en Escuadra)
Parámetros de entrada: p(lista), q(lista), dir (cadena) ancho(entero), alto(entero), ang(real),
Variables de coordenadas y medidas: p, q, ancho, alto, ang, p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11.

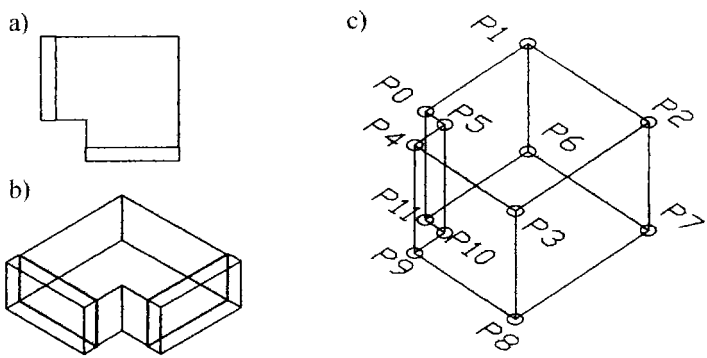


Figura III.2.9 Dibujo generado por la función *CodoEscRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *Desviacion* (Desviación Redonda)
Parámetros de entrada: p(lista), q(lista), dir (cadena) diam(entero), desv(real).
Variables de coordenadas y medidas: p0, p1, p2, p3, p4, p5, pa, pb, pr1, pr2, h, z, r, diam.

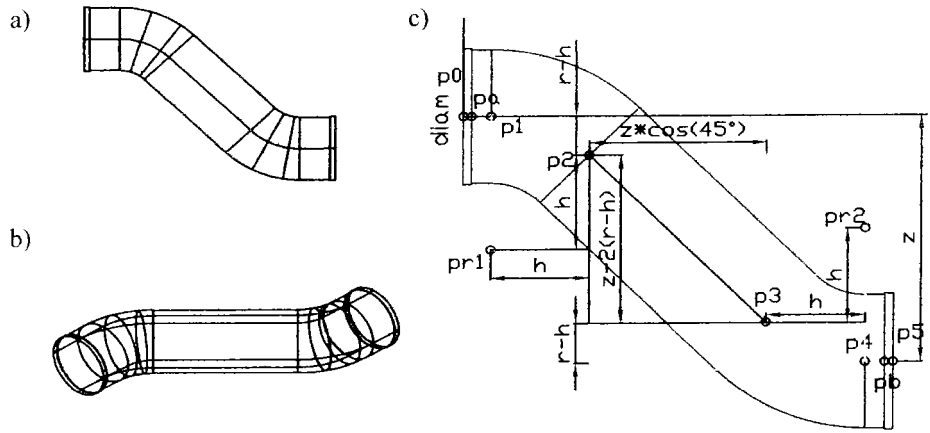


Figura III.2.10 Dibujo generado por la función *Desviacion*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *DesviacionRect* (Desviación Rectangular)
Parámetros de entrada: p(lista), q(lista), dir (cadena) ancho(entero), alto(entero), ang(real), relrd(real)
Variables de coordenadas y medidas: Las variables de para *Desviacion* con alto y ancho en lugar de diam.

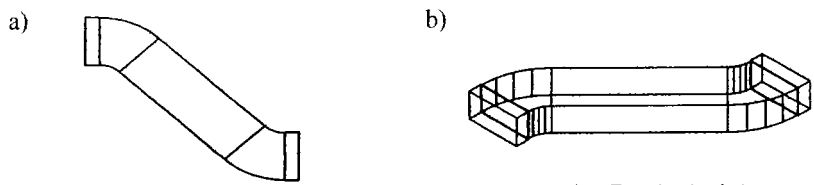


Figura III.2.11 Dibujo generado por la función *DesviacionRect*. a) vista superior b) vista isométrica.

Nombre de la Función: *TapaRed* (Tapa Redonda)
Parámetros de entrada: p(lista), q(lista), diam(entero),
Variables de coordenadas y medidas: p, q, diam.

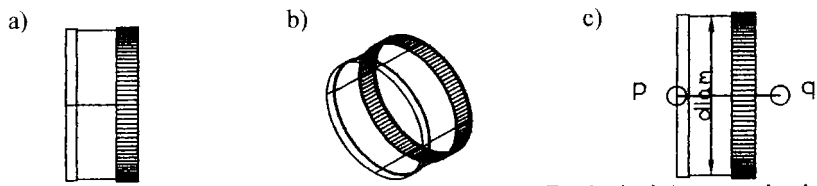


Figura III.2.12 Dibujo generado por la función *TapaRed*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *TapaRect* (Tapa Rectangular)
Parámetros de entrada: p(lista), q(lista), ancho(entero),alto(entero).
Variables de coordenadas y medidas:p, q, ancho, alto.

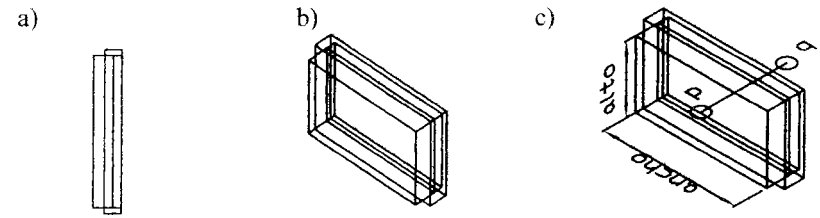


Figura III.2.13 Dibujo generado por la función *TapaRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *Reduccion* (Reducción Redonda)
Parámetros de entrada: p(lista), q(lista), diam1(entero), diam2(entero).
Variables de coordenadas y medidas:p, q, diam1,diam2.

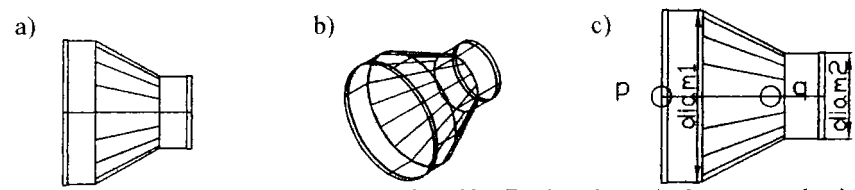


Figura III.2.14 Dibujo generado por la función *Reduccion*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *ReduccionEx* (Reducción Excéntrica Redonda)
Parámetros de entrada: p(lista), q(lista), diam1 (entero) diam2(entero).
Variables de coordenadas y medidas: Las mismas que para *Reduccion*.

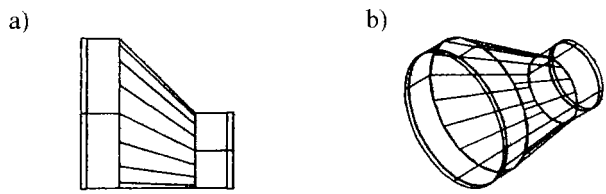


Figura III.2.15 Dibujo generado por la función *ReduccionEx*. a) vista superior b) vista isométrica.

Nombre de la Función: *ReduccionRect* (Reducción Rectangular)
Parámetros de entrada: p(lista), q(lista), ancho1(entero),alto1(entero), ancho2(entero), alto2(entero).
Variables de coordenadas y medidas: p0,p1,p2,p3, ancho 1 y 2,alto 1 y 2.

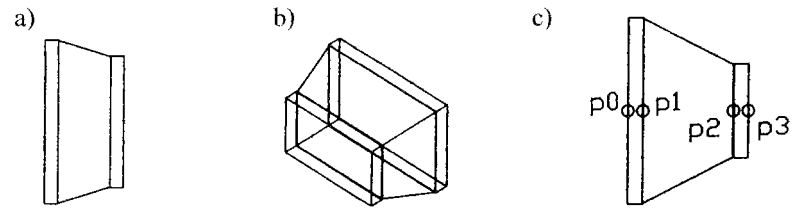


Figura III.2.16 Dibujo generado por la función *ReduccionRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *ReduccionEXRect* (Reducción Excéntrica Rectangular)
Parámetros de entrada: p(lista), q(lista), ancho1(entero),alto1(entero), ancho2(entero), alto2(entero).
Variables de coordenadas y medidas:p, q, ancho 1 y 2, alto1 y 2, p0, p1, p2, p3.

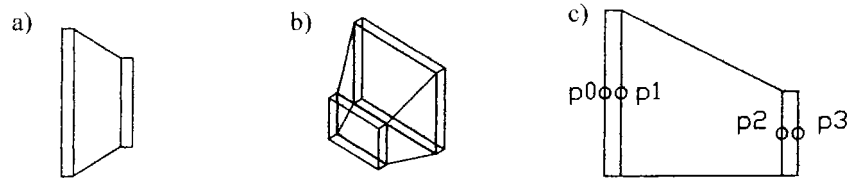


Figura III.2.17 Dibujo generado por la función *ReduccionEXRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *ReduccionEXEXRect* (Reducción Bi-excéntrica Rectangular)
Parámetros de entrada: p(lista), q(lista), ancho1(entero),alto1(entero), ancho2(entero), alto2(entero), alineación(cadena).
Variables de coordenadas y medidas:p, q, ancho1, alto1, ancho2, alto2, p1, p2, p3, p4, p5, p6, p7, p8.

