



UNIVERSIDAD NACIONAL
AUTÓNOMA DE
MÉXICO

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

**PROGRAMA DE MAESTRÍA Y DOCTORADO EN
INGENIERÍA**

FACULTAD DE INGENIERÍA

**SISTEMA AUTOMATIZADO DE ENSAMBLE
CON VISIÓN ARTIFICIAL PARA CELDAS DE
MANUFACTURA FLEXIBLE**

TESIS

QUE PARA OPTAR POR EL GRADO DE:

MAESTRO EN INGENIERÍA

MECÁNICA-MECATRÓNICA

PRESENTA:

ARTURO IGLESIAS ZÁRATE

DIRECTOR DE TESIS:

DR. GENGIS KANHG TOLEDO RAMÍREZ

TUTOR:

DR. JESÚS MANUEL DORADOR GONZÁLEZ



Año 2010

Agradecimientos

Quiero agradecer de manera especial a mi director de tesis Dr. Gengis Kanhg Toledo Ramírez, quien sin conocerme confió en mí y me ayudó desde el inicio de este proyecto con sus consejos, recomendaciones y su vasta experiencia en el área de la ingeniería mecánica y específicamente en la disciplina de la visión artificial. Por el tiempo y entusiasmo que dedicaste al presente trabajo hasta verlo concluido, “Gracias Gengis”.

A todos mis maestros de la UNAM que durante el tiempo que duró el curso de maestría siempre se mostraron dispuestos a impartir sus conocimientos, colaborar desinteresadamente con nosotros en todo lo concerniente en lograr el éxito en nuestras materias y sobre todo porque siempre se mostraron además de ser excelentes profesores como verdaderos amigos. Gracias a todos ellos. De manera especial quiero agradecer al Dr. Jesús Manuel Dorador González, director de la maestría en Ingeniería Mecánica campo disciplinario de la Mecatrónica, por su paciencia, interés, dedicación y por el esfuerzo realizado para que todos los alumnos de la maestría aprovecháramos al máximo el tiempo, por ayudarnos a resolver de la mejor manera posible los problemas que de manera especial se nos fueron presentando a cada uno de nosotros en el trayecto de nuestros estudios. Agradezco también a la M.I. Itzel Flores, colaboradora del Dr. Dorador, quien se encargó de auxiliarnos en los múltiples y tediosos trámites administrativos del proceso de titulación.

A todo el personal docente y administrativo del ITSSLP,C, por su colaboración y apoyo al facilitarnos el uso de instalaciones, equipo y laboratorios. En especial quiero agradecer el apoyo recibido por el Ing. Rubén Loredó Limón, responsable de laboratorios del área de Mecatrónica del ITSSLP,C por su colaboración en la fase de implementación del sistema en lo concerniente al subsistema de Transporte y Manipulación, que con sus conocimientos, destreza y paciencia logramos la comunicación con el resto de los subsistemas. A mi compañero y amigo Ing. Carlos Francisco Leura González, por su colaboración en el diseño y fabricación de dos variantes de dedos para los dos modelos de efector final con los que cuenta el robot manipulador de la CMA del Tecnológico. Gracias Francisco.

A mi esposa Margarita, mis hijas Liliana, Laura y Lorena por su amor y comprensión por el tiempo que he dejado de estar con ellas dedicándolo a la realización de éste trabajo, muchas gracias, las Amo.

Finalmente y no por ello menos importante, a DIOS por permitirme ver terminados mis estudios y obtener el grado de Maestro en Ingeniería Mecatrónica.

Índice general

Resumen

Abstract

1. Introducción

1.1 Configuración de la celda del ITSSLP,C.	1
1.2 Sistemas de manufactura flexible.	5
1.3 Propuesta.	5
1.3.1 Objetivo.	5
1.3.2 Justificación.	6
1.3.3 Metodología.	6
1.4 Organización de la tesis.	7

2. Necesidades del usuario y determinación de especificaciones

2.1 Necesidades del usuario.	9
2.2 Especificaciones.	10

3. Diseño conceptual

3.1 Descripción general del proyecto.	13
3.1.1 Metodología IDEF0.	14
3.2 Análisis y establecimiento de funciones.	15
3.2.1 Descripción de cada subsistema.	17
3.2.2 Alternativas de solución.	18
3.2.2.1 Alternativas para el Subsistema Visión Artificial (SVA).	18
3.2.2.2 Alternativas para el Subsistema Programación de Procesos de Manufactura (SPPM).	20
3.2.2.3 Alternativas para el Subsistema Manejo de Piezas (SMP).	20
3.2.2.4 Alternativas para el Subsistema Estación de Maquinado (SEM).	20
3.2.3 Evaluación de alternativas.	20
3.2.3.1 Para SVA.	21
3.2.3.2 Para SPPM.	21
3.2.3.3 Para SMP.	21
3.3 Resultados en la evaluación de alternativas..	22

4. Diseño de configuración

4.1 Introducción.	23
4.2 Elementos del CMA que se utilizaron en el proyecto.	23
4.2.1 Estación de transporte.	23
4.2.2 Estación de visión.	23
4.2.3 Estación de manipulación y montaje.	24
4.2.4 Mesa de trabajo.	25
4.3 Imágenes capturadas por la cámara.	26
4.4 Coordenadas y matrices homogéneas.	28
4.4.1 Composición de matrices homogéneas.	31
4.5 Cálculo de la posición y orientación de una pieza.	31

5. Subsistema de visión artificial (SVA)	
5.1 Sistemas de visión artificial.	37
5.1.1 Descripción de cada nivel y etapa.	38
5.1.1.1 Visión de bajo nivel.	38
5.1.1.2 Técnicas de preproceso.	39
5.1.1.3 Visión de nivel intermedio.	40
5.1.1.4 Descripción.	41
5.1.1.5 Ángulo de inclinación de las piezas.	43
5.1.1.6 Visión de nivel alto.	45
5.1.1.7 Reconocimiento de patrones.	46
5.1.1.8 Reconocimiento usando clasificador de distancia mínima.	46
5.2 Implementación del SVA.	47
5.2.1 Adquisición de imagen.	47
5.2.2 Preproceso de imagen.	48
5.2.3 Segmentación de imagen..	49
5.2.4 Descripción de imagen.	50
5.2.5 Reconocimiento de patrones.	55
6. Subsistema de programación de procesos de manufactura (SPPM)	
6.1 Módulos que integran el SPPM.	59
6.1.1 Módulo de aprendizaje.	59
6.1.2 Módulo de creación de plantilla para ensamble.	61
6.1.3 Módulo proceso actual de ensamble.	63
6.1.4 Módulo de calibración.	64
6.1.5 Módulo ejecuta ensamble.	66
7. Subsistema de transporte y manipulación de piezas (SMP)	
7.1 Estación de transporte.	71
7.2 Estación de manipulación.	72
7.2.1 Estructura de un robot industrial.	72
7.2.2 Generalidades del robot Mitsubishi RV-2AJ.	74
7.3 Comunicación entre el robot y el subsistema SPPM..	77
8. Pruebas y resultados.	
8.1 Antecedentes.	79
8.2 Plantillas utilizadas en las pruebas.	80
8.3 Pruebas.	81
8.3.1 Plantilla uno.	82
8.3.2 Plantilla dos.	85
8.4 Resultados..	88
Conclusiones.	89
Trabajo futuro..	91
Apéndices	
A. Ejemplo sistemas de referencia.	95
B. Código fuente de funciones de los módulos SPPM.	99
C. Programa Melfa-Basic IV en operaciones tomar y dejar.	117
Bibliografía	119
Glosario	121

Índice de figuras

1.1 Centro de Manufactura Avanzada del Instituto Tecnológico Superior de San Luis Potosí, Capital..	2
1.2 Estación de distribución.	3
1.3 Estación de verificación.	3
1.4 Estación de manipulación.	3
1.5 Estación de procesamiento.	3
1.6 Estación de visión.	4
1.7 Estación de manipulación y montaje.	4
1.8 Estación de clasificación.	4
1.9 Estación de transporte.	4
1.10 Sistema de manufactura flexible asistido por visión artificial.	6
3.1 Árbol de funciones generales.	15
3.2 Función principal.	16
3.3 Líneas de flujo de información y subsistemas.	16
3.4 Robot Mitsubishi RV-2AJ.	20
4.1 Estaciones de trabajo del CMA que participan en el sistema.	24
4.2 Paralelismo entre los planos de imagen, mesa de trabajo y banda transportadora.	25
4.3 Mapeo de coordenadas de dispositivo en coordenadas del mundo real.	26
4.4 Estaciones de visión, transporte y de manipulación y ensamble.	27
4.5 Sistema coordinado base del robot (x0,y0,z0) y de la mesa de trabajo (x1,y1,z1).	28
4.6 Configuración en la estación de ensamble.	32
4.7 Sistema coordinado del efector final con sus dos giros.	33
4.8 El Ángulo γ entre las rectas AC y BC es el valor de alabeo buscado..	34
5.1 Representación esquemática de un sistema de visión genérico.	38
5.2 Arreglo bidimensional (M x N) de elementos de imagen.	39
5.3 Escena segmentada en cuatro regiones.	40
5.4 Distancia de un punto en la región a una recta.	44
5.5 Transformación del sistema referido en la imagen a otro bajo la cámara.	57
6.1 Implementación del SPPM fuera de línea.	60
6.2 Archivo de texto de patrones.	61
6.3 Ejecución de la función crea_plantilla () capturando una escena con tres patrones.	62
6.4 Proceso de asignación de orden de ensamblado de las piezas.	63
6.5 Piezas en la escena de trabajo para ensamblar una plantilla.	67
6.6 Datos de piezas localizadas y valores transmitidos al robot.	67
7.1 Estaciones de trabajo del CMA.	71
7.2 Panel de control alambrado para el sistema de transporte.	72
7.3 Articulaciones de un brazo robot en similitud a las de un brazo humano.	73
7.4 Partes y periféricos asociados al brazo robot Mitsubishi RV-2AJ.	74

7.5 Espacio de trabajo del robot Mitsubishi RV-2AJ.	76
8.1 Plantilla de ensamble uno.	79
8.2 Plantilla de ensamble dos.	80
8.3 Arreglo en memoria de los elementos que forman la plantilla uno.	81
8.4 Arreglo en memoria de los elementos que forman la plantilla dos.	81
8.5 Piezas en la escena para ensamblar la plantilla uno.	83
8.6 Información de escena proporcionada por la función analiza_escena().	84
8.7 Información para manipulación en el ensamble de la plantilla uno.	84
8.8 Dibujo digitalizado en ensamble de plantilla uno.	86
8.9 Escena en primera vuelta para ensamble de plantilla dos.	87
8.10 Escena en segunda vuelta para ensamble de plantilla dos..	87
8.11 Cálculo de los parámetros y el estado del ensamble de la plantilla dos..	88
A.1 Coordenadas de un punto respecto de dos sistemas diferentes.	95
A.2 Alineación de sistemas coordenados usando criterio uno.	96
A.3 Alineación de sistemas coordenados usando criterio dos.	97

Índice de tablas

3.1 Identificación y selección de principios de trabajo para cada función del sistema.	.	.	.	22
5.1 Correspondencia entre niveles y sus etapas en un sistema de visión artificial.	.	.	.	38
5.2 Selección de descriptores.	.	.	.	52
5.3 Patrones con sus descriptores seleccionados.	.	.	.	54
7.1 Características y especificaciones del robot Mitsubishi modelo RV-2AJ.	.	.	.	74
8.1 Piezas utilizadas en el proceso de ensamble de las plantillas uno y dos.	.	.	.	80
8.2 Valores de calibración y su justificación.	.	.	.	82

Sistema automatizado de ensamble con visión artificial para celdas de manufactura flexible

Tesis de Maestría

Resumen

Arturo Iglesias Zárate

Facultad de Ingeniería - Área Mecánica
Universidad Nacional Autónoma de México

El propósito del presente trabajo fue crear un sistema de cómputo que muestre de manera clara y objetiva el concepto fundamental de los procesos de manufactura flexible, mediante la programación y ejecución de procesos de ensamble. El sistema desarrollado incluye un subsistema de visión para reconocimiento y ubicación de piezas destinadas a ser ensambladas y que se encuentren aleatoriamente distribuidas sobre una mesa de trabajo montada sobre una charola que se desplaza a su vez sobre una banda transportadora. El subsistema de visión analiza la escena una vez que la charola se detiene frente a la estación de ensamble. Se trata de un sistema administrador y un subsistema de visión con interfaces a los subsistemas previamente existentes de transporte y de manipulación. El proyecto fue desarrollado para la celda de manufactura avanzada (CMA) del Instituto Tecnológico Superior de San Luis Potosí, Capital (ITSSLP,C).

El sistema determina para cada pieza en la escena su ubicación respecto del sistema coordinado del robot y en base a un plan de ensamble previamente programado, comunica al control del robot la información necesaria para tomar cada pieza de la escena y colocarla en su posición final de ensamble. El sistema se probó con dos procesos diferentes de ensamble o plantillas de ensamble. El tamaño aproximado de las piezas con las que puede trabajar el sistema varía entre 1 y 8 cm de largo y entre 1 y 3 cm de ancho. El sistema es capaz de reconocer de forma satisfactoria piezas previamente entrenadas en su módulo de aprendizaje. El sistema determina para cada pieza reconocida su posición espacial con un error aproximado de ± 0.33 mm, así como al ángulo que el efector final del manipulador debe alcanzar para lograr tomar la pieza. En esta tarea se lograron precisiones de $\pm 0.10^\circ$.

Las plantillas de ensamble se diseñan fácilmente mediante el sistema y se pueden modificar de manera simple editando el archivo de texto que las define. Es posible editar el nombre de una o varias piezas, modificar su valor de prioridad en el proceso de ensamble, cambiar cualquier valor ya sea de las coordenadas de su centro de masa o bien del ángulo de inclinación de la pieza, puede eliminar o añadir nuevas piezas al proceso. Todo ello con el fin de enriquecer y despertar en el lector el interés por los procesos de manufactura flexibles.

El resultado fue un sistema modular, fácil de operar y didáctico. El presente trabajo es además un ejemplo de la implementación de un sistema de manufactura flexible sobre un sistema de fabricación tradicional.

Automated assembly system with machine vision for flexible manufacturing cells

Master Thesis

Abstract

Arturo Iglesias Zárate

Faculty of Engineering - Mechanical Area
Universidad Nacional Autónoma de México

The purpose of this study was to create a computer system that clearly and objectively shows the fundamental concept of flexible manufacturing processes, through the programming and execution of assembly processes. The system developed includes a vision subsystem for recognition and location of parts to be assembled and which are randomly distributed on a work table set up on a tray which at the same time moves on a conveyor belt. The vision subsystem analyzes the scene once the platter stops in front of the assembly station. It is a system administrator and a vision subsystem with interfaces to preexisting subsystems of transportation and handling. The project was developed for the advanced manufacturing cell (AMC) of the Instituto Tecnológico Superior de San Luis Potosi, Capital (ITSSLP, C).

For each piece at the scene the system determines its location in regards to the robot's coordinate system and based on a previously scheduled assembly plan, it communicates the information required to the robot controller in order to pick up each piece of the scene and place it in its final assembly position. The system was tested with two different processes of assembly or assembly templates. The approximate size of the pieces which you can work the system with varies between 1 and 8 cm long and between 1 and 3 cm wide. The system is capable of satisfactorily recognizing previously trained pieces in its learning module. The system determines for each acknowledged piece its spatial position with an approximate error of ± 0.33 mm, as well as the angle that the end effector of the manipulator must reach to be able to take part. During this task, accuracies of $\pm 0.10^\circ$ were achieved.

The assembly templates are easily designed through the system and can be modified in simple manner by editing the text file that defines them. It is possible to edit the name of one or more pieces, modify their priority value in the assembly process, change any value whether it's the coordinates of its center of mass or the angle of inclination of the piece, you can delete or add new parts to the process. All of this to enrich and awaken the reader's interest in flexible manufacturing processes

The result was an easy to operate and didactic modular system. This current task is also an example of implementing a flexible manufacturing system over a traditional manufacturing system.

Capítulo 1

Introducción

El mercado globalizado de productos y procesos industriales cada vez más competido, exigente y dinámico, demanda productos y procesos productivos con un alto grado de automatización, donde se sustituye la mano de obra humana por máquinas cada vez más precisas, funcionales y dotadas con un mayor grado de inteligencia, para llevar a cabo procesos que requieren flexibilidad, adaptabilidad, ambientes dinámicos que cumplan con los estándares de calidad más exigentes y que además sean económicos.

Por otra parte, la educación es uno de los recursos más poderosos para lograr un desarrollo integral humano y una mejora de los estándares de confort y niveles de vida en general, al satisfacer necesidades básicas como vivienda, educación, empleo bien remunerado, seguridad y paz social que la población requiere para vivir dignamente y establecer niveles de desarrollo y crecimiento integral. Al lograr que la población mexicana tenga un buen nivel educativo y cultural automáticamente se convierte en terreno fértil para continuar fomentando valores en el núcleo social más importante como lo es el seno familiar. Valores que nos impidan caer en vicios, corrupción, delincuencia, que llevan a crear ambientes de inseguridad y elevan los índices delictivos a niveles difíciles de combatir. Un pueblo con un buen nivel educativo y cultural donde se ha mejorado el nivel de vida de sus habitantes tiene muchas posibilidades de lograr altos niveles de competitividad. Estos niveles de competitividad nos permitirán crear e implantar tecnología propia que nos permita satisfacer las necesidades del mercado nacional y además competir exitosamente en los mercados internacionales.

El Instituto Tecnológico Superior de San Luis Potosí, Capital (ITSSLP,C), al igual que muchas universidades y centros educativos en todo el país forman profesionales a nivel licenciatura en áreas de ingeniería industrial, mecatrónica y sistemas computacionales y a nivel maestría en mecatrónica y ciencias de la educación.

El ITSSLP,C como otras instituciones educativas públicas y privadas, cuenta con un centro de manufactura avanzada (CMA). Este CMA tiene la desventaja que los sistemas tienen una arquitectura muy cerrada, limitando así a los estudiantes a no tener libertad de modificar, reconfigurar o proponer nuevas tareas y procesos que los estrictamente proporcionados por los fabricantes y proveedores de los equipos, fomentando así la dependencia tecnológica.

Este trabajo presenta un sistema flexible, modular, fácilmente programable, didáctico y sobre todo destinado a motivar a los alumnos a aprender y a despertar en ellos el interés por los procesos de manufactura flexible.

1.1 Configuración de la celda del Instituto Tecnológico Superior de San Luis Potosí ITSSLP,C.-

Al centro de manufactura avanzada (CMA) del ITSSLP,C, lo integran nueve áreas enlazadas a través de un sistema de transporte [1]. Éstas tienen el propósito de mostrar las diferentes etapas en la fabricación de un cilindro de carrera corta. Esta celda forma la base para la información técnica en general, proporciona

la plataforma perfecta para analizar, comprender y dominar la interacción de la mecánica, la ingeniería eléctrica, la tecnología de control y las interfaces de comunicación necesarias en la formación de profesionistas en los perfiles de interés del Tecnológico, véase Fig. 1.1

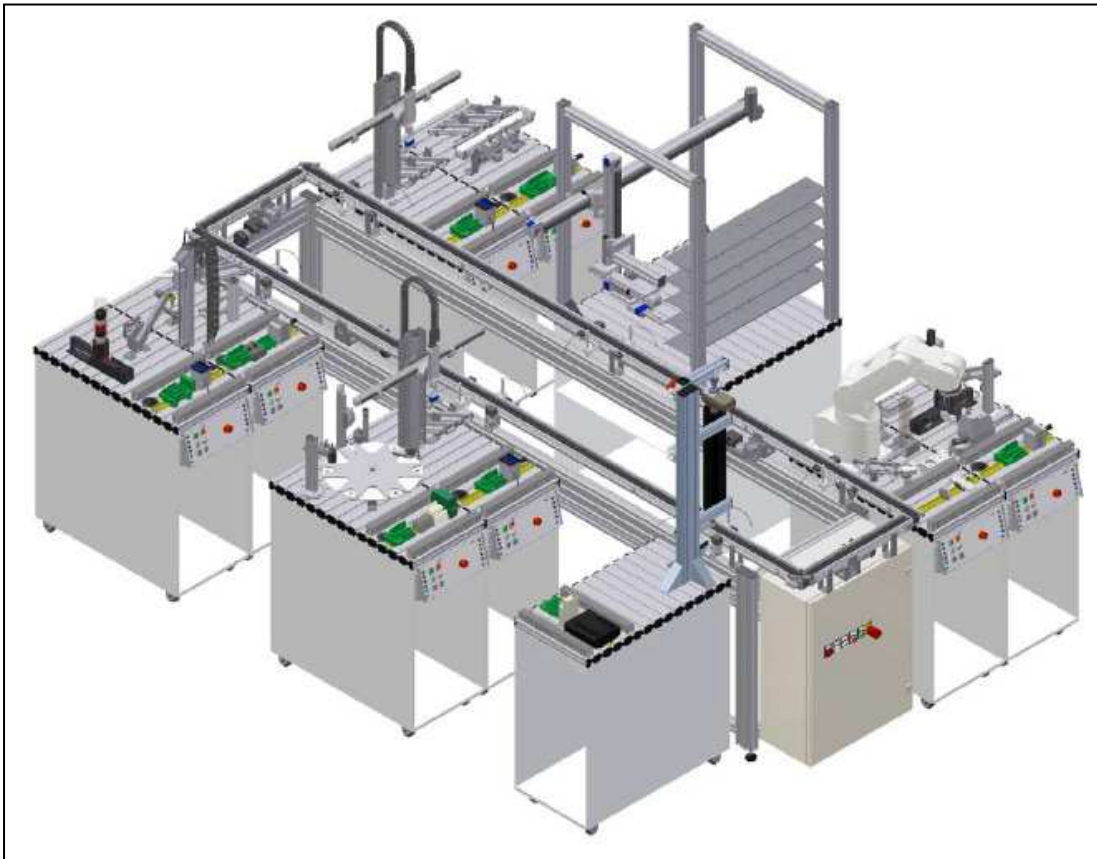


Figura 1.1.- Centro de Manufactura Avanzada del ITSSLP,C.

Estaciones de trabajo de la celda de manufactura avanzada del ITSSLP,C:

Distribución.- Esta estación introduce las piezas a la celda y las desplaza a la siguiente estación que es la de verificación, véase Fig.1.2

Verificación.- Las piezas son transportadas según su clasificación, esto se realiza por medio de sensores capaces de distinguir un color de otro en base al espectro luminoso de la luz que los cuerpos reflejan. Un sensor del tipo ultrasónico mide la altura de la pieza y las defectuosas son rechazadas y separadas del proceso a través de una rampa de desvío por medio de un pistón, véase Fig.1.3

Manipulación.- Está equipada con un manipulador cartesiano de dos grados de libertad con movimientos horizontal y vertical. Aquí las piezas son detectadas por medio de un sensor de presencia y transportadas a la siguiente estación, véase Fig.1.4

Procesamiento.- En esta estación las piezas son verificadas y taladradas en una mesa giratoria, véase Fig.1.5

Visión.- La estación de visión es utilizada con fines de inspección para detectar piezas defectuosas en tareas de aseguramiento de la calidad para control de la producción, véase Fig.1.6

Manipulación y montaje.- Un robot Mitsubishi modelo RV-2AJ, que es un robot industrial tipo PUMA de cinco grados de libertad [2], con alcance máximo de 65 cm y capacidad de carga de 2.5 kg, une las diferentes piezas obteniendo el cilindro montado, véase Fig.1.7

Clasificación.- Esta estación clasifica las piezas utilizando sensores de presencia instalados en tres rampas, de esta manera puede detectar si la rampa esta llena, véase Fig. 1.8

Transporte.- Integrada por una banda transportadora encargada de transportar las piezas de trabajo a lo largo de las nueve estaciones que integran la celda durante su proceso de fabricación y que habrán de convertirse en el producto terminado, cilindro de carrera corta en este caso, véase Fig.1.9

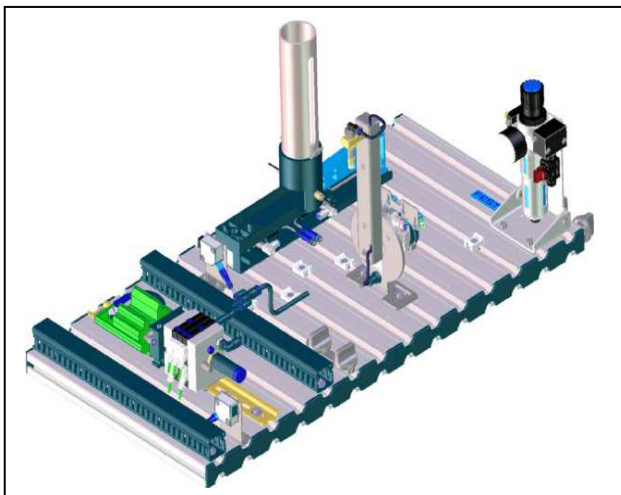


Figura 1.2 Distribución.

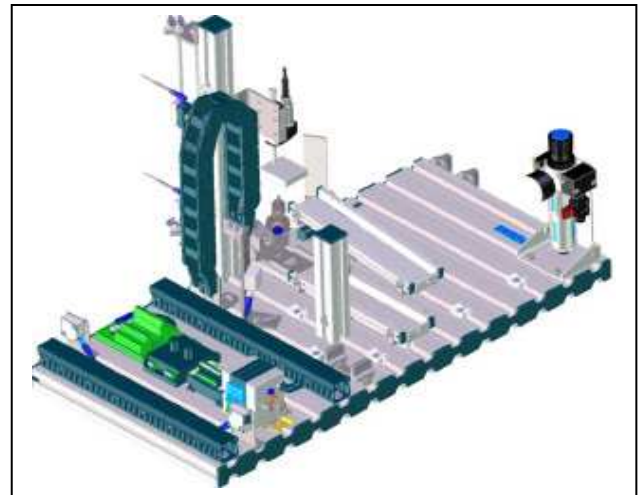


Figura 1.3 Verificación.



Figura 1.4 Manipulación.

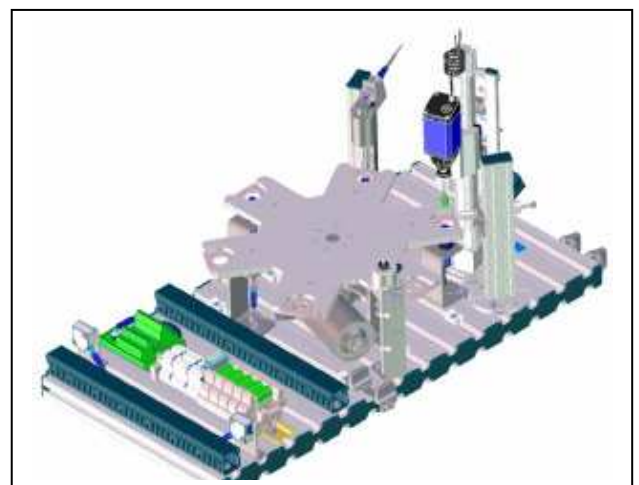


Figura 1.5. Procesamiento.



Figura 1.6 Visión.



Figura 1.7 Manipulación y montaje.

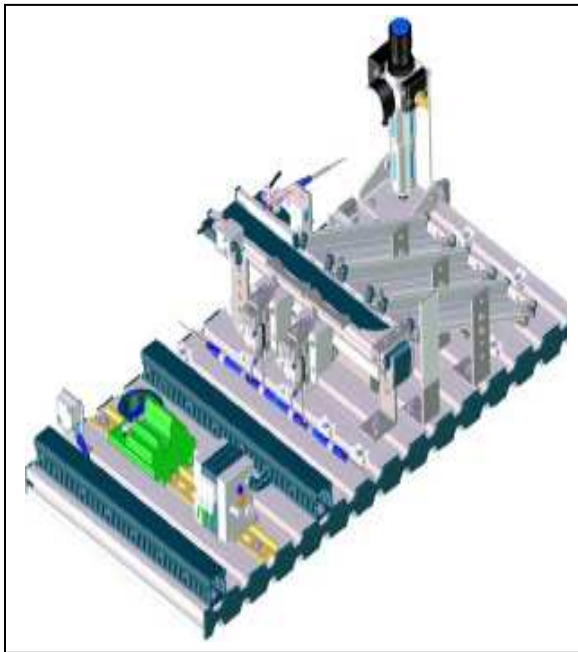


Figura 1.8 Clasificación.



Figura 1.9 Transporte.

Además del hardware y software que conforman la celda, CMA cuenta con una sección para capacitación con 15 computadoras personales conectadas en red, pizarrón electrónico y seis módulos de capacitación para prácticas de neumática e hidráulica industrial.

Además del CMA, otra área con la que cuenta el ITSSLP,C es el laboratorio de PLC (controlador lógico programable) con 15 mesas, cada una provista de una PC y un tablero de trabajo con un PLC de la marca Allen Bradley integrado para prácticas.

1.2 Sistemas de manufactura flexible.- Un sistema de manufactura flexible es un sistema de fabricación integrado por máquinas-herramientas preferentemente del tipo CNC (Control numérico computarizado) enlazadas mediante un sistema de transporte (i.e. banda) y otro de manejo de materiales (i.e. robot manipulador). Puede contener además un sistema de visión para reconocimiento y localización de piezas para efectos de manipulación. Un FMS tiene la posibilidad de realizar diversas tareas como maquinado y ensamble a diferentes piezas o productos sin la necesidad de cambiar o reconfigurar los equipos del sistema [3,4].

Estos sistemas pueden ser casi tan flexibles y de mayor complejidad que un taller de trabajo y al mismo tiempo tener la capacidad de alcanzar la eficacia de una línea de ensamble bien balanceada.

Los FMS pueden disponer de un sistema de manejo de materiales automatizado que transporta las piezas de una máquina a otra hacia dentro y fuera del sistema. Puede tratarse de vehículos guiados automáticamente (AGV por sus siglas en inglés) las cuales son plataformas dotadas de un sistema de locomoción a base de ruedas y cuyo sistema de control utiliza técnicas de radiofrecuencia para guiar el vehículo siguiendo un cable oculto en el piso, el cual marca la trayectoria a seguir por el AGV.

El empleo de los FMS permite flexibilidad productiva, gestión en tiempo real y acelerado nivel de automatización general, así que una celda en línea es en resumen aceptar el ingreso de materia prima y obtener productos terminados.

Los temas de control de un FMS involucran el monitoreo en tiempo real, para asegurarse de que el sistema se desempeñe conforme a lo programado y que se ha logrado la producción esperada.

1.3.- Propuesta.- La CMA del ITSSLP,C constituye un ejemplo de una planta productiva totalmente automatizada a escala didáctica. Si se desea manufacturar otro producto diferente al "*cilindro de carrera corta*", se tendrían que hacer cambios estructurales y de control significativos en cada una de las estaciones que la conforman, aún si los cambios se realizaran en el mismo producto. Un concepto aún más innovador y que llega a impactar más en el estudiante de ingeniería es el de los sistemas de manufactura flexibles (SMF o FMS por sus siglas en inglés). A continuación se presentan las características generales del proyecto a desarrollar.

1.3.1 Objetivo.- Diseño e implementación de un sistema automatizado de ensamble de piezas con visión artificial para celdas de manufactura flexible. El proyecto se desarrollará para la CMA del ITSSLP,C y utilizará algunas estaciones de trabajo de la actual celda, así como software que será desarrollado principalmente para los subsistemas de visión y para el de programación de procesos de manufactura.

1.3.2 Justificación.- El presente trabajo pretende implementar un sistema que muestre de manera simple a los alumnos del ITSSLP,C los conceptos fundamentales de los procesos de manufactura flexible, tales como las operaciones de manipulación “tomar y dejar” con robots, operaciones de maquinado más elementales tales como taladrado, redondeo de esquinas, ranurado, etc.. Estas operaciones se programan y aplican sobre piezas planas de geometría regular que al ser manipuladas por el robot con el que cuenta la celda, forman un ensamble o producto final. La característica sobresaliente de éste sistema es que cuenta con un subsistema de visión propio, que le permite reconocer las piezas que se encuentran libremente distribuidas en una mesa de trabajo, así como determinar su ubicación exacta (centro de masa y orientación de la pieza) y referir finalmente dicha ubicación respecto al sistema coordinado base del robot para efectos de manipulación. El contar con un subsistema de visión artificial le permite al sistema obtener información suficiente para decidir si ha de mover la pieza hacia su lugar final de ensamble o deberá llevarla antes a una estación de maquinado CNC donde se le habrá de aplicar alguna operación previa.

Éste proyecto formará parte del CMA y servirá a alumnos y profesores que cursen e impartan conocimientos en materias de ingeniería industrial, automatización, manufactura, celdas flexibles, control digital, etc. No pretende reemplazar a los sistemas existentes en la celda sino más bien complementarlos y se construirá principalmente con recursos (software y hardware) propios del CMA del ITSSLP,C.

La Figura 1.10 muestra de manera simplificada los elementos que conformarán el presente trabajo los cuales serán presentados de manera amplia en capítulos posteriores.