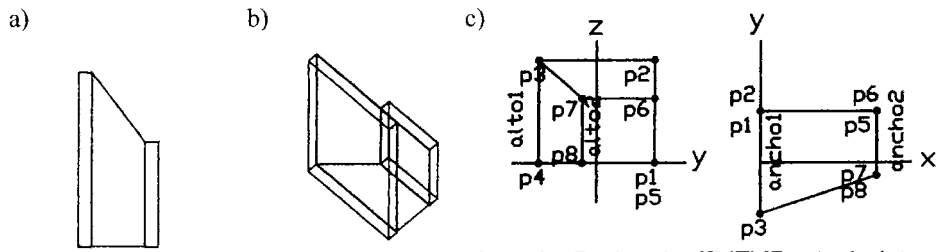


Figura III.2.18 Dibujo generado por la función *ReduccionEXEXRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *CTS* (Cabeza de Toro Simple)
Parámetros de entrada: p(lista), q(lista), diam(entero).
Variables de coordenadas y medidas: p, q, diam, p1, p2, p3.

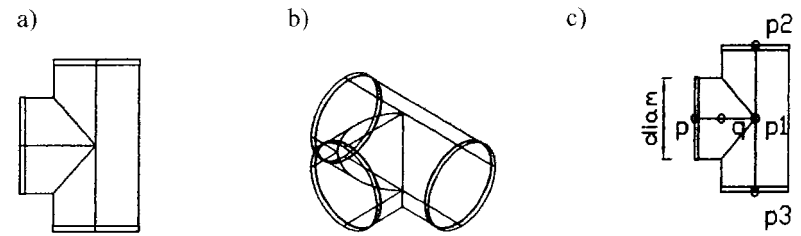


Figura III.2.19 Dibujo generado por la función *CTS*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *CTR* (Cabeza de Toro en Reducción)
Parámetros de entrada: p(lista), q(lista),diam1(entero), diam2(entero), diam3(entero).
Variables de coordenadas y medidas: p,q, diam 1 y 2 , p1, p2, p3, p4, p5, p6, p7.

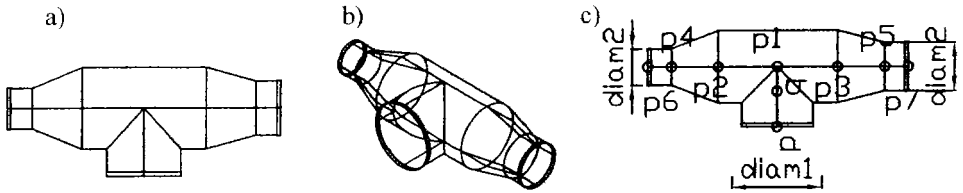


Figura III.2.20 Dibujo generado por la función *CTR*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *CTSRect* (Cabeza de Toro Simple Rectangular)
Parámetros de entrada: p(lista), q(lista), ancho(entero),alto(entero).
Variables de coordenadas y medidas:p, q, ancho, alto, cuello, p1, p2, p3, p4 ,p5, p6, p7, p8, p9, p10, p11, p12, p13, p14, p15.

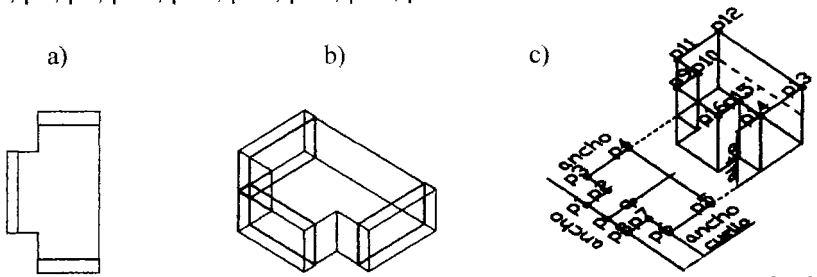


Figura III.2.21 Dibujo generado por la función *CTSRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *CTRRectRed* (Cabeza de Toro en Reducción Rectangular)
Parámetros de entrada: p(lista), q(lista), ancho1(entero), ancho2(entero), ancho3(entero), alto(entero).
Variables de coordenadas y medidas: ancho1, ancho2, ancho3, alto, más los mismos puntos usados en *CTRRect*.

a) b)

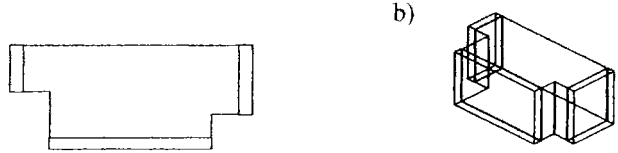


Figura III.2.22 Dibujo generado por la función *CTRRectRed*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *CIE* (Cruz con Injerto Eficiente)
Parámetros de entrada: p(lista), q(lista),diam1(entero), diam2(entero), diam3(entero).
Variables de coordenadas y medidas: p, q, diam1, diam2, diam2, p1, p2, p3.

a) b) c)

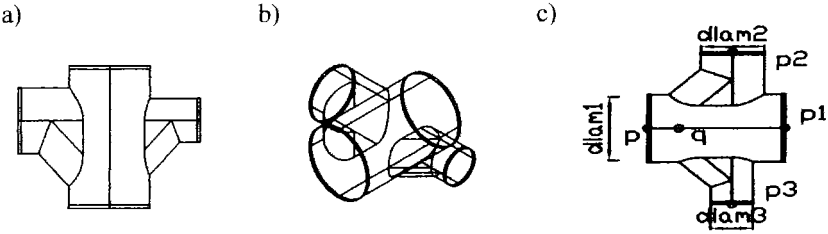


Figura III.2.23 Dibujo generado por la función *CIE*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *CIER* (Cruz con Injerto Eficiente en Reducción)
Parámetros de entrada: p(lista), q(lista), diam1(entero), diam2(entero), diam3(entero), diam4(entero).
Variables de coordenadas y medidas: p, q, diam1, diam2, diam3, diam4, p1, p2, p3, p4, p5.

a) b) c)

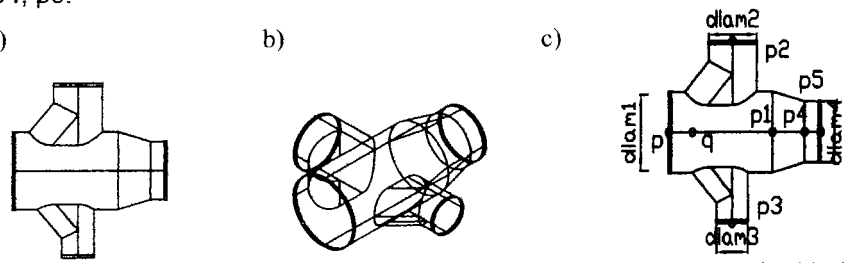


Figura III.2.24 Dibujo generado por la función *CIER*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *TIE* (Te con Injerto Eficiente)
Parámetros de entrada: p(lista), q(lista), diam1(entero), diam2(entero).
Variables de coordenadas y medidas: p, q, diam1, diam2, p1, p2.

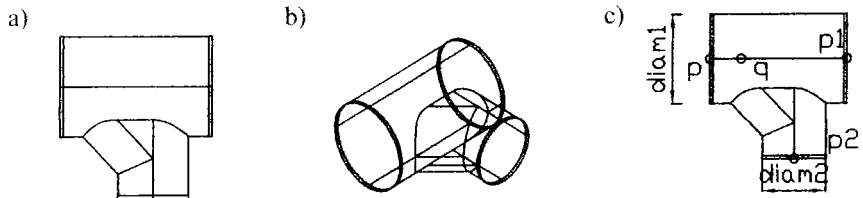


Figura III.2.25 Dibujo generado por la función *TIE*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *TIER* (Te con Injerto Eficiente en Reducción)
Parámetros de entrada: p(lista), q(lista),diam1(entero), diam2(entero), diam3(entero).
Variables de coordenadas y medidas: p, q, diam1, diam2, diam3, p1, p2, p3, p4.

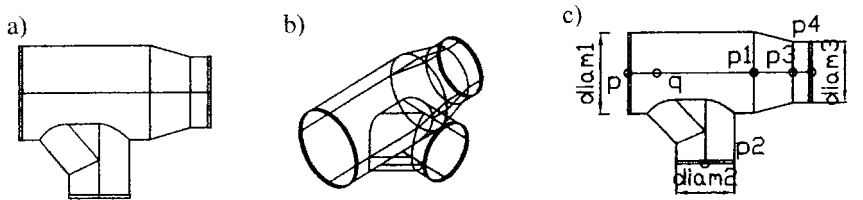


Figura III.2.26 Dibujo generado por la función *TIER*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *TIR* (Te con Injerto Redondo)
Parámetros de entrada: p(lista), q(lista), diam1(entero), diam2(entero).
Variables de coordenadas y medidas: p, q, diam1, diam2, p1, p2.

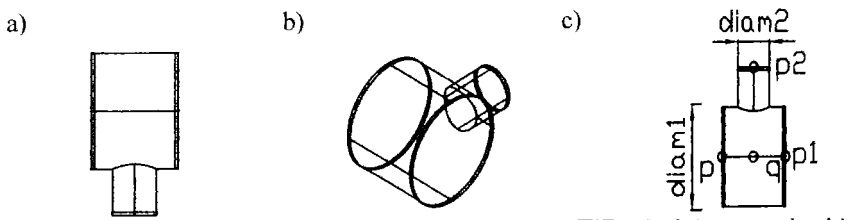


Figura III.2.27 Dibujo generado por la función *TIR*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *TIRR* (Te con Injerto Redondo en Reducción)
Parámetros de entrada: p(lista), q(lista), diam1(entero), diam2(entero), diam3(entero).
Variables de coordenadas y medidas: p, q, diam1, diam2, diam3, p1, p3, p3, p4.

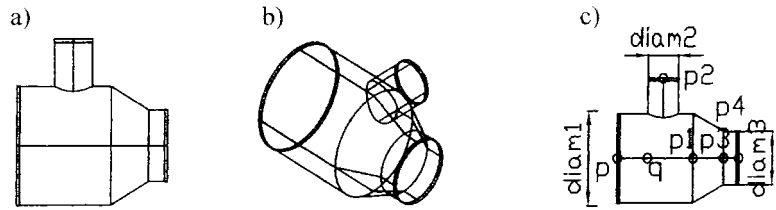


Figura III.2.28 Dibujo generado por la función *TIRR*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *TCR* (Transformación de Ducto Rectangular a Redondo)
Parámetros de entrada: p(lista), q(lista), diam(entero), ancho(entero), alto(entero), long(entero).
Variables de coordenadas y medidas: ancho, alto, p0, p1, p2, p3, p4, p5.

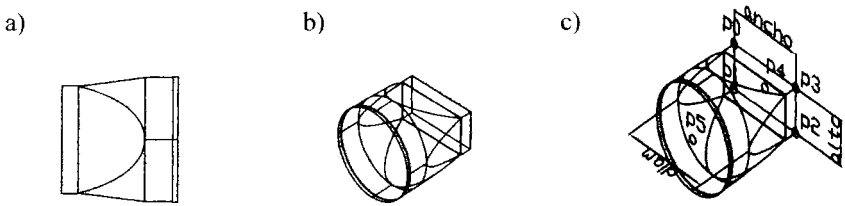


Figura III.2.29 Dibujo generado por la función *TCR*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *SALM* (Cuello de lona)
Parámetros de entrada: p(lista), q(lista), ancho(entero), alto(entero), long(entero).
Variables de coordenadas y medidas: p, q, ancho, alto, long.

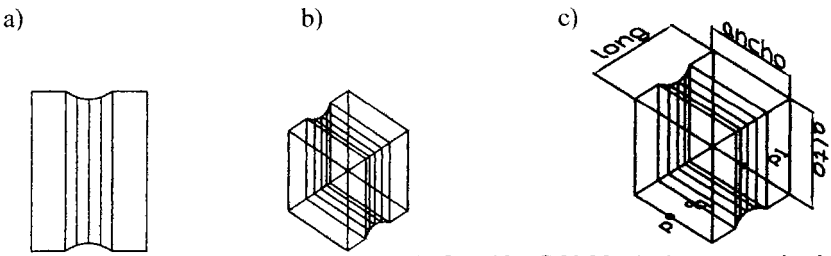


Figura III.2.30 Dibujo generado por la función *SALM*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *YIC* (Ye con Injerto Cónico)
Parámetros de entrada: p(lista), q(lista), diam1(entero), diam2(entero), diam3(entero), opcion (cadena).
Variables de coordenadas y medidas: p, q, diam1, diam2, diam3, p1, p2, p3, p4, p5, p6, p7, p8, p9.

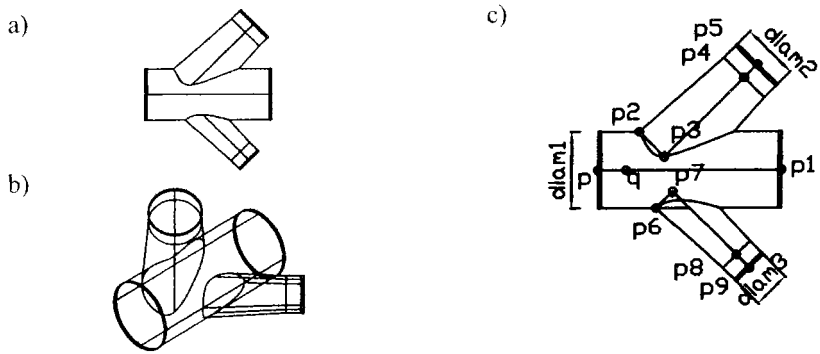


Figura III.2.31 Dibujo generado por la función *YIC*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *YICR* (Ye con Injerto Cónico en Reducción)
Parámetros de entrada: p(lista), q(lista), diam1(entero), diam2(entero), diam3(entero), diam4(entero), opcion(cadena).
Variables de coordenadas y medidas: p, q, diam1, diam2, diam3, diam4, p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, p11.

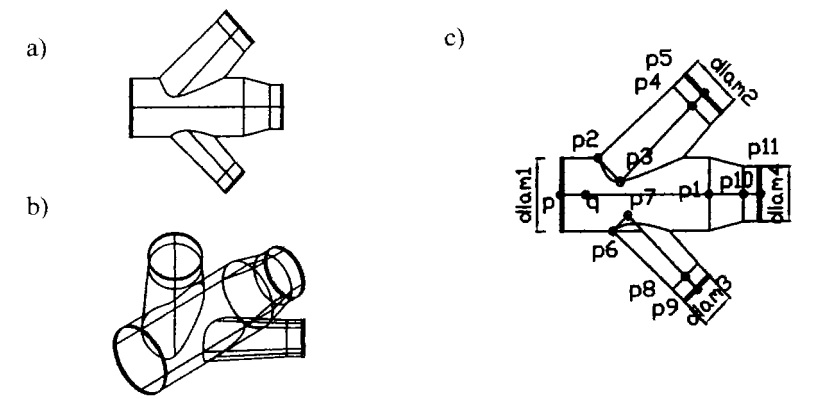


Figura III.2.32 Dibujo generado por la función *YICR*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *YeCCRect* (Ye Codo-Codo Rectangular)
Parámetros de entrada: p(lista), q(lista), dir (cadena) ancho(entero), alto (entero), desv1(real) desvs1(entero), desv2(real) desvs2(entero), ang1(real),ang2(real), relrd1(real), relrd2(real).
Variables de coordenadas y medidas: ancho, alto, dsvs1, desvs2, pr1, pr2, p0, p1, p2, p3, p4, p5, p6, p7, p8.

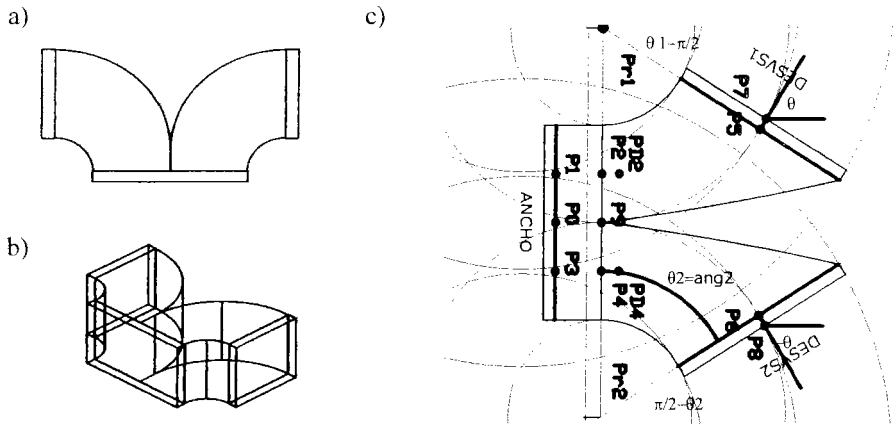


Figura III.2.33 Dibujo generado por la función *YeCCRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *YeCDRect* (Ye Codo-Ducto Rectangular)
Parámetros de entrada: p(lista), q(lista), ancho(entero), alto(entero), desv1(real), desvs1(entero), ang(real), relrd(real).
Variables de coordenadas y medidas: ancho, alto, desvs1, pr1, r1, p0, p1, p2, p3, p4, p5, p6, p7, p8, p9, p10.