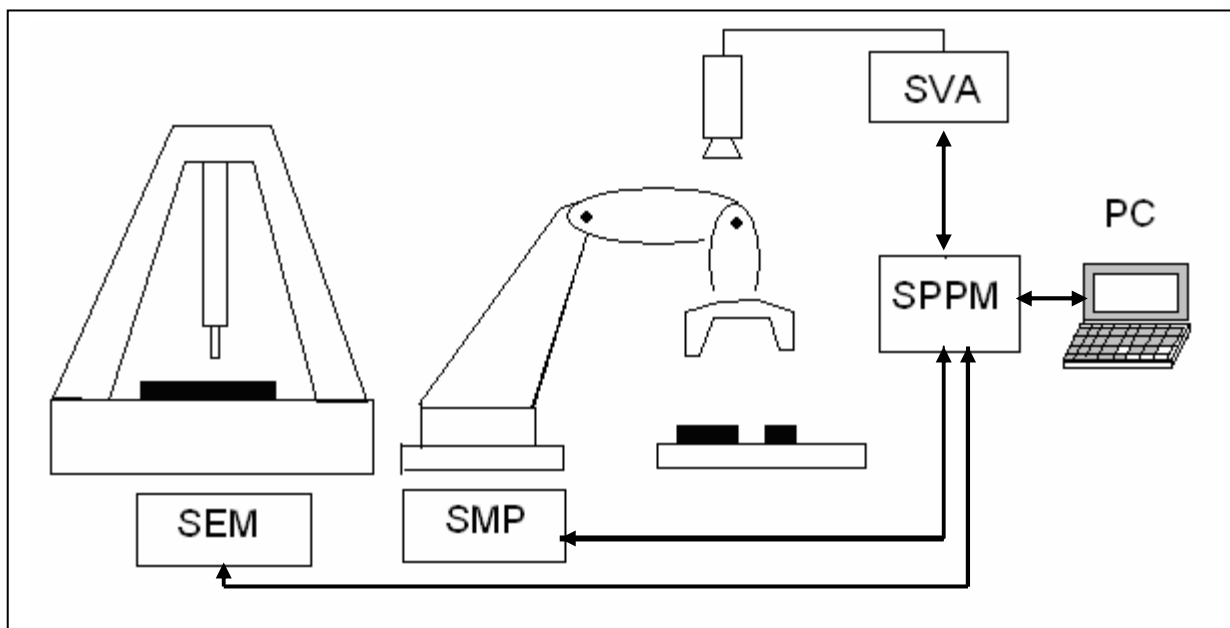


Figura 1.10 Sistema de manufactura flexible asistido por visión artificial.

1.3.3 Metodología.- El sistema completo está conformado por un subsistema de visión artificial (SVA), un subsistema de manipulación de piezas (SMP), un subsistema estación de maquinado (SEM), y un subsistema de programación de procesos de

manufactura (SPPM). El sistema, sus componentes principales así como su interacción se muestran en la Fig.1.10.

El alcance del presente trabajo no incluye el subsistema SEM, ya que actualmente el CMA del ITSSLP,C no cuenta con equipos CNC.

Partimos del diseño conceptual presentado en el Cap. 3, en él se muestra esquemáticamente la función principal, subfunciones así como su interrelación. Para cada subfunción se evalúan las diversas alternativas en su implementación, tanto en software como en hardware, considerando la restricción vista en el apartado anterior que considera que para el desarrollo del presente trabajo se deberá hacer preferentemente uso de los recursos existentes en el CMA del ITSSLP,C. El SVA reconoce las piezas que se localizan en la escena y determina su ubicación (centro de masa y ángulo de inclinación) respecto al sistema coordinado que se localiza sobre la mesa de trabajo y que abarca el área capturada por la cámara, mediante transformaciones homogéneas se obtienen las coordenadas 3D de las piezas referidas respecto al sistema base del robot para efectos de manipulación. De forma general podemos decir que el SPPM ejecuta de manera continua en el CMA un proceso de ensamblado, la mesa de trabajo contiene una serie de piezas dispuestas en forma aleatoria que serán ensambladas, se coloca bajo la cámara, el SPPM llama a trabajar al SVA, de esta forma, se toma y analiza la escena, se determina que piezas participan en el proceso así como su ubicación exacta. Esta información es recibida por el SPPM el cual calcula los valores requeridos por el efector final del robot y los transmite al SMP para “tomar” las piezas de la escena y “dejarlas” en su posición final de ensamble, obteniendo así el producto terminado.

1.4 Organización de la tesis.- El Cap. 1 inicia con una breve introducción de lo que es el CMA del ITSSLP,C, lugar donde se realizó la tesis, en él se muestran características y equipo con el que cuenta, ya que parte de ése equipo se utilizó en el proyecto. Posteriormente una pequeña introducción a los sistemas de manufactura flexible, para dar paso de lleno a la propuesta presentada en el Apartado 1.3. En el Cap. 2 se analizan las necesidades del usuario y se establecen las especificaciones a cumplir en el presente trabajo. En el Cap. 3 se aborda el diseño conceptual del sistema haciendo uso de la metodología IDEF0. Mediante este esquema se representa gráficamente la función principal, sus subfunciones y la interrelación que guardan entre ellas, indicando además los elementos de entrada, controles, medios y elementos de salida. Se evalúan para cada subfunción, las diversas opciones para su implementación y se determina cual y porqué es la más conveniente. En el Cap. 4 trataremos el tema del diseño de configuración que es la disposición física que presentan los diferentes elementos que componen el sistema así como su correcta y adecuada referencia espacial y geométrica. El Cap. 5 lo dedicaremos exclusivamente al SVA, se dará una breve explicación de qué es y en que consiste un sistema de visión genérico para dar paso al diseño, desarrollo e implementación del nuestro en base a la reutilización de librerías de funciones de procesamiento de imagen digital ya programadas en lenguaje “C” en ambiente MatLab. El Cap. 6 trata sobre el SPPM que es el subsistema que programa y controla los procesos de manufactura flexible. El subsistema de transporte y manipulación de piezas se aborda en el Cap. 7. Finalmente el Cap. 8 muestra las pruebas realizadas al sistema así como los resultados obtenidos. Para terminar mostramos las conclusiones del proyecto así como los trabajos a futuro.

Capítulo 2

Necesidades del usuario y determinación de especificaciones

La falta de flexibilidad en los procesos que se llevan a cabo en el centro de manufactura avanzada (CMA) quedó expuesto en el capítulo anterior cuando presentamos la celda como “un sistema de transporte con el propósito de mostrar las diferentes etapas en la fabricación de un cilindro de carrera corta”. De tal manera que si queremos utilizarla para crear otra pieza diferente, tendríamos que realizar modificaciones en la estructura de algunas o todas las estaciones de trabajo. Es de reconocer que la manera en que está constituido el CMA permite mostrar a los alumnos un claro ejemplo de un proceso industrial totalmente automatizado para fabricación en línea de un solo producto, sin embargo esto contrapone al concepto de manufactura flexible que queremos mostrar en nuestro proyecto, en los cuales se pueden fabricar piezas totalmente diferentes con mínimos cambios en sus estaciones de trabajo.

Algunas limitaciones de la actual celda de manufactura son:

- El tipo de pieza a fabricar tiene que ser siempre la misma.
- Las piezas se ubican en lugares fijos sobre una placa de trabajo provista de orificios que sirven de contra formas para la sujeción de las piezas, por lo que no se permite aleatoriedad en su posicionamiento.
- El sistema de iluminación es de ambiente rígido, ya que no soporta cambios en intensidad, sombras o reflejos muy comunes en ambientes industriales.
- Las características relacionadas con el aprendizaje y reconocimiento de formas libres, es nulo o bien muy limitado.

2.1 Necesidades del usuario.- Si consideramos que los usuarios del sistema a desarrollar son los estudiantes y profesores del CMA del ITSSLP,C, éstos requieren un sistema que sea:

- Flexible. El sistema actual es sólo una celda de manufactura didáctica, sin embargo es rígida ya que sólo es posible observar la fabricación de un único producto. Los usuarios requieren una CMF en la cual tengan la flexibilidad en procesos y productos de manera que puedan realizar diversas prácticas y comprender mejor los procesos de manufactura flexible.
- Modular. El usuario podrá de manera sencilla añadir o retirar una o varias estaciones de trabajo para desarrollar prácticas específicas, caso concreto para nuestro proyecto sólo se utilizó la estación de transporte, estación de manipulación y del sistema de visión que forma parte de la instalación original de

la CMA. Otro ejemplo lo tendremos cuando se integren al CMA los equipos de control numérico computarizado (CNC).

- Fácilmente programable. Mediante interfaces gráficas del usuario (GUI de sus siglas en inglés), el usuario podrá modificar la configuración del sistema de ensamblado aún estando operando en línea, de modo que podrá ensamblar un producto diferente si así lo desea o simplemente modificar la ubicación de la plantilla de ensamblado del producto actual.
- Didáctico. Con ejemplos sencillos se pretende motivar a los alumnos a aprender y despertar en ellos el interés por los procesos de manufactura flexible.

2.2 Especificaciones.- El sistema deberá ser diseñado tomando en cuenta las siguientes especificaciones:

- El sistema contará con un subsistema de visión artificial (SVA) para imágenes bidimensionales (2D), capaz de reconocer objetos planos de forma libre y alargada de aproximadamente 1 a 8 cm, con anchos entre 0.50 a 3 cm (dependiendo del tipo de efector final a utilizar, contamos con dos variantes), no deberán estar amontonados ni siquiera tocarse uno con otro, no hay límite en cuanto al número de piezas que podrán aparecer en la escena, pero sí debemos tomar en cuenta que mientras mayor sea la cantidad de piezas que aparecen en la escena, mayor también será el tiempo de procesamiento que se requerirá para su análisis, además el robot debe tener suficiente espacio para tomar las piezas disminuyendo el riesgo de colisión. Se utilizará un sistema de iluminación con luz posterior con el fin de producir imágenes con alto contraste que el sistema convierte en imágenes binarias que son imágenes con sólo dos tonos, blanco para los objetos y negro para el fondo. El SVA será capaz de reconocer cada una de las piezas en la escena en base a una serie de patrones almacenados, calculará además su posición y orientación y comunicará esta información al subsistema coordinador que es el subsistema programador de procesos de manufactura (SPPM)
- El sistema deberá ser capaz de crear, modificar y ejecutar diferentes procesos de manufactura flexible (ensamble y/o maquinado) sobre las piezas reconocidas en el área de trabajo. Esto permite reprogramar tareas, tales como quitar o añadir piezas a un ensamble determinado, cambiar la secuencia de ensamblado, modificar la posición y orientación de la plantilla de ensamble o bien modificar la cantidad y/o el tipo de operaciones de maquinado que han de aplicarse a determinadas piezas antes de iniciar un proceso de ensamble.
- El sistema moverá piezas mediante el uso de un manipulador automático programable (robot) ejecutando rutinas “tomar y dejar” a las piezas que formen parte del proceso de ensamble previamente programado.

- El sistema podrá modificar su entorno de trabajo al ser capaz de añadir o retirar estaciones de trabajo, ya sean manipuladores, centros de maquinado, alimentadores de piezas, sensores, etc.
- El sistema deberá ser didáctico ya que está dirigido a estudiantes y profesores de materias afines a los procesos de manufactura, celdas flexibles de manufactura, automatización industrial, robótica, sensores, control digital y otras. Específicamente por didáctico se entiende que el sistema debe permitir la creación de diferentes y variadas prácticas académicas como pueden ser programar y ejecutar una serie de ensambles diferentes utilizando las mismas piezas, reprogramar tareas de maquinado a piezas antes de ser ensambladas, uso de PLC para modificar tareas de transporte y manipulación, prueba de nuevos algoritmos de visión, prueba de rutinas CNC, etc.

En el siguiente capítulo se abordará el tema de diseño conceptual donde a partir de las necesidades y especificaciones del usuario se propone un modelo que las represente y materialice.

Capítulo 3

Diseño Conceptual

La fase de diseño consiste en concebir un producto o un proceso a partir de las especificaciones técnicas o parámetros de diseño [5]. Estos son magnitudes físicas o directrices que el proceso debe respetar y verse reflejadas en el producto final. En el capítulo anterior abordamos precisamente éstas directrices a las que llamamos especificaciones de diseño establecidas como resultado de una serie de necesidades del usuario. El presente capítulo trataremos sobre el diseño conceptual del sistema.

3.1.- Descripción general del proyecto.- El propósito del trabajo fue crear un sistema que muestre de manera muy objetiva a los alumnos del ITSSLP,C el concepto fundamental de los procesos de manufactura flexible, entre ellas las operaciones de manipulación de piezas, las operaciones de maquinado más elementales tales como taladrado, redondeo de esquinas, ranurado, etc. previamente programadas sobre piezas planas de geometría regular que puedan ser manipuladas por el robot con el que cuenta la celda para formar un ensamble o producto. Una característica sobresaliente de éste sistema es que cuenta con el sensor de visión, importante sensor que le permite reconocer y ubicar las piezas que se encuentran libremente distribuidas en una mesa de trabajo, refiriendo su ubicación respecto del sistema coordinado del robot o sistema mundial, con el fin de que el sistema determine si ha de mover la pieza hacia su lugar final de ensamble o llevarla antes a una estación de maquinado donde se le habrá de aplicar alguna operación previa a su ensamblado final [6].

En términos generales, se trata de un sistema de cómputo con interfaces al controlador del robot y al sistema de visión del CMA, éste último con características propias de un sistema de inspección con fines de control de calidad de la producción, se encargará de capturar las imágenes de las escenas a ser procesadas e interpretadas por nuestro sistema. Debido a que el ITSSLP,C no cuenta con equipos CNC, la característica de incluir operaciones de maquinado se dejó fuera del alcance de la tesis y sólo se hace referencia en el último capítulo en el apartado de trabajos a futuro. La integración de los equipos CNC al CMA dará sentido a la característica de modularidad del sistema, una de las especificaciones establecidas en nuestro proyecto.

El objetivo de nuestro proyecto es didáctico y por tal motivo las operaciones de maquinado serán operaciones básicas, así como la forma y materiales de las piezas serán las adecuadas para que el robot las pueda manejar y el ensamble pueda ser muy objetivo, visual e ilustrativo. El sistema maneja ensambles con 1 a 10 piezas, de un tamaño aproximado de 1 a 8 cm. de largo por 1 a 3 cm. de ancho y altura variable, de forma libre, un mismo ensamble puede ser realizarlo varias veces modificando el orden de ensamblado o alterando el número y las operaciones de maquinado, etc. Todo ello con el fin de enriquecer y despertar el interés en los alumnos por los procesos flexibles de manufactura. A continuación se dará una breve descripción de la metodología utilizada para representar el diseño conceptual de nuestro sistema.

3.1.1- Metodología de desarrollo IDEF0.- La traducción literal de las siglas IDEF0 es definición de la integración para la modelización de las funciones (*Integration Definition for Function Modeling*). IDEF0 consiste en una serie de normas que definen la metodología para la representación de funciones modelizadas [7].

Estos modelos son diagramas jerárquicos junto con texto y referencias, representados mediante rectángulos o cajas y una serie de flechas. Una característica importante de IDEF0 es que como concepto de modelización introduce gradualmente cada vez más niveles de detalle a través de la estructura del modelo. Como características importantes podemos decir que son una forma unificada de representar funciones o sistemas, su lenguaje es simple, pero preciso, permite establecer límites de representación de detalle establecido universalmente. Algunos de sus elementos más importantes son:

- Diagrama A-0: Diagrama de contexto de IDEF0 de una sola caja, contiene la función de más alto nivel a ser modelizada, así como sus entradas, salidas, controles y mecanismos.
- Flecha: Línea directa compuesta por uno o varios segmentos que modeliza un conducto para datos u objetos desde una fuente a un destino. Existen cuatro tipos de flechas que son entradas, salidas, controles y mecanismos.
- Caja: Rectángulo que contiene un nombre y un número usado para representar una función.
- Nombre de caja: Verbo o frase verbal ubicada en el interior de una caja IDEF0 para describir la función modelizada.
- Número de caja: Número que va desde 0 a 6 que se sitúa dentro de la esquina inferior derecha de una caja IDEF0 para fines de identificación.
- Flecha de Entrada: Se ubican en la parte izquierda de la caja, representan datos u objetos que el sistema transformará en salidas. Pueden ser requisitos, estados, necesidades, facturas, etc.
- Flecha de Salida: Se ubican en la parte derecha de la caja, representan datos u objetos que el sistema transformó.
- Flecha de Control: Apuntan a la parte superior de la caja y representan las condiciones que se requieren para producir una salida correcta (i.e. manuales de calidad, políticas, procedimientos, etc.)
- Flecha de Mecanismo: Todo aquello que se requiere para desarrollar una función, se sitúan en la parte baja de la caja IDEF0. Muestran las interrelaciones con otros procesos y los recursos externos necesarios para el proceso.

3.2.- **Análisis y establecimiento de funciones.**- Para su solución el problema principal se dividió en partes, véase Fig. 3.1, de lo cual se desprendió la propuesta que se esquematiza mediante el uso de la metodología IDEF0 para representación de la función principal, véase Fig. 3.2, así como el desglose de la función principal en una serie de subfunciones para un mejor entendimiento del problema y lograr una mejor solución al mismo, véase Fig. 3.3.

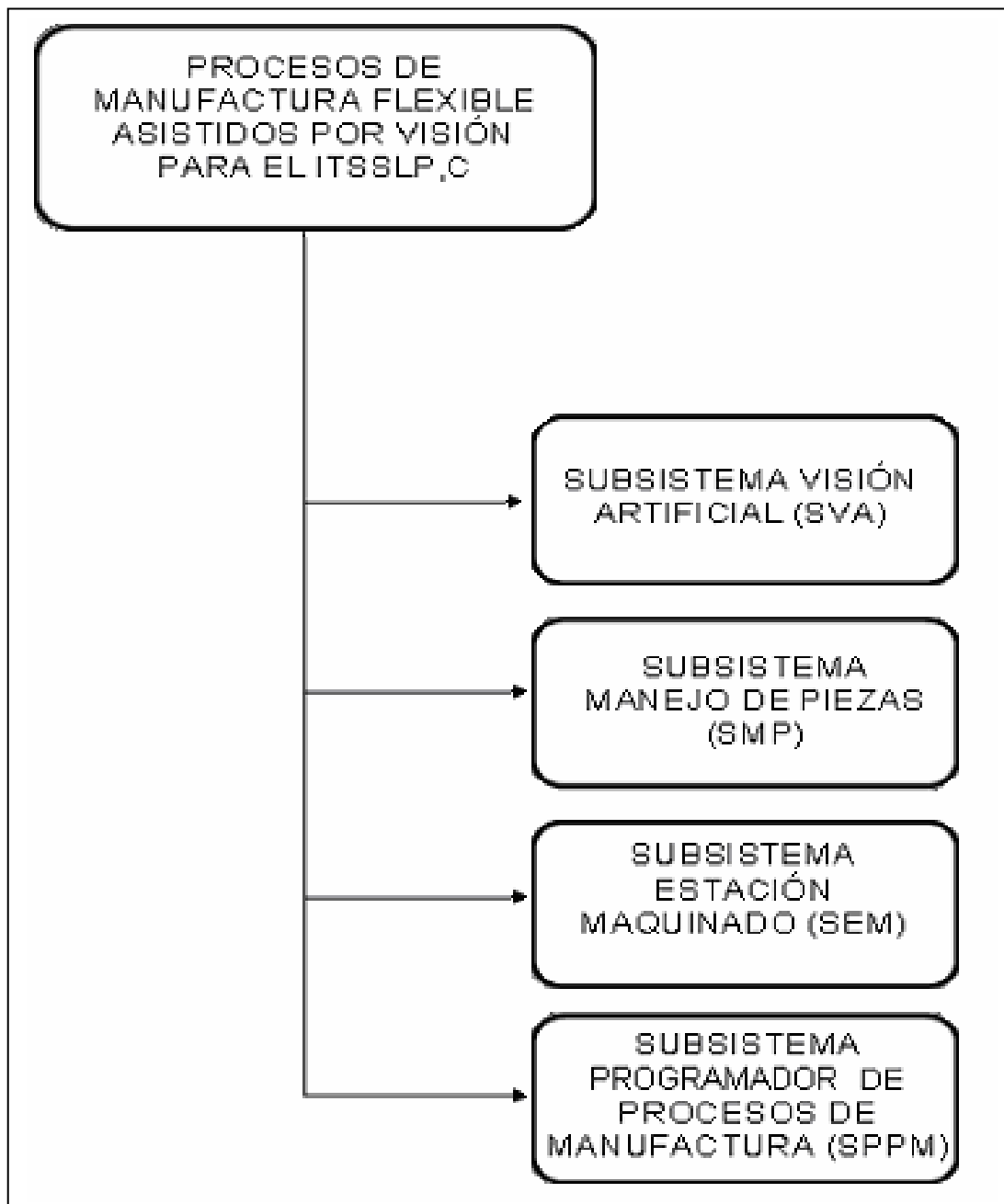


Figura 3.1 Árbol de funciones generales.

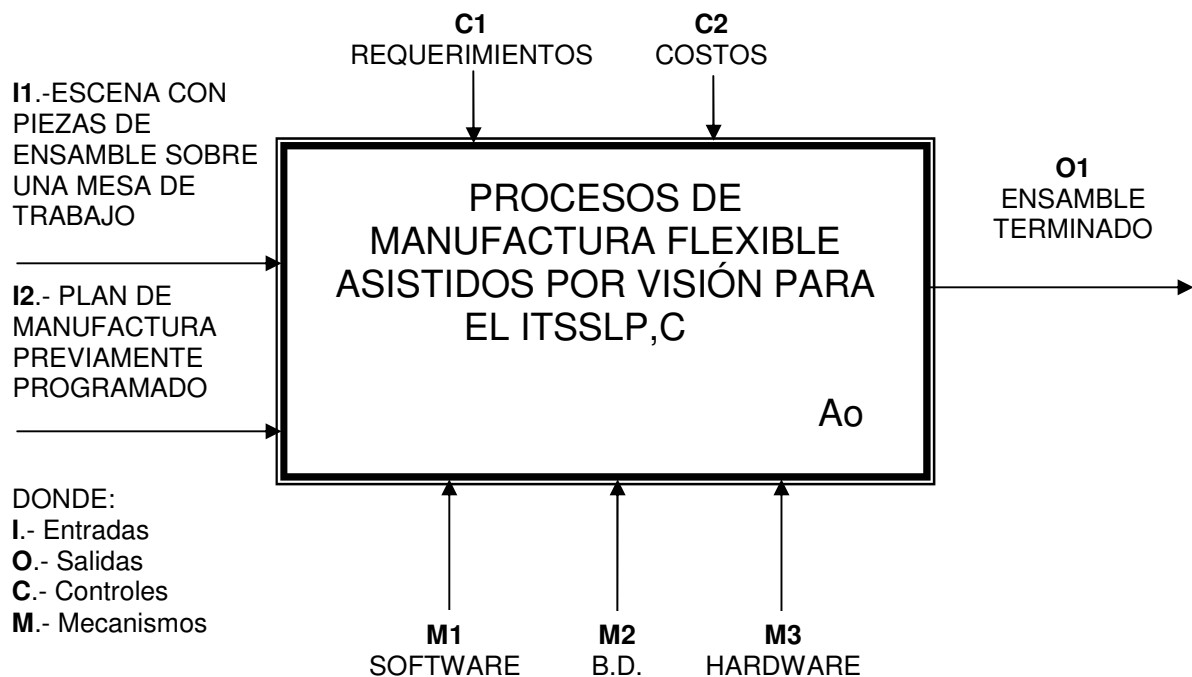


Figura 3.2 Función principal del sistema desarrollado

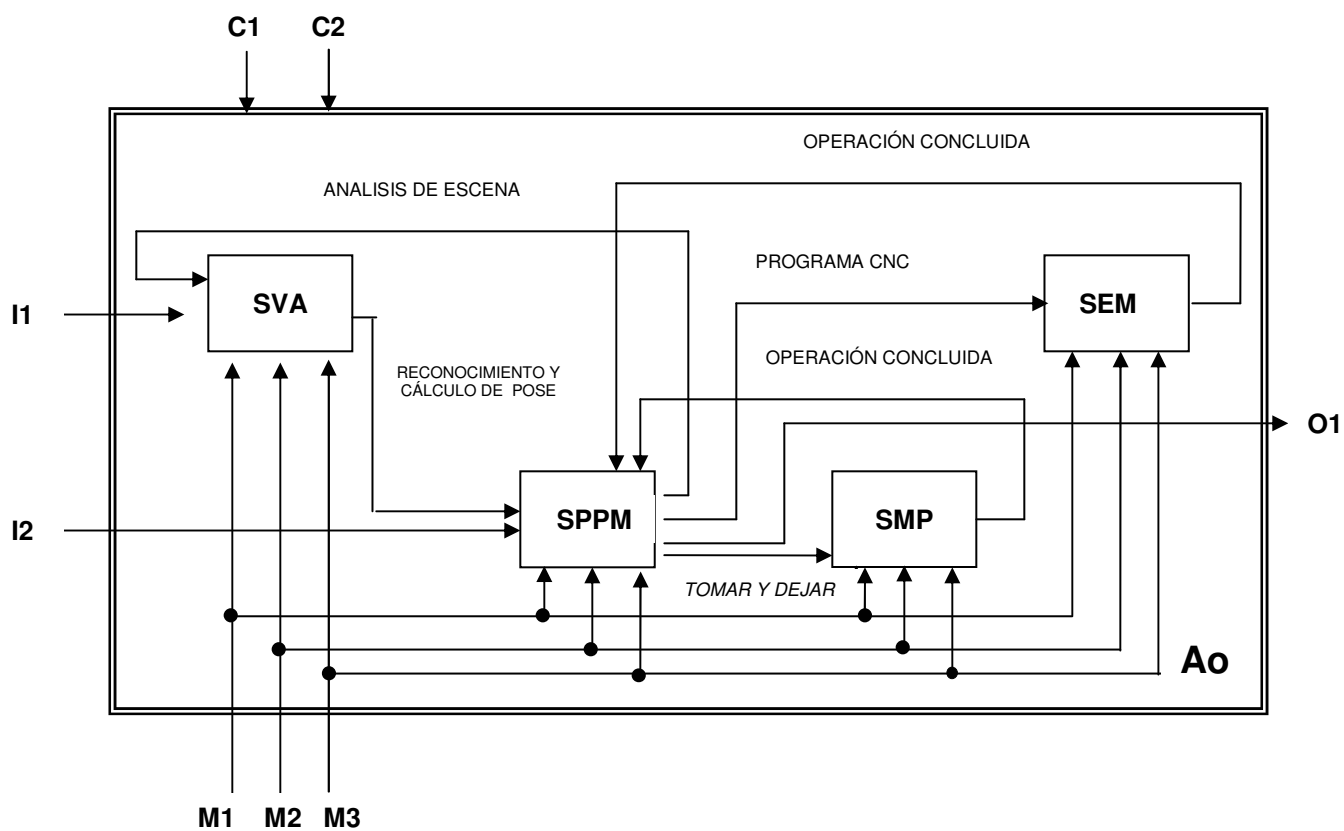


Figura 3.3 Líneas de flujo de información y subsistemas.

3.2.1 Descripción de cada subsistema.- De manera generalizada a continuación se dan características inherentes a cada subsistema que son y en que consisten, posteriormente se presentan una serie de alternativas a analizar para su implementación. Los medios con los que se cuentan para la implementación del sistema están restringidos a los exclusivamente proporcionados por la CMA, ya que no se cuenta con recursos propios para la realización del proyecto.

Subsistema de Visión Artificial (SVA).- Previo a la ejecución de cualquier proceso de manufactura, se debe contar con información suficiente para asegurar que se logrará el reconocimiento de las piezas que forman parte del proceso de ensamble que se desea ejecutar. Para lograr esto el SVA contará con una etapa previa llamada “proceso de aprendizaje”, con esto el SVA podrá reconocer de manera adecuada las piezas de trabajo en la escena y posteriormente determinar en cada caso la posición (coordenadas del centro de masa de la pieza) y la orientación (ángulo que el eje de simetría longitudinal o eje de mayor elongación de la pieza tiene respecto a la horizontal), valores requeridos por el SPPM para su almacenamiento y manipulación en operaciones de manejo de piezas, con el fin de trasladar la pieza a una estación de maquinado o bien, a la posición final de ensamble.

Subsistema Programación de Procesos de Manufactura (SPPM).- Es un programa de cómputo encargado de administrar y controlar los procesos de manufactura, con interfase a una base de datos donde se almacena información referente a cada pieza que forma parte del proceso y que se relaciona con:

- Posición y orientación que cada pieza ocupa dentro del proceso de ensamble.
- Orden que le corresponde en la secuencia de ensamble.
- Número de operación de tarea de maquinado como pueden ser barrenado, fileteado, redondeado de esquina o ranurado en caso de ser requeridas, así como identificación y secuencia de la o las estaciones de trabajo que se encargarán de dichas operaciones.
- Programa CNC correspondiente a cada operación de maquinado.

Con el SPPM se diseñan, administran y controlan todos los procesos de manufactura flexible, este módulo es el encargado de controlar y administrador los procesos a ejecutarse y es el que mantiene continuamente comunicación con el resto de los subsistemas.

Subsistema de Manejo de Piezas (SMP).- Lo constituye básicamente un manipulador automático programable también llamado robot, el cual se encargará de realizar todas las operaciones de manejo de piezas, para ello el SPPM obtiene del SVA el nombre de cada pieza y su posición dentro de la escena de trabajo, el SPPM busca en su base de datos y determina para cada pieza reconocida que forma parte del proceso que se está ejecutando, su posición final posterior a la conclusión del proceso, la cual puede ser una estación de maquinado para realizar una o más operaciones previas a su ensamble (torno, fresado, taladrado) o su ubicación final sobre la plantilla de ensamble. Es cuando el SPPM para cada una de las piezas hace interfase con el robot y le asigna las dos posiciones, la inicial sobre la mesa de trabajo y la final (área de ensamble ó estación de maquinado SEM) esto mediante operaciones básicas de manejo de piezas llamadas operaciones “tomar y dejar”. En caso de que la pieza haya sido trasladada a un SEM,

cuando terminen las operaciones de maquinado, el robot toma la pieza de dicha estación y la coloca finalmente en su posición de ensamble.

Subsistema Estación de Maquinado (SEM).- Esta parte del modelo lo constituye uno o más equipos de CNC, los cuales podrán trabajar en colaboración con el sistema cuando a futuro lleguen a ser adquiridos por el ITSSLP,C. Por el momento y para los fines del presente trabajo, el SEM no será incluido en nuestro proyecto.

3.2.2 Alternativas de solución.- Con el objeto de determinar cuales serán los mecanismos a utilizar en cada subsistema, se deben analizar todas y cada una de las opciones disponibles a nuestro alcance proporcionados por el CMA, considerando su conveniencia, factibilidad de implementación, así como sus costos. Este último constituye uno de los controles del sistema importantes, debido a que la integración del sistema se propuso realizar contando únicamente con los recursos de hardware y software propios del CMA del ITSSLP,C.

3.2.2.1 Alternativas para el subsistema de Visión Artificial SVA.- En la presente sección se exponen opciones a nuestro alcance y que además cumplan con los requerimientos y necesidades presentadas en el capítulo dos, para implementar el subsistema de visión artificial. Presentamos las siguientes alternativas para este subsistema:

- Máquina de visión de la celda de manufactura.
- Sistema de Visión Técnica [8] desarrollado para el Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET) de la UNAM.
- Desarrollo de software.

Máquina de visión de CMA del ITSSLP,C.- La celda cuenta con un sistema de inspección de piezas por visión de la marca Cognex modelo DVT LEGEND 510 con imágenes binarias (blanco y negro) y resolución de 480x640 pixels, básicamente es un sistema de visión orientado a tareas de inspección [9] que es capaz de detectar defectos, ausencia o presencia de partes, toma mediciones como radios, longitudes, áreas, cuenta y reconoce piezas. Trabaja bajo la siguiente idea básica:

Tomar una secuencia de imágenes de la pieza correcta a fabricar, la cual se almacenará como patrón.

Se define que es lo que se va a inspeccionar en la imagen (radios, longitudes, contar objetos, etc.) y se establecen las condiciones o márgenes de aceptación para que las piezas a inspeccionar sean analizadas y el sistema pueda determinar un estado de aceptación ó rechazo que comunicará al módulo de entradas/salidas.

El sistema de visión procesa la imagen y determina una señal de PASA ó FALLA, el cual comunica mediante un módulo de entradas y salidas por conexión ethernet al módulo de control de la celda.

El sistema funciona, pero está limitado a tareas de inspección para control de la calidad de los productos producidos en la celda, el campo visual de la cámara es reducido y está limitado a piezas no mayores a 4 cm por lado, trabaja sólo con un producto a la vez ya que si se desea trabajar con dos o más requiere de especificar un identificador de producto y asignarlo al proceso activo. Debido a limitaciones técnicas del sistema, el máximo número de productos distintos con los que se puede trabajar es de cuatro y mediante un proceso de selección, sólo se trabaja uno a la vez.

Esto constituye una limitante para los requerimientos establecidos en nuestro proyecto, ya que requerimos el reconocimiento de 8 a 10 piezas a la vez y con tamaños posiblemente mayores a los permitidos por este sistema y con campos visuales que abarquen un área rectangular de aproximadamente 10 cm x 15 cm como mesa de trabajo.

Sistema de Visión Técnica.- Sistema de visión para reconocimiento y ubicación de piezas para tareas de micromanipulación y micro ensamble dentro de un proyecto de creación de micro fábricas totalmente automáticas y autorreproducibles para el Laboratorio de Computación Neuronal del Centro de Ciencias Aplicadas y Desarrollo Tecnológico de la Universidad Nacional Autónoma de México. Utiliza algoritmos neuronales de los clasificadores LIRA y PCNC para el reconocimiento de piezas bajo condiciones variadas de iluminación, oclusión parcial de piezas y posicionamiento libre en la escena [8].

Este sistema tiene buen desempeño en tareas de reconocimiento bajo condiciones naturales de trabajo que usualmente tenemos en las líneas de fabricación y ensamble de las fábricas, sin embargo en la determinación de la orientación de la pieza tiene cierta deficiencia. Trabaja en un entorno de programación visual de alto nivel C++ y no tiene implementada, en ambiente MS-Windows, la forma de operarlo fuera de línea mediante llamadas al módulo ejecutable por medio de argumentos desde otra aplicación particular. Esto sería de gran valor para nuestro proyecto, ya que de esta manera llamaríamos a trabajar al sistema de visión como un subprograma al que le proporcionaríamos el archivo de imagen de la escena en la mesa de trabajo y la referencia a la base de conocimiento donde se almacenan los patrones de las piezas y el subprograma nos daría de regreso el identificador de todas y cada una de las piezas así como su posición y orientación en la mesa de trabajo, datos que serían utilizados por el SPPM para comunicar al SMP datos relativos a su posición actual y con los datos de su posición final de ensamble el SMP podrá realizar operaciones básicas tomar y dejar para desplazar cada pieza en su rutina de ensamble.