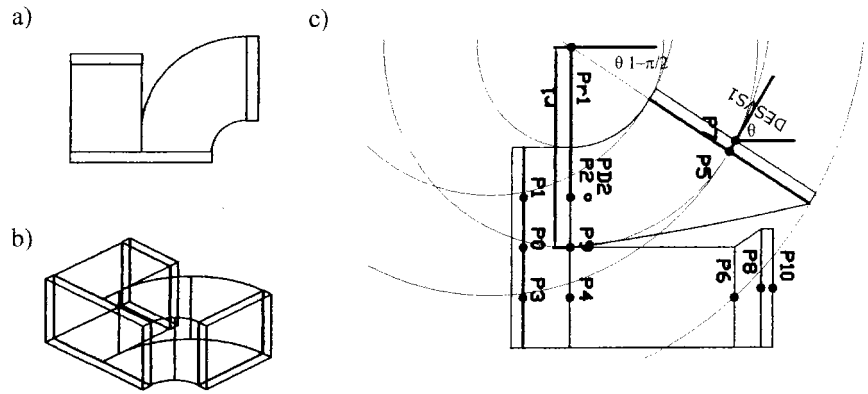


Figura III.2.34 Dibujo generado por la función *YeCDRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *YeCDCRect* (Ye Codo-Ducto-Codo Rectangular)
Parámetros de entrada: p(lista), q(lista), dir (cadena) ancho(entero), alto (entero), desv1(real) desvs1(entero), desv2(real), desvs2(entero), ang1(real),ang2(real), relrd1(real), relrd2(real).
Variables de coordenadas y medidas: ancho, alto, desvs1, pr1, r1, pr2, r2, p0, p1, p2, p3, p4, p5, p6, p7, p8, p9, p10, pterm1, pterm2, pterm2.

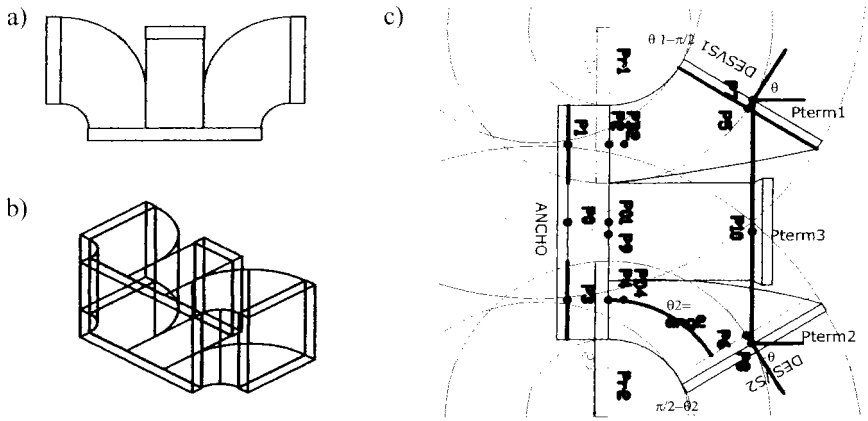


Figura III.2.35 Dibujo generado por la función *YeCDCRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *DifusorCuad* (Difusor cuadrado)
Parámetros de entrada: p(lista), q(lista), dir (cadena) ancho(entero)
Variables de coordenadas y medidas: p, q, ancho, p0, p1, p2, p3, p4, p5, vias.

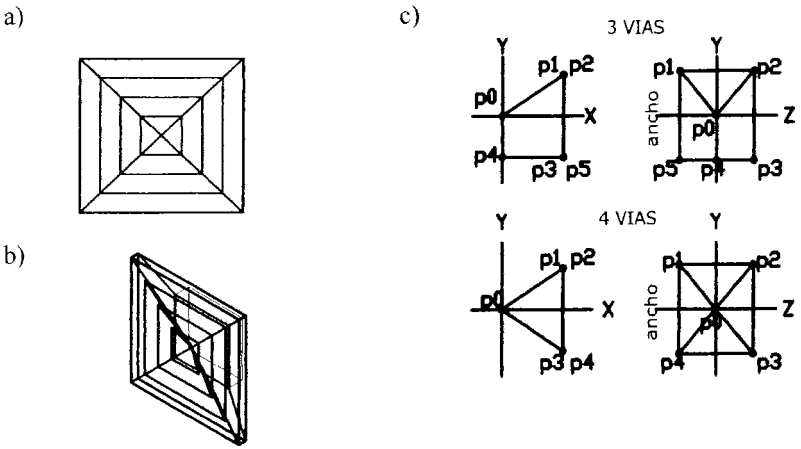


Figura III.2.36 Dibujo generado por la función *DifusorCuad*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *Difusor* (Difusor Redondo)
Parámetros de entrada: p(lista), q(lista), diam1(entero), diam2(entero).
Variables de coordenadas y medidas: p, q, diam 1 y 2, p1, p3, p3, h1, h2, r.

a) b) c)

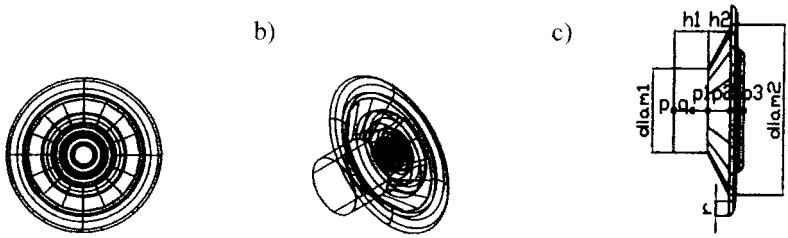


Figura III.2.37 Dibujo generado por la función *Difusor*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *RejillaRect* (Rejilla Rectangular)
Parámetros de entrada: p(lista), q(lista), ancho(entero), alto(entero).
Variables de coordenadas y medidas: p, q, ancho, alto, p1, pbi, pbj, gr.

a) b) c)

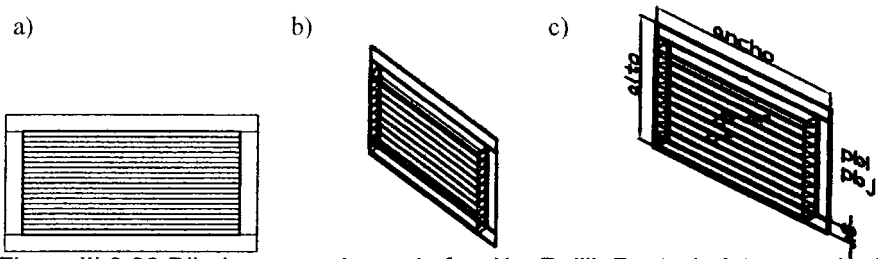


Figura III.2.38 Dibujo generado por la función *RejillaRect*. a) vista superior b) vista isométrica c) coordenadas.

Nombre de la Función: *UMA* (Unidad Manejadora de Aire)
Parámetros de entrada: p(lista), q(lista), ancho(entero), alto(entero) seccion(lista).
Variables de coordenadas y medidas: p, q, ancho, alto, seccion.

a) b) c)

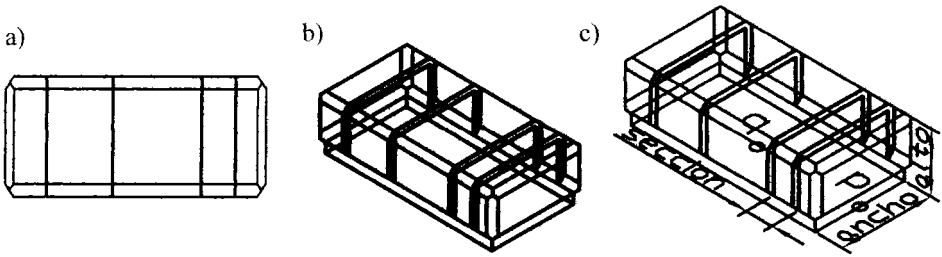


Figura III.2.39 Dibujo generado por la función *UMA*. a) vista superior b) vista isométrica c) coordenadas.

Como se ha mencionado, tenemos también funciones de ayuda al diseño, estás son la siguientes:

Función *CambiarMedidas*.

Esta función, en los casos en que es posible aplicarla, sustituye a la tarea de crear un dibujo nuevamente, ya sea con los comandos originales de AutoCAD o con la librería de funciones introducidas. Toma como parámetro un objeto seleccionado por el usuario con el ratón; este objeto queda identificado por su "nombre de entidad" **[E-1]**, de forma que se pueden recuperar los datos actuales del objeto y volver a crear otro con diferentes medidas, para crear el efecto que fue el objeto original el que se modificó.

Función *DetectarOrigen*.

En este caso, se llama a la función cuando el usuario escoge la opción "*Seleccionar objeto base*" que aparece después de que selecciona un comando de dibujo, si entonces se selecciona un objeto dibujado por las librerías, se reconocerán los datos de este. Con esto se tiene la ventaja de pasar automáticamente la alineación de un objeto a otro que se va a dibujar, además de que se toman las medidas del objeto seleccionado como base al nuevo dibujo. Toma como parámetro el nombre de entidad un objeto seleccionado con el ratón.

Función *LlegarA*.

Se crea esta función para poder modificar la longitud de un objeto gráfico del tipo "ducto" en base a un punto seleccionado en la pantalla. Con el fin de hacer más sencilla la tarea de dibujo, esta función permite que, en caso de que el punto seleccionado no se encuentre dentro de la prolongación del objeto, el programa agregue un codo y un ducto adicional, de forma que la trayectoria de objeto al extenderse se doble y coincida con el punto seleccionado como se puede apreciar en la figura III.2.39. En el caso de que el objeto no cambie de dirección debido al punto escogido, la función no pedirá información adicional, si el objeto debe cambiar de dirección se pedirá un punto adicional para determinar desde qué ángulo deberá llegarse a ese punto. Como los objetos se encuentran en un espacio tridimensional, la vista que tiene el usuario puede no corresponder con el sistema de coordenadas (UCS) de dibujo. El objeto puede encontrarse en un plano distinto al que se tiene en la vista, por ejemplo; en un ducto inclinado la coordenada Z varía con respecto al plano en el cual el usuario seleccionó los puntos; como la pantalla del monitor es plana, el usuario no tiene un medio directo de seleccionar puntos con diferentes coordenadas en el eje Z, considerando a los ejes X e Y como dentro del plano que se presenta en la pantalla. Para solucionar esto, se toma la proyección de los puntos seleccionados sobre el plano en que se encuentre el objeto, y se obtiene un ángulo adicional que representa la torsión del plano para lograr una alineación con el punto seleccionado. Las ecuaciones para encontrar las coordenadas absolutas (WCS) de tales puntos se tomaron de la referencia bibliográfica **[B-4]**.

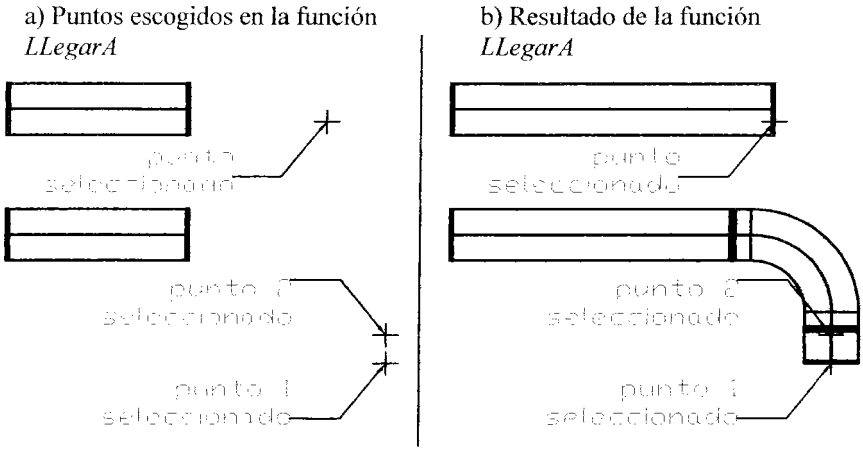


Figura III.2.39. Resultado de aplicar la función *LlegarA*.

Función *DibujarMedidas*.

Esta función genera automáticamente el texto correspondiente a la medida nominal de un objeto y lo alinea con dicho objeto. Toma como parámetro un "conjunto de selección de objetos" **[E-1]**, de tal forma que se puede ejecutar sobre cualquier cantidad de objetos que hayan sido seleccionados en la pantalla. El resultado de la aplicación de esta función sobre un pequeño conjunto de objetos se puede ver en la figura III.2.40.

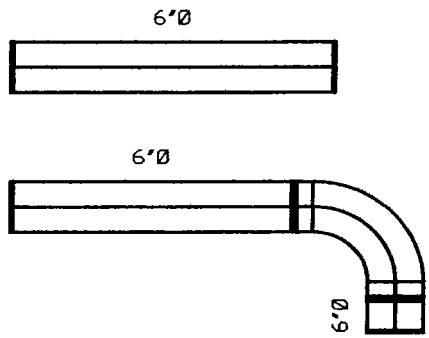
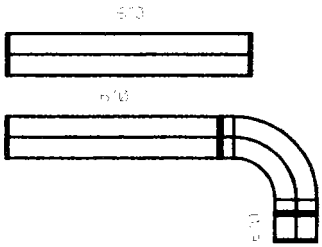


Figura III.2.40. Resultado de aplicar la Función *DibujarMedidas*.

Función *GenerarInforme*.

Con el fin de realizar un conteo automático de los elementos dibujados y de sus características, se implementa esta función. Toma como parámetro un conjunto de selección de objetos y como salida produce el dibujo de una lista con las características dichos elementos (figura III.2.41), o bien un archivo de texto con esta información.

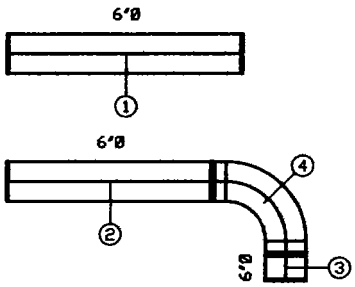


No.	TAMAÑO	LONG./CANT.	DESCRIPCION
3	6"	8.07 '	DUCTO REDONDO
2	6"	34.81 '	DUCTO REDONDO
1	6"	39.24 '	DUCTO REDONDO
4	6"	1	CODO REDONDO A 90° 9" DE RADIO AL CENTRO

Figura III.2.41. Resultado de aplicar la Función *GenerarInforme*.

Función *DibujarNumeración*.

Con esta función se dibuja sobre cada objeto seleccionado él número que le corresponde dentro de una lista creada por la función *GenerarInforme*. Se identifica así cada objeto con una línea y un número dentro de un círculo (figura III.2.42). Aquí se toma como parámetro un conjunto de selección de objetos.



No.	TAMAÑO	LONG./CANT.	DESCRIPCION
3	6"	8.07 '	DUCTO REDONDO
2	6"	34.81 '	DUCTO REDONDO
1	6"	39.24 '	DUCTO REDONDO
4	6"	1	CODO REDONDO A 90° 9" DE RADIO AL CENTRO

Figura III.2.42. Resultado de aplicar la Función *DibujarNumeración*.

Función *GirarObjeto*.

Esta función se implemento para ofrecer una opción adicional del comando de AutoCAD *Rotate3D* . En este caso el programa calcula el eje de rotación y lo hace coincidir con el eje del objeto seleccionado.

Como una ayuda adicional al diseño, se consideró la característica de que los dibujos creados sigan siendo compatibles con los comandos ya existentes de AutoCAD; esto es por el hecho de que existen aplicaciones comerciales de diseño de plantas industriales basadas en AutoCAD que no guardan esta compatibilidad **[B-6]**, sino que para manipular sus objetos incorporan nuevos comandos, con lo que se pierde la ventaja que tiene el usuario familiarizado con los comandos originales además de que el número de comandos que tiene que aprender el usuario aumenta drásticamente.

Con respecto a las funciones de interacción con el usuario se crearon cuadros de dialogo, menús y botones que se describen a continuación. Los cuadros de dialogo fueron definidos en el lenguaje DCL y se controlan mediante AutoLISP. Las barras de herramientas y los menús se crearon siguiendo el mismo formato que tiene el menú original de AutoCAD en el archivo *acad.mnu*.

Menú Principal.

Con el fin de tener acceso rápido a la librería de funciones se agregó un menú dentro de la barra de menús de (figura III.2.43), que se divide en submenús para cada una de las funciones creadas.

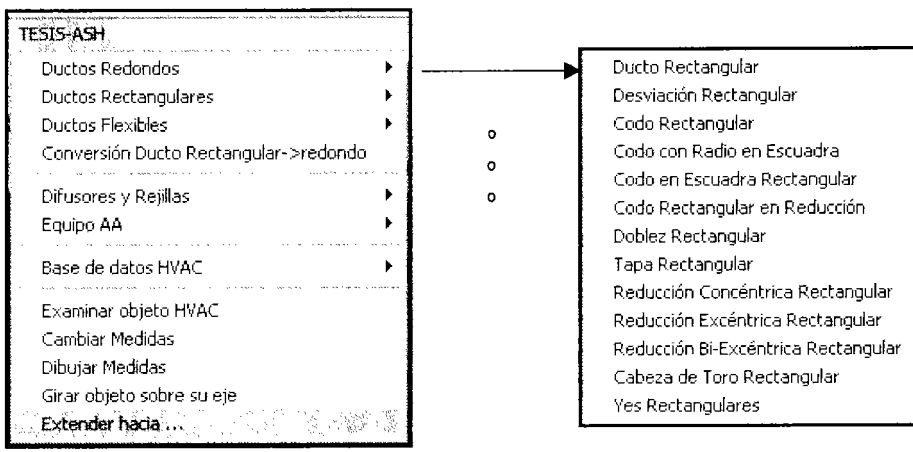


Figura III.2.43 Menú principal de las librerías HVAC.

Barras de Herramientas.

Se incorporaron también botones de acceso para las funciones creadas, estos botones se agruparon en barras de herramientas correspondientes a las categorías siguientes:

- Ductos Redondos
- Ductos Rectangulares
- Ductos Flexibles
- Base de Datos
- Equipo Aire Acondicionado
- Difusores y Rejillas
- Utilidades HVAC

Los iconos mostrados en cada barra de herramientas pueden apreciarse en la figura III.2.44.