Elaboración de programas de aplicación en ambiente MatLab.- Matlab de MathWorks, Inc. es un lenguaje comercial de computación técnica de alto desempeño integra cómputo, programación y visualización en un medio ambiente amigable y con notación matemática matricial común. El nombre se deriva de Matrix Laboratory se compone de familias de aplicaciones específicas llamadas toolboxes, donde Image Processing Toolbox (IPT) es una colección de funciones que extienden las capacidades de Matlab para la solución de problemas de procesamiento digital de imagen. Además [10] proporciona otro conjunto de funciones adicionales llamada DIPUM (Digital Image Processing Using Matlab) que en conjunto con IPT proporcionan un buena fuente de información y recursos para construir un SVA

3.2.2.2 Alternativas para el subsistema programación de procesos de manufactura SPPM.- Este subsistema es el encargado de administrar, comunicar y ejecutar coordinadamente las operaciones para que el proceso de ensamblado automatizado se lleve a cabo de manera correcta.

3.2.2.3 Alternativas para el subsistema de manejo de piezas SMP.- Un SMP está integrado por al menos un brazo robótico, en nuestro caso el que se utilizó es el robot, véase Figura 3.4, así como sus características principales en la Tabla 7.1, adquirido como parte esencial de la celda en el CMA y cuenta con los siguientes elementos:

Brazo robótico Mitsubishi RV-2AJ, controlador del robot, botonera y una PC. La forma de manipular el brazo puede ser: Por medio de la botonera (*teach pendant*), operación automática corre cíclicamente programas almacenados en el controlador y mediante control externo vía PC.

MELFA-BASIC es un sencillo lenguaje de programación para controlar al robot, el programa corre asociado a un archivo de posiciones enseñadas grabadas previamente mediante el uso de la botonera, todo dentro de un entorno cerrado de simulación y control de la celda llamado COSIMIR suministrado por el fabricante [1,11]. El programa de instrucciones compilado es bajado al controlador del robot para su almacenamiento y ejecución.

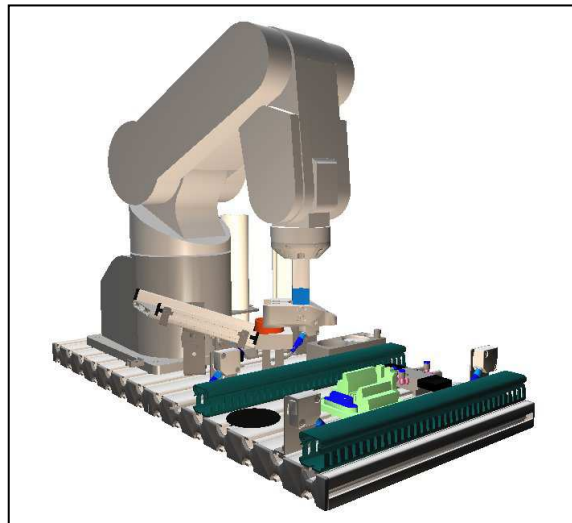


Figura 3.4 Robot Mitsubishi RV-2AJ.

3.2.2.4 Alternativas para el subsistema estación de maquinado (SEM).- Actualmente no se cuenta con estos equipos CNC en el ITSSLP,C , (véase Apartado 3.1) por lo tanto este subsistema no forma parte del presente trabajo.

3.2.3 Evaluación de las alternativas.- En la presente sección se analizará cada una de las alternativas de solución en cada subsistema y se definirá en cada caso, cual de ellas se

adopta para integrar lo que será el diseño preliminar, también llamado diseño de configuración con el que se trabajará de aquí en adelante hasta el término del proyecto.

3.2.3.1 Para SVA.- La máquina de visión del CMA diseñado particularmente para tareas de inspección y control de calidad, no puede formar parte de nuestro sistema (véase Apartado 3.2.2.1), sin embargo se utiliza su dispositivo de adquisición de imagen para capturar las escenas donde aparecen las piezas de trabajo en nuestro proyecto. La segunda alternativa, Sistema de Visión Técnica, no es viable debido a que no contamos con medios para comunicarnos fuera de línea con dicho sistema y obtener información en tiempo real desde nuestra aplicación, este sistema cuenta con comunicación vía línea de comandos con una implementación desarrollada por sus autores bajo el sistema operativo Linux. Nos decidimos por la tercera alternativa, desarrollo de software en lenguaje C en el ambiente de MatLab con la ayuda de rutinas disponibles en librerías IPT y DIPUM mencionadas anteriormente y por medio de ellas implementaremos módulos para cada una de las fases que se requieren en una máquina de visión con propósitos de reconocimiento de formas bidimensionales que corresponden a los objetos dispuestos libremente en la mesa de trabajo. Estos módulos son:

- Adquisición o toma de imagen.
- Preprocesamiento de imagen (mejora y restauración).
- Segmentación de imagen (separa los objetos del fondo).
- Descripción y extracción de características de cada objeto segmentado.
- Reconocimiento, clasificación e interpretación de la información obtenida.

Además se programaron módulos para calibrar el sistema con la cámara y otro para almacenar información en la base de conocimiento referente a los patrones durante la fase de entrenamiento o aprendizaje.

3.2.3.2 Para SPPM.- Dadas las características inherentes a este subsistema encargado de administrar, controlar y comunicar al resto de los subsistemas, se elaboró un programa de cómputo programado en lenguaje C en el entorno de Matlab, homologando así el desarrollo e implementación con el resto de los subsistemas.

3.2.3.3 Para SMP.- En el apartado 3.2.2.3, se mostraron las características generales del brazo robótico del CMA y sus diferentes maneras de operarlo, de todas ellas la última modalidad es la que más se acerca a la forma en la cual queremos controlar al robot desde una aplicación particular, se requirió que pueda correr fuera de línea respecto al entorno cerrado de operación y simulación llamado COSIMIR, para ello utilizamos otra forma de comunicar y operar el robot, esto fue posible haciendo uso del modo inmediato que nos proporciona una herramienta de software suministrada por el fabricante llamada *Movemaster command line* la cual utiliza comunicación serial RS-232 para acceder directamente al controlador del robot y ejecutar las instrucciones suministradas vía puerto serial. Puerto al cual podemos acceder desde nuestra aplicación particular programada en lenguaje C en el entorno Matlab. El subsistema SPPM comunicará mediante instrucciones básicas del robot para ejecutar los movimientos necesarios para llevar a cabo las operaciones “tomar y dejar” requeridas.

En resumen, el diseño preliminar lo podemos representar mediante la Tabla 3.1 que nos muestra cada función así como los medios que utiliza para su estructuración.

FUNCION	MEDIOS		
	PRINCIPIO DE TRABAJO	ELEMENTO FISICO	CARACTERÍSTICAS
SUBSISTEMA DE VISION ARTIFICIAL (SVA)	SOFTWARE PARA EL SISTEMA DE VISIÓN CON FINES DE RECONOCIMIENTO Y UBICACIÓN DE PIEZAS.	CAMARA Y SOFTWARE COMERCIAL MATLAB CON LIBRERIAS IPT Y DIPUM.	PROGRAMADO EN LENGUAJE C DE MATLAB v7, INSTALADO EN LA PC QUE CONTROLA LA CMA.
SUBSISTEMA DE MANEJO DE PIEZAS (SMP)	MANIPULADOR AUTOMÁTICO PROGRAMABLE (ROBOT) SISTEMA DE TRANSPORTE DE PIEZAS.	ROBOT MITSUBICHI MOD. RV-2AJ. BANDA TRANSPORTADORA CONTROLADA POR PLC.	ALCANCE: 410 mm CARGA MAX. : 2 Kg GRADOS DE LIBERTAD: 5 REPETIBILIDAD: ± 0.02 mm VEL. MAX. : 2100 mm/seg PESO DEL ROBOT: 17 Kg
SUBSISTEMA PROGRAMADOR DE PROCESOS DE MANUFACTURA (SPPM)	SOFTWARE PARA ADMINISTRAR Y CONTROLAR LOS PROCESOS DE MANUFACTURA FLEXIBLE.	SOFTWARE DE APLICACIÓN QUE SE EJECUTARÁ EN UNA PC.	SOFTWARE PROGRAMADO EN LENGUAJE DE ALTO NIVEL PARA ADMINISTRAR LOS PROCESOS DE MANUFACTURA ASÍ COMO INTERFACES PARA COMUNICARSE CON EL RESTO DE LOS SUBSISTEMAS.

Tabla 3.1 Identificación y selección de principios de trabajo para cada función del sistema.

3.3.- Resultados en la evaluación de alternativas.- Se evaluaron las diversas alternativas disponibles para cada uno de los subsistemas SVA, SPPM y SMP (véase Apartado 3.2.3) y se determinaron los elementos físicos a utilizar en la implementación de cada subsistema (véase Tabla 3.1). Una vez determinados los elementos físicos a utilizar en cada uno de ellos y con el propósito de detallar el sistema, en los siguientes capítulos se tratará cada uno de los subsistemas por separado. Al final el capítulo 8 englobará a los tres para mostrar el sistema trabajando en su conjunto. En este mismo capítulo se demostrará y comprobará la validez del sistema mediante un demo que ejecute el ensamblado de dos plantillas distintas que utilizan las mismas piezas en su proceso. Al final presentaré los resultados obtenidos, conclusiones y trabajo futuro.

Capítulo 4

Diseño de Configuración

4.1 Introducción.- En el presente capítulo abordaremos el tema de configuración del sistema que es la disposición física que presentan los diferentes elementos que componen el sistema así como su correcta y adecuada referencia espacial y geométrica.

El diseño adoptado en el presente trabajo está delimitado al uso de los elementos existentes en la CMA del Instituto, que está integrada por varias estaciones de trabajo, véase Sección 1.2. Las estaciones utilizadas para nuestro sistema son las estaciones de transporte, la estación de visión y la estación de manipulación y montaje las cuales se describen a continuación.

4.2 Elementos del CMA que se utilizaron en el proyecto.- Una de las características principales en nuestro proyecto y que comentamos en el capítulo dos, es el hecho de utilizar como parte de nuestro proyecto algunas de las estaciones de trabajo de la actual celda del ITSSLP,C,. Pretendemos demostrar que mediante un sistema de visión artificial trabajando en conjunto a elementos de transporte y manejo de piezas, obtendremos un sistema flexible de manufactura en contraste con el sistema rígido de fabricación con el que actualmente cuenta el tecnológico.

4.2.1 Estación de transporte.- Integrada por una banda transportadora que conduce charolas rectangulares de 110 x 165 mm. sobre las cuales se colocan las piezas de trabajo. El control de la banda en su interacción con sensores de presencia, pistones para detener el avance de la banda, así como intercambio de información con otras estaciones tales como el sistema de visión de la CMA y el robot manipulador Mitsubishi RV-2AJ se realiza mediante un PLC, véase Fig. 4.1.

4.2.2 Estación de visión.- Integrada por sistema de visión de la marca Cognex modelo DVT-Legend 510 cuyas características ya fueron descritas en el Apartado 3.2.2.1. Ésta estación sólo se utilizó para tomar imágenes del área de trabajo y almacenarlas en disco. Cabe hacer mención que se hicieron algunas modificaciones en esta estación, tales como retirar un tubo de metal cilíndrico que la CMA utiliza para un mejor control de la iluminación ambiental, ya que aísla cualquier interferencia luminosa externa para hacer más eficiente el sistema de iluminación integrado a la cámara. Otra modificación necesaria fue fabricar una extensión al soporte vertical que sostiene la cámara, esto con el propósito de aumentar la altura de la cámara en aproximadamente 35 centímetros para así incrementar las dimensiones del campo visual sobre la mesa de trabajo, para así lograr visualizar los 110 mm. de ancho de las charolas que transportan las piezas de trabajo.

4.2.3 Estación de manipulación y montaje.- Constituido básicamente por el robot y su controlador, al igual que la estación de visión mantiene comunicación con el PLC que controla la celda. Al circular las charolas sobre la banda, ésta se detiene junto al robot cuando un sensor de presencia detecta la charola, en ese momento, el control pasa al manipulador iniciando la ejecución del programa almacenado en el control del manipulador destinado a ejecutar una serie de operaciones de “tomar y dejar” sobre las piezas que se encuentran en la charola. En tanto el SPPM leyó la imagen, la preprocesó, segmentó, describió e identificó las piezas en la escena así como calculó la posición en la que se encuentran en la mesa de trabajo respecto al sistema coordinado del robot. Estos valores son transmitidos por el SPPM vía puerto serial al programa que controla el robot, junto con las posiciones finales de ensamblado para que éste aplique repetidamente operaciones “tomar y dejar” y se obtenga finalmente el proceso automatizado de ensamble asistido por visión artificial, que es el objetivo del presente proyecto.

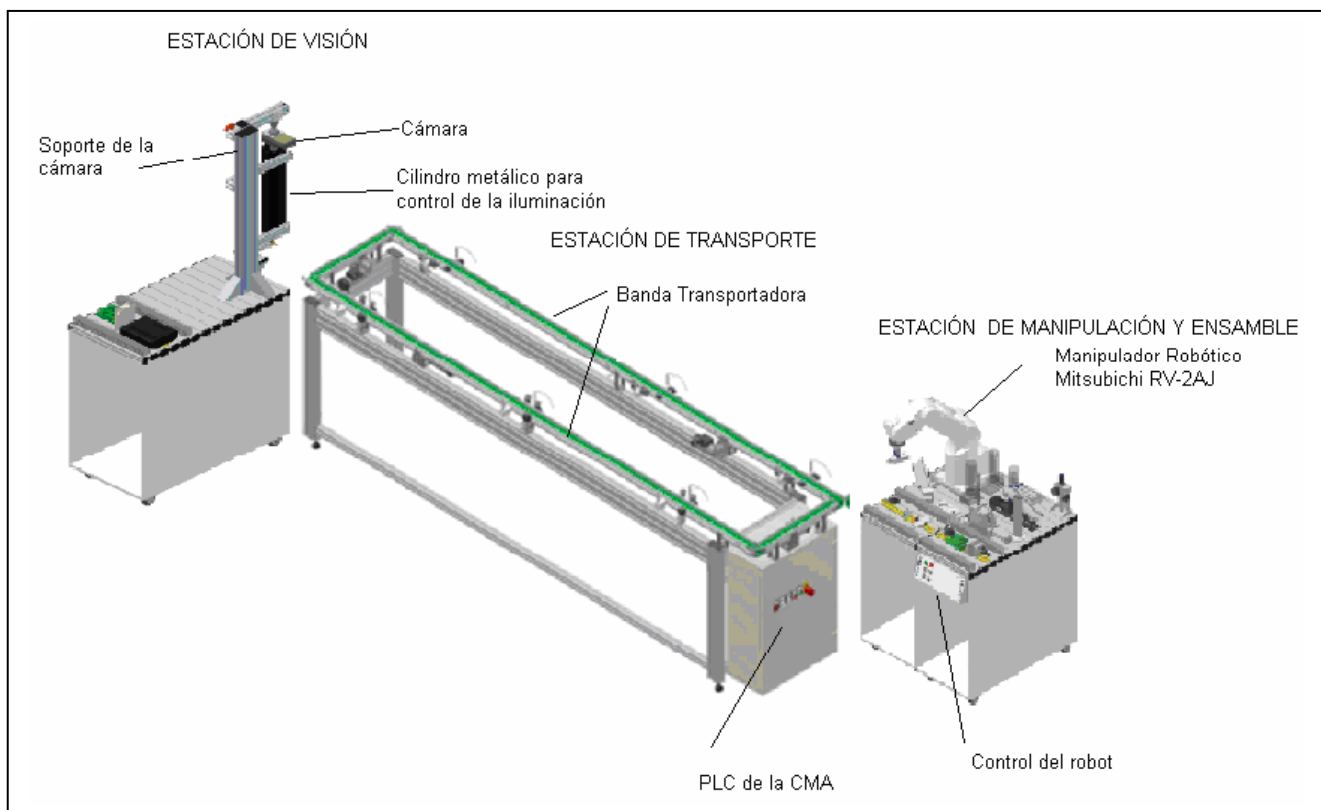


Figura 4.1 Estaciones de trabajo del CMA que participan en el sistema.

En la Fig. 4.1, la cámara se ubica por encima de la banda transportadora de tal manera que el plano sensor de imagen de la cámara, la cinta de la banda transportadora y por consecuencia la superficie de la mesa de trabajo, son todos planos paralelos, el eje X del plano de imagen es también paralelo al eje longitudinal de la banda, véase Fig. 4.2

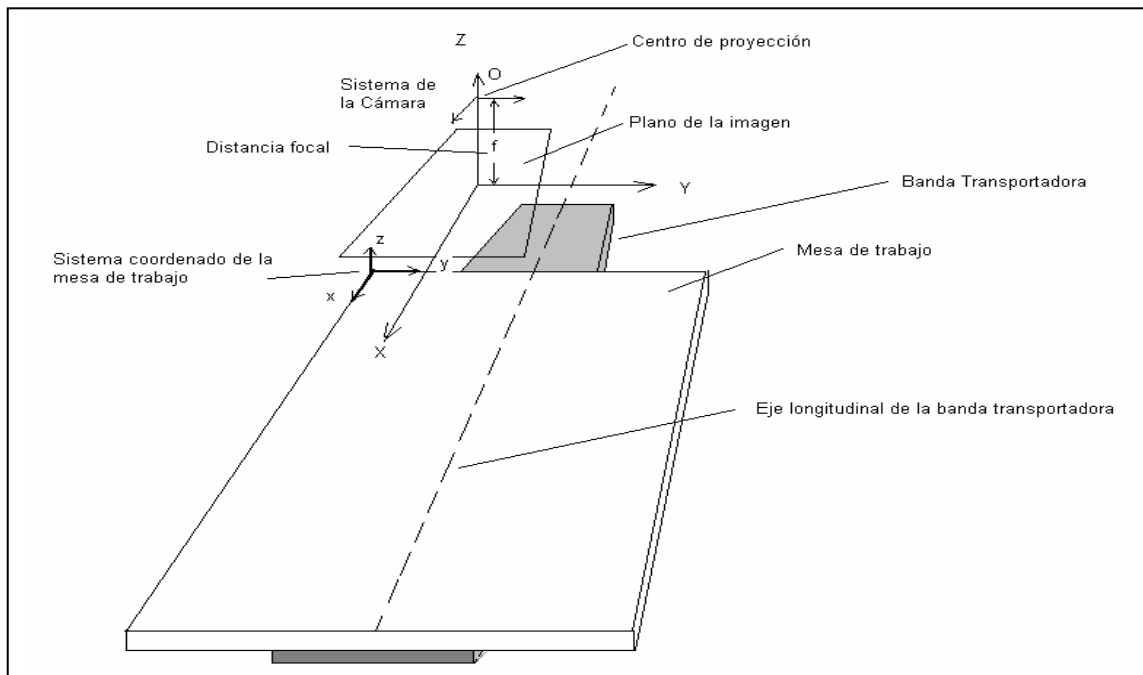


Figura 4.2 Paralelismo entre los planos de imagen, mesa de trabajo y banda transportadora.

4.2.4 Mesa de trabajo.- El Subsistema de visión tiene la finalidad de determinar la identidad así como la ubicación correcta para cada una de las piezas que se localizan en la mesa de trabajo. La mesa de trabajo es una caja de forma rectangular cuya superficie es una placa de acrílico translúcido blanco de 100 x 230 mm, bajo esta placa se colocó una fuente de luz blanca, la cual permite obtener imágenes de buena calidad en cuanto a nitidez y contraste. La mesa de trabajo se coloca sobre la charola que se desplaza por medio de la banda transportadora, transportando la mesa de trabajo de la estación de visión a la estación de manipulación y ensamble para luego retornar en forma cíclica a la estación de visión.

En el trayecto que la charola viaja de la estación de visión a la estación de manipulación, el SPPM deberá planificar la tarea de ensamblado y transmitir al control del robot la información para que pueda realizar las operaciones tomar y dejar sobre las piezas y logre ejecutar el ensamble programado.

Usamos la cámara del sistema DVT Legend 510, para obtener la imagen de la escena, la cual genera en base a un disparo que de manera externa se realiza cuando el PLC del CMA después de haber detectado la presencia de una charola, acciona el pistón que la detiene, la imagen es almacenada con un nombre genérico en formato de imagen .BMP (Windows Bit MaP o mapa de bits) llamada: escena001.bmp. El formato BMP es el único formato que se tiene disponible en el software de la cámara, sin embargo, las librerías de funciones que utilizamos en el presente trabajo, IPT y DIPUM, permiten convertir las imágenes a formatos más eficientes tales como: JPEG (Joint Photographic Experts Group), GIF (Graphics Intechange Format) y PNG (Portable Network Graphics).

Posterior a la toma de imagen, el PLC libera el pistón que detenía el avance de la charola y al mismo tiempo lanza la aplicación de nuestro sistema, véase Fig. 4.4, ejecución que

termina en el momento en que se establece el canal de comunicación entre el SPPM y el control del robot mediante comunicación serial RS-232, para transmitir valores de posición (coordenadas x,y,z) del centro de masa de cada pieza a ensamblar, así como los valores que determinan la orientación del efector final, de tal manera que se logre el posicionamiento esperado, para ello el control de robot ejecuta un proceso cíclico que se repite tantas veces como piezas deban manipularse y el conjunto de acciones en cada ciclo consiste en el llamado de solo dos subrutinas, una para la operación de “*tomar*” y la otra para la de “*dejar*”.

4.3 Imágenes capturadas por la cámara.- Las imágenes que la cámara toma son almacenadas en un archivo con formato BMP cuyo nombre es genérico, por ejemplo: escena001.bmp. del cual conocemos su resolución, es decir el tamaño de la imagen en píxeles, por ejemplo: 480 x 640, denotando el número de renglones por el número de columnas que ocupa la imagen. Si el campo visual de la cámara es conocido, podemos pensar en que abarcará el ancho de la mesa de trabajo por una distancia vertical, cuyos valores en mm pueden ser medidos y así tener los elementos necesarios para hacer un mapeo de las coordenadas de dispositivo (píxeles) a coordenadas del mundo real (milímetros) del campo visual capturado por la imagen, mediante las siguientes relaciones:

$$X_1 = X_p (XWmax) / (XPmax)$$

$$Y_1 = Y_p (YWmax) / (YPmax)$$

La Figura 4.3, nos muestra los valores asociados a las relaciones anteriores.

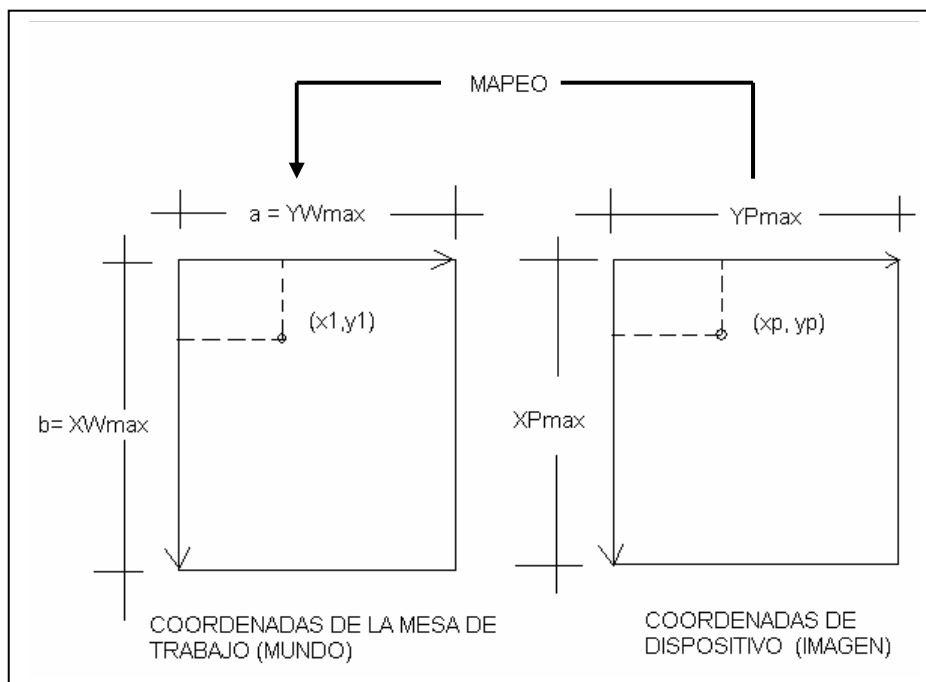


Figura 4.3 Mapeo de coordenadas de dispositivo en coordenadas del mundo real.

En esta forma calculamos los valores coordenados de cualquier punto en la escena en coordenadas del mundo real (X_1, Y_1) en nuestro caso expresadas en milímetros, teniendo

como datos los valores de ese mismo punto en coordenadas de dispositivo (X_p, Y_p) expresadas en píxeles. Este punto a que se hace referencia puede ser por ejemplo las coordenadas del centro de gravedad de la pieza, el cual representa al objeto mismo por medio de esa referencia. Además de proporcionarnos las coordenadas del centro de masa de cada pieza que se localiza en la escena sobre la mesa de trabajo, el SVA nos proporciona el nombre de la pieza reconocida o en su defecto la etiqueta “desconocido” en el caso que el sistema determine que ninguno de los patrones almacenados en la base de conocimiento de patrones corresponde con la pieza por reconocer. Otro importante dato proporcionado por el SVA es el ángulo que forma el eje de mayor elongación de la pieza respecto al eje Y de un sistema coordenado derecho con origen en la esquina superior izquierda del campo visual que sobre la mesa de trabajo capta la cámara. Cabe mencionar que la cámara se dispuso con un giro de -90° respecto al sistema coordenado de la mano derecha de la mesa de trabajo, esto con la finalidad que la totalidad del campo visual tomado por la cámara cayera dentro de la mesa de trabajo. El origen del sistema coordenado de la mesa de trabajo se ubica en el punto (0,0), es decir en la esquina superior izquierda del área de trabajo, véase Fig. 4.5, que en nuestro caso está referido a la charola que se monta sobre la banda transportadora y de esa manera desplaza la mesa de trabajo a la siguiente estación una vez que la cámara ha tomado una fotografía de la escena de trabajo. Esta escena está compuesta por una serie de piezas ubicadas de manera aleatoria (libre) sobre la mesa. Cuando la charola llega a la estación de manipulación y ensamble, que es donde se localiza el robot, un sensor de presencia activa la señal que recibe el PLC de la estación de transporte y éste lanza el pistón que detiene el avance de la charola. La siguiente figura nos muestra las tres estaciones trabajando en conjunto.

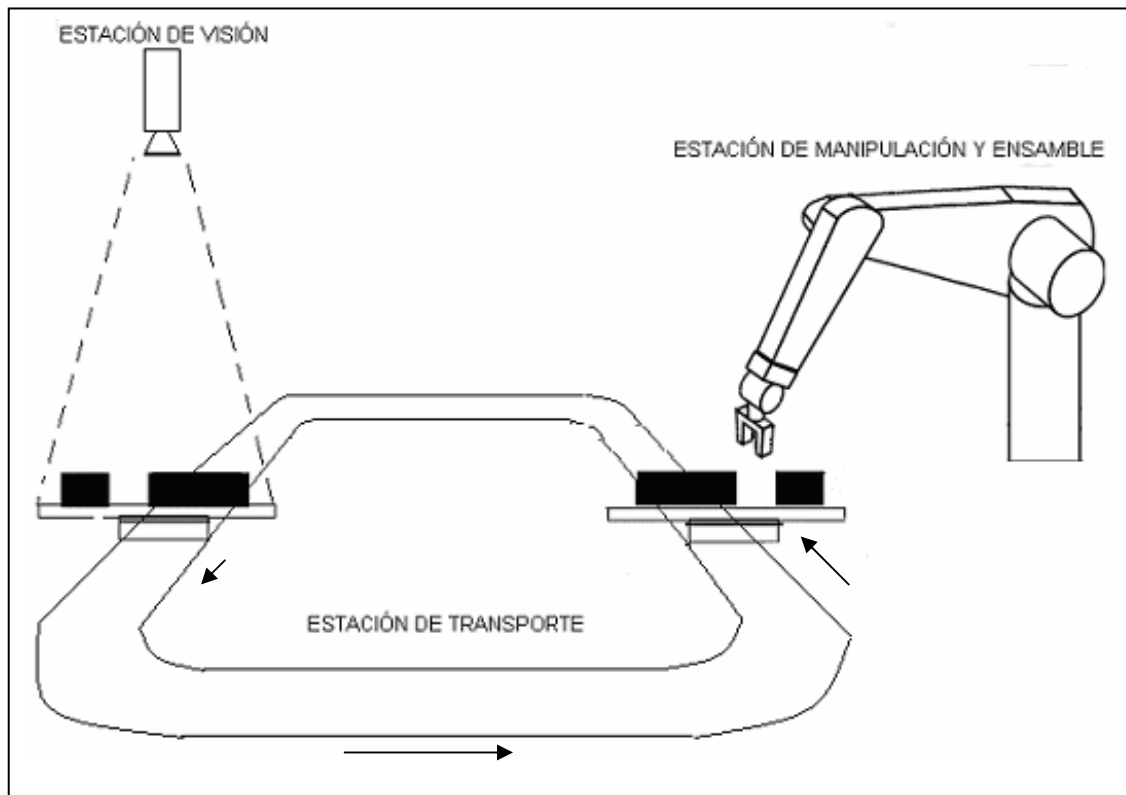


Figura 4.4 Estaciones de visión, transporte y de manipulación y ensamble.

Dada la información en coordenadas de dispositivo o píxeles proveniente de la imagen de la escena, obtenemos los valores coordenados del mismo punto en unidades del mundo real o

milímetros. Cuando la charola llega a la estación de manipulación y ensamble, la charola tiene la posición mostrada en la Figura 4.5.

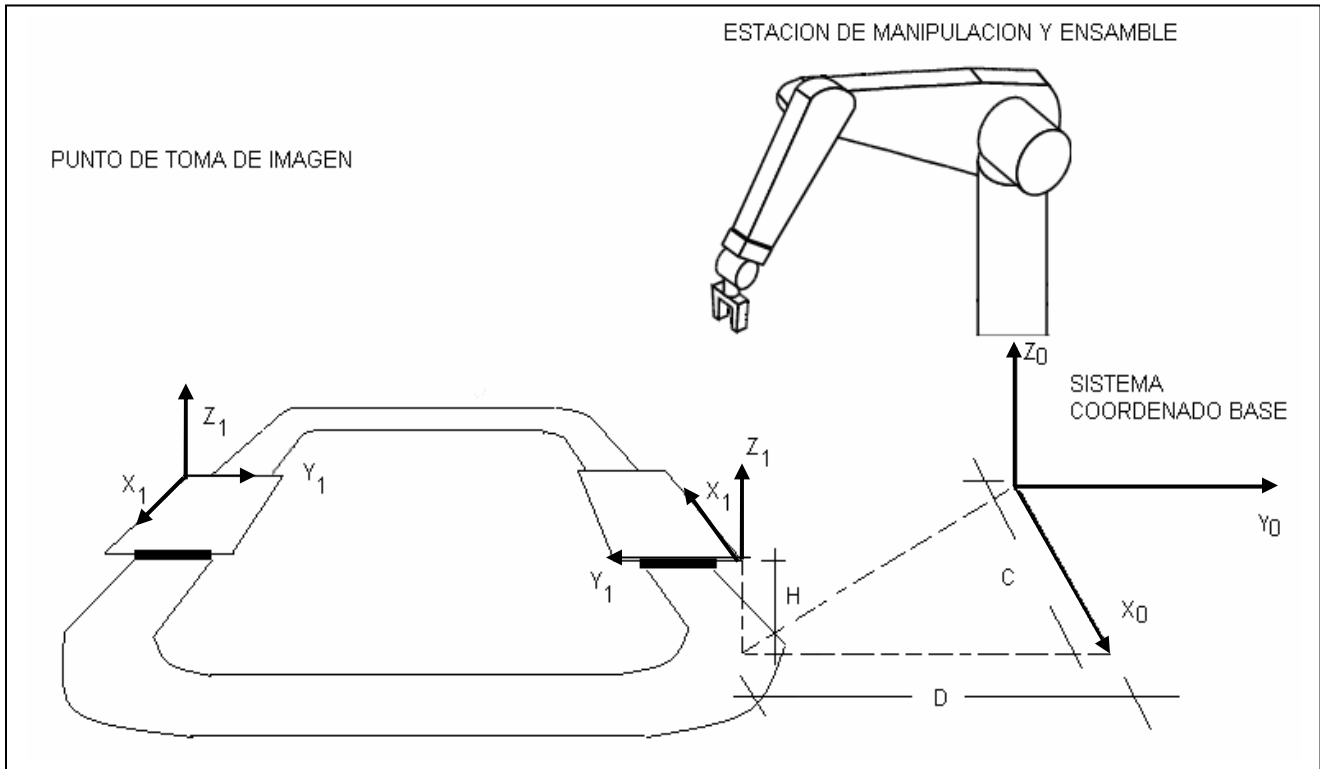


Figura 4.5 Sistema coordinado base del robot (x_0, y_0, z_0) y de la mesa de trabajo (x_1, y_1, z_1) .

Si embargo estamos haciendo referencia a una serie de sistemas coordinados, los cuales debemos uniformizar, es decir, referir respecto al sistema coordinado de la mesa de trabajo y posteriormente respecto del sistema base o sistema del robot. La manipulación de piezas llevada a cabo por un robot implica el movimiento espacial de su extremo. Asimismo para que el robot pueda tomar una pieza, es necesario conocer la posición y orientación de ésta con respecto a la base del robot. Para ello veremos a continuación el concepto de coordenadas homogéneas y basados en este concepto podremos referir finalmente todos los puntos respecto al sistema coordinado base del robot.

4.4 Coordenadas y matrices homogéneas.- Un espacio n -dimensional se encuentra representado en coordenadas homogéneas por $(n+1)$ dimensiones, de tal manera que un vector $\mathbf{p}(x, y, z)$ se representa por $\mathbf{p}(wx, wy, wz, w)$, donde w tiene un valor arbitrario y representa un factor de escala. De forma general, un vector $\mathbf{p}=ai + bj + ck$, donde \mathbf{i} , \mathbf{j} , \mathbf{k} son los vectores unitarios de los ejes OX , OY y OZ del sistema de referencia $OXYZ$, se representa en coordenadas homogéneas mediante un vector columna:

$$\mathbf{P} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} aw \\ bw \\ cw \\ w \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \\ 1 \end{bmatrix}$$

Por ejemplo, el vector $2i+3j+4k$ se puede representar en coordenadas homogéneas como $[2,3,4,1]^T$ o como $[4,6,8,2]^T$.