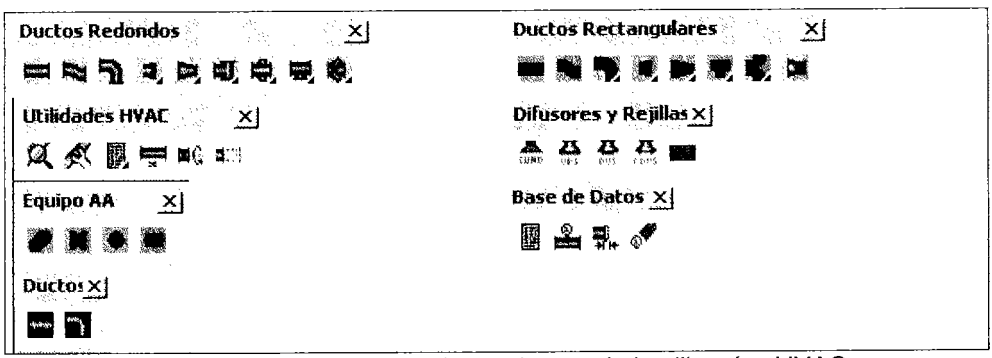


Figura III.2.44 Barras de Herramientas de las librerías HVAC.

Cuadro de diálogo *VerObjetos.*

Este cuadro de diálogo es controlado por la función del mismo nombre y sirve para mostrar al usuario información correspondiente a un objeto de dibujo creado con las librerías. El contenido de las etiquetas y comportamiento de sus elementos varía según el objeto seleccionado (figura III.2.45).

Cuadro de diálogo *GenerarInforme.*
Con este cuadro de diálogo (figura III.2.46) se le solicita al usuario la entrada de datos para la función *GenerarInforme*; contiene como opciones a los parámetros que se pasan al llamar a ésta.

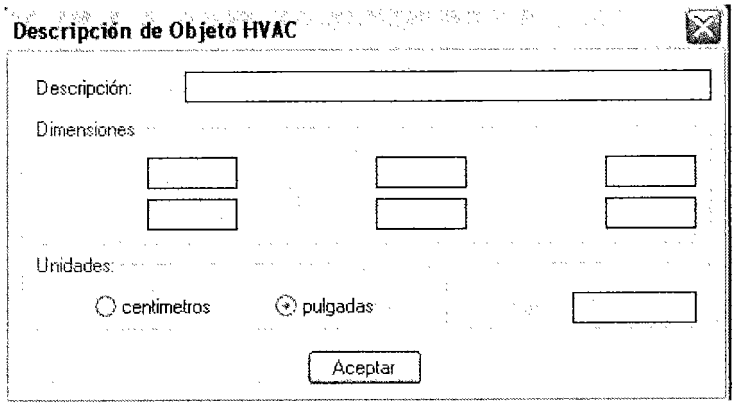


Figura III.2.45. Cuadro de diálogo *VerObjetos*.

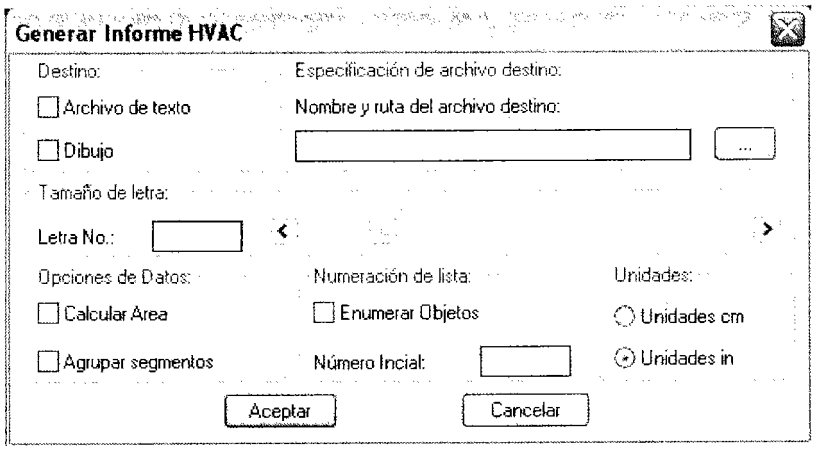


Figura III.2.46. Cuadro de diálogo *GenerarInforme*.

Cuadro de diálogo *DibujarMedidas*.

En este cuadro (figura III.2.47) de diálogo se pregunta por las opciones de la función *DibujarMedidas*.

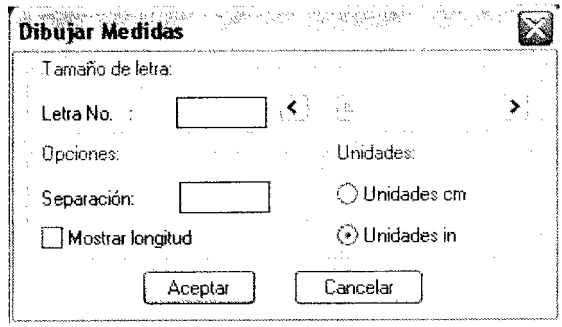


Figura III.2.47. Cuadro de diálogo *DibujarMedidas*.

Cuadro de diálogo *DibujarNumeración.*
Mediante este cuadro se pregunta por las opciones de la función *DibujarNumeración* (figura III.2.48).

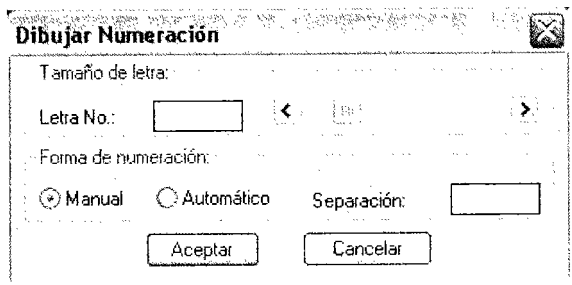


Figura III.2.48. Cuadro de diálogo *DibujarNumeración.*

Cuadro de diálogo *DimensionesAdicionales.*

En este cuadro (figura III.2.49) se pide el valor de las dimensiones de las pestañas de los ductos que se agregan a su longitud en las funciones *DibujarMedidas* y GenerarInforme.

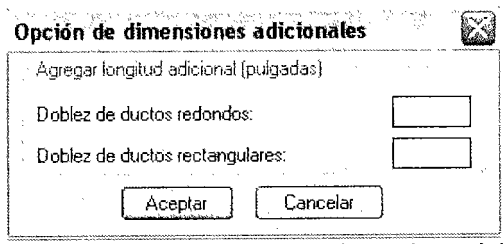


Figura III.2.49. Cuadro de diálogo *DimensionesAdicionales.*

En todos los cuadro de diálogo anteriores se proporcionan valores por omisión, además de que se guarda la última configuración de valores suministrada, por lo que en la mayoría de los casos sólo se tienen que hacer pequeños ajustes para mandar a llamar a los comandos respectivos.

Aparte de esto, se implementaron también funciones para interactuar entre el usuario y las funciones de dibujo; la mayoría de estas son rutinas sencillas que piden datos al usuario a través de la línea de comandos, no toman ningún parámetro y dan como resultado la llamada a una función que realiza un dibujo específico; dentro del código en AutoLISP se identifican de la forma:

ObtenDatos-*función*

Donde *función* se sustituye por la cadena de caracteres correspondiente al nombre de la función para la cual se piden los datos.

III. 3 Medidas de eficiencia de las librerías HVAC.

De manera resumida, tenemos que se crearon 9 librerías gráficas, cada barra de herramientas que se mostró en la figura III.2.44 corresponde a una librería; a éstas se suman una librería con funciones de interacción y una más de funciones auxiliares. Las librerías mencionadas agrupan en total a 76 funciones, de las cuales 46 corresponden a comandos accesibles directamente para el usuario, el resto corresponden a funciones auxiliares a las que recurren internamente los comandos para su funcionamiento.

Con el fin de tener una percepción cuantitativa de las mejoras que se agregan al programar las nuevas funciones complementarias para AutoCAD, se presenta a continuación un análisis del número de operaciones que se requieren para realizar la misma tarea con los comandos originales de AutoCAD y con los nuevos comandos creados en AutoLISP.

Se considera primero, que las operaciones de dibujo utilizadas, se agrupan bajo alguno de los tipos mencionados a continuación, los nombres asignados no constituyen una notación estándar, solamente se toman como una referencia:

- **Comandos:** De esta forma identificamos a la indicación por parte del usuario de una instrucción de AutoCAD, ya sea por medio de su escritura en la línea de comandos, selección desde un menú, selección por medio de un botón, o mediante la asignación de un método abreviado de teclado.

- **Coordenadas exactas:** Utilizamos esta expresión para referirnos a la selección por parte del usuario de un punto dentro del espacio de dibujo a través de la especificación de sus coordenadas en tres dimensiones, ya sea desde el teclado o por medio del ratón.

- **Coordenadas aproximadas:** Llamaremos de esta forma a la especificación de un punto de manera aproximada, en el espacio de dibujo por medio del ratón. En este caso, las coordenadas exactas pueden variar por tratarse de puntos próximos a un punto dado. La determinación de si un punto es próximo a otro punto o no, se obtiene de acuerdo a la escala de la vista que tiene el usuario del espacio de dibujo.

- **Medidas:** De esta forma indicamos que el usuario especifica una dimensión de cierto objeto de dibujo mediante la introducción desde el teclado de un número real o entero.