Se define como matriz de transformación homogénea T a una matriz de 4×4 que representa la transformación de un vector de coordenadas homogéneas de un sistema a otro.

$$T = \begin{bmatrix} \mathbf{R}_{3 \times 3} & \mathbf{p}_{3 \times 1} \\ \mathbf{f}_{1 \times 3} & \mathbf{w}_{1 \times 1} \end{bmatrix} = \begin{bmatrix} \text{Rotación} & \text{Traslación} \\ \text{Perspectiva} & \text{Escalado} \end{bmatrix}$$

Podemos considerar que una matriz homogénea se haya compuesta de cuatro submatrices de distinto tamaño: una submatriz $\mathbf{R}_{3 \times 3}$ que corresponde a una matriz de rotación; una submatriz $\mathbf{p}_{3 \times 1}$ que corresponde a un vector de traslación; una submatriz $\mathbf{f}_{1 \times 3}$ que representa una transformación de perspectiva y una submatriz $\mathbf{w}_{1 \times 1}$ que representa un escalado global. En robótica sólo interesa conocer el valor de $\mathbf{R}_{3 \times 3}$ y de $\mathbf{p}_{3 \times 1}$ considerando las componentes $\mathbf{f}_{1 \times 3}$ nulas y las de $\mathbf{w}_{1 \times 1}$ la unidad. Al tratarse de una matriz de 4×4 los vectores sobre los que se aplique deberán contar con 4 dimensiones, que serán las coordenadas homogéneas del vector tridimensional de que se trate.

Aplicación de matrices homogéneas.-

$$T = \begin{bmatrix} \mathbf{R}_{3 \times 3} & \mathbf{p}_{3 \times 1} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \text{Rotación} & \text{Traslación} \\ 0 & 1 \end{bmatrix}$$

Representa la orientación y posición de un sistema $O'UVW$ rotado y trasladado con respecto a un sistema de referencia $OXYZ$. Esta matriz sirve para conocer las coordenadas (r_x, r_y, r_z) del vector r en el sistema $OXYZ$ a partir de sus coordenadas (r_u, r_v, r_w) en el sistema $O'UVW$:

$$\begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix} = T \begin{bmatrix} r_u \\ r_v \\ r_w \\ 1 \end{bmatrix}$$

También se puede utilizar para expresar la rotación y traslación de un vector respecto de un sistema de referencia fijo $OXYZ$, de tal manera que un vector r_{xyz} rotado según $\mathbf{R}_{3 \times 3}$ y trasladado según $\mathbf{p}_{3 \times 1}$ se convierte en el vector r'_{xyz} dado por:

$$\begin{bmatrix} r'_x \\ r'_y \\ r'_z \\ 1 \end{bmatrix} = T \begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix}$$

En resumen, una matriz de transformación homogénea se puede aplicar para:

- Representar la posición y orientación de un sistema girado y trasladado $O'UVW$ con respecto a un sistema fijo de referencia $OXYZ$.
- Transformar un vector expresado en coordenadas con respecto a un sistema $O'UVW$ a su expresión en coordenadas del sistema de referencia $OXYZ$.
- Rotar y trasladar un vector respecto a un sistema de referencia fijo.

Traslación.- Suponer que el sistema $O'UVW$ únicamente se encuentra trasladado un vector $\mathbf{p} = p_x \mathbf{i} + p_y \mathbf{j} + p_z \mathbf{k}$ con respecto al sistema $OXYZ$. La matriz T corresponderá a una matriz homogénea de traslación:

$$T(\mathbf{p}) = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Un vector \mathbf{r} cualquiera representado en el sistema $O'UVW$ por r_{uvw} , tendrá como componentes del vector con respecto al sistema $OXYZ$:

$$\begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_u \\ r_v \\ r_w \\ 1 \end{bmatrix} = \begin{bmatrix} r_u + p_x \\ r_v + p_y \\ r_w + p_z \\ 1 \end{bmatrix}$$

Rotación.- Suponer ahora que el sistema $O'UVW$ sólo se encuentra rotado respecto al sistema $OXYZ$, la submatriz de rotación $R_{3 \times 3}$ será la que defina la rotación. Se pueden definir tres matrices homogéneas básicas de rotación según se realice ésta respecto a cada uno de los tres ejes coordenados OX , OY y OZ del sistema de referencia $OXYZ$:

$$T(\mathbf{x}, \alpha) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \alpha & -\text{sen } \alpha & 0 \\ 0 & \text{sen } \alpha & \cos \alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T(\mathbf{y}, \varphi) = \begin{bmatrix} \cos \varphi & 0 & \text{sen } \varphi & 0 \\ 0 & 1 & 0 & 0 \\ -\text{sen } \varphi & 0 & \cos \varphi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T(\mathbf{z}, \theta) = \begin{bmatrix} \cos \theta & -\text{sen } \theta & 0 & 0 \\ \text{sen } \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.4.1 Composición de matrices homogéneas.- Una de las principales utilidades de las matrices homogéneas es que podemos hacer composiciones para describir diversos giros y traslaciones sobre un sistema de referencia especificado, por ejemplo, una matriz que representa un giro de un ángulo α sobre el eje OX , seguido de un giro de ángulo φ sobre el eje OY y de un giro de un ángulo θ sobre el eje OZ , puede obtenerse por la composición de las matrices básicas de rotación:

$$T=T(z, \theta) T(y, \varphi) T(x, \alpha)=\begin{bmatrix} C\theta & -S\theta & 0 & 0 \\ S\theta & C\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C\varphi & 0 & S\varphi & 0 \\ 0 & 1 & 0 & 0 \\ -S\varphi & 0 & C\varphi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha & -S\alpha & 0 \\ 0 & S\alpha & C\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Donde $C\theta$ representa $\text{Cos } \theta$ y $S\theta$ representa $\text{Sen } \theta$

Debido a que el producto de matrices no es conmutativo, tampoco lo es la composición de transformaciones. El ejemplo anterior los ejes sobre los que se realizan las operaciones corresponden al sistema fijo de referencia $OXYZ$. También es posible componer matrices de transformación de manera que las operaciones estén referidas en todo momento al sistema que esté moviéndose. Para ello bastará ir concatenando matrices en orden inverso

De manera general, a la hora de componer diversas transformaciones mediante matrices homogéneas, se han de tener en cuenta los siguientes criterios:

- 1) Si el sistema fijo $OXYZ$ y el sistema transformado $O'UVW$ son coincidentes la matriz homogénea de transformación será la matriz 4x4 identidad I_4 .
- 2) Si el sistema $O'UVW$ se obtiene mediante rotaciones y traslaciones definidas con respecto al sistema fijo $OXYZ$, la matriz homogénea que representa cada transformación se deberá premultiplicar sobre las matrices de las transformaciones previas.
- 3) Si el sistema $O'UVW$ se obtiene sobre rotaciones y traslaciones definidas con respecto al sistema móvil, la matriz homogénea que representa cada transformación se deberá postmultiplicar sobre las matrices de las transformaciones previas.

En el Apéndice A encontrará un ejemplo resuelto con los criterios 2 y 3 descritos en el párrafo anterior, en él se demuestra que ambos producen el mismo resultado. Pudimos haber utilizado cualquiera de ellos en la implementación de nuestro sistema, utilizamos el criterio 3, dada la naturalidad que presenta en la ejecución de los productos matriciales al efectuarse de izquierda a derecha. No se aplica el criterio 1 en el ejemplo ya que no corresponde al caso especial descrito en él.

4.5 Cálculo de posición y orientación de una pieza.- En base al diseño conceptual del sistema (véase Figura 3.3), el SVA nos proporciona el nombre de la pieza en estudio, su centro de gravedad o posición y su orientación o ángulo en grados (véase Apartado 5.1.1.4) respecto a una línea paralela al eje Y_2 del sistema $X_2Y_2Z_2$, que es el sistema coordinado del campo visual sobre la mesa de trabajo y que es tomado por la cámara cuando el PLC detecta la presencia de la charola al momento que pasa sobre la banda transportadora y bajo el campo visual de la cámara. En la Fig. 4.6 presentamos los

sistemas coordenados $X_0Y_0Z_0$ ó base del robot, $X_1Y_1Z_1$ ó sistema coordenado de la mesa de trabajo, y el $X_2Y_2Z_2$ comentado en esta sección.

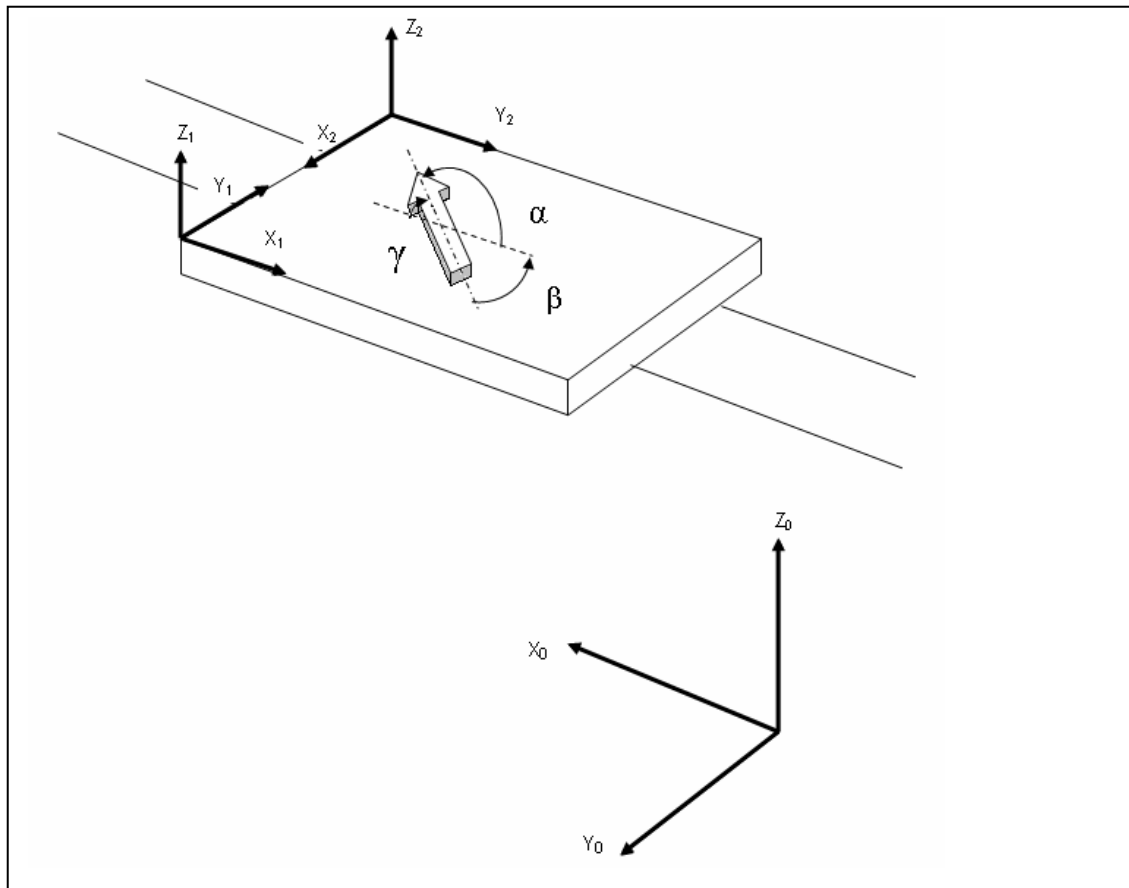


Figura 4.6 Configuración en la estación de ensamble, charola, banda transportadora, sistemas coordenados base del robot, mesa de trabajo y campo visual tomado por la cámara.

Sea α el ángulo que el eje de mayor elongación de la pieza forma con la paralela al eje OY_2 y que pasa por el centro de gravedad de la pieza. Cabe hacer mención que éste ángulo proporcionado por el SVA, toma en cuenta el sentido correcto de la pieza dispuesta sobre la mesa y no tan sólo la dirección que éste eje de mayor elongación tiene respecto a la horizontal [12-13], si observamos la Fig. 4.6 aparece una pieza de trabajo en forma de flecha, se diría, que el ángulo β representa la inclinación de la pieza, esto sin dar importancia al sentido que tenga, de ese modo sería indistinta la posición de la pieza así dispuesta a que si estuviera rotada un ángulo de 180° en relación a la posición en la que se encuentra. Abordaremos más sobre el tema cuando se discuta en el próximo capítulo el SVA.

Para posicionar un objeto en el espacio tridimensional se requieren seis parámetros o grados de libertad, los primeros tres corresponden a las coordenadas x,y,z del efector final y los restantes son para posicionar el efector final, y particularmente la herramienta sobre la pieza a manipular, son tres ángulos: alabeo, cabeceo y guiñada, mejor conocidos por su denominación en inglés como *roll*, *pitch* y *yaw* respectivamente. Alabeo es el giro del efector final respecto al eje local en la muñeca OZ (eje J_6 del robot) véase Fig. 4.7, cabeceo es el

giro del efector final respecto al eje local en la muñeca OY el robot Mitsubishi modelo RV-2AJ no tiene implementado este giro. Guiñada es el giro del efector final respecto al eje local en la muñeca OX (eje J5). El robot Mitsubishi RV-2AJ es un robot de cinco grados de libertad ya que no cuenta con el eje J4.

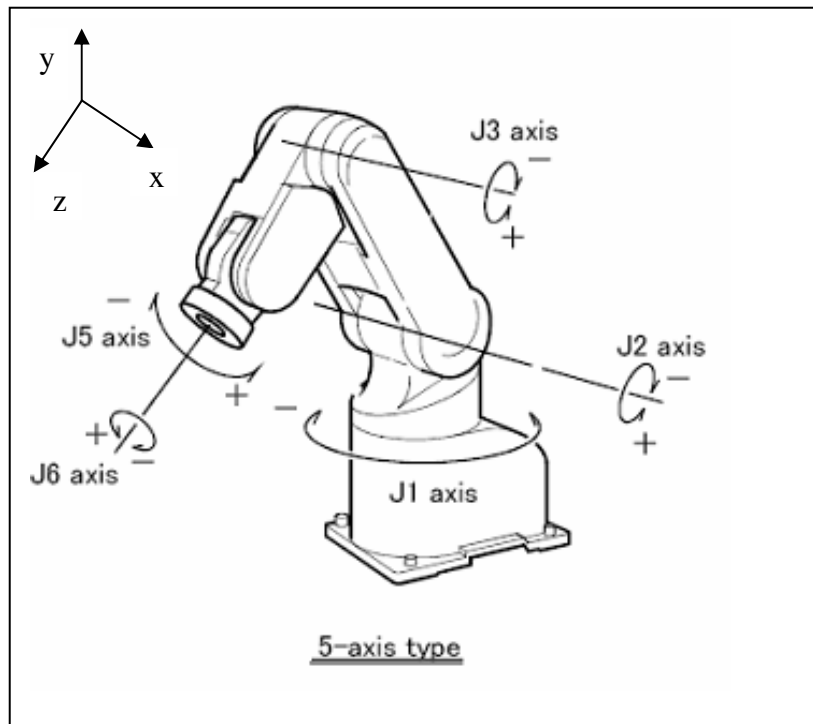


Figura 4.7 Sistema coordinado del efector final con sus dos giros.

Esto permite mejorar el desempeño en su control al simplificar el sistema reduciendo un grado de libertad la complejidad del sistema tomando en cuenta que no pierde generalidad ya que se adapta a la mayor parte de los entornos industriales, para nuestro proyecto resulta conveniente ya que trabajaremos sobre la mesa de trabajo que es una superficie paralela al plano OXY del sistema base del robot. De esta manera decimos que nuestro modelo de robot utiliza para posicionar y orientar una pieza las coordenadas x, y, z del centro de gravedad de la pieza, el ángulo alabeo y finalmente la guiñada. La guiñada en las operaciones de “tomar y dejar” que llevaremos a cabo en el proceso de ensamble de piezas, será siempre el mismo y con un valor de 180° , haciendo que el eje OZ del sistema local del efector final permanezca siempre vertical y paralelo al eje OZ del sistema base y dirigido en el sentido negativo de dicho eje.

Como ilustra la Fig. 4.6, los valores que el SVA proporciona en cuanto a las coordenadas (x, y) del centro de gravedad de la pieza, estos están referidos respecto al sistema $X_2Y_2Z_2$. Esto se logra por medio de composición de transformaciones con matrices homogéneas. Dichos puntos son referidos respecto al sistema coordinado de la mesa de trabajo o sistema $X_1Y_1Z_1$ y ésta a su vez con respecto al sistema coordinado base del robot o sistema $X_0Y_0Z_0$. De lo anterior, para posicionar definitivamente la pieza requerimos determinar los últimos dos grados de libertad alabeo y guiñada, debido a que la guiñada es constante durante toda la operación de tomar y dejar con un valor de 180° como lo

explicamos en el párrafo anterior, nos resta calcular el valor del alabeo con el objeto de orientar el efector final para que éste pueda “tomar y dejar” las piezas a manipular.

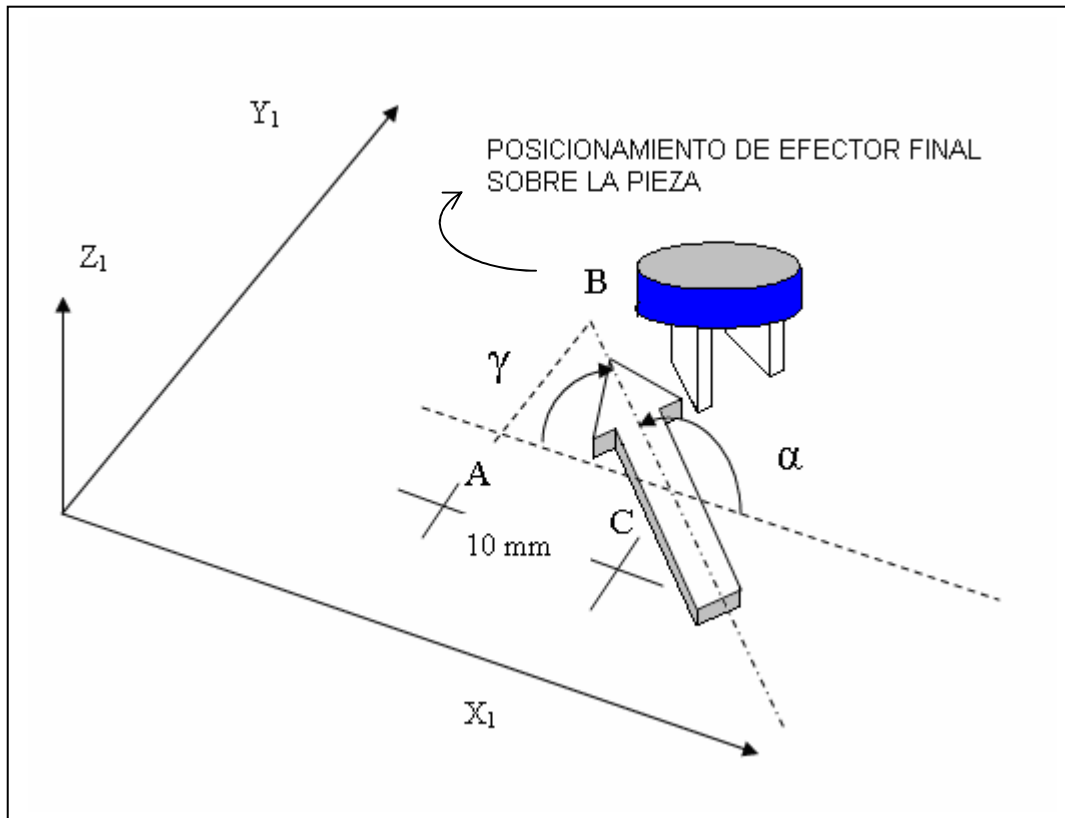


Figura 4.8 El Ángulo γ entre las rectas AC y BC es el valor de alabeo buscado.

Sea C en la Fig. 4.8, el centro de gravedad de la pieza, AC un segmento de recta localizado en el plano X_1Y_1 del sistema coordenado de la mesa de trabajo y cuyo plano es paralelo al plano X_0Y_0 , es también paralela al eje OX del sistema base del robot. Cuando el efector final se posiciona sobre del punto C para tomar una pieza, la referencia a 0° del giro alabeo coincide con esta línea a cada momento mientras el efector final ejecuta los primeros tres grados de libertad para posicionarse, así el ángulo γ en la Fig. 4.8, corresponde al giro alabeo. Este giro es positivo si se hace hacia la derecha de la línea de referencia y negativo si el giro se hace hacia la izquierda.

El objetivo ahora es determinar el ángulo entre las dos rectas AC y BC , véase Fig. 4.8. La recta BC es una línea coincidente con el eje de mayor elongación de la pieza. Para determinar éste ángulo requerimos conocer el valor de las pendientes de ambas rectas, para ello trabajaremos en el plano X_1Y_1 del sistema mesa de trabajo y consideraremos dos casos: cuando el ángulo α tiene un valor entre 90° y 270° , la coordenada x tanto del punto A como del punto B tendrán el mismo valor y las determinamos en base a la coordenada conocida x del punto C (centro de gravedad proporcionado por el SVA) a la que restamos una cantidad arbitraria (por ejemplo 10 mm), la ordenada del punto A es la misma que la del punto C , nos resta calcular la ordenada del punto B . La ecuación de la recta que pasa por dos puntos $P1(x1, y1)$ y $P2(x2, y2)$ es: $y = (y2-y1)/(x2-x1)(x-x1)+y1$. Haciendo uso de ésta ecuación para los puntos B y C , despejamos la única incógnita en la ecuación, la ordenada del punto B . Conocidas las coordenadas de A y B , se calculan las pendientes de

las rectas AC y BC , a las que denominaremos $m1$ y $m2$ respectivamente. Haciendo uso de la siguiente ecuación se determina el ángulo más pequeño entre las dos rectas.

$$\gamma = \left\| \operatorname{atan} \frac{m2-m1}{1 + m1 m2} \right\| \quad (\text{ec. 4.1})$$

Si α tiene un valor entre 90° y 180° , el valor de γ será positivo y el efector final girará a la derecha a partir de la referencia 0° de la línea AC , si α se encuentra entre 180° y 270° , el valor de γ será negativo y el efector final girará a la izquierda de la referencia mencionada, los casos especiales son cuando α vale 90° , γ valdrá 90° también, cuando α vale 180° , γ toma el valor de 0° y el efector final no se mueve, cuando α vale 270° , γ adquiere el valor de -90° y el efector final girará a la izquierda.

Cuando el ángulo α presenta valores mayores a 270° pero menores a 90° , significa que la pieza está orientada hacia el sentido positivo de X_1 y por tanto el cálculo de las abscisas de los puntos A y B se harán sumando y no restando 10mm a la abscisa conocida del punto C , el calculo posterior es similar al caso anterior, sólo que se trabaja con el ángulo suplementario de γ no con el valor mismo del ángulo. Si α está entre 0° y 90° el efector final gira a la derecha un ángulo igual a $180 - \gamma$, si α se encuentra entre 270° y 360° , girará a la izquierda un ángulo igual al suplementario de γ . El caso especial es cuando α vale 0° , el efector final gira 180° y puede ser tanto por la derecha como por la izquierda.

La operación de tomar una pieza por parte del manipulador va seguida por otra operación similar a ésta que es la de dejar, repitiéndose este ciclo para todas y cada una de las piezas que se encuentren en la escena sobre la mesa de trabajo y que previo análisis por parte del SPPM se determina que forman parte del proceso de ensamblado en base a un plan previamente almacenado. En la operación de dejar una pieza se presenta una característica adicional no considerada antes y es que la plantilla de ensamblado podrá ubicarse en cualquier posición y orientación sobre la mesa de trabajo, con esto el cálculo de la orientación final de la pieza debe ser corregido con el valor del ángulo que presenta la posición final de la plantilla respecto del eje OZ de la mesa de trabajo.

Hasta aquí se ha dado la descripción de la configuración del sistema. En el siguiente capítulo se abordará el estudio del Subsistema de Visión Artificial (SVA), el cual determina que piezas aparecen en la escena de trabajo y su ubicación exacta.

Capítulo 5

Subsistema de Visión Artificial (SVA)

El presente capítulo inicia con una breve descripción general de un sistema de visión artificial genérico para dar paso a la descripción particular de nuestro sistema de visión. Se trata brevemente que es y en que consiste un sistema de visión artificial, cuales son los elementos básicos que lo constituyen y los procesos inmersos en ellos. En cuanto a nuestro sistema se describe a detalle como funciona. Parte importante del sistema es el papel que diversas funciones de las librerías de procesamiento digital de imágenes de MatLab presentadas en las librerías IPT (Image Processing ToolBox) y DIPUM (Digital Image Processing Using MatLab) desempeñan, explicándose su operación, características y como han sido utilizadas.

El Subsistema de Visión Artificial tiene la finalidad de reconocer y ubicar en forma correcta cada una de las piezas que se localizan en la mesa de trabajo o charola que se desplaza por medio de la banda transportadora. El SPPM (Subsistema de Programación de Procesos de Manufactura) llama a trabajar al SVA durante la implementación de nuestro proyecto y tras obtener datos de identidad, posición y orientación de cada pieza que se localiza en la escena sobre la mesa de trabajo, deberá planificar la tarea de ensamble y transmitir al control del robot información necesaria para ejecutar las operaciones “tomar y dejar” sobre las piezas realizando así la tarea de ensamble previamente programada.

5.1 Sistemas de visión artificial.- La visión artificial puede entenderse como “*Los procesos de obtención, caracterización e interpretación de información de imágenes tomadas de un mundo tridimensional*” [14] y son estos procesos los que constituyen verdaderos sistemas autónomos de percepción que a su vez se pueden subdividir en tres niveles y éstos en seis etapas principales, véase Fig. 5.1:

Las etapas ejecutadas por un sistema de visión genérico son:

- Adquisición de imagen.
- Preproceso de imagen.
- Segmentación de la escena.
- Representación y descripción.
- Reconocimiento.
- Interpretación.

Los cuales corresponden a tres grandes niveles que tiene un sistema de visión y que son:

- Visión de bajo nivel
- Visión de nivel intermedio
- Visión de alto nivel

La correspondencia entre niveles y sus etapas, véase Tabla 5.1.

ETAPA	NIVEL
Adquisición de Imagen	Bajo
Preproceso de Imagen	Bajo
Segmentación	Intermedio
Representación y descripción	Intermedio
Reconocimiento	Alto
Interpretación	Alto

Tabla 5.1 Correspondencia entre niveles y sus etapas en un sistema de visión artificial.

Una representación esquemática del contenido de la Tabla 5.1 la podemos apreciar en la Figura 5.1.

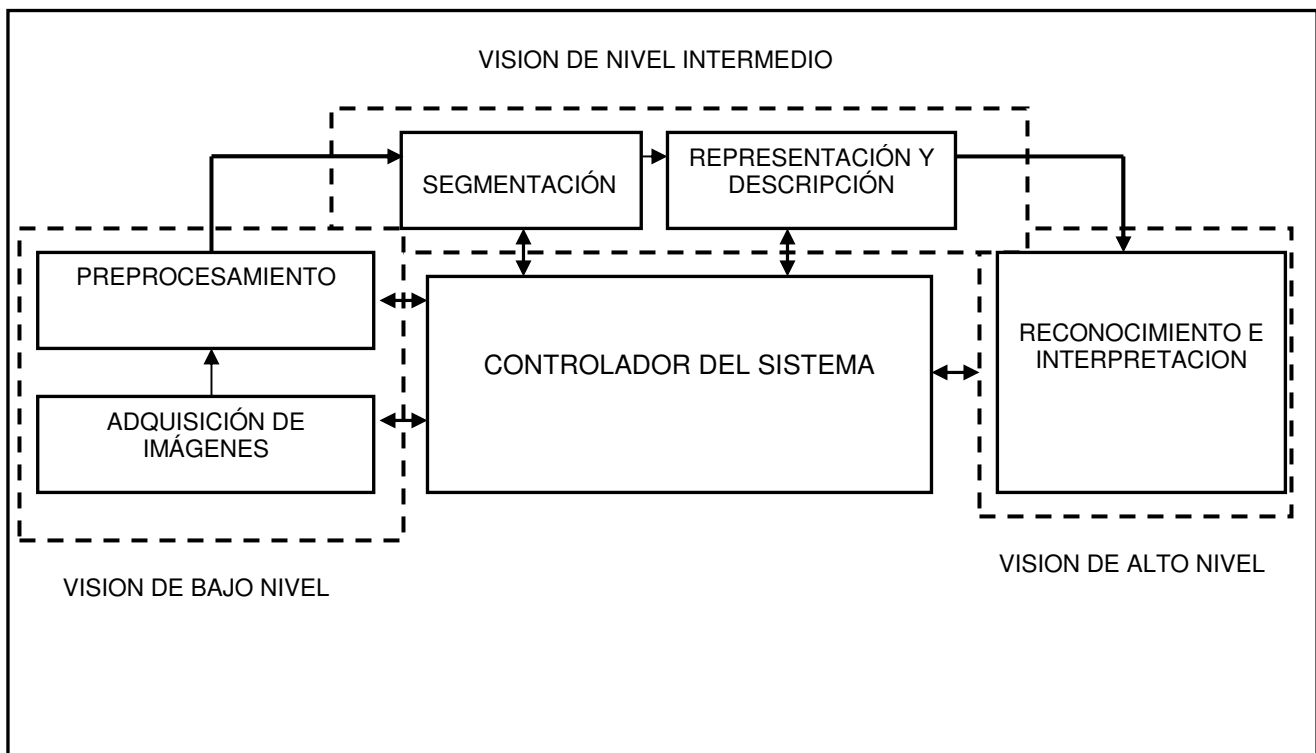


Figura 5.1 Representación esquemática de un sistema de visión genérico.

5.1.1 Descripción de cada nivel y etapa.

5.1.1.1 Visión de bajo nivel.- Son funciones consideradas como reacciones automáticas, no requieren inteligencia por parte del sistema de análisis de imágenes. La adquisición y el preproceso de imágenes son funciones que operan a este nivel. **La adquisición de imagen** consiste en obtener la representación de la escena mediante un arreglo de elementos fotosensitivos (sensores de imagen) cuyo nivel de intensidad lumínica es convertido a valores numéricos mediante un proceso llamado cuantización. Este proceso consiste en asignar tonos o niveles de gris a cada elemento del arreglo de imagen (proceso

llamado muestreo espacial) en un rango preestablecido según la resolución utilizada por ejemplo de 0 a 255, donde los valores extremos representan el negro y blanco intenso respectivamente. Este proceso de barrer la imagen completa lo realiza en frecuencias del orden de los 75 ciclos por segundo aproximadamente, también llamado razón de muestreo.

5.1.1.2 Las técnicas de preproceso.- son operaciones en el dominio espacial o en el dominio de las frecuencias [3,10,14], que tienen por objeto mejorar la calidad de imagen adquirida en cuanto a brillo y contraste o bien hacer resaltar ciertas características tales como contornos o bordes de los objetos en la escena, suavizado de imagen, eliminación de ruido como puntos aislados o datos espurios provenientes de las operaciones de muestreo y cuantización de imagen, procesos ya mencionados en esta misma sección.

La imagen resultante finalmente se representa mediante un arreglo bidimensional de M renglones por N columnas en un sistema coordenado que tiene su origen en la esquina superior izquierda con ejes X en el sentido de los renglones o sentido vertical y el eje Y en el sentido de las columnas o sentido horizontal, cada elemento de imagen se le llama píxel, véase Fig. 5.2.

Esta etapa termina en el momento que se obtiene una imagen binaria, es decir una imagen con solo dos tonos negro para el fondo y blanco para los píxeles que forman las imágenes de los objetos en la escena. Para lograr esto se recurre a una técnica de preproceso llamada binarización, la cual basada en la distribución estadística de los tonos de gris (histogramas de intensidad) de los puntos de la imagen original y sus frecuencias, se determina el tono de gris que mejor separa los cúmulos en la curva de distribución que corresponden al fondo respecto a los que corresponden a los objetos en la escena [15]. Así los píxeles en la imagen original que tengan un tono menor a ese valor límite o también llamado valor de umbral, tomarán en la imagen binaria el valor de 0, mientras que los píxeles con valores iguales o mayores tendrán en la imagen binaria el valor de 1.

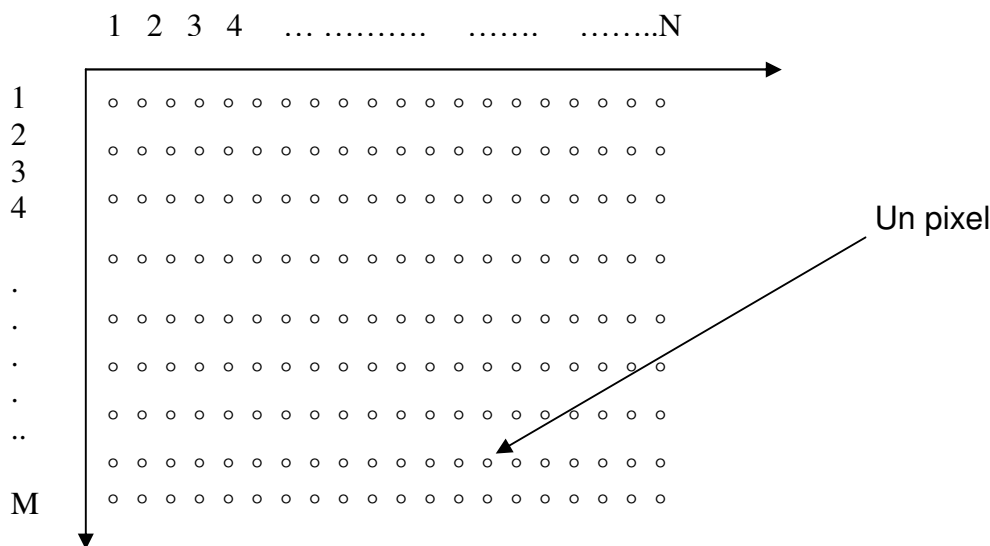


Figura 5.2 Arreglo bidimensional $[M \times N]$ de elementos de imagen.

5.1.1.3 Visión de nivel intermedio.- Consiste en la extracción y caracterización de componentes de la imagen (i.e. regiones) que se obtiene de un proceso de bajo nivel. La segmentación y la descripción se consideran funciones que pertenecen a este nivel. **La segmentación** consiste en separar los diferentes objetos que componen una escena como elementos inconexos del fondo. Una forma de hacerlo es utilizando imágenes que contienen solo dos tonos blanco y negro también llamadas imágenes binarias cuyos elementos son 0 y 1, el 0 para etiquetar el fondo de la escena y 1 para todos aquellos píxeles que forman parte de un objeto. El resultado de la segmentación es un conjunto de n piezas o regiones (que para nuestro caso corresponden a piezas u objetos por ensamblar) cada una de ellas etiquetada con un número, es decir todos los píxeles que están conectados entre sí formando un objeto tienen una misma etiqueta, véase Fig. 5.3.

Las técnicas más comunes [14,16] en segmentación son:

- Búsqueda de las partes uniformes (similitud).
- Búsqueda de las zonas donde se produce un cambio(discontinuidad)
- Segmentación de regiones:
 - § Umbralización.
 - § Crecimiento de regiones.
 - § Partir y unir.

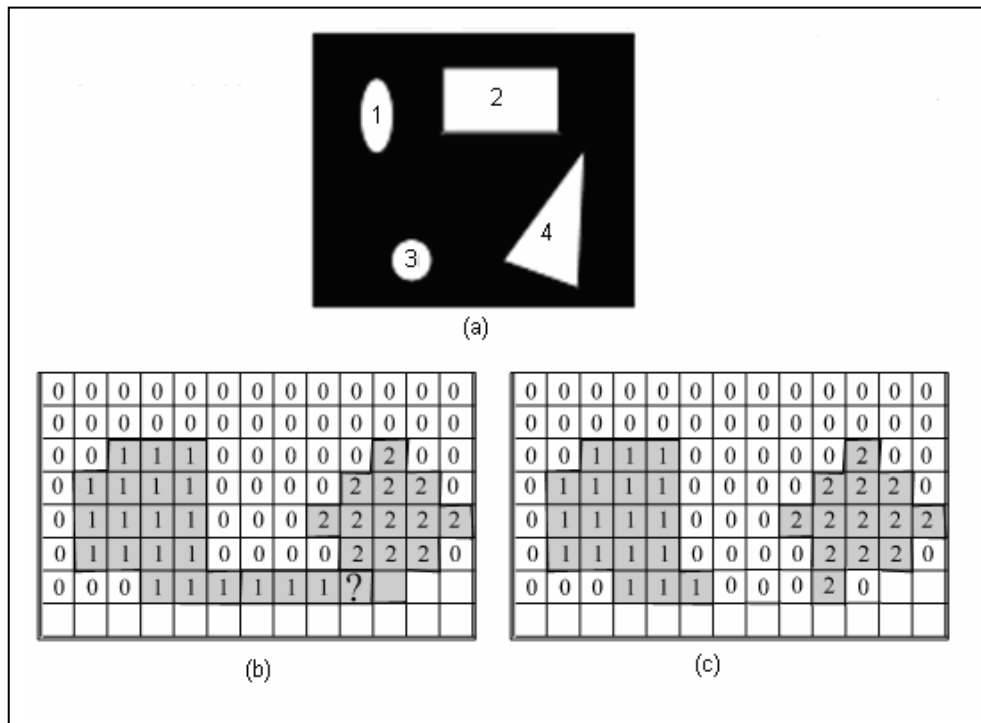


Figura 5.3 (a) Escena segmentada en cuatro regiones. (b) Existencia de ambigüedades, (c) Etiquetado de dos segmentos en la escena.

La técnica que utiliza el grupo de funciones DIPUM que utilizaremos en nuestro proyecto pertenece al de crecimiento de regiones [10], que se caracteriza por partir de puntos iniciales llamados puntos semilla recorre la imagen completa siguiendo un orden establecido

y busca vecinos que tengan la misma propiedad, en este caso de conectividad (4 conectada u 8 conectados, dependiendo si consideramos a un pixel ubicado en cualquier posición norte, sur, este u oeste respecto de uno central, o bien si además tomamos en cuenta sus posiciones intermedias, que es el caso de 8 conectados). Si los puntos están conectados, pertenecen a la misma región y comparten propiedades similares a los puntos iniciales, de no ser así, pueden formar una nueva región. Se continúa con el recorrido visitando el resto de los puntos de la imagen. Se puede llegar a un punto en el cual se determine que ambas regiones pertenecen a una sola región y no a dos en la misma escena dependiendo del criterio de conectividad adoptado, véase Fig. 5.3 (b). Este método iterativo termina cuando ya no existen puntos en la escena por analizar.

Después de segmentar una imagen en regiones, el conjunto resultante de píxeles segmentados se representa y describe en forma adecuada para su posterior proceso.

Representar una región implica dos posibilidades:

- 1) Hacerlo en términos de sus características externas (contornos).
- 2) Hacerlo en términos de sus características internas (píxeles que comprenden la región).

5.1.1.4 Descripción.- Consiste en obtener una serie de valores característicos de un objeto que serán utilizados para su posterior reconocimiento [14]. Por regla general, los descriptores deben ser independientes del tamaño, localización y orientación que tenga el objeto, es decir, deben ser invariantes a cambios de escala, traslación y rotación, además deben contener suficiente información de discriminación para diferenciar un objeto de otro.

La descripción es un tema central en el diseño de los sistemas de visión, ya que los descriptores influyen en la complejidad y rendimiento de los algoritmos de reconocimiento. Los descriptores se clasifican en descriptores de frontera y descriptores de región [10].

Los descriptores de frontera son los que describen un objeto en base a los puntos que se localizan en su contorno o frontera, tales como:

- Códigos de cadena.- Representan una frontera mediante un conjunto de segmentos de longitud y dirección especificadas.
- Signaturas.- Representación funcional unidimensional de una frontera, tales como la representación de la distancia del centro a la frontera como una función de un ángulo.
- Números de contorno.- Estudia diversas características de una frontera codificada en forma de cadena, tales como: número de contorno, eje mayor, eje menor, excentricidad de frontera, etc.
- Descriptores de Fourier.- Utilizan la transformada de Fourier unidimensional discreta para describir una frontera bidimensional.

Descriptores de Región.- Describen un objeto mediante sus características internas, a diferencia de los anteriores que se basan en características de sus fronteras.

Gran número de sistemas de visión industrial se basan en descriptores de región de naturaleza sencilla y por tanto atractivos computacionalmente hablando [14], alguno de ellos son:

- Área.- Se define como el número de píxeles en el interior de su frontera, útil cuando la geometría de la imagen es fija y los objetos se ubican a una misma distancia de la cámara.
- Ejes mayor y menor.- Útiles para obtener la orientación de un objeto, El cociente de las longitudes de éstos ejes se denomina excentricidad de la región y es un descriptor importante de la forma del objeto.
- Perímetro.- Es la longitud de la frontera. Suele utilizarse con mayor frecuencia mediante el concepto de compacidad, se define como la razón del cuadrado del perímetro a su área (p^2 / A).
- Número de Euler.- Descriptor de forma o también llamado topológico, se define como la diferencia del número de regiones conectadas menos el número de huecos. De tal manera que la letra (A) tiene un número de Euler de $1-1=0$, mientras en la letra (B) será $1-2= - 1$.
- Momentos invariantes.- Descriptores de región invariantes a cambios de escala, traslación y rotación.

Momentos Invariantes.- Los momentos 2-D de orden $(p+q)$ de una imagen digital $f(x,y)$ se define como:

$$m_{pq} = \sum_x \sum_y x^p y^q f(x,y) \quad (\text{ec. 5.1})$$

Para $p,q=0$, la sumatoria de los puntos o elementos de imagen $f(x,y)$ representa el área

$$m_{00} = \sum_x \sum_y x^0 y^0 f(x,y) = \sum_x \sum_y f(x,y) = \text{Área} \quad (\text{ec. 5.1.1})$$

$$m_{10} = \sum_x \sum_y x^1 y^0 f(x,y) = \sum_x \sum_y x f(x,y) \quad (\text{ec. 5.1.2})$$

$$m_{01} = \sum_x \sum_y x^0 y^1 f(x,y) = \sum_x \sum_y y f(x,y) \quad (\text{ec. 5.1.3})$$

$$\bar{x} = m_{10} / m_{00} \quad ; \quad \bar{y} = m_{01} / m_{00} \quad (\text{ec. 5.2})$$

para $p,q = 0,1,2,\dots$ donde las sumatorias son sobre los valores de las coordenadas x , y espaciales de la imagen. Así mediante las ecuaciones 5.2 obtenemos las coordenadas del centro de masa del objeto. Para lograr independencia a cambios de traslación, ubicaremos nuestro sistema de referencia en el centro de masa del objeto, transformando la ecuación 5.1 en la 5.3. El correspondiente momento central se define como:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x,y) \quad (\text{ec. 5.3})$$

para $p,q = 0,1,2,\dots$

Los momentos centrales deben ser normalizados para obtener independencia a cambios de rotación y de escala [10,12,14]. Los momentos centrales normalizados de orden $(p+q)$ se definen como:

$$\eta_{pq} = \mu_{pq} / \mu_{00}^{\gamma} \quad (\text{ec. 5.4})$$

Donde:

$$\gamma = (p+q)/2 + 1; \quad \text{para } p+q = 2,3,\dots \quad (\text{ec. 5.5})$$

Un conjunto de siete momentos invariantes 2-D insensibles a cambios de escala, traslación y rotación se derivan de estas ecuaciones.

$$\begin{aligned} \Phi_1 &= \eta_{20} + \eta_{02} \\ \Phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\ \Phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\ \Phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\ &\dots \\ &\dots \end{aligned} \quad (\text{ec. 5.6})$$

Utilizaremos los siguientes momentos centrales y normales de orden 2 para determinar el ángulo de inclinación de la pieza, de la ec. 5.4:

$$\eta_{20} = \mu_{20} / \mu_{00}^2$$

$$\eta_{02} = \mu_{02} / \mu_{00}^2$$

De la ec. 5.5:

$$\gamma = (2+0)/2 + 1 = 2$$

De la ec. 5.3

$$\mu_{20} = \sum_x \sum_y (x - \bar{x})^2 (y - \bar{y})^0 f(x,y) = \sum_x \sum_y (x - \bar{x})^2 f(x,y) = a \quad (\text{ec. 5.7})$$

$$\mu_{02} = \sum_x \sum_y (x - \bar{x})^0 (y - \bar{y})^2 f(x,y) = \sum_x \sum_y (y - \bar{y})^2 f(x,y) = c \quad (\text{ec. 5.8})$$

$$\mu_{11} = \sum_x \sum_y (x - \bar{x})^1 (y - \bar{y})^1 f(x,y) = b \quad (\text{ec. 5.9})$$

$$\mu_{00} = \sum_x \sum_y (x - \bar{x})^0 (y - \bar{y})^0 f(x,y) = \sum_x \sum_y f(x,y) = m_{00} \quad (\text{ec. 5.10})$$

5.1.1.5 Ángulo de inclinación de las piezas.- La distancia de un pixel cualquiera que pertenezca a una región respecto a una línea recta (véase Fig. 5.4) se define por:

$$r = x \operatorname{sen}\theta - y \operatorname{cos}\theta + \rho \quad (\text{ec. 5.11})$$

Donde θ es la inclinación de la línea respecto al eje X y ρ es la distancia perpendicular de la línea respecto al origen del sistema coordenado XY .

Si un objeto es alargado tendrá un eje natural en la dirección de la elongación y será un eje respecto al cual el objeto tenga el menor momento de inercia (momento de orden 2). En ingeniería mecánica se hace referencia a los ejes de mayor y menor inercia como los ejes principales de un objeto. Podemos utilizar la dirección del eje de menor inercia para determinar la dirección del objeto [17]. Esto falla sólo si no se tiene un único eje de menor inercia, tal como ocurre con un objeto simétrico central (cuadrado, disco, etc.).

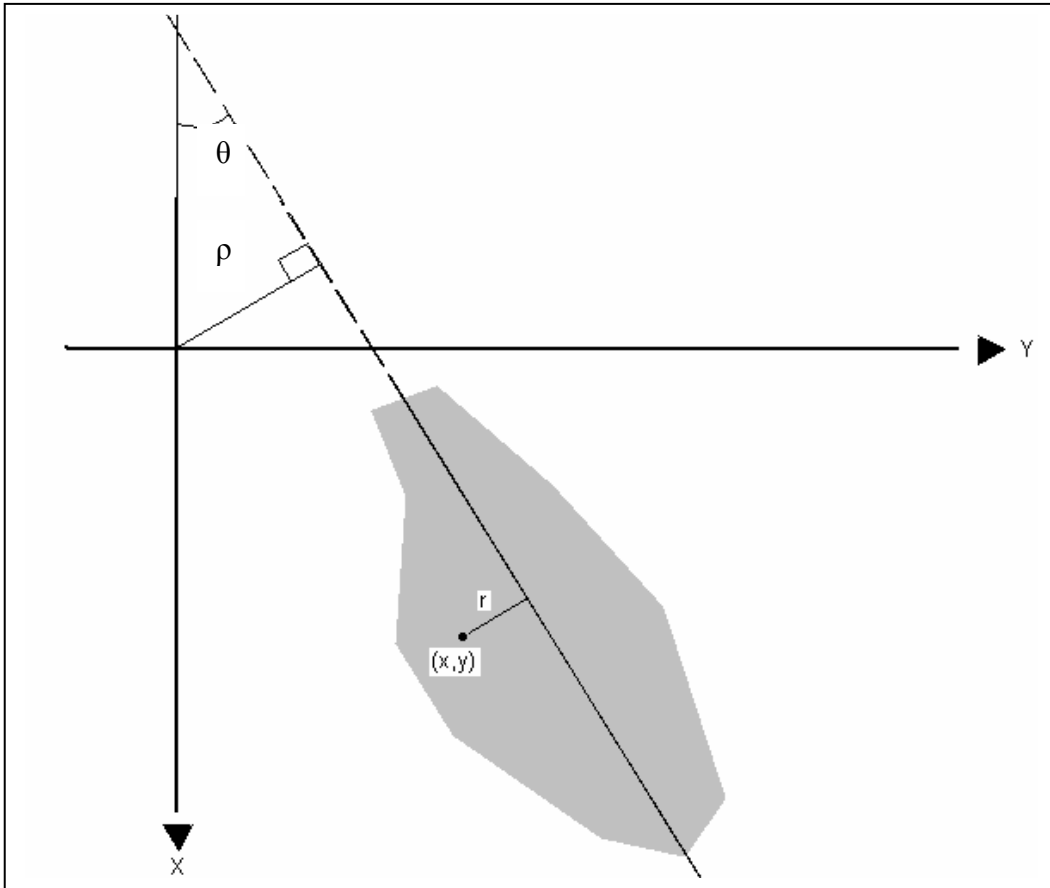


Figura 5.4 Distancia de un punto en la región a una recta.

El eje de menor inercia pasa a través del centro de masa de la región a considerar, si trasladamos el origen del sistema coordenado a éste punto, la distancia ρ del nuevo origen al eje de menor inercia es cero. El momento de inercia de la región respecto a la línea recta con inclinación θ respecto al eje X es:

$$I = \sum_x \sum_y f(x,y) r^2 \quad (\text{ec. 5.12})$$

De ecs. 5.10 y 5.11 aplicada a la ec. 5.12, tenemos:

$$I = \sum_x \sum_y f(x,y) (x' \text{sen}\theta - y' \text{cos}\theta + \rho)^2 \quad (\text{ec. 5.12.1})$$

Dado que trasladamos el origen del sistema coordenado al centro de masa

$$\rho=0, \quad x' = x - \bar{x}, \quad y' = y - \bar{y}, \quad \text{por lo tanto:}$$

$$I = \text{sen}^2\theta \left[\sum_x \sum_y x'^2 f(x,y) \right] - 2 \text{sen}\theta \cos\theta \left[\sum_x \sum_y x' y' f(x,y) \right] + \cos^2\theta \left[\sum_x \sum_y y'^2 f(x,y) \right] \quad (\text{ec. 5.13})$$

Sustituyendo las ecuaciones 5.7) a 5.9) tenemos:

$$I = a \text{sen}^2\theta - 2b \text{sen}\theta \cos\theta + c \cos^2\theta \quad (\text{ec. 5.14})$$

Podemos encontrar la orientación θ de los ejes de mayor y menor momento de inercia derivando la ec. 5.14 respecto a θ e igualando a cero el resultado.

$$d(I)/d\theta = 0 \quad (\text{ec. 5.15})$$

$$d(I)/d\theta = a(2 \text{sen}\theta \cos\theta) - 2b(\text{sen}\theta(-\text{sen}\theta) + \cos\theta \cos\theta) + c(2 \cos\theta)(-\text{sen}\theta)$$

$$0 = 2 \text{sen}\theta \cos\theta (a - c) - 2b (\cos^2\theta - \text{sen}^2\theta) \quad (\text{ec. 5.15.1})$$

Haciendo uso de las siguientes identidades trigonométricas:

$$\text{sen}2\theta = 2 \text{sen}\theta \cos\theta \quad (\text{ec. 5.15.2})$$

$$\cos2\theta = \cos^2\theta - \text{sen}^2\theta \quad (\text{ec. 5.15.3})$$

Tenemos:

$$0 = \text{sen}2\theta (a - c) - 2b (\cos2\theta) \quad (\text{ec. 5.16})$$

$$\text{sen}2\theta / \cos2\theta = 2b / (a - c) = \tan 2\theta \quad (\text{ec. 5.16.1})$$

$$\theta = (1/2) \text{atan} (2b / (a-c)) \quad (\text{ec. 5.17})$$

5.1.1.6 Visión de nivel alto.- Comprende el reconocimiento y la interpretación. El problema de **reconocimiento** de objetos se puede definir como problema de etiquetado basado en modelos de objetos conocidos [10,14]. Dada una imagen que contiene uno o más objetos de interés (además del fondo) y una base de conocimiento de etiquetas correspondientes a un conjunto de modelos conocidos, el sistema deberá asignar etiquetas correctas a las regiones o componentes de la imagen.

5.1.1.7 Reconocimiento de patrones.- Un patrón es un arreglo de descriptores [10]. El concepto característica, se usa en la literatura de reconocimiento de patrones para denotar un descriptor. Clase de patrones es una familia de patrones que comparten un conjunto de propiedades comunes, se denotan por w_1, w_2, \dots, w_W , donde W es el número de clases. Reconocimiento de patrones por máquinas comprende técnicas para asignar patrones a sus respectivas clases automáticamente. Las dos principales estructuras para patrones son vectores (para descriptores cuantitativos) y cadenas (para descriptores estructurales).

$$x^T = [x_1, x_2, \dots, x_n] \quad (\text{ec. 5.18})$$

Donde: x^T denota transpuesta del vector x , x_i representa el i -ésimo descriptor y n el total de descriptores asociados con cada patrón.

Distancia Euclidiana.- Entre dos vectores X y Y de dimensión n , es definido por el escalar :

$$d(x, y) = \|x - y\| = \|y - x\| = [(x_1 - y_1)^2 + \dots + (x_n - y_n)^2]^{1/2} \quad (\text{ec. 5.19})$$

Esta expresión es simplemente la Norma de la diferencia entre los dos vectores.

Reconocimiento basado en métodos de decisión teórica.- Son métodos que se basan en el uso de funciones de decisión (también llamados discriminantes).

Hagamos $x = (x_1, x_2, \dots, x_n)^T$ represente un vector patrón n -dimensional, para W clases de patrón w_1, w_2, \dots, w_W , el problema básico de reconocimiento de patrones de decisión teórica es encontrar W funciones de decisión $d_1(x), d_2(x), \dots, d_W(x)$, con la propiedad que si un patrón x pertenece a la clase w_i , entonces:

$$d_i(x) > d_j(x), \quad j=1, 2, \dots, W; \quad j \neq i \quad (\text{ec. 5.20})$$

O sea, un patrón desconocido x , se dice pertenece a la i -ésima clase si, sustituyendo x en todas las funciones de decisión, $d_i(x)$ es el valor numérico mayor.

5.1.1.8 Reconocimiento usando clasificador de distancia mínima.- Suponer que cada clase patrón w_j es caracterizado por un vector de media m_j , usamos el vector media de cada población de vectores de entrenamiento como representativos de esa clase de vectores.

$$m_j = \frac{1}{N_j} \sum_{x \in w_j} x, \quad j = 1, 2, \dots, W \quad (\text{ec. 5.21})$$

Donde N es el número de vectores patrón de entrenamiento para la clase w_j , W es el número de clases patrón. Una manera de determinar la clase a la que pertenece un vector patrón desconocido es asignarlo a la clase de su prototipo más cercano usando distancia

euclidiana como una medida de proximidad o similaridad, esto se reduce a calcular medidas de distancia.

$$D_j(x) = \|x - m_j\|, j = 1, 2, \dots, W \quad (\text{ec. 5.22})$$

Asignamos x a la clase w_j si $D_j(x)$ es la más pequeña. La distancia más pequeña indica la mejor comparación.

Los vectores promedio se organizan como renglón de una matriz M , entonces el cálculo de distancia de un patrón arbitrario x a cada uno de los vectores promedio es la raíz cuadrada de la sumatoria de las distancias calculadas por la ec. 5.19. Debido a que todas las distancias son positivas, se puede simplificar la expresión ignorando la operación raíz cuadrada, así la distancia mínima determina la clase perteneciente del vector x .

5.2 Implementación del SVA.- Una vez vistos los elementos en los que divide un sistema de visión, nos enfocaremos en esta sección a mostrar la manera de implementar nuestro propio sistema de visión en base a los módulos ya programados de las librerías IPT y DIPUM, (véase apartado 3.2.2.1).

Algunos autores [3,14] consideran a los sistemas de iluminación como parte de la etapa de adquisición de imagen, ya que una mala calidad de iluminación en la escena provoca mayor complejidad en el desarrollo de las técnicas que habrán de aplicarse en la etapa de preproceso para su mejoramiento y facilidad de postproceso. Se ha utilizado un sistema de iluminación denominado posterior, consistente en una mesa de acrílico blanco provista en su interior de una fuente luminosa blanca. Este sistema de iluminación produce imágenes con alto nivel de contraste que facilita el proceso de conversión de la imagen de tonos de gris a tener solamente dos tonos blanco y negro (imagen binaria), además elimina sombras que de lo contrario harían imposible la segmentación de la escena.

Funciones de librería IPT y DIPUM utilizadas en el sistema:

5.2.1 Adquisición de imagen.- Con las funciones de librería para lectura y escritura de imágenes, se elaboró la siguiente función que representa en nuestro sistema la etapa de Adquisición de imagen, la cual se llama a operar de la siguiente manera:

$$f = \text{Adq_imagen}(\text{'archivo'}, \text{'ext'}, ri, rf, ci, cf)$$

Obtiene una imagen (f) en tonos de gris, recortada y en formato jpg, dada una imagen original de nombre "archivo", a color y en cualquier formato válido de imagen con extensión 'ext'. La imagen de salida tendrá un tamaño: ($rf-ri$) (donde rf : renglón final, ri : renglón inicial) renglones por ($cf-ci$) (donde cf : columna final, ci : columna inicial) columnas. De tal manera que las coordenadas de la esquina superior izquierda del área recortada respecto a la imagen original son (ri, ci), así como las coordenadas de la esquina inferior derecha serán del área recortada respecto a la imagen original son (rf, cf).

El pseudocódigo para esta función es el siguiente:

- Imagen leída = $\text{imread}(\text{nombre de archivo})$;
- Recortar imagen leída en subimagen ($ri:rf, ci:cf$)

- Grabar en disco la imagen recortada
- Regresa la imagen recortada a la función que llama

5.2.2 Preproceso de imagen.- Al igual que en el punto anterior, se presentan funciones de librería de IPT y de DIPUM que utilizaremos en la implementación de nuestra propia función de preproceso, encargada de mejorar la calidad de las imágenes que contienen las escenas y que servirán de entrada en procesos posteriores de nuestro SVA.

`fg=rgb2gray(f)`

Esta función convierte una imagen (f) a color en una imagen en tonos de gris.

`se=strel('disk',n)`

Esta función tiene el propósito de eliminar ruido para mejorar la calidad de la imagen. Crea una matriz de convolución que se aplicará a la imagen. Toma en cuenta la vecindad en una subárea en forma de disco con un radio de n píxeles que será aplicado como filtro. En nuestro sistema se utilizó un valor n= 2.

`fg1=imclose(imopen (fg, se), se)`

Utiliza la forma estructural generada por strel para aplicarla mediante técnicas de convolución a la imagen por mejorar mediante operaciones morfológicas de erosión y dilatación, véase [10] y Cap.9.

`fbw=im2bw(fg1, graythresh (fg1))`

Convierte una imagen de tonos de gris ya preprocesada a una imagen binaria de solo dos tonos mediante la aplicación de técnicas de histogramas de intensidad de niveles de gris [10,14], que garantiza determinar el tono de gris que mejor separa los objetos del fondo en una imagen, de tal modo que todos los píxeles cuyo nivel de gris sea menor a ese umbral se convertirán en puntos de valor 0 o sea objetos y los píxeles con tonos de gris mayores al umbral serán puntos de valor 1 o píxeles de fondo [15].

$f_{bin} = imcomplement(f_{bw})$

Obtiene la imagen invertida de la imagen de entrada, es decir los píxeles de fondo se convierten a valor 0 y los de los objetos en la escena serán puestos a 1.

Haciendo uso de estas funciones básicas de librería, creamos en nuestro proyecto la función que constituye la etapa de preproceso de imagen y que describimos a continuación:

$fb = Pre_imagen(f)$

Preprocesa la imagen (f) y obtiene como salida una imagen binarizada (fb). Aplica un corrimiento en la distribución de su histograma para considerar la serie de niveles de gris distribuidas entre el 35 y el 100 por ciento de su representación inicial.

Se eligió ése rango para obtener mayor realce de la imagen. Aplica además filtro espacial llamado Top Hat para un suavizado y mejora de la imagen. Posteriormente aplica binarización de la imagen mediante umbral, obteniendo así una imagen con dos tonos, fondo blanco y objetos negros, finalmente invierte la imagen binaria que es regresada a la función que hace la llamada.

5.2.3 Segmentación de imagen.- Posterior al preproceso y binarización de imagen, la siguiente parte del proceso consiste en la segmentación de imagen, para ello disponemos de una función que toma como dato de entrada la imagen binaria y la conectividad a utilizar (4 u 8 conectados), siendo 8 el valor por omisión, y la salida de la función es una matriz de etiquetas L , seguido por la variable n , que nos proporciona el número de elementos o regiones independientes que encontró en la escena. Una limitante en nuestro sistema, las piezas que aparezcan sobre la mesa de trabajo deberán estar separadas, ya que provocaría confusión en las formas.

$[L, n] = bwlabel(f)$

Esta línea del programa constituye la etapa de segmentación de imagen.

5.2.4 Descripción de imagen.- La herramienta fundamental del IPT para cálculo de descriptores de región es la función:

$$D = \text{regionprops}(L, \text{properties})$$

Donde:

L.- Es la matriz de etiquetas obtenida en la segmentación.

D.- Arreglo de estructura de longitud $\max(L(:))$, los campos de la estructura denotan diferentes medidas de cada región especificadas en propiedades (*properties*).

El argumento propiedades puede ser un lista de cadenas de texto separadas por comas, un arreglo conteniendo cadenas, la cadena 'all' o la cadena 'basic'. En el caso de elegir la cadena 'all' entonces todos los descriptores se calculan, si se utiliza la palabra 'basic' los descriptores calculados son: área, centro de gravedad y caja envolvente. La librería IPT usa *x,y* para indicar coordenadas horizontal y vertical con origen en la esquina superior izquierda y los valores se incrementan a la derecha y abajo del origen respectivamente. El siguiente ejemplo muestra la manera de obtener el número de regiones y los valores de sus áreas dada una imagen binaria *B*.

Convierte la imagen *B* a una matriz de etiquetas:

$$B = \text{bwlabel}(B)$$

Obtiene el área y la caja envolvente de cada región:

$$D = \text{regionprops}(B, \text{'area', 'boundingbox'})$$

Extrae el valor de área de cada región:

$$w = [D.Area]$$

Obtiene el número de regiones en la escena:

$$NR = \text{length}(w)$$

Donde los elementos del vector *w* son las áreas de las regiones y *NR* es el número de regiones. Otros descriptores de región útiles para nuestro sistema son:

- Eccentricity.- Escalar cuyo valor es la excentricidad de una elipse con los mismos momentos de segundo orden que la región y se define como la razón de distancias entre el foco de la elipse y su longitud de eje mayor. Su valor varia entre 0 y 1, . 0 para un círculo y 1 para un segmento de recta.
- EquivDiameter.- Es el diámetro de un círculo con la misma área que la región.
- Extent.- Proporción de los píxeles del boundingbox que pertenecen a la región. Escalar que se calcula como el área dividida entre el área del boundingbox.
- Orientation.- Ángulo en grados entre el eje *x*'s y el eje mayor de la elipse que tiene el mismo momento de segundo orden que la región.

- EulerNumber.- Escalar que se calcula como la diferencia entre el número de objetos menos el número de huecos en ellos.
- Solidity.- Escalar que se calcula como la proporción de los píxeles de la concha convexa que pertenecen también a la región. Numéricamente es el cociente del área de la región entre el área de la concha convexa.

Otros descriptores de región muy usados en el área de robótica y visión artificial por más de cuatro décadas son los Momentos Invariantes de HU [18], son 7 y son insensibles a cambios de posición, rotación y de escala, véase Ec. 5.6.

La función que calcula los momentos invariantes se llama *invmoments*().

$$Phi = invmoments (f)$$

Donde *f* es la imagen de entrada y *Phi* es un vector renglón de 7 elementos conteniendo los momentos invariantes definidos.

$$Phi = abs(log (invmoments (f)))$$

El uso de *logaritmo* reduce el rango dinámico y el valor absoluto [10] elimina tener que lidiar con números complejos que resultan del cálculo de *logaritmo* de momentos invariantes negativos. Ya que lo que cuenta aquí es la invarianza de los momentos y no su signo.

Se realizaron varias pruebas para seleccionar los descriptores a utilizar en el sistema. Las pruebas consistieron en leer para cada objeto una serie de imágenes donde cada una de ellas contiene al objeto en posiciones diferentes y para cada imagen se calculan descriptores y las coordenadas del centroide del objeto, los valores se registran en la Tabla 5.2.

Con el propósito de determinar cuales serían los descriptores que mejor se comportan en la fase de descripción de escenas dentro de la implementación del SVA, se utilizó un escáner de cama para tomar imágenes de las piezas hechas a base de figuras de papel y que simularían las piezas reales con las que trabajaría posteriormente el sistema. Cada figura se coloca en una posición y orientación aleatoria, se midió en milímetros la ubicación real del centro de masa, así como el ángulo en grados que la pieza en su eje más alargado forma con la horizontal y que aparecen en la Tabla 5.2 como valores propuestos. El criterio que se tomó para elegir los mejores descriptores fue el menor valor en porcentaje de variabilidad respecto a su media aritmética para cada uno de los descriptores.