- **Textos:** Con esto identificamos a la operación en la cual el usuario introduce un texto por medio del teclado o mediante las instrucciones de copiar y pegar. Se presupone, que se trata de un texto corto de no más de 256 caracteres.

- **Consultas:** Denominamos así a la obtención por parte del usuario, de datos que no se encuentren explícitamente determinados dentro de un diseño, por ejemplo, la medición de cierta distancia entre dos elementos de dibujo, la conversión de una medida, medición del ángulo entre dos figuras, entre otros; estas operaciones no necesariamente tienen que realizarse dentro del entorno de diseño, pero los datos obtenidos son imprescindibles para elaborar el dibujo de un objeto con la precisión requerida en un trabajo de ingeniería.

- **Elecciones:** Con esto nos referimos a las operaciones en las que el usuario realiza la selección de una opción desde la línea de comandos de AutoCAD o desde un cuadro de dialogo.

En el caso de funciones que realizan algún dibujo específico, se consideran las funciones más representativas de la librería creada, incluyendo a la función desde en punto de vista técnico más sencilla, la función más complicada, y alguna función intermedia, considerando que los demás casos caen dentro de este rango, es decir, no sustituyen a más ni a menos operaciones. El resto de las funciones son analizadas como casos por separado.

Hay que tomar en cuenta que la cantidad de operaciones utilizadas puede variar de acuerdo a la habilidad y técnica utilizada por el usuario, por esta razón se especifica el tipo de comandos manejados con el número de éstos entre paréntesis.

Función *Ducto*.

Este es el caso de función que se considera que realiza el dibujo más sencillo.

Operaciones originales de AutoCAD.

Tipo de operación	**Cantidad de operaciones**	**Total**
Comandos:	*Cylinder* (1), *Scale* (1), *Copy*(2), *Rotate3D* (1);	5
Coordenadas exactas:	6	6
Coordenadas aproximadas:	3	3
Medidas:	3	3
Consultas:	Medir longitud (1), Conversión de unidades (1).	2
		19

Operaciones requeridas por la función.

Tipo de operación	**Cantidad de operaciones**	**Total**
Comandos:	*Ducto* (1).	1
Coordenadas exactas:	2	2
Medidas:	1	1
		4

Tabla III.3.1 Operaciones sustituidas por la función *Ducto*.

Función *Codo*.

Operaciones originales de AutoCAD requeridas.

Tipo de operación.	**Cantidad de operaciones**	**Total**
Comandos:	*Cylinder* (1), *Circle*(1), *Revolve*(1), *Copy*(1), *Rotate3D*(2), *Move*(1);	7
Coordenadas exactas:	9	9
Coordenadas aproximadas:	5	5
Medidas:	4	4
Consultas:	Conversión de unidades (2). Calcular radio de revolución (1).	3
		28

Operaciones requeridas por la función.

Tipo de operación	**Cantidad de operaciones**	**Total**
Comandos:	*Codo*(1).	1
Coordenadas exactas:	2	2
Medidas:	3	3
Elecciones:	3	3
		9

Tabla III.3.2 Operaciones sustituidas por la función *Codo*.

Función *YeCDCRect*.

Operaciones originales de AutoCAD.

Tipo de operación	**Cantidad de operaciones**	**Total**
Comandos:	*Pline*(2), *Box*(5), *Extrude*(1), *Rotate3D*(2), *Move*(3);	13
Coordenadas exactas:	28	28
Coordenadas aproximadas:	12	12
Medidas:	10	10
Consultas:	Conversión de unidades (5). Calcular radio de giro (2).Medición de distancia (7).	14
Elecciones:	8	8
		85

Operaciones requeridas por la función.

Tipo de operación	**Cantidad de operaciones**	**Total**
Comandos:	*YeCDCRect* (1).	1
Coordenadas exactas:	2	2
Medidas:	10	10
Elecciones:	6	6
		19

Tabla III.3.3 Operaciones sustituidas por la función *YeCDCRect*.

Se tratará ahora con las funciones que sirven para simplificar el trabajo del usuario en la creación del modelo tridimensional de un diseño específico, así como de llevar la contabilización e identificación de las piezas utilizadas en el diseño. Como las operaciones en estos casos se realizan sobre diferentes objetos gráficos, la cantidad de operaciones requeridas para realizar una tarea varía. Por lo anteriormente mencionado, se utiliza en la cuenta de operaciones una notación con la forma *n-m*, donde *n* es el mínimo número de operaciones requeridas y *m*, representa al máximo.

Función *VerObjeto*.

Al utilizar este nuevo comando, se sustituyen operaciones de consulta de datos sobre los objetos gráficos, también se sustituyen operaciones de conversión de medidas, pues los datos se presentan con la opción de cambiar de unidades.

Operaciones originales de AutoCAD por objeto.

Tipo de operación	Cantidad de operaciones	Total
Consultas:	Medir longitud (2-13), Conversión de unidades (2-13) Calculo de área (1).	5-27
		5-27

Operaciones requeridas por la función.

Tipo de operación	Cantidad de operaciones	Total
Comandos:	*VerObjetos*(1).	1
Coordenadas aproximadas:	1	1
		2

Tabla III.3.4 Operaciones sustituidas por la función *VerObjeto*.

Función *CambiarMedidas*.

Esta función se tiene las variantes de que sustituye a las operaciones de crear un nuevo dibujo con los comandos originales de AutoCAD, o bien a las operaciones de crear un nuevo dibujo utilizando las nuevas librerías; ambos casos aparecen en la siguiente tabla.

Número de operaciones originales de AutoCAD requeridas	Número de operaciones de las librerías requeridas	Cantidad de operaciones requeridas por la función
19-28	4-9	2-5

Tabla III.3.5 Operaciones sustituidas por la función *CambiarMedidas*.

Función *DetectarOrigen*.

Cuando el usuario escoge la opción "*Seleccionar objeto base*" se sustituyen las operaciones de acuerdo a la siguiente tabla.

Operaciones originales de AutoCAD por objeto.

Tipo de operación	Cantidad de operaciones	Total
Coordenadas exactas:	2	2
Medidas:	1-2	1-2
		3-4

Operaciones requeridas por la función.

Tipo de operación	Cantidad de operaciones	Total
Coordenadas aproximadas:	1	1
		1

Tabla III.3.6 Operaciones sustituidas por la función *DetectarOrigen*.

Función *DibujarMedidas*.

Al dibujar las medidas de los objetos gráficos que representan ductos de manera automática se sustituye a las siguientes operaciones:

Operaciones originales de AutoCAD por objeto.

Tipo de operación	Cantidad de operaciones	Total
Comandos:	*Text* (1)	1
Coordenadas exactas:	2	2
Medidas:	1	2
Textos:	1	1
Consultas:	Medir longitud (1-2), Conversión de unidades (1-2).	2-4
Elecciones:	1	1
		7-9

Operaciones requeridas por la función para la totalidad de objetos seleccionados.

Tipo de operación	Cantidad de operaciones	Total
Comandos:	*DibujarMedidas* (1).	1
Coordenadas aproximadas:	2	2
Elecciones:	1	1
		4

Tabla III.3.7 Operaciones sustituidas por la función *DibujarMedidas*.

Se recalca el hecho de que las cantidades presentadas en la tabla anterior para los comandos originales de AutoCAD, corresponden a cada objeto sobre cual se va a operar; las cantidades para las librerías HVAC corresponden a la totalidad de los objetos que seleccione el usuario.

Función *GenerarInforme*.

La generación automática de un informe de los elementos utilizados, implica principalmente un ahorro de consultas de datos, así como de texto escrito. Se considera en la tabla un informe con todas las opciones que suministra el programa.

Operaciones originales de AutoCAD por objeto.

Tipo de operación	Cantidad de operaciones	Total
Comandos:	*Text* (4)	1
Coordenadas exactas:	8	2
Medidas:	4	2
Textos:	4	1
Consultas:	Medir longitud (2-7), Conversión de unidades (2-7), Cálculo de área(1).	5-15
Elecciones:	4	4
		15-29

Operaciones requeridas por la función para la totalidad de objetos seleccionados.

Tipo de operación	Cantidad de operaciones	Total
Comandos:	*GenerarInforme* (1).	1
Coordenadas Exactas:	1	1
Medidas:	1	1
Textos:	1	1
Elecciones:	6	6
		10

Tabla III.3.8 Operaciones sustituidas por la función *GenerarInforme*.

Función *DibujarNumeración*.

Esta nuevo comando, requiere una llamada previa de la función *GenerarInforme*, al asignar un número a cada objeto, se facilita su identificación, por lo que con esta función se ahorran operaciones de consulta, texto y dibujo.

Operaciones originales de AutoCAD por objeto.

Tipo de operación	Cantidad de operaciones	Total
Comandos:	*Text* (1), *Circle*(1), *Line*(1)	3
Coordenadas exactas:	5	5
Medidas:	2	2
Textos:	1	1
Elecciones:	1	1
		12

Operaciones requeridas por la función para la totalidad de objetos seleccionados.