OBJETO: FLECHA 70 mm, TAMAÑO DE IMAGEN: 659x424 PÍXELES, CAMPO VISUAL: 200 mm VERT. X: 150 mm HORIZ																							
No.	Imagen	Area(Píxeles)	Ang. Incl. °	C.G. Propuesto(mm)			C.G. Calculado(mm)			e1	EquipDiamet	Solidity	Descriptores invariantes de HU										
				X	Y	Z	X	Y	Z				e2	e3	e4	e5	e6	e7	Eccentricity	Extent			
1	f1_r106_c76_90.jpg	4276	89.94	106	76		217.03	107.706619	76.7794811	0.131800	73.780000	0.548100	2.504100	2.942900	5.666000	3.123700	11.804900	0.987700	0.382000				
2	f1_60_c110_88.jpg	4112	89.37	50	110		142.137	50.854025	111.792453	0.064000	72.357000	0.542800	0.282100	2.530000	2.961000	3.102400	11.301000	0.988800	0.232700				
3	f1_36_c29_54.jpg	3856	55.18	36	29		96.8107	34.637102	29.286692	0.067000	70.070000	0.540700	0.222700	2.190000	2.564300	2.676400	9.736500	0.988800	0.177300				
4	f1_r118_c37_15.jpg	4189	17.16	118	37		334.22	119.57818	35.7318366	0.128800	72.857000	0.564300	3.404000	3.265600	3.773700	7.293400	3.946800	11.156800	0.989600				
5	f1_r110_c78_35.jpg	4042	40.34	110	78		312.287	111.73069	78.0791038	0.086800	71.738700	0.538000	0.284300	2.430000	2.871500	3.000000	10.790000	0.988300	0.172400				
6	f1_r154_c103_12.jpg	4053	10.19	154	103		436.0962	292.68	156.028905	105.545891	0.066000	71.882000	0.540300	2.638000	3.063000	5.913700	3.190500	10.832500	0.989400				
7	f1_r141_c115_50.jpg	4163	45.6	141	115		399.831	323.535	143.162296	114.486137	0.106700	72.809000	0.561400	0.299500	2.886000	3.360300	6.484000	3.519500	10.427300				
8	f1_r125_c40_65.jpg	3971	66.07	125	40		346.949	116.6915	124.847565	41.2823703	0.062200	71.106000	0.539300	0.215000	2.674800	5.145800	2.782800	10.142700	0.988600				
9	f1_r1_c41_20.jpg	3922	21.97	31	41		87.6851	112.4041	31.3721288	39.7656014	0.076600	70.666000	0.535200	0.239000	2.565200	5.817100	3.136600	10.461000	0.989200				
10	f1_r69_c87_55.jpg	3957	57.34	89	87		255.0662	245.289	91.2580322	86.7767889	0.067200	70.980000	0.530900	0.226400	2.281800	2.899500	5.190200	2.813100	10.206000				
	Media	4052.1					Abs(C.G. Prop. - C.G. Calc.)			0.09073	71.81999	0.54356	2.55776	2.99272	5.70817	3.12858	10.66577	0.9888	0.22993				
	al Abs Dif. Con la media	223.9					1.70661896	0.77948113	0.04107	1.96001	0.00454	0.09089	0.05366	0.04982	0.10217	0.00488	1.11913	0.0011	0.15207				
		59.9					0.85402504	1.79245283	0.00667	0.53701	0.00076	0.01159	0.02776	0.03172	0.00087	0.02618	0.61523	0	0.00277				
		196.1					1.362898003	0.29659198	0.02373	1.74999	0.00286	0.04781	0.36776	0.42942	0.82657	0.45218	0.94927	0	0.06263				
		116.9					1.5781753	1.28816038	0.03807	1.03701	0.02074	0.06989	0.7074	0.78088	1.52523	0.81832	0.47003	0.006	0.02657				
		10.1					1.73689034	0.07910377	0.00513	0.08129	0.00656	0.00621	0.12776	0.12122	0.24557	0.12868	0.0423	0.006	0.06753				
		0.9					2.02890519	0.54589857	0.00573	0.01621	0.00326	0.01621	0.08024	0.07028	0.14553	0.06192	1.4673	0.005	0.07287				
		110.9					2.05223614	0.54188321	0.01597	0.98901	0.01784	0.02899	0.33224	0.36758	0.71583	0.38492	0.25847	0.0004	0.06663				
		81.1					0.15241603	1.28237028	0.02863	0.71399	0.00886	0.05651	0.29076	0.31792	0.62237	0.34578	0.54307	0.0002	0.02523				
		130.1					0.3721288	1.23438869	0.01413	1.15398	0.00836	0.03151	0.02744	0.02348	0.04883	0.00792	0.22477	0.0004	0.01133				
		95.1					2.2980322	0.2323113	0.02363	0.88999	0.01286	0.04411	0.27386	0.29322	0.57797	0.31548	0.47977	0.0004	0.06988				
							Error Promedio en mm																
	romojo de Dif.	102.5					1.40938673	0.80436439	0.020156	0.90785	0.00824	0.040272	0.228732	0.248464	0.487104	0.254616	0.49107	0.00044	0.050856				
		2.530%							22.215%	1.324%	1.587%	14.887%	8.943%	8.302%	8.445%	8.138%	4.586%	0.044%	22.118%				
							Factores de			$F_x = (200/659) =$			0.303641898484082										
							Escala			$F_y = (150/424) =$			0.35377358										
							Donde:																
							e1 a e7			Momentos invariantes de HU													
							EquipDiamet			Diámetro equivalente en unidades de píxel de un círculo cuya área es igual a la del objeto en estudio													
							Solidity			Escalar, que se calcula como la razón de el área del objeto entre el área de su concha convexa.													
							Ang. Incl.			Concha convexa es el polígono con el menor número de lados que es capaz de contener un objeto dado.													
							Eccentricity			Ángulo de inclinación del eje de mayor elongación de la pieza respecto al eje horizontal de la mesa de trabajo (Eje y s)													
							Extent			Escalar, cociente del área de la elipse que tiene los mismos valores de momento de segundo orden que la región, razón del foco de la elipse a su eje de mayor longitud. El valor es 0 para un círculo, 1 para un segmento de recta.													
										Escalar, cociente del área de la región dividida por área de la caja circunscrita (Bounding box).													
										Los mejores descriptores son los que presentan el menor porcentaje de variación respecto al valor medio													
										Malos descriptores													

Tabla 5.2 Selección de descriptores.

Como podemos observar en la Tabla 5.2, al revisar cada pieza analizada se puede apreciar que la gran mayoría de los casos los descriptores que mejor se comportan, es decir los que tienen porcentajes menores de variación son: área, diámetro equivalente, solidez y el primer momento invariante de HU. Cabe hacer mención en esta misma tabla, que los porcentajes de variación en el cálculo de las coordenadas del centro de gravedad de cada caso en cada pieza en unidades del mundo real (mm en nuestro caso), en base a los valores proporcionados por el sistema con la función: *regionprops*(), en valores de píxel y haciendo uso de factores de escala en cada uno de los ejes (*x,y*), son muy pequeños, del orden del 0.856% lo cual para una distancia de 100 mm esto representa 0.856 mm, es decir menos de un milímetro de error en el cálculo de las coordenadas del centro de gravedad de una pieza en unidades del mundo real.

En base a este análisis de descriptores, se determinó utilizar los siguientes descriptores: Diámetro equivalente (*ed*), Solidez (*sd*), Número de Euler (*eu*) y el primer momento invariante de HU (*phi*). En la Tabla 5.3 se presenta el cálculo de valores de los descriptores seleccionados para todas y cada una de las piezas que se utilizarán en los procesos de ensamble. En algunas piezas se incluyeron las imágenes correspondientes.

La función que en nuestro sistema equivale a la etapa de descripción de imagen es:

$$[areas, ed, sd, ex, ec, eu, phi, cm, ang] = Desc_imagen(L, n, ri, rf, ci, cf)$$

Obtiene además de los descriptores que utilizaremos, el centro de masa y orientación de cada pieza en la escena.

La siguiente instrucción obtiene los descriptores listados en base a la matriz de etiquetas (*L*) obtenida en la fase de segmentación, el resultado lo asigna a la variable de estructura (*D*).

$$D=regionprops(L, 'Area', 'Centroid', 'EquivDiameter', 'Solidity', 'Orientation', 'Extent', 'Eccentricity', 'Euler')$$

Las áreas de las (*n*) regiones en píxeles las obtenemos mediante:

$$areas=[D.Area]$$

De manera similar al cálculo de las áreas de cada objeto segmentado, se obtienen los valores de cada descriptor los cuales retornan de la función, a la función que realiza el llamado, obteniendo así la descripción completa de la escena.

OBJETO: RECTÁNGULO DE 50x10 mm, TAMAÑO DE IMAGEN: 559x424 PÍXELES, CAMPO VISUAL: 200 mm VERT. X 150 mm HORIZ.														
No.	Imagen	Area(Píxeles)	C.G. Propuesto(mm)			C.G. Píxeles			C.G. Calculado(mm)			EquiDiameter	Solidity	
			X	Y		X	Y		X	Y				ø1
1	rec_c38_c44_20.jpg	4557	38	44	105	124.7	37.5670841	44.115566	0.908600	76.171900	0.954300			
2	rec_c54_c120_100.jpg	4510	54	120	149.15	343.98	53.3631485	121.691038	0.904200	75.778000	0.956300			
3	rec_c139_c110_160.jpg	4641	139	110	392.89	310.41	140.588873	109.814858	0.937900	76.870700	0.953600			
4	rec_c136_c63_90.jpg	4476	136	63	382.33	178.809	136.790698	63.2579009	0.881200	75.491900	0.971800			
5	rec_c140_c34_50.jpg	4511	140	34	391.529	92.665	140.081932	32.7824292	0.892300	75.786400	0.957100			
6	rec_c175_c27_0.jpg	4621	75	27	210.146	75.513	75.1864043	26.7145047	0.866800	76.709900	0.932800			
7	rec_c24_c23_130.jpg	4688	24	23	66.678	68.2226	23.8661717	24.1363638	0.929200	77.094000	0.963500			
8	rec_c154_c132_130.jpg	4768	154	132	426.05	378.1549	152.432916	133.781215	0.950000	77.036200	0.946600			
Media		4594					Abs(C.G. Prop. - C.G. Calc.)		0.9073875	76.367375	0.9545			
Val Abs Dif. Con la media		37					0.43291592	0.11556604	0.0011125	0.195475	0.0002			
		84					0.63885152	1.69103774	0.0031875	0.589375	0.0018			
		47					1.56887299	0.18514151	0.0305125	0.503325	0.0009			
		118					0.79069767	0.25790094	0.0261875	0.875475	0.0173			
		83					0.08193202	1.21757075	0.0150875	0.580975	0.0026			
		27					0.18640429	0.28649528	0.0616875	0.342525	0.0217			
		74					0.14382826	1.13635377	0.0218125	0.728625	0.009			
		174					1.56708408	1.78121462	0.0426125	0.668825	0.0079			
Promedio de Difs.		80.5					Error Promedio en mm		0.0240125	0.560325	0.007675			
% Promedio de Difs.		1.752%							2.646%	0.734%	0.804%			
Factores de Escala $F_x = (200/559) = 0.35778175 \text{ mm/pixel}$ $F_y = (150/424) = 0.35377358 \text{ mm/pixel}$														
Donde:														
		ø1	Primer momento invariante de HU											
		EquiDiameter	Diámetro equivalente en unidades de pixel de un círculo cuya área es igual a la del objeto en estudio											
		Solidity	Escalar, que se calcula como la razón de el área del objeto entre el área de su concha convexa.											
			Concha convexa es el polígono con el menor número de lados que es capaz de contener un objeto dado.											
			Valores representativos (promedios) de los descriptores que forman el vector característico para efectos de reconocimiento											
			Porcentaje de variación respecto a la media.											

Tabla 5.3 Patrones con sus descriptores seleccionados.

5.2.5 Reconocimiento de patrones.- Inicialmente se aplicó el método de decisión teórica de reconocimiento usando el clasificador de distancia mínima [19-21], (véase Apartado 5.1.1.8).

Sin embargo este método al ser numérico cien por ciento, incurre en errores cuando la pieza por reconocer se aproxima numéricamente hablando más a un patrón diferente al que verdaderamente le correspondería. Para eliminar esto proponemos modificar el procedimiento combinando la eficiencia de los métodos de decisión teórica con un método geométrico que toma en cuenta las imágenes binarias de los patrones de las piezas en su posición horizontal y sentido inicial al que nos referiremos a continuación.

Pseudocódigo mejorado de la función de *analiza_escena()*:

- Tomar imagen.
- Preprocesarla.
- Segmentar imagen.
- Obtener descriptores para cada una de las piezas en la escena.
- Calcular factores de escala vertical y horizontal para convertir dimensiones píxel a milímetros.
- Convertir valores del centro de masa en píxeles a dimensiones del mundo real.
- Leer la base de datos de patrones.
- Para cada pieza en la escena hacer lo siguiente:
 - Tomar los valores del centro de masa y ángulo de inclinación de la pieza respecto a una paralela al eje X de la imagen tomada por la cámara.
 - Aislar la pieza en estudio.
 - Aplicar una rotación respecto al eje Z igual al a su ángulo de inclinación pero en sentido opuesto, de tal forma que adopte una posición horizontal.
 - § Trasladar la imagen resultante de la pieza a la esquina superior izquierda de la escena.
 - Buscamos en la base de patrones las piezas con un tamaño similar en $\pm 5\%$ y topológicamente iguales (igual número de Euler).
 - Mientras el patrón cumpla con la condición anterior, hacemos:
 - § Leer la imagen del patrón con inclinación de 0° .
 - § Comparar las imágenes de la pieza por reconocer con la del patrón actual.
 - § Obtener la diferencia absoluta de imágenes correspondientes.

- § Aplicamos adelgazamiento a la imagen resultante para contrarrestar posibles diferencias en el tamaño de las piezas debido a fluctuaciones al momento de tomar la imagen.
- § Se obtiene el umbral a utilizar en la diferencia de imágenes (10% del tamaño del patrón).
- § Giramos 180° la imagen del patrón respecto al eje Z y volvemos a comparar con la imagen de la pieza mediante la diferencia absoluta de ambas.
- § Aplicamos adelgazamiento a la imagen resultante para contrarrestar posibles diferencias en el tamaño de las piezas debido a fluctuaciones al momento de tomar la imagen.
- § Si la 1ª. Diferencia es menor al umbral y menor a la 2ª. Diferencia
 - El patrón compara satisfactoriamente y el sentido de la pieza corresponde también al original del patrón.
- § Si la 2ª. Diferencia es menor al umbral y menor a la 1ª. Diferencia
 - El patrón compara satisfactoriamente y el sentido de la pieza es opuesta a la que presenta el patrón en su definición (girada 180°).
- § Si tanto la 1ª. Diferencia como la 2ª. Diferencia son mayores al umbral
 - El patrón no compara satisfactoriamente.
- Si agotamos la base de datos de patrón y no compara, asignamos el nombre “Desconocido” a la pieza.
- Transformar coordenadas mediante la matriz que se describe en la sección siguiente.
- Continuar análisis con la siguiente pieza segmentada hasta terminar.
- Regresa a la función que llama, para cada pieza:
 1. Nombre del patrón que compara satisfactoriamente.
 2. Coordenadas finales del centro de masa.
 3. Ángulo final de inclinación del eje de mayor elongación de la pieza respecto la horizontal.
 4. Sentido.
 5. Además del número total de piezas en la escena.
- Fin del algoritmo.

La función se llama desde cualquier punto de nuestro sistema mediante la instrucción:

[piezas, npz]=Analiza_escena3(imagen, ext, ri, rf, ci, cf, CVv, CVh)

Uno de los valores que esta función toma para trabajar son los que corresponden a las dimensiones vertical (*CVv*) y horizontal (*CVh*) del campo visual en valores del mundo real (milímetros en nuestro caso) comprendido en el rectángulo del puerto de visión definido por los puntos (*ri,rf*) y (*ci,cf*) comentados al inicio de esta sección [22].

La referencia de los ejes coordenados en una imagen procesada por la librería IPT tomando el origen en la esquina superior izquierda de la imagen, el eje X hacia la derecha y el eje Y hacia abajo y el eje Z hacia adentro del papel, véase Fig. 5.5(a). Para transformar coordenadas a otro sistema que ubicaremos sobre la mesa de trabajo justo bajo la cámara donde su eje X apunte hacia abajo y el eje Y hacia la derecha y con el eje Z saliendo del papel, véase Fig. 5.5(b). Debemos obtener la matriz de transformación y para ello rotamos

el sistema de la imagen 90° respecto a su eje Z, seguido de un giro de 180° con respecto a su eje X.

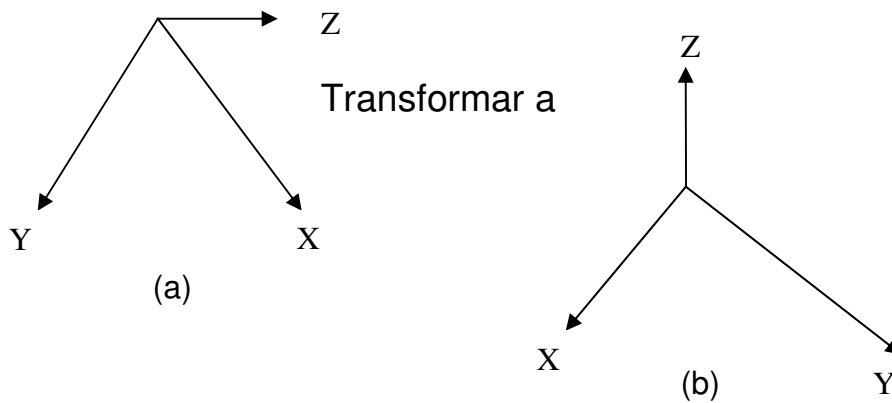


Figura 5.5 (a) Transformación del sistema referido en la imagen a uno (b) ubicado en la mesa de trabajo y bajo la cámara.

De esta manera decimos que la **interpretación** lo constituye, toda aquella información obtenida y que puede ser considerada como información relevante o bien, que pueda ser considerada como dato de entrada en otros procesos. En nuestro caso podemos decir que la información de salida de la función que analiza la escena (patrón, centro de masa, inclinación, sentido) es relevante en el desarrollo del proceso de ejecución de ensamblado, así como en el proceso que crea las plantillas de ensamble.

En el siguiente capítulo daremos una descripción detallada del SPPM, módulos que lo integran, manera de operar el programa creado para esta subfunción y corrida de un prototipo, con objeto de mostrar en que forma podemos "Ejecutar tareas robotizadas de ensamble guiadas por el sistema de visión artificial".

Capítulo 6

Subsistema de Programación de Procesos de Manufactura (SPPM)

El presente capítulo se refiere al principal subsistema del sistema desarrollado. Este actúa como el controlador y administrador del resto de los subsistemas; organiza, planifica y ejecuta todas las tareas que constituyen el proceso de maquinado y ensamble de los productos finales. El presente trabajo no considera el subsistema llamado Estación de Maquinado (SEM), concentrándose exclusivamente en las tareas de reconocimiento, localización y ensamblado de partes que darán como resultado un producto final. Se describen además las funciones que comprenden el subsistema, tales como calibración, etapa de aprendizaje, generación de plantilla de ensamble, programación y ejecución del proceso de ensamblado. Todas estas funciones implementadas dentro de un programa interactivo en lenguaje "C" en ambiente MatLab, Versión 7 (véase Apéndice B). Estas funciones sirven para operar el sistema fuera de línea, es decir, mientras se calibra, entrena y se programan plantillas de ensamblado. Se entrena el sistema para incorporar patrones nuevos a la base de conocimiento. Se genera una nueva plantilla o se programa para su ejecución. El módulo que corre durante el proceso en línea es llamado a operar por el PLC, mismo que controla la cámara, el sistema de transporte (banda transportadora) y el sistema de manipulación de partes (manipulador robótico). El PLC se encuentra funcionando de manera continua.

El SPPM programado se puede apreciar en la Fig. 6.1, donde se encuentra una pieza que será dada de alta en la base de datos de patrones, podemos observar varias escenas que corresponden a la misma pieza colocada en diferentes posiciones.

6.1 Módulos que integran el SPPM.- Está constituido por los siguientes módulos: calibración, aprendizaje, crea plantilla, especificación del proceso actual y ejecución del proceso. A continuación se describen cada uno de ellos.

6.1.1 Módulo de Aprendizaje.- Es la fase en la que el sistema da de alta la información correspondiente a las clases que integran la base de conocimiento de los patrones. Para dar de alta una clase o patrón, se requiere una serie de imágenes en que aparezca la pieza correspondiente en diferentes posiciones sobre el campo visual que comprende la mesa de trabajo, dichas imágenes tienen un nombre base y añadido a este nombre base, un número que forma parte de una secuencia con un valor inicial, un número final y un valor de incremento, los cuales constituyen los datos de entrada de la función. Para cada imagen se aplican las siguientes fases: adquisición de imagen, preproceso de imagen, segmentación de la escena y descripción. En esta última fase se obtienen los descriptores seleccionados para el presente trabajo (véase Apartado 5.2), de los cuales se obtiene el valor promedio de cada uno de ellos almacenándolos en la base de conocimiento de clases, para su posterior uso en tareas de reconocimiento de piezas.

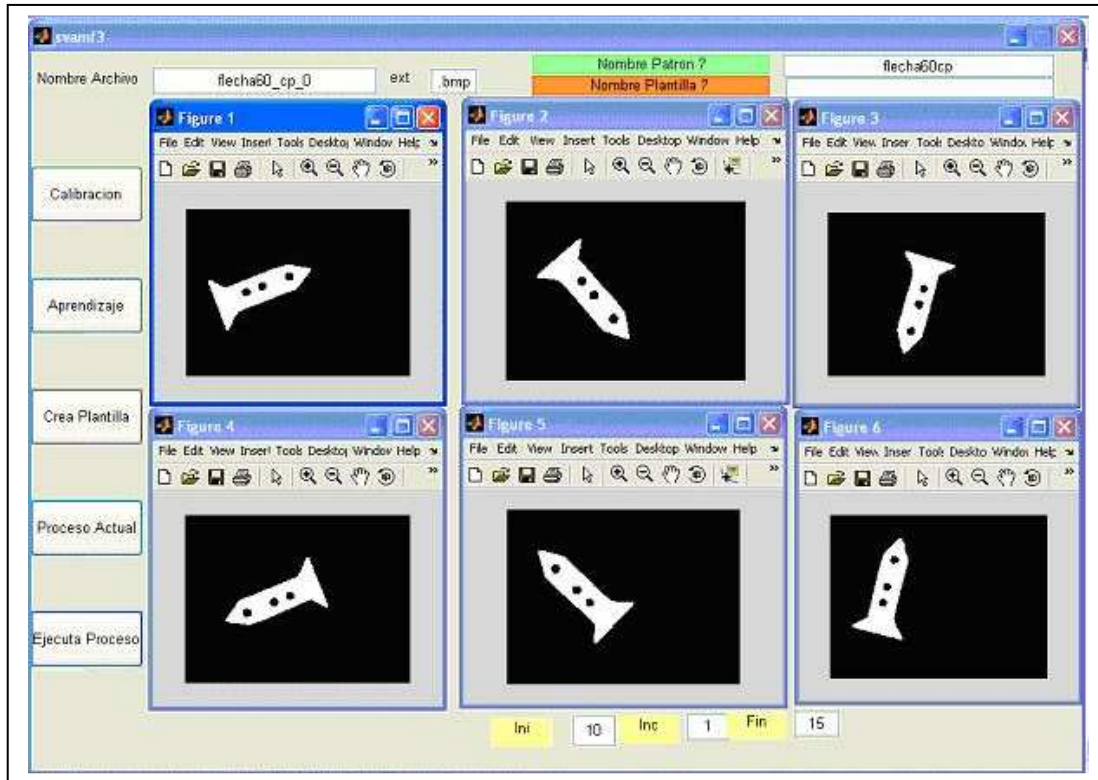


Figura 6.1 Implementación del SPPM fuera de línea.

La función para implementar la fase de aprendizaje en el SPPM es la siguiente:

Aprendizaje_gui(nombre_patron, imagen, ini, inc, fin, ext, ri, rf, ci, cf)

Esta función toma como datos de entrada el nombre que se le asignará al patrón (*nombre_patron*); el nombre base de los archivos de imagen incluida la ruta (*imagen*), donde aparece la pieza que será dada de alta en la base de datos de patrones; los valores inicio (*ini*), incremento (*inc*) y fin (*fin*) de la parte numérica del nombre de los archivos de imagen; la extensión correspondiente al formato de imagen que se utiliza en los archivos de imagen por leer (*ext*) y por último los valores renglón inicial (*ri*), renglón final (*rf*), columna inicial (*ci*) y columna final (*cf*) de la subimagen a considerar sobre las imágenes leídas, véase Apartado 5.2.1. La función almacena en disco en el archivo de texto (llamado *patrones.txt*), el nombre del nuevo patrón así como los valores correspondientes a sus descriptores. En la Fig. 6.2 se presenta un ejemplo de este archivo, debido a que la información se obtiene de varias imágenes, se almacena sólo el valor promedio para cada uno de ellos.

Pseudocódigo de la función:

Para cada imagen de entrada aplicar:

- Adquisición de imagen.
- Preproceso de imagen.
- Segmentación de imagen.

- Descripción de imagen.
 - § Obtener los descriptores (Apartado 5.2).
 - § Almacenar en memoria los valores obtenidos.
- Obtener la media aritmética de cada descriptor para cada una de las imágenes analizadas.
- Preguntar al usuario la altura en milímetros que tiene la pieza correspondiente al nuevo patrón.
- Almacenar en disco en el archivo de texto información del nombre, descriptores y altura de la pieza.
- De la última imagen:
 - Obtener la transformación de imagen que hace girar la pieza un ángulo contrario a su valor de inclinación y que por consiguiente ubica la pieza paralela al eje coordinado horizontal en la imagen.
 - Aplicar la transformación descrita en el punto anterior (giro respecto al eje Z)
 - Recortar la pieza
 - Almacenar en disco la imagen recortada asignándole el nombre del patrón.
 - Fin

Patrón	EquivDiameter	Solidity	Extent	Eccentricity	EulerNumber	InvMoments(1)	Altura
cuadro10sp	61.91	0.9704	0.652062	0.365839	1.000000	1.796848	11.000000
rec40x13sp	128.72	0.9806	0.537785	0.940025	1.000000	1.301850	13.000000
circu20_cp	100.85	0.9372	0.744347	0.243825	0.000000	1.731177	12.000000
circu20_sp	103.70	0.9906	0.782391	0.217296	1.000000	1.837384	12.000000
hrombo45sp	95.26	0.9716	0.550652	0.952987	1.000000	1.215903	12.000000
trape80_sp	172.01	0.9839	0.430957	0.974180	1.000000	0.986184	12.000000
flecha60sp	162.74	0.7205	0.366182	0.954014	1.000000	1.062471	12.000000
flecha60cp	161.67	0.7089	0.358710	0.948592	-2.000000	1.011066	13.000000
trape80_cp	172.86	0.9675	0.468965	0.970344	0.000000	1.018011	13.000000
rombo45_cp	93.21	0.9156	0.365937	0.952473	1.000000	1.133758	12.000000

Figura 6.2 Archivo de texto de patrones conteniendo nombre del patrón, seis descriptores y altura de la pieza en milímetros.

6.1.2 Módulo de creación de plantilla para ensamble.- Función cuyo objetivo es el de generar un archivo de texto con los nombres de las piezas que participan en el proceso de ensamble, la posición y orientación de cada una de ellas respecto al sistema coordinado 2D sobre la mesa de trabajo (sistema 2 según la Fig. 4.6). El archivo posee además información de la prioridad que tiene cada pieza en el proceso mismo de ensamblado. Los valores de ubicación de las piezas respecto al sistema coordinado citado son relativos al mismo.

Al ejecutarse una tarea de ensamblado real, debe especificarse previo a la ejecución de la misma, la posición espacial (x,y,z) sobre la mesa de trabajo del origen del sistema coordinado de la plantilla de ensamble, así como su orientación (giro) respecto al eje Z, que es el eje perpendicular a la mesa de trabajo. De esta forma una misma plantilla puede ser ensamblada varias veces en diferentes ubicaciones variando únicamente su posición.

Esta función llama a la función *analiza_escena()*, descrita en la sección 5.2.5, proporciona como salida un arreglo en memoria que almacena el nombre de la pieza, su centro de masa, el ángulo de inclinación (véase Apartado 5.1.1.5) del eje de simetría de la pieza con mayor elongación respecto a la horizontal (medido en grados), el sentido que tiene la pieza en la escena respecto al patrón y la prioridad que deberá tener la pieza al momento de ser ensamblada. Estos valores son calculados para todas y cada una de las piezas que se localizan en la escena sobre la mesa de trabajo. En la Fig. 6.3 se muestra una escena captada del sistema al crear una plantilla de ensamble. El contenido de la estructura en memoria que contiene la información de las piezas en la escena de trabajo se muestra en la Fig. 6.4.

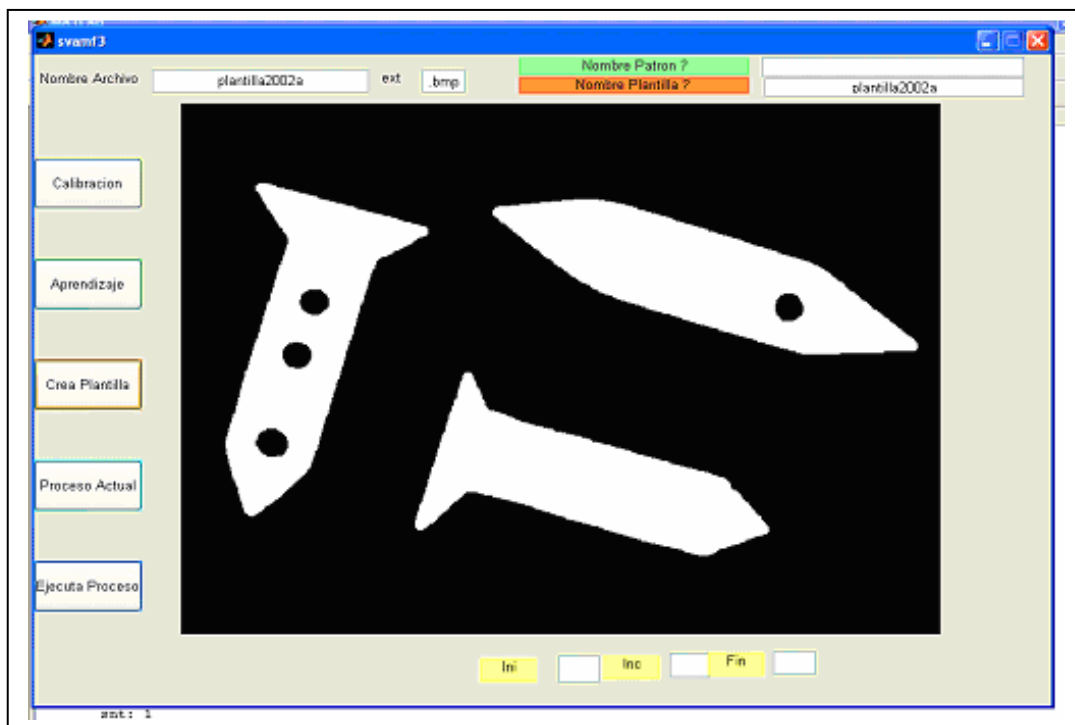
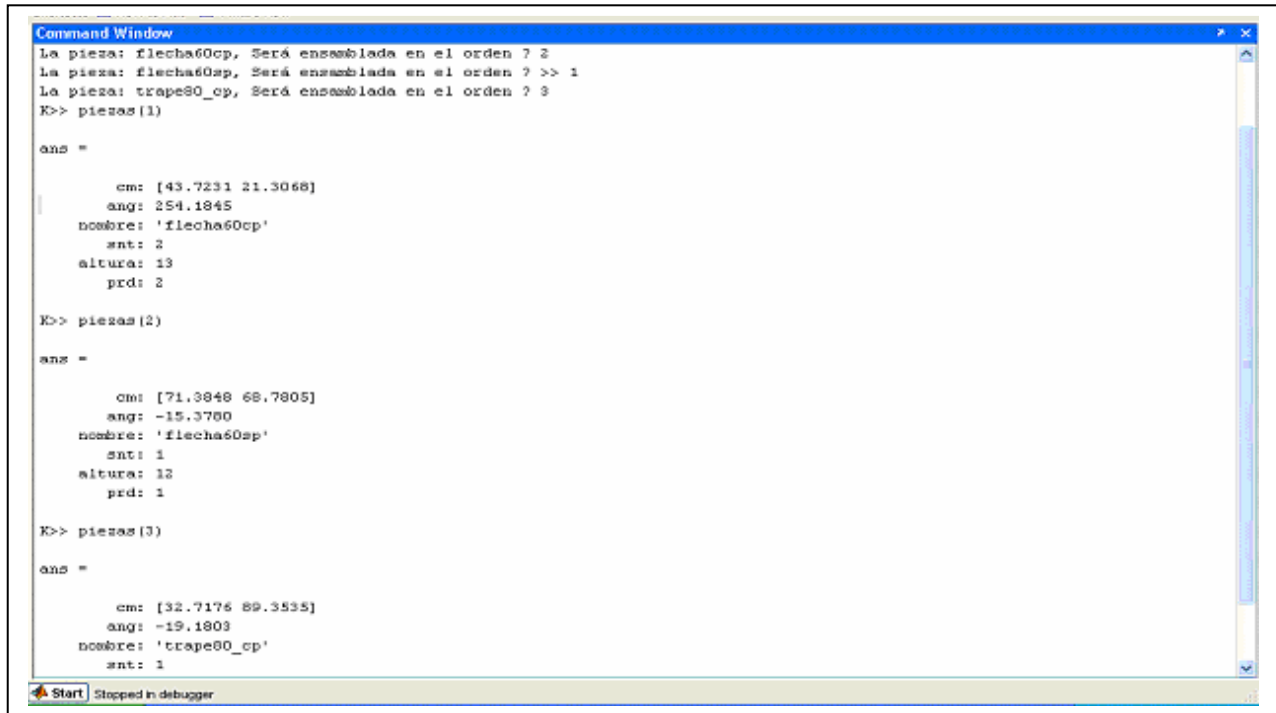


Figura 6.3 Ejecución de la función *crea_plantilla()* capturando una escena con tres patrones.



```
Command Window
La pieza: flecha60cp, Será ensamblada en el orden ? 2
La pieza: flecha60sp, Será ensamblada en el orden ? >> 1
La pieza: trape80_cp, Será ensamblada en el orden ? 3
K>> piezas(1)

ans =

    cm: [43.7231 21.3068]
    ang: 254.1845
    nombre: 'flecha60cp'
    xnt: 2
    altura: 13
    prd: 2

K>> piezas(2)

ans =

    cm: [71.3848 68.7805]
    ang: -15.3780
    nombre: 'flecha60sp'
    xnt: 1
    altura: 13
    prd: 1

K>> piezas(3)

ans =

    cm: [32.7176 89.3535]
    ang: -19.1803
    nombre: 'trape80_cp'
    xnt: 1

Start Stopped in debugger
```

Figura 6.4 Proceso de asignación de orden de ensamblado de las piezas e información que regresa la función *analiza_escena()*.

Pseudocódigo de la función es el siguiente:

- Analiza escena
- Para cada pieza identificada en la escena solicita al usuario el orden en el cual será ensamblada.
- Ordena la información en sentido ascendente según por la prioridad de ensamblado de las piezas.
- Almacenar en disco la información de plantilla.

6.1.3 Módulo proceso actual de ensamble.- Esta función almacena información para el sistema especificando el nombre de la plantilla actual que ha de ser ensamblada, la posición espacial del origen del sistema local de plantilla medido respecto al sistema coordinado de la mesa de trabajo, así como el giro respecto al eje Z de la plantilla en su posición final de ensamble respecto al sistema de la mesa de trabajo. Este eje Z es perpendicular al plano de la plantilla. Esta información se guarda en un archivo de texto llamado “*proceso.txt*”

El pseudocódigo para esta función es el siguiente:

- Solicita el nombre de la plantilla.
- Solicita al usuario las coordenadas (x, y, z) que el origen del sistema plantilla tendrá en su posición final de ensamble, referido respecto al sistema de la mesa de trabajo. Tendrá el mismo efecto que si trasladamos el origen de la mesa de trabajo al origen del sistema coordinado de la plantilla en su posición final de ensamblado, (apartado 4.2), véase solución 1 en Apéndice A.
- Solicita al usuario el giro (en grados) que el sistema mesa de trabajo debe aplicar respecto a su eje z , para alinearse al sistema de plantilla en su posición final.
- Almacenamos en disco la información de la plantilla con su ubicación final

Las coordenadas (x, y, z), de la plantilla (referidas respecto al sistema de la mesa de trabajo) y el giro respecto al eje Z (necesario para alinear la mesa de trabajo al sistema de plantilla) son los elementos requeridos para obtener una matriz de transformación. Esta matriz será utilizada para convertir puntos coordinados referidos a la plantilla respecto de la mesa de trabajo. Para lo cual se hace uso de otra transformación similar, la cual referirá estos mismos puntos respecto ahora del sistema base del robot.