Tipo de operación	Cantidad de operaciones	Total
Comandos:	*DibujarNumeracion* (1).	1
Coordenadas aproximadas:	2	2
Medidas:	2	2
		5

Tabla III.3.9 Operaciones sustituidas por la función *DibujarNumeracion*.

Función *GirarObjeto*.

Aquí solo existe una pequeña mejora con respecto al comando original de AutoCAD *Rotate3D*, se incluye porque el girar un objeto en el espacio tridimensional es una operación muy común dentro de un diseño, queda a elección del usuario el utilizar o no el nuevo comando.

Operaciones originales de AutoCAD por objeto.

Tipo de operación	**Cantidad de operaciones**	**Total**
Comandos:	*Rotate3D*(1)	1
Coordenadas exactas:	2	2
Coordenadas aproximadas:	1	1
Medidas:	1	1
		5

Operaciones requeridas por la función.

Tipo de operación	**Cantidad de operaciones**	**Total**
Comandos:	*GirarObj*(1)	1
Coordenadas aproximadas:	1	1
Medidas:	1	1
		3

Tabla III.3.10 Operaciones sustituidas por la función *GirarObjeto*.

Las desventajas que aparecen, a cambio de ahorrar dos operaciones, son que la nueva función no se puede aplicar a todo tipo de objetos, sino sólo a los que han sido dibujados automáticamente (ya que el programa registra estos objetos); además de que únicamente se realiza la rotación a lo largo del eje del objeto.

Función *LlegarA*.

Esta función es de gran ayuda al realizar un diseño tridimensional, tanto por la cantidad de operaciones a las que sustituye como por la frecuencia con que se requiere cambiar las dimensiones de un objeto o extender una línea de flujo de aire hacia un punto dado. En la mayoría de los casos, al no usar esta función, se tendrá que crear un nuevo objeto si se quiere cambiar su longitud. En el caso de que la extensión sea hacia un punto que requiera cambio de dirección se tiene un ahorro mayor de operaciones.

Operaciones sustituidas, caso sin cambio de dirección.		
Operaciones originales de AutoCAD requeridas	Operaciones de las librerías requeridas	Cantidad de operaciones requeridas por la función
19	4	3
Operaciones sustituidas, caso con cambio de dirección.		
Operaciones originales de AutoCAD requeridas	Operaciones de las librerías requeridas	Cantidad de operaciones requeridas por la función
66	17	5

Tabla III.3.11 Operaciones sustituidas por la función *LlegarA*.

Cómo se podrá observar, el número de operaciones requeridas por el usuario se reduce en forma considerable en cada caso en el que se aplican las librerías creadas. Con esto se comprueba la eficiencia de las librerías en aplicaciones HVAC, en las que al hacer uso de ellas en conjunto con los comandos originales de AutCAD, los cuales no se pretende sustituir, sino complementar, se logra cumplir con el objetivo propuesto para esta tesis.

En la figura inferior se puede apreciar una aplicación de las librerías en conjunto con las características de AutoCAD que consiste en la creación de un modelo sencillo; esta figura se presenta con perspectiva isométrica y render tipo *Gouraud Shaded*.

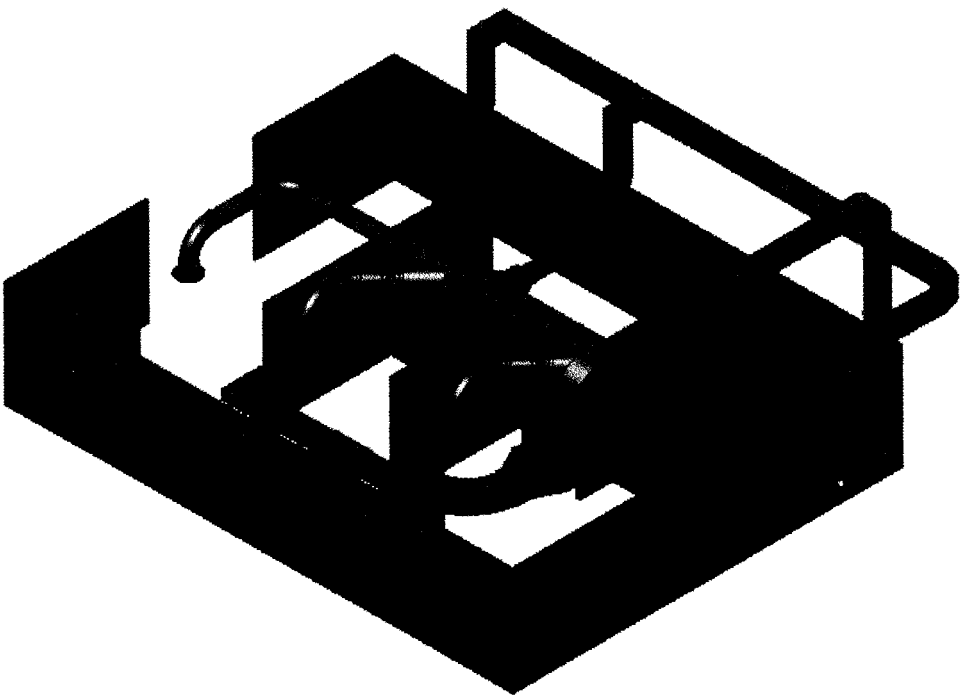


Figura III.3.1 Ejemplo de aplicación de las librerías HVAC.

CONCLUSIONES.

Observamos que al aplicar técnicas de programación sobre el entorno CAD utilizado como base, AutoCAD, para el área de interés de este trabajo, ingeniería HVAC; se consigue realizar los dibujos correspondientes a un determinado diseño, de una manera mucho más rápida que utilizando los comandos originales de AutoCAD para realizar los dibujos en base a figuras primitivas de propósito general.

Al representar los elementos de un diseño de manera tridimensional se logró un nivel de detalle bastante aceptable en los dibujos, que no se limitan a una simple representación simbólica, sino que permiten apreciar una vista de lo que será la realización física de un proyecto, consumiendo sólo una fracción del tiempo que tomaría realizar el dibujo sin la ayuda de las librerías.

Se tiene una reducción de los errores causados por el factor humano, ya que la mayor parte de un dibujo técnico del tipo aquí tratado, así como la contabilización de materiales y descripción de los elementos del dibujo se consiguen mediante automatización; si se parte desde un buen diseño, el error humano prácticamente desaparece.

Cuando se produce un cambio dentro de un diseño, con la utilización de la librerías gráficas creadas resulta muy sencillo reflejar estos cambios en el dibujo y en los datos descriptivos del mismo.

El hecho de contar con el código fuente de las librerías permite posteriores adaptaciones de las herramientas de dibujo creadas, hacia una gran cantidad de necesidades que puedan aparecer en el futuro, que es algo que no se puede hacer con ningún software comercial orientado hacia la ingeniería HVAC hasta el momento.

BIBLIOGRAFÍA.

Información en formato impreso.

[B-1] Carrier Air Conditioning Company. 1990. Manual de aire acondicionado. Ed. Macrocombo Barcelona España. ISBN 84-267-0115-9.

[B-2] Cogollar José Luis. Domine Autocad 2002. Ed. Alfaomega RAMA. México 2002. ISBN 970-15-0825-4

[B-3] Finkelstein Ellen. Autocad 14 ¡Soluciones! Ed. Osborne McGraw-Hill España 1999. ISBN 84-481-2121-X.

[B-4] Hasser Norman B., La Salle Joseph P., Sullivan Joseph A. 1990. Análisis Matemático Volumen 2. Ed. Trillas. ISBN968-24-3882-9.

[B-5] Mayne John A. LISP. Primer lenguaje para programadores de computadoras. Ed. Megabyte. Noriega Editores México 1994. ISBN 968-18-5029-7

[B-6] Rebis Industrial Workgroup Software. 1992. Autoplant Piping and Plant Desing Software. Designer User Reference. V.11.5. CA USA.

[B-7] Sheet Metal and Air Conditioning Contractors' National Association Inc. SMACNA CAD Standard. Second Edition. Julio 2001. USA.

[B-8] Tajadura Z. J. Antonio, López F. Javier. 1999 Autocad 2000 Avanzado. Ed. McGraw-Hill. Madrid España. ISBN 84-481-2430-8.

Información en formato electrónico.

[E-1] Autodesk Inc. 2000. Autocad Visual Lisp and ActiveX Help.

Información en Internet.

[I-1] www.abvent.com
[I-2] www.alias.com
[I-3] www.arriscad.com
[I-4] www.autodesk.com
[I-5] www.avid.com
[I-6] www.caligari.com
[I-7] www.catia.com
[I-8] www.capvidia.be
[I-9] www.formz.com
[I-10] www.graphisoft.com
[I-11] www.imsisoft.com
[I-12] www.intellicad.org
[I-13] www.kubotekusa.com
[I-14] www.lisp.org
[I-15] www.lispworks.com
[I-16] www.mcneel.com
[I-17] www.nemetschek.net
[I-18] www.newtek.com
[I-19] www.openmind.de
[I-20] www.ptc.com
[I-21] www.schemers.org
[I-22] www.sidefx.com
[I-23] www.softimage.com
[I-24] www.ugsolution.com
[I-25] www.vx.com