Cada vez que se desee modificar los valores de posición y orientación de la plantilla actual de ensamble podremos hacerlo mediante este módulo, o también si se desea cambiar la plantilla actual por alguna otra previamente creada.

6.1.4 Módulo de calibración.- Proceso mediante el cual se establecen y almacenan una serie de parámetros necesarios para el correcto funcionamiento del sistema SPPM y que se clasifican como: parámetros del campo visual, parámetros de ubicación de la mesa de trabajo y finalmente parámetros de ubicación del sistema de adquisición de imágenes (cámara). La aplicación de éste módulo presenta los tres grupos de parámetros, para que el usuario elija cual grupo de valores desea modificar. Al inicio de la ejecución del programa los valores de calibración son leídos de un archivo (llamado "*calibración.txt*"). El sistema permite además modificar los parámetros de calibración del mismo.

Los valores de campo visual (véase Apartado 4.3), se refieren al dispositivo de despliegue, archivo de imagen digital correspondiente a la escena capturada por la cámara sobre la mesa de trabajo. Estos valores corresponden a valores de renglón, columna que delimitan el área rectangular de interés en la imagen los cuales se listan a continuación con los datos numéricos que se utilizaron en el sistema.

ri: renglón inicial = 1
rf: renglón final = 479
ci: columna inicial = 1
cf: columna final = 639

Para imágenes de 480 píxeles verticales por 640 píxeles horizontales utilizadas en el sistema, suponemos aquí que la totalidad de la imagen tomada por la cámara cae sobre una sección de la mesa de trabajo.

Como valores del campo visual consideramos también:

CVV: campo visual vertical

CVh: campo visual horizontal.

Estos dos últimos valores son las dimensiones vertical y horizontal reales en milímetros del campo visual que corresponden a la sección rectangular sobre la mesa de trabajo y bajo el campo visual de la cámara.

$$XP_{max} = rf - ri$$

$$YP_{max} = cf - ci$$

$$XW_{max} = CVv$$

$$YW_{max} = CVh$$

Del Apartado 4.3,

$$X_1 = X_p (XW_{max}) / (XP_{max})$$

$$Y_1 = Y_p (YW_{max}) / (YP_{max})$$

Sustituyendo valores:

$$X_1 = X_p (CVv) / (rf - ri)$$

$$Y_1 = Y_p (CVh) / (cf - ci)$$

Con esto calculamos las coordenadas reales (X_1 , Y_1) en unidades de milímetro de un punto, particularmente el centro de masa de la pieza, dados los valores en coordenadas de dispositivo (píxeles) en la imagen del mismo punto.

Los parámetros del segundo grupo a considerar en el proceso de calibración determinan la ubicación de la mesa de trabajo respecto al sistema coordinado base o sistema del robot. Son las coordenadas (dx, dy, dz) del origen del sistema de la mesa de trabajo referidas al sistema base y los ángulos A , B y C (ángulos en grados) que el sistema base del robot debe girar respecto a su eje X , Y , Z respectivamente. Se utiliza para alinearse con el sistema coordinado de la mesa de trabajo. En base a la metodología mostrada en el Apartado 4.2 obtenemos la matriz de transformación para referir puntos dados en coordenadas de la mesa de trabajo a puntos respecto al sistema base del robot.

Para el tercer y último grupo de parámetros en el proceso de calibración, el objetivo es obtener la matriz de transformación que refiere puntos dados en coordenadas del sistema captado por la cámara para ser convertidos a coordenadas de la mesa de trabajo, de manera similar al grupo anterior, los parámetros son : coordenadas (dxi, dyi, dzi) del origen del sistema de cámara respecto al sistema de mesa de trabajo y giro en grados del sistema coordinado de la mesa para lograr alinearse con el sistema de cámara.

Estos valores pueden ser actualizados en cualquier momento y vueltos a grabar por el sistema una vez que han sido modificados.

6.1.5 Módulo ejecuta ensamble.- Es la función que ejecuta el proceso de ensamblar una plantilla seleccionada en una ubicación definida por el usuario, utilizando para ello las piezas que se localizan dispuestas de modo aleatorio sobre la mesa de trabajo. La cantidad de piezas que aparecen en la escena pueden ser mas, pueden ser menos o exactamente las que el proceso requiere para lograr el ensamblado total del producto, solamente las piezas que requiera el proceso deberán ser manipuladas por el robot en el estricto orden en base a la prioridad asignada en la programación de plantilla (véase Fig. 6.5), donde aparecen un mayor número de piezas que las que son requeridas para su ensamble. Las piezas entonces podrán ser “*tomadas*” con base en su centro de masa y con una inclinación en correspondencia a la que presenta la parte mas alargada de la pieza, para ser “*liberadas*” en su posición y orientación correctas sobre la plantilla de ensamble. Una vez ensambladas las piezas localizadas en la escena, el PLC que controla las estaciones de transporte y manipulación libera la charola sobre la cual se encuentra la mesa de trabajo. Ésta continúa su trayecto circular para volver nuevamente a la estación de toma de imagen, antes de lo cual, se alimenta la charola con nuevas piezas que serán ensambladas sobre la misma plantilla si aún faltan piezas por ensamblar o a una nueva plantilla con las mismas características a la anterior. En el momento que una plantilla se ensambla completamente, ésta debe ser retirada (incluso puede utilizarse el mismo robot para esta tarea) para continuar con el ensamblado de otra plantilla que podría ser igual a la anterior o reprogramada por el usuario. Puede utilizarse la misma plantilla pero en distinta ubicación o bien utilizar una plantilla diferente. Esto último justifica la propuesta del presente trabajo de tesis al considerar procesos de manufactura flexible en la celda del ITSSLP,C y no procesos fijos como inicialmente se tenían, de acuerdo a lo comentado en la Sección 1.4.

Todas las funciones descritas hasta aquí, se ejecutan fuera de línea, esto es cuando no se encuentra operando el proceso continuo de toma de imagen, transporte de material, manipulación de piezas (todo ello controlado por el PLC). Cuando el proceso esta operando (en línea), es el PLC quien debe llamar a la función *ejecuta_ensamble_en_linea*() justo cuando empieza a detenerse la charola que transporta la mesa de trabajo al aproximarse a la estación de manipulación y ensamblado, de tal forma que cuando la charola logra detenerse, el sistema ya procesó la información de reconocimiento y localización de piezas y está a punto de establecer comunicación con el controlador del robot para transmitir los datos (*x, y, z, alabeo, guiñada*) correspondientes a la posición de cada pieza por ensamblar que se localiza en la escena, tanto para la operación de “tomar” como para la de “dejar”, como puede verse en la Fig. 6.6, iniciando así la etapa de manipulación de piezas y lograr finalmente ensamblar la plantilla programada. Además de transmitir estos valores al control del robot, el sistema almacena esta información en un archivo de texto llamado “*puntos.txt*”.

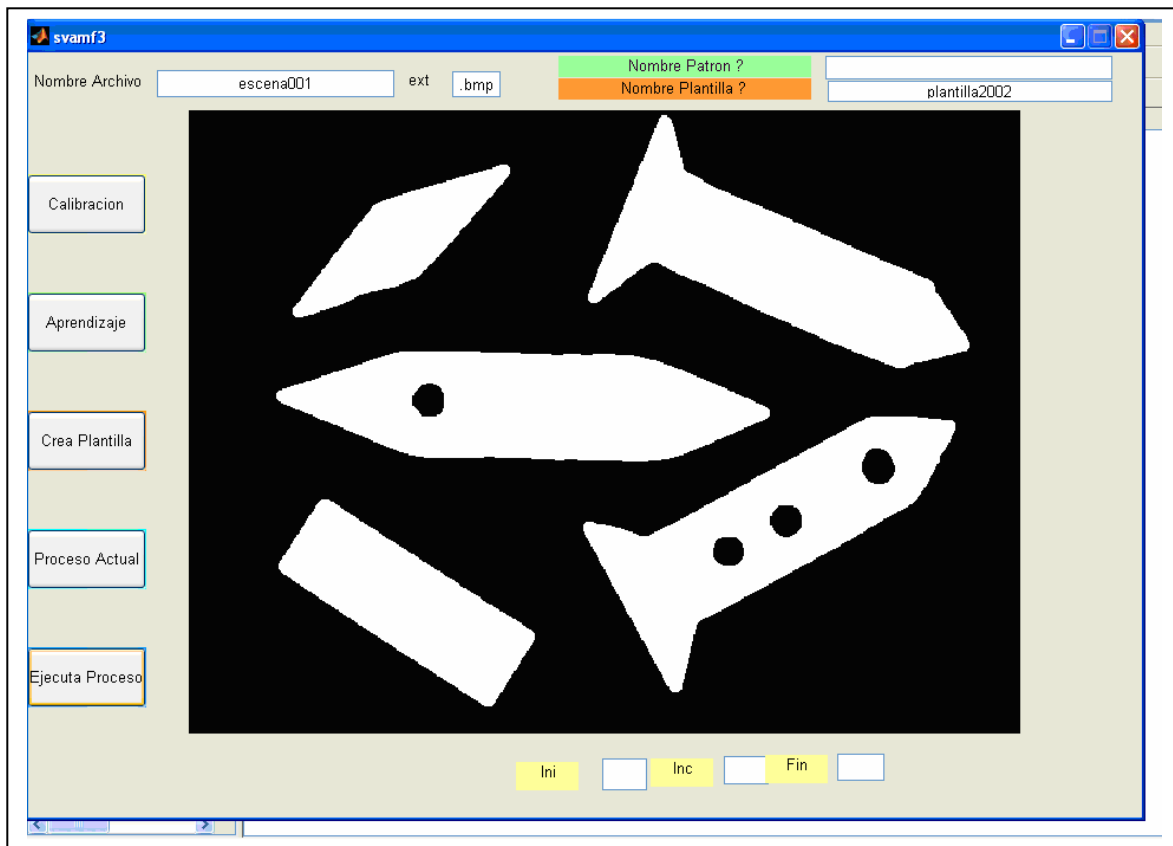


Figura 6.5 Piezas en la escena de trabajo para ensamblar plantilla de Figura 6.3.

```

Command Window
K>> piezas(4)

ans =

      cm: [67.6652 89.8430]
      ang: 29.4374
      nombre: 'flecha60cp'
      snt: 1
      altura: 13
      disp: 'u'

K>> piezas(5)

ans =

      cm: [24.8960 91.4170]
      ang: -22.9479
      nombre: 'flecha60sp'
      snt: 1
      altura: 12
      disp: 'u'

K>> type puntos.txt

199.88   -236.39   340.50    2.63    3.14
103.44   -282.75   340.50    0.28    3.14
198.30   -279.16   339.50   -2.74    3.14
131.10   -235.28   339.50   -1.30    3.14
236.52   -256.51   340.50   -3.11    3.14
 92.44   -214.71   340.50   -1.24    3.14

```

Figura 6.6 Datos de piezas localizadas en la escena de la Figura 6.5 y valores transmitidos al robot.

El pseudocódigo para la función *ejecuta_ensamble_en_linea()* es el siguiente:

- Lee calibración.
- Lee del proceso actual el nombre y la ubicación de plantilla para su ensamble.
- Lee del archivo de plantilla de ensamble para cada pieza:
 - nombre
 - centro de masa
 - ángulo de inclinación
 - prioridad de ensamble
- Calcula las matrices de transformación (Apartado 4.4.1)
 - Matriz para referir puntos respecto al sistema base, dadas sus coordenadas respecto de la mesa de trabajo.
 - Matriz para referir puntos respecto al sistema mesa de trabajo, dadas sus coordenadas respecto al sistema de imagen.
 - Matriz para referir puntos respecto al sistema mesa de trabajo, dadas sus coordenadas respecto al sistema de plantilla en su ubicación final de ensamble.
- Analiza la escena. Se obtiene como salida las piezas reconocidas en la escena, posición y orientación de ellas referidas respecto al sistema de imagen.
- Para cada pieza que forma parte de la plantilla de ensamble.
 - Se busca entre las piezas localizadas en la escena siempre y cuando sea la misma pieza y esté aún disponible (no utilizada antes)
 - § Obtener el centro de masa y altura de la pieza.
 - § Transformar centro de masa 3D respecto al sistema mesa de trabajo.
 - § Transformar centro de masa 3D respecto al sistema base.
 - § Cálculo de la posición y orientación de la pieza (Apartado 4.5).
 - § Almacenar en un arreglo de memoria las coordenadas 3D del centro de masa donde se localiza la pieza, el giro del efector final (alabeo) y la guiñada.
 - § Para la misma pieza se determinan los valores de ubicación para su posición final sobre la plantilla de ensamble referidas también respecto al sistema base.
 - § Almacenar en un arreglo de memoria las coordenadas 3D del centro de masa donde se localizará la pieza, el giro del efector final (alabeo) y la guiñada.
 - § Marcar la pieza en la escena como ya utilizada.
 - Continúa hasta agotar la totalidad de piezas en la escena.
- Continuar hasta agotar la totalidad de piezas en la plantilla.
- Se establece comunicación serial entre el sistema y el control del robot.
- Para cada pieza encontrada que participa en el proceso de ensamble hacer
 - Mientras que el robot esté listo, enviar valores x , y , z , alabeo, guiñada de la posición “tomar”
 - En este punto el robot se mueve para “tomar” la pieza.
 - Mientras que el robot esté listo, enviar valores x , y , z , alabeo, guiñada de la posición “dejar”.
 - En este punto el robot se mueve para “dejar” la pieza.
- Continuar hasta terminar con las piezas que forman parte del proceso de ensamble.
- El sistema envía al control del robot la indicación que el proceso ha terminado.

Con esto damos por terminado este capítulo. En el siguiente capítulo daremos una breve descripción del Subsistema de Manipulación de Piezas (SMP) integrado por la banda transportadora, los diversos sensores y actuadores para controlar el movimiento de la banda, el robot manipulador con su sistema de control propio, el PLC que controla las estaciones de toma de imagen, transporte y manipulación de partes y que además llama a operar la aplicación *“Ejecuta_esamble_en_linea()”*.

Capítulo 7

Subsistema de Transporte y Manipulación de Piezas (SMP)

El presente capítulo describe del SMP que constituye un elemento importante en nuestro sistema (Apartado 3.2.1). Está integrado por una banda transportadora y por un manipulador robótico. La banda transportadora conduce charolas metálicas rectangulares sobre las cuales se colocan las piezas por ensamblar y el manipulador robótico mueve las piezas de su posición inicial sobre la mesa de trabajo a su posición final de ensamble. Por tratarse de componentes que existen en la CMA del ITSSLP,C y que se encuentran actualmente operando en varias aplicaciones de la celda, las trataremos de manera general y con enfoque directo a nuestro proyecto.

7.1 Estación de transporte.- Requiere para su control un PLC sistema Siemens S7, la transmisión de datos interna de la estación la realiza el sistema de red ASi (Actuators and Sensors Interface) así como la comunicación con otras estaciones. En el sistema aquí presentado la comunicación con la estación de adquisición de imagen y con la estación de manipulación y ensamble se realiza por medio de un módulo de entradas y salidas ó módulo I/O. En la Fig. 7.1 se presentan la estación de transporte, la estación de adquisición de imagen (cámara) y la estación de manipulación y ensamble (robot).

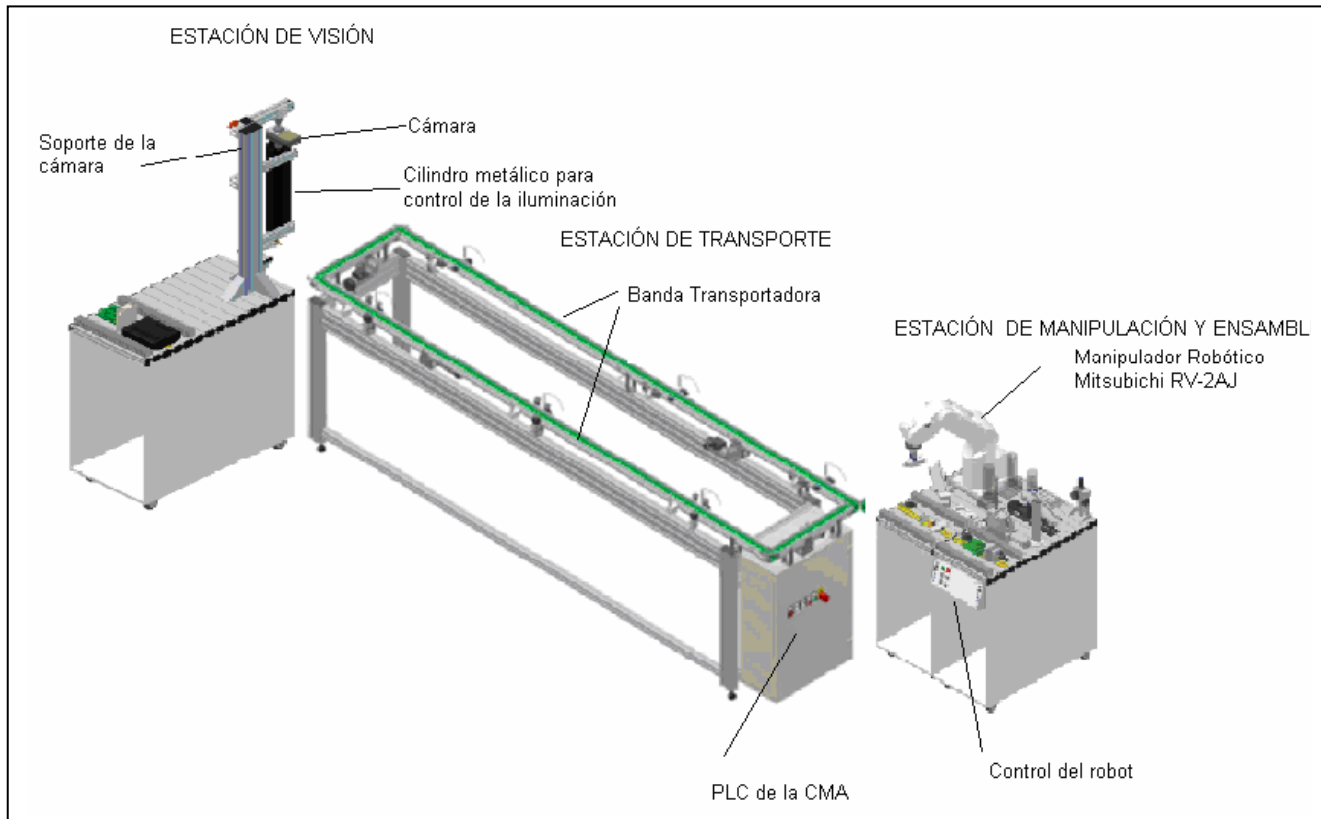


Figura 7.1 Estaciones de trabajo del CMA que forman parte de nuestro sistema.

Las líneas de comunicación interna en el sistema de transporte están físicamente alambradas en el panel de control, véase Fig. 7.2. Éstas son utilizadas para controlar los sensores que detectan la presencia de las charolas y los pistones que detienen el avance de las mismas. La cámara y el robot están directamente conectados a los dispositivos de control respectivos por medio del módulo de entradas y salidas. Cabe mencionar que la manera de utilizar y programar esta estación queda fuera del alcance de la tesis.

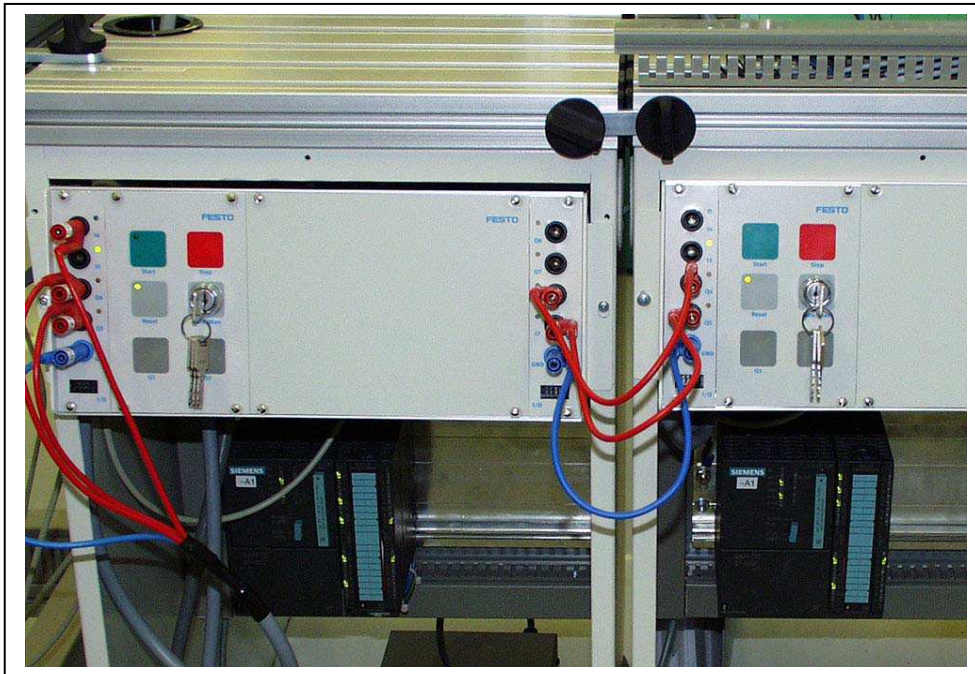


Figura 7.2 Panel de control alambrado para el sistema de transporte.

7.2 Estación de manipulación.- La definición de robot, según el Instituto Norteamericano de Robótica es *"un manipulador multifuncional y reprogramable, diseñado para mover materiales, piezas, herramientas o dispositivos especiales, mediante movimientos programados y variables que permiten llevar a cabo diversas tareas"*.

Existen gran variedad de robots por lo que es difícil estandarizar elementos comunes a todos. Puede decirse que la mayoría dispone de un esqueleto o chasis, que puede ser interno o externo, motores, piezas que permiten su movilidad, sistemas de agarre y manipulación y una fuente de alimentación, normalmente eléctrica.

7.2.1 Estructura de un robot industrial.- El diseño de un brazo robot industrial se inspira en el brazo humano, véase Fig. 7.3, con algunas diferencias, según su estructura y composición. Por ejemplo, un brazo robótico puede extenderse telescópicamente. También existen brazos robóticos que puedan articularse como la trompa de un elefante. Las pinzas están diseñadas tratando de imitar la función y estructura de la mano humana. Muchos robots están equipados con pinzas especializadas para agarrar dispositivos concretos, como una gradilla de tubos de ensayo o un soldador de arco.

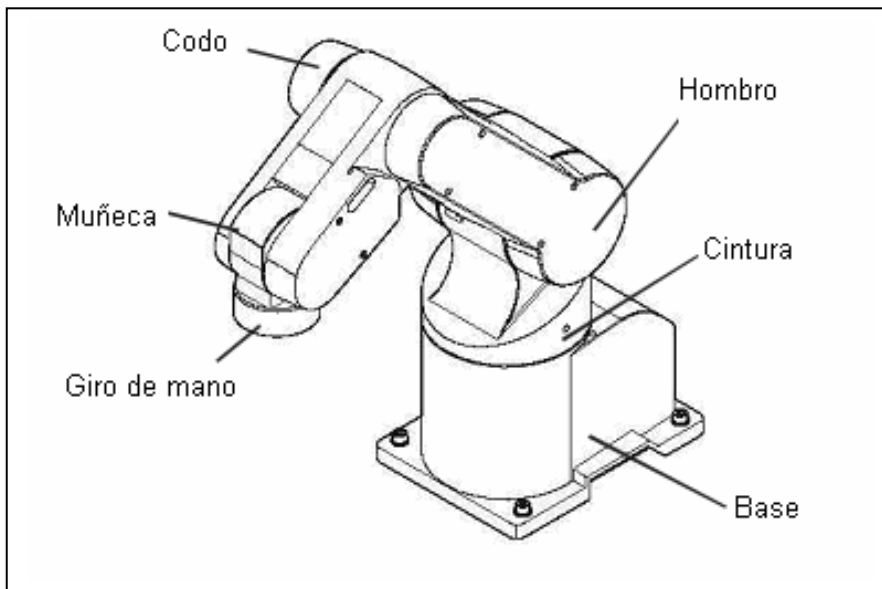


Figura 7.3 Articulaciones de un brazo robot en similitud a las de un brazo humano.

Las articulaciones de un brazo robótico suelen moverse mediante motores eléctricos, aunque pueden utilizarse otras técnicas como la neumática y la hidráulica. En la mayoría de los robots, la pinza se mueve de una posición a otra cambiando su orientación. La unidad de control calcula los ángulos de articulación necesarios para llevar la pinza a la posición deseada, un proceso conocido como cinemática inversa.

La mayoría de los brazos multiarticulados están equipados con servo controladores, o controladores por retroalimentación, que reciben datos de la unidad de control. Cada articulación del brazo tiene un dispositivo que mide su ángulo (encoder) y envía ese dato al controlador. Si el ángulo real del brazo no es igual al ángulo calculado para la posición deseada, el servo controlador mueve la articulación hasta que el ángulo del brazo coincida con el ángulo calculado. Este tipo de control se denomina control de lazo cerrado. La Fig. 7.4 muestra de forma esquemática las diferentes partes y periféricos asociados a un brazo robot [1]. La estructura mecánica y la unidad de control forman la estructura principal, que dependiendo de la aplicación a realizar se le acoplará la herramienta adecuada para la tarea programada. La unidad de control se puede ampliar para interconectar mediante buses de comunicación o entradas/salidas con otros elementos o herramientas que compartan la CMA.

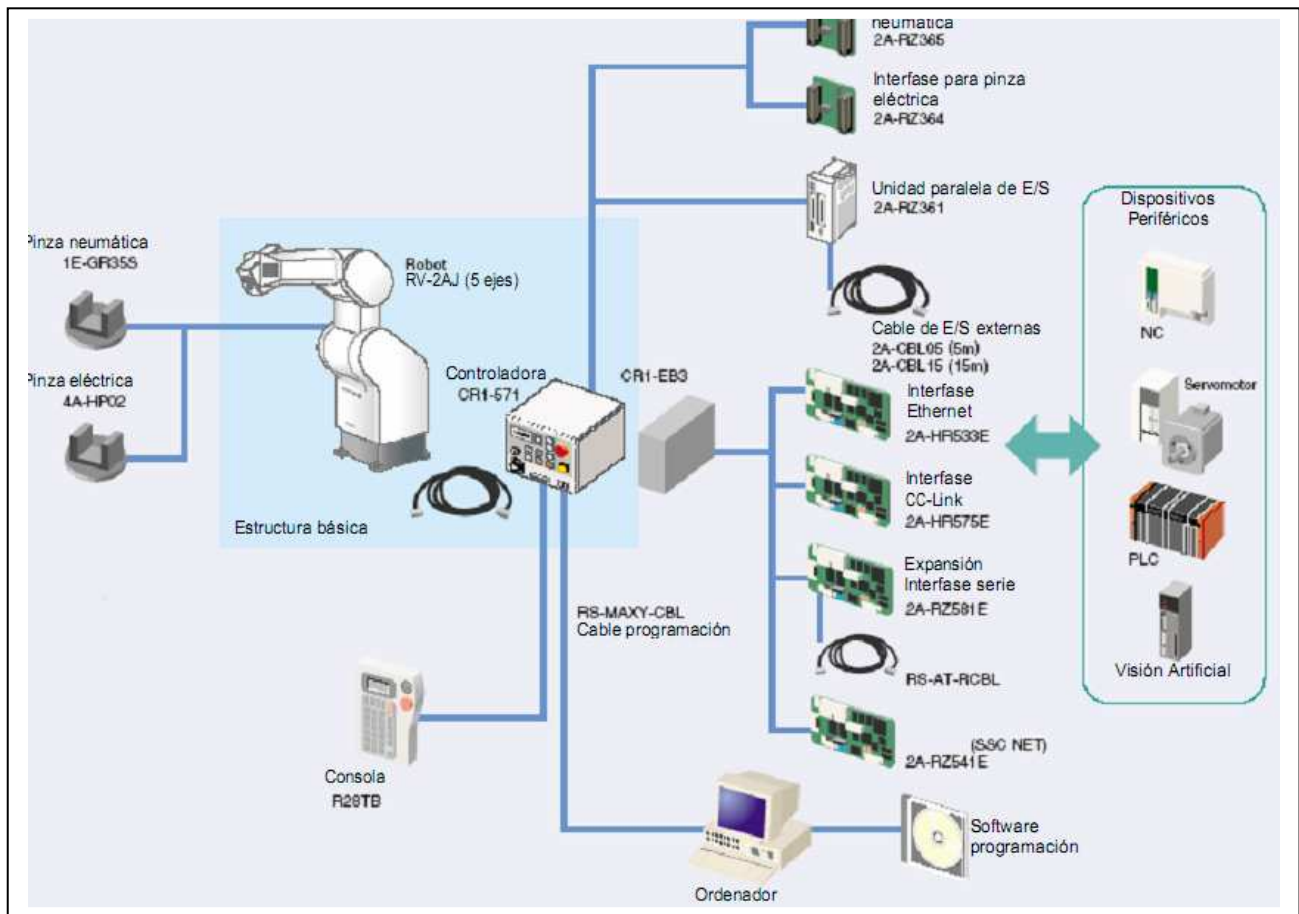


Figura 7.4 Partes y periféricos asociados al brazo robot Mitsubishi RV-2AJ.

CARACTERÍSTICA	ESPECIFICACIÓN
Grados de libertad	5
Carga máxima	2 Kg
Repetitividad	± 0.02mm
Velocidad máxima	2100 mm/seg
Alcance de la pinza	482 mm
Alcance desde Q en mm	A410, B285, C190, D300
Alcance desde Q en grados	E150
Alcance desde Q en mm	R1 220, R2 410

Tabla 7.1 Características y especificaciones del robot modelo RV-2AJ de Mitsubishi, referirse a la Fig. 7.5

7.2.2 Generalidades del robot Mitsubishi RV-2AJ.- El diseño del robot RV-2AJ es apropiado en entornos con poco espacio de maniobra para mover cargas que no superen los 2 kg. de peso, el alcance máximo con la pinza hacia abajo es de 410 mm desarrollando una velocidad de 2100 mm/seg con una repetibilidad de $\pm 0.02\text{mm}$ [1]. Las características y especificaciones del robot Mitsubishi RV-2AJ, se muestran en la Tabla 7.1, donde Q es el punto de unión del extremo del manipulador o muñeca al efector final.

Los servomotores de corriente alterna con encoders de posición absolutos garantizan una fiabilidad y un bajo costo de mantenimiento únicos en su tipo. Los encoders que utiliza permiten al control apagar el robot en cualquier momento y al conectar nuevamente la alimentación, se podrá seguir trabajando a partir de su posición actual.

El controlador utiliza un CPU de 64 bits, lo cual le permite ejecución en paralelo de hasta 32 programas en modo multitarea [2], es decir mientras se mueve puede al mismo tiempo recibir datos, activar o desactivar entradas y salidas, realizar cálculos además de 28 tareas más aún disponibles.

Las comunicaciones constan de un puerto RS-232 además de 16 entradas y salidas digitales, entrada para red Ethernet protocolo TCP/IP, entrada para red CC-Link de Mitsubishi, el cual permite intercambio inmediato de datos entre el robot y un PLC.

Grados de libertad son las dimensiones del mundo de un manipulador, cualquier movimiento de una pieza en el espacio la podemos descomponer en 6 movimientos básicos, véase Apartado 4.3.

En robótica es fundamental entender los conceptos de resolución, exactitud y repetibilidad. Por ello se definen a continuación:

- Resolución.- Movimiento mínimo que puede producir el robot. Está condicionado por la mecánica y el tipo de control.
- Exactitud.- Capacidad de un robot para situar el efector final en un punto señalado dentro de su volumen de trabajo. La exactitud es mayor cuando el robot trabaja cerca de su base.
- Repetibilidad.- Capacidad para volver repetidas veces al mismo punto.

El campo de trabajo es el volumen dentro del cual el robot puede situar el extremo de su muñeca. Está limitado por las envolventes que se producen al mover los ejes del robot entre sus posiciones mínimas y máximas. El campo de trabajo de un robot influye en el grado de accesibilidad de éste a las diferentes máquinas o elementos de la instalación, motivo por el cual cuando se desea robotizar una instalación es necesario estudiar la distribución de elementos en el entorno del robot para que aquellas piezas que deban ser manipuladas estén ubicadas dentro de su volumen de trabajo. La Fig. 7.5 muestra esquemáticamente el volumen de trabajo del robot y sus variables se muestran en la Tabla 7.1.

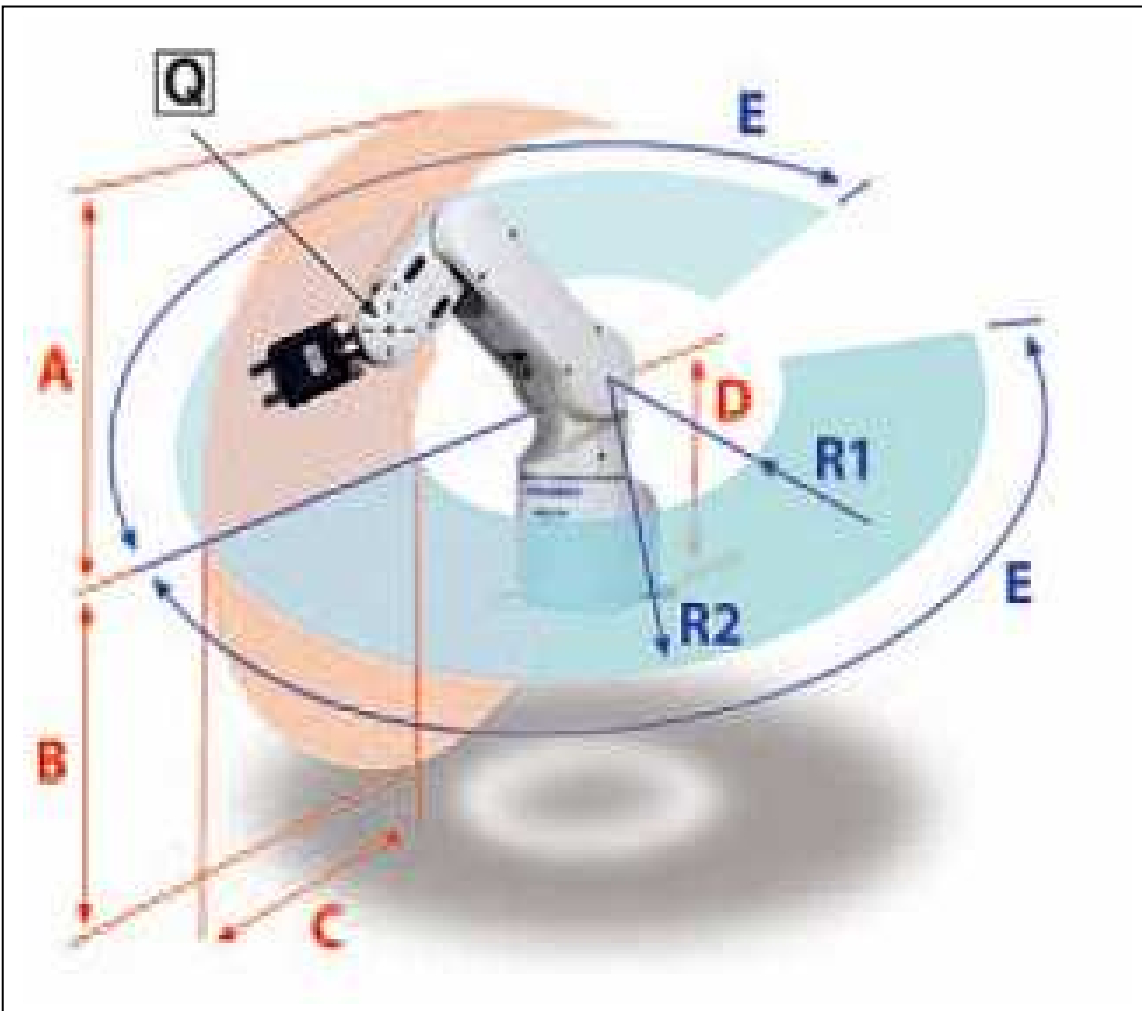


Figura 7.5 Esquema que muestra el espacio de trabajo del robot Mitsubishi RV-2AJ.

Los elementos básicos que integran el sistema robot son:

- El brazo robot Mitsubishi RV-2AJ.
- El controlador del robot.
- Una botonera de control.
- Una PC.

Cuando queremos enseñar puntos al robot para almacenarlos y asignarlos a un programa específico, usamos la botonera [2], con ella podemos mover al robot girando cada una de sus articulaciones o bien moviendo el efector final a lo largo de cada uno de los ejes coordenados x,y,z del sistema base del robot, así como los giros permitidos de su efector final que son alabeo y guiñada (véase Apartado 4.5).

El modo “Operación automática Auto(op.)” es usado para correr programas almacenados en la memoria del controlador. En éste el programa corre cíclicamente una vez iniciado.

El modo “Operación automática externa Auto(ext.)” sirve para correr programas ejecutándose en una PC ya sea en el software diseñado para este manipulador (COSIMIR) en un lenguaje propio llamado MELFA-BASIC IV que es un código similar al Basic [1,2,11] con comandos de movimiento para posicionar el efector final en los puntos de destino. Esta misma forma de operación se aplica en nuestro caso en que tenemos un programa en lenguaje “C” corriendo desde un entorno de MatLab y que se comunica mediante puerto serial RS-232 con el programa en código MELFA-BASIC IV almacenado en el control de del robot. Esta operación sirve para enviar y recibir información necesaria para mover las piezas de acuerdo a la plantilla de ensamble que se esté ejecutando en ese momento por el SPPM, véase Apartado 6.1.5.

7.3 Comunicación entre el robot y el subsistema SPPM.- El programa almacenado en el control del robot se muestra en el Apéndice C. El PLC activa al robot después de detener la banda sobre la cual se ubica la charola que porta la mesa de trabajo con las piezas, al activarse se mantiene en espera de que el SPPM analice la escena y envíe al control del robot los datos de posición y orientación de cada pieza de interés mediante un sencillo protocolo de comunicación. Cuando se ha completado el proceso de transmisión de información, se cierra el puerto de comunicación, el robot termina de mover la última pieza y el PLC libera el pistón que detenía la charola para que ésta pueda continuar su camino.

Así concluimos el capítulo referente al sistema de transporte y manipulación; en el siguiente capítulo se describirá una prueba al sistema y comentaremos los resultados y conclusiones. Además visualizaremos los posibles trabajos a futuro relacionados con el presente trabajo.

Capítulo 8

Pruebas y resultados

8.1 Antecedentes.- El contenido de éste capítulo es útil para conocer de una manera general el software del sistema y utilizarlo de manera amplia para diseñar y programar nuevas y variadas rutinas de ensamble. Aborda la totalidad de los módulos en los que se divide el SPPM abarcando todas las posibles formas en que éstos se presentan.

En este capítulo se presentan los resultados de las pruebas hechas al sistema desarrollado, éstas consisten en programar y ejecutar los procesos de ensamble descritos anteriormente en el Capítulo 6, para dos plantillas diferentes.

Una plantilla de ensamble está formada por una serie de piezas individuales, las cuales deberán estar colocadas o ensambladas en posiciones previamente programadas, dando como resultado un producto final (véase Apartado 6.1.2). El proceso para obtener este producto final se denomina proceso de ensamble. Las pruebas realizadas con el sistema que se presentan consisten en la programación y ejecución de dos plantillas de ensamble distintas (véase Fig. 8.1 y Fig. 8.2), ambas están formadas por las mismas piezas, sin embargo las piezas están dispuestas de modo diferente y por lo tanto constituyen dos productos diferentes. Las piezas que intervienen en ambas plantillas se presentan en la Tabla 8.1.

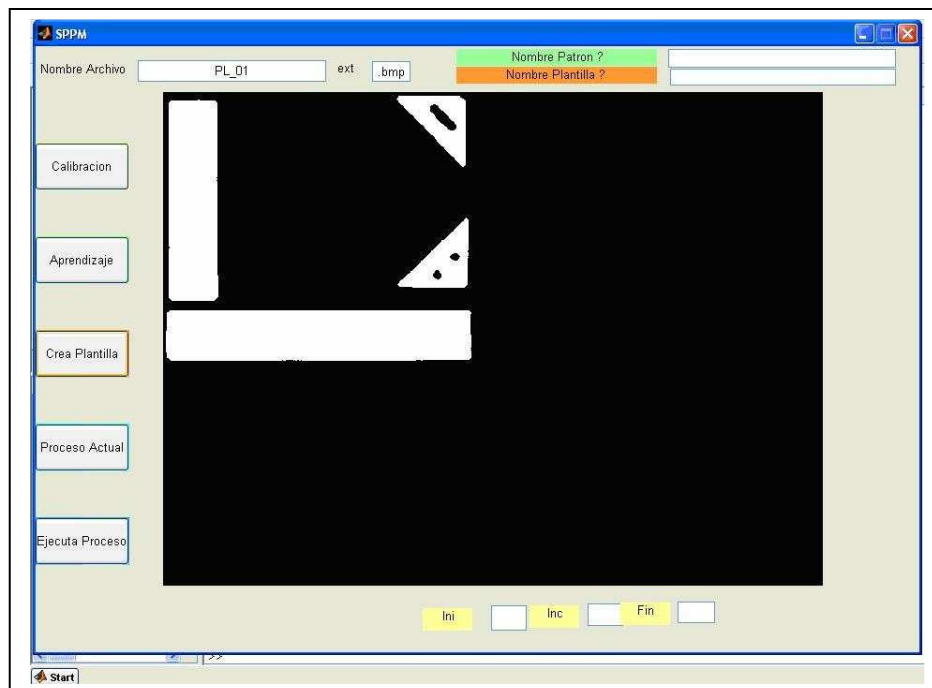


Figura 8.1 Definición y visualización en el software del sistema de la Plantilla de ensamble uno.

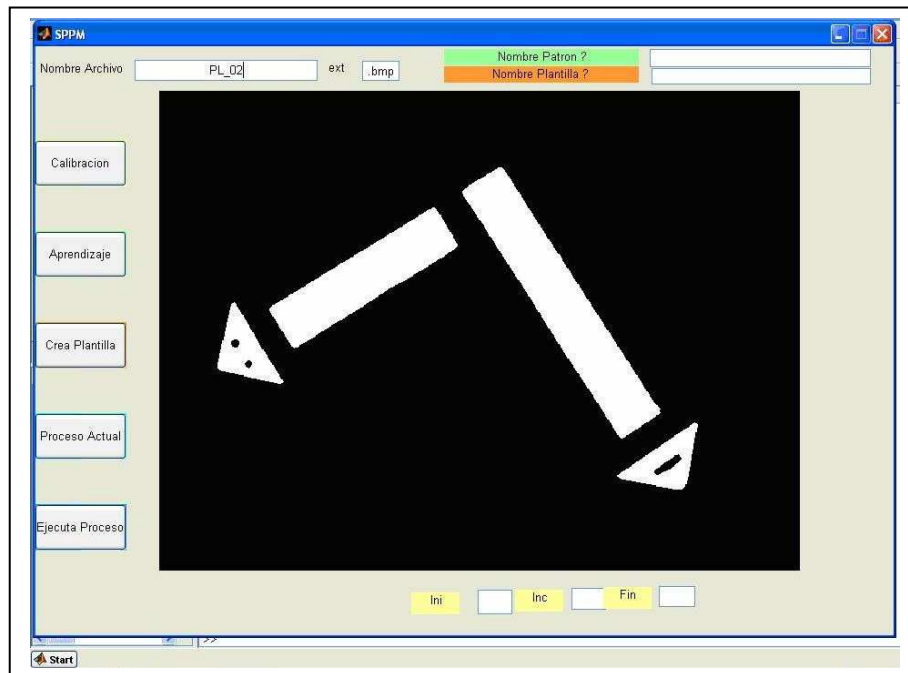


Figura 8.2 Definición y visualización en el software del sistema de la Plantilla de ensamble dos.

Patrón	Descripción	Imagen
triang2_pr	Triángulo con dos perforaciones.	
triang_ran	Triángulo con ranura horizontal.	
rec_men_sp	Rectángulo menor sin perforaciones.	
rec_may_sp	Rectángulo mayor sin perforaciones.	

Tabla 8.1 Piezas utilizadas en el proceso de ensamble de las plantillas uno y dos.

8.2 Plantillas utilizadas en las pruebas.- La ubicación de las piezas en una plantilla se describe en el Apartado 6.1.2. Las coordenadas de ubicación de las piezas correspondientes así como otros parámetros para las plantillas uno y dos se muestran en las Fig. 8.3 y 8.4 respectivamente.

8.3 Pruebas.- Las pruebas se realizaron en la celda de manufactura avanzada (CMA) del Instituto Tecnológico Superior de San Luis Potosí, Capital (ITSSLP,C). Las plantillas de ensamble se crearon con el módulo SPPM del sistema. Es este mismo módulo el que designa cual será el proceso actual por ejecutar, referirse al Apartado 6.1.2. Los valores de calibración (véase Apartado 6.1.4) así como su justificación para las dos plantillas utilizadas en las pruebas, se muestran en la Tabla 8.2.

```

Command Window
nombre: 'triang2_pr'
param: [34.8000 56.6500 224.3000]
prd: 1
st: 'P'

K>> pas_pl(2)

ans =

nombre: 'rec_may_sp'
param: [49.4600 31.4600 0.0700]
prd: 2
st: 'P'

K>> pas_pl(3)

ans =

nombre: 'triang_ran'
param: [6.0200 56.1900 -46.3800]
prd: 3
st: 'P'

K>> pas_pl(4)

ans =

nombre: 'rec_men_sp'
param: [22.1500 6.1500 -89.9100]
prd: 4
st: 'P'

K>>
Start | Stopped in debugger

```

Figura 8.3 Estructura de datos en RAM donde se describen los elementos que forman la plantilla uno.

```

Command Window
nombre: 'triang2_pr'
param: [54.3900 17.8700 120.7200]
prd: 1
st: 'P'

K>> pas_pl(2)

ans =

nombre: 'triang_ran'
param: [77.6000 105.2300 214.4900]
prd: 2
st: 'P'

K>> pas_pl(3)

ans =

nombre: 'rec_men_sp'
param: [39.0700 42.5300 31.7500]
prd: 3
st: 'P'

K>> pas_pl(4)

ans =

nombre: 'rec_may_sp'
param: [44.2300 83.5600 -57.2100]
prd: 4
st: 'P'

Start | Busy

```

Figura 8.4 Estructura de datos en RAM donde se describen los elementos que forman la plantilla dos.

Concepto	Valores	Justificación
Tamaño de la imagen capturada por la cámara.	$ri = 1$ $rf = 479$ $ci = 1$ $cf = 639$	Considera la imagen completa, 480x640 píxeles.
Campo visual	$CVv = 100$ $CVh = 133$	Con objeto de abarcar la mayor superficie posible de la mesa de trabajo, para la posición de la cámara y enfoque adecuado, es un rectángulo de 100x133 mm por lado.
Valores de traslación del sistema base para alinearse con el sistema mesa de trabajo (mm).	$dx = 289.72$ $dy = -204.06$ $dz = 332.5$	Requerido para cálculo de matrices de transformación.
Giro en cada uno de los ejes del sistema base para alinearse con el sistema mesa de trabajo.	$A = 0^\circ$ $B = 0^\circ$ $C = 180^\circ$	Requerido para cálculo de matrices de transformación.
Valores de traslación en cada eje y giro en Z del sistema mesa de trabajo para alinearse con el sistema de la imagen.	$dxi = 0$ $d yi = 100$ $d zi = 0$ $C gi = -90^\circ$	Requerido para cálculo de matrices de transformación.

Tabla 8.2 Valores de calibración y su justificación.

8.3.1 Plantilla uno.- Cuando se selecciona la plantilla uno como la guía para ensamble, se proporcionan también los valores de su ubicación (véase Apartado 6.1.3) respecto al sistema coordenado de la mesa de trabajo, que para esta prueba fueron:

$$\begin{aligned}
 xp &= 230 \text{ mm} \\
 yp &= 100 \text{ mm} \\
 zp &= 0 \text{ mm} \\
 Rz &= 180^\circ
 \end{aligned}$$

La Figura 8.1 muestra la imagen de la plantilla uno, la Figura 8.3 muestra información de cada pieza en la plantilla referidos a su sistema local de coordenadas; *nombre de la pieza*; *param* se refiere a las coordenadas (x,y) de su centro de masa y al ángulo de la pieza respecto a la horizontal; *prd* es la prioridad que tiene la pieza en el proceso en cuanto al orden de ensamblado y *st* es el estatus que guarda la pieza por el momento.

Mediante el botón “Ejecuta ensamble” en el SPPM, el sistema toma una imagen de la escena cuyo ejemplo se muestra en la Figura 8.5 y ejecuta los procesos de reconocimiento y cálculo de posiciones actuales y de destino de las piezas que participan en el proceso de ensamble de la plantilla en proceso.

La función *analiza_escena3()*, véase Apartado 5.2.5, es llamada a operar por el módulo “Ejecuta_ensamble”. Esta función proporciona la cantidad de piezas que aparecen en la escena además de sus coordenadas y orientación (véase Fig. 8.6).

La Figura 8.8 muestra una imagen digitalizada de un dibujo a escala 1:2 (un milímetro en el papel representan dos milímetros en el mundo real) que se realizó en papel milimétrico, donde se representan los sistemas coordenados del robot (x_0, y_0, z_0), de la mesa de trabajo (x_1, y_1, z_1), de la imagen de la escena captada por la cámara (x_2, y_2, z_2) y de la plantilla en su ubicación de ensamblado (x_3, y_3, z_3). Con el propósito de determinar la precisión con la que el sistema calcula los valores que ha de transmitir al control del robot para que éste realice las tareas de tomar y dejar para cada pieza, en ésta Figura se han representado las piezas que aparecen en la Fig. 8.5 dibujando sobre la hoja de papel los datos de las piezas (x, y , y su ángulo de inclinación) que se muestran en la Fig. 8.6 respecto al sistema coordenado (x_2, y_2, z_2). De igual manera en la Fig. 8.8 se dibujaron las piezas ya ensambladas de la plantilla uno como se representan en la Fig. 8.1 en base a la información que se muestra en la Fig. 8.3, respecto al sistema coordenado (x_3, y_3, z_3). Las coordenadas de estos puntos sobre la hoja de papel milimétrico podrán ser comparadas con los valores transmitidos al control del robot y que el sistema almacena en un archivo de texto llamado “puntos.txt” (véase Fig.8.7). Los valores transmitidos al robot son ($x, y, z, alabeo, guiñada$) y están referidos respecto al sistema coordenado del robot (x_0, y_0, z_0).

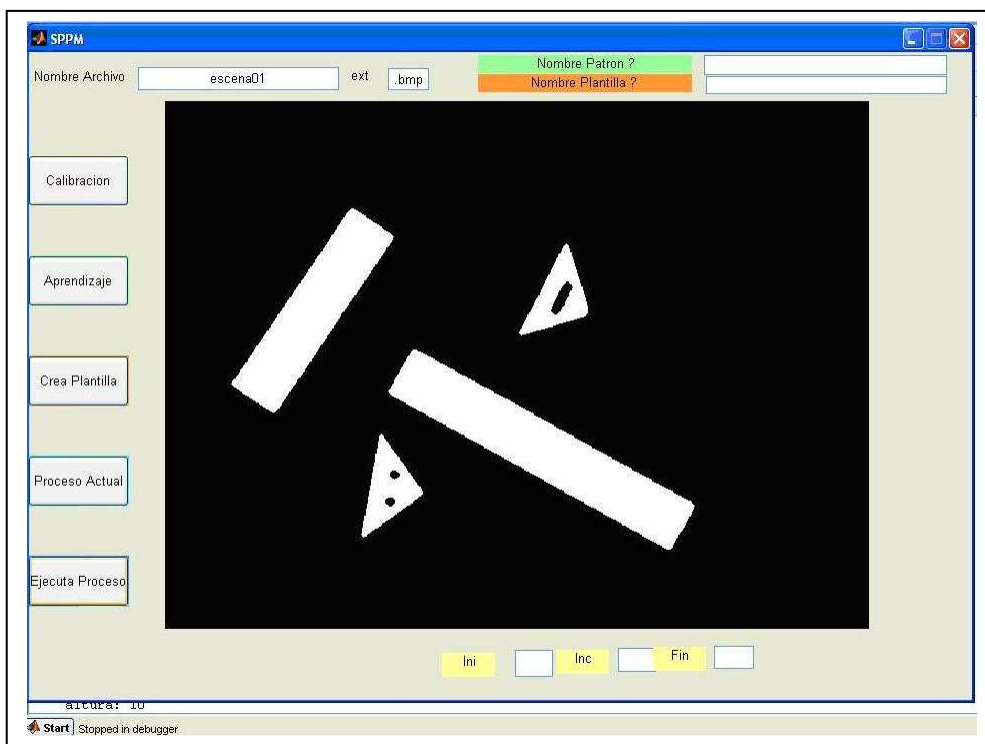
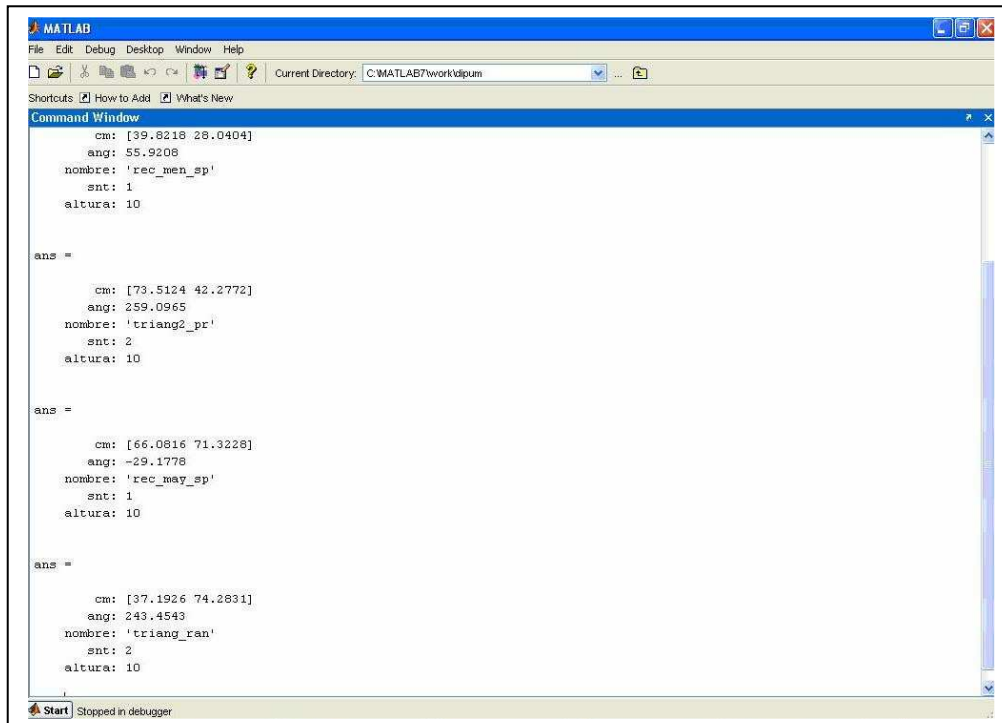


Figura 8.5 Piezas en la escena para ensamblar la plantilla uno.



```

MATLAB
File Edit Debug Desktop Window Help
Current Directory: C:\MATLAB7\work\dipum

Shortcuts How to Add What's New

Command Window

cm: [39.8218 28.0404]
ang: 55.9208
nombre: 'rec_men_sp'
snt: 1
altura: 10

ans =

cm: [73.5124 42.2772]
ang: 259.0965
nombre: 'triang2_pr'
snt: 2
altura: 10

ans =

cm: [66.0816 71.3228]
ang: -29.1778
nombre: 'rec_may_sp'
snt: 1
altura: 10

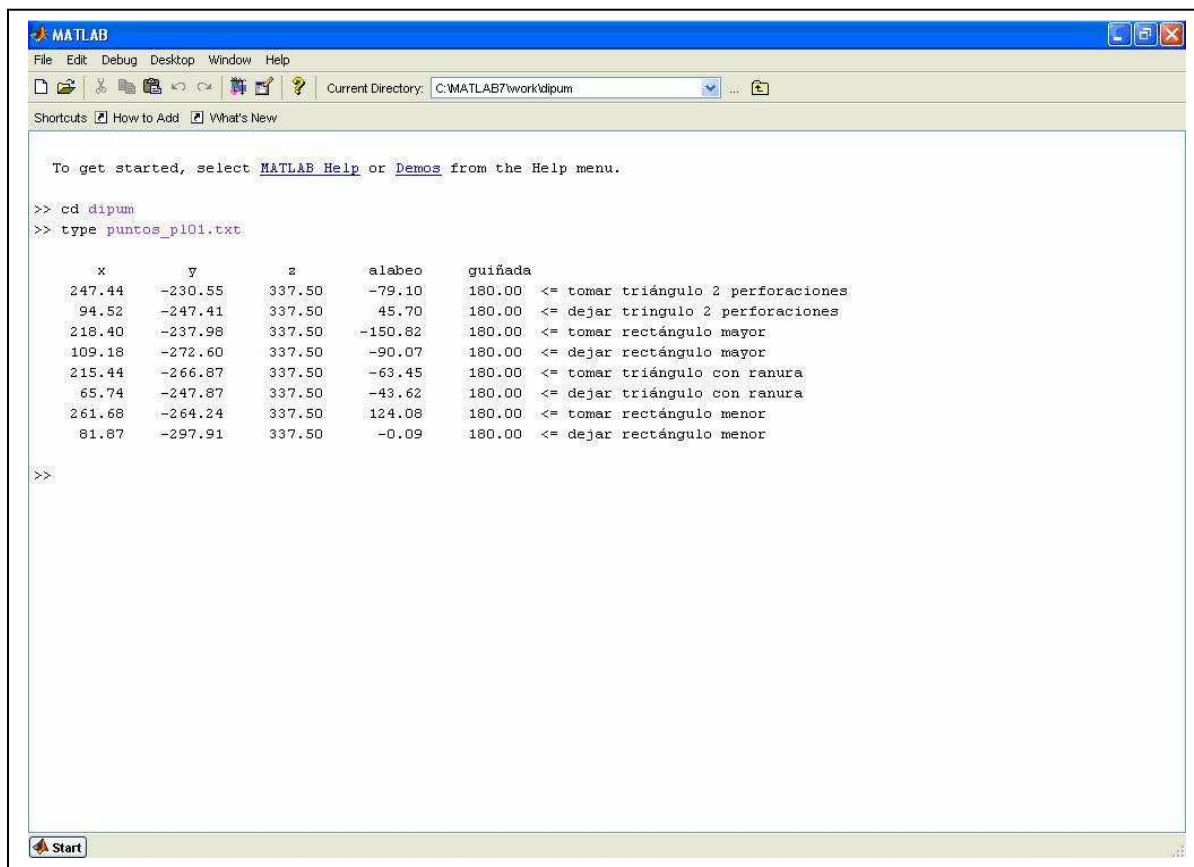
ans =

cm: [37.1926 74.2831]
ang: 243.4543
nombre: 'triang_ran'
snt: 2
altura: 10

Start Stopped in debugger

```

Figura 8.6 La Información de cada pieza reconocida es proporcionada por la función *analiza_escena3()*.



```

MATLAB
File Edit Debug Desktop Window Help
Current Directory: C:\MATLAB7\work\dipum

Shortcuts How to Add What's New

To get started, select MATLAB Help or Demos from the Help menu.

>> cd dipum
>> type puntos_p101.txt

x      y      z      alabeo      guiñada
247.44 -230.55 337.50 -79.10      180.00 <= tomar triángulo 2 perforaciones
94.52  -247.41 337.50 45.70       180.00 <= dejar triángulo 2 perforaciones
218.40 -237.98 337.50 -150.82     180.00 <= tomar rectángulo mayor
109.18 -272.60 337.50 -90.07      180.00 <= dejar rectángulo mayor
215.44 -266.87 337.50 -63.45      180.00 <= tomar triángulo con ranura
65.74  -247.87 337.50 -43.62      180.00 <= dejar triángulo con ranura
261.68 -264.24 337.50 124.08     180.00 <= tomar rectángulo menor
81.87  -297.91 337.50 -0.09       180.00 <= dejar rectángulo menor

>>

```

Figura 8.7 Información para manipulación en el ensamble de la plantilla uno.

Nótese que en el proceso de ensamble de la plantilla uno, las piezas que se disponen en la escena son justamente las que se requieren para llevar a cabo el proceso de ensamble.

8.3.2 Plantilla dos.- En el ensamble de la plantilla dos (véase Fig. 8.2) se requirió que la mesa de trabajo colocada sobre la charola que viaja en la banda del SMP, desarrollara dos vueltas en torno a su trayectoria de desplazamiento, en cada una de ellas se alimentó con varias piezas de trabajo de las cuales sólo dos en cada ocasión formaban parte del proceso de ensamble (véase Figs.8.9 y 8.10). La banda transportadora es finita y describe una trayectoria cíclica de forma rectangular (véase Figs. 4.1 y 4.4), para el ensamble de la plantilla dos se requirieron dos ciclos, sin embargo el número de ciclos para completar un ensamble varía en cada caso, dependiendo de la manera en que las piezas que participan en el ensamble sean alimentadas en la charola que viaja sobre la banda. Así el número de ciclos de la banda varía de 1 a n . Esto es una contribución más al sistema para ser considerado un sistema flexible de manufactura.

Al finalizar la primer vuelta, el sistema almacena en un archivo el resultado del proceso o estatus, en éste aparece el nombre de la plantilla; el orden de prioridad para cada pieza y la situación en la que se encuentra la pieza al finalizar el proceso que puede ser la letra "T", si terminó de ensamblarse o la letra "P", si está pendiente de ensamblarse. Esta información es leída inmediatamente antes de iniciar el siguiente ciclo, así el sistema sabe que piezas deberá tomar para completar el proceso de ensamble. En el segundo ciclo se colocaron las dos piezas restantes y tanto en la escena del primero como en la del segundo ciclo se colocaron piezas adicionales que el sistema detecta y reconoce siempre y cuando hayan sido previamente dadas de alta en el SPPM mediante el módulo de aprendizaje. Estas piezas que se añadieron no forman parte de la plantilla en proceso, el sistema simplemente las ignora, por tanto permanecerán en sus posiciones originales todo el tiempo. Con una sencilla modificación al sistema, estas piezas que no intervienen en el proceso, pueden también ser retiradas y depositadas en un contenedor por el propio robot, reafirmando así una vez más la característica de flexibilidad con que cuenta el sistema.

La Fig. 8.11 muestra resultados del proceso de ensamble de la plantilla dos. El origen del sistema coordinado de la plantilla se localizará en las coordenadas espaciales (230, 100, 0) con un giro en Z de 180° respecto al sistema de la mesa de trabajo. Se muestran además los datos a transmitir al control de robot para posicionar las únicas piezas que encontró en la escena de primera vuelta y que participan de éste proceso, que son el rectángulo menor y el triángulo con ranura horizontal. En seguida se muestra el estado del proceso después de la primera vuelta, en el que observamos que las piezas pendientes por ensamblar son la primera y la cuarta en base al ordenamiento mostrado en la Fig. 8.4, las cuales corresponden al triángulo con dos perforaciones y al rectángulo mayor. En la segunda vuelta los valores calculados para el control del robot corresponden a éstas últimas dos piezas, finalmente la Fig. 8.11 muestra el estado del proceso terminada la segunda vuelta, se observa que todas las piezas ya han sido ensambladas.

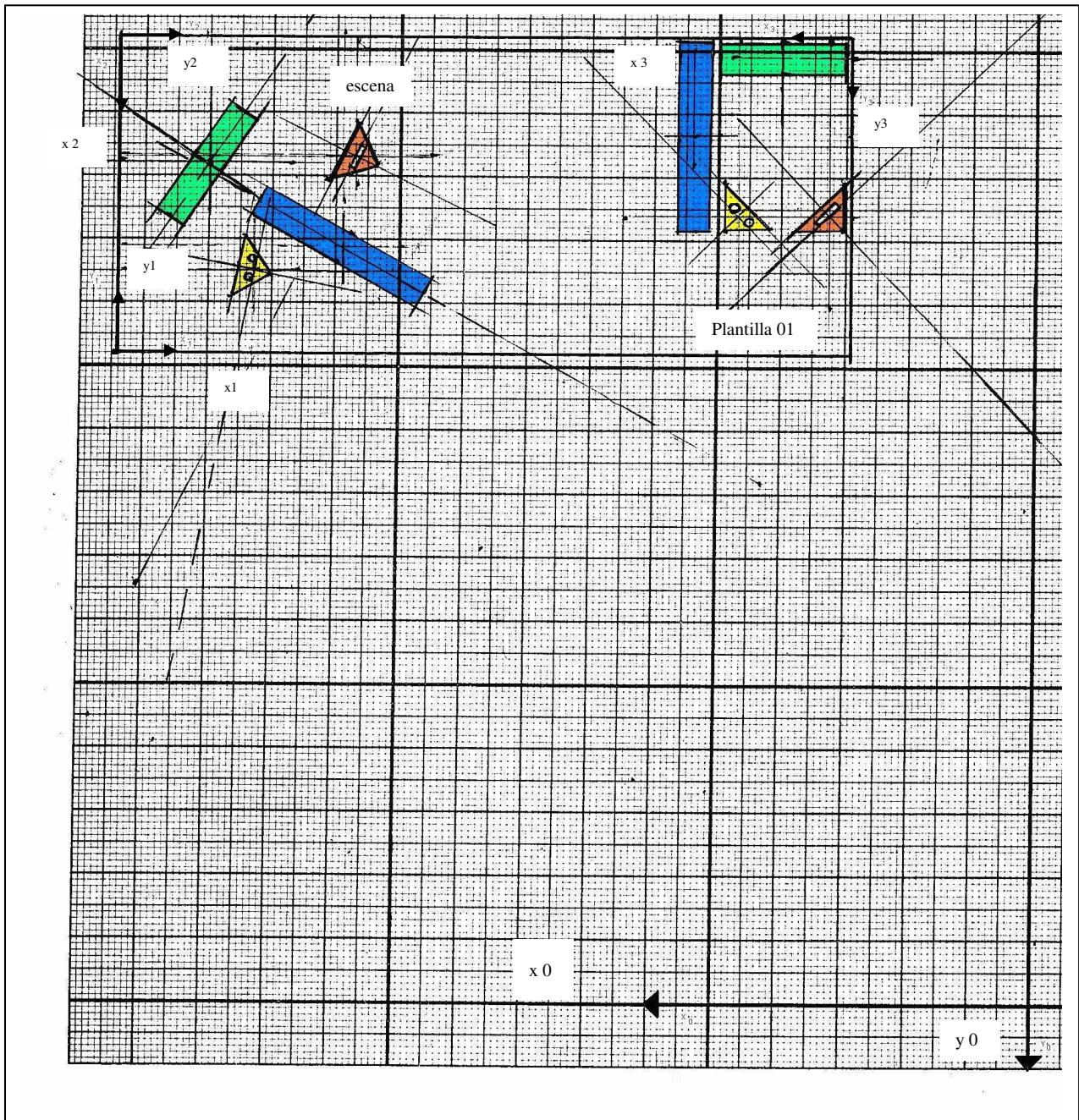


Figura 8.8 Dibujo digitalizado que muestra las piezas en la escena en el proceso de ensamble de la plantilla uno además de la plantilla finalmente ensamblada.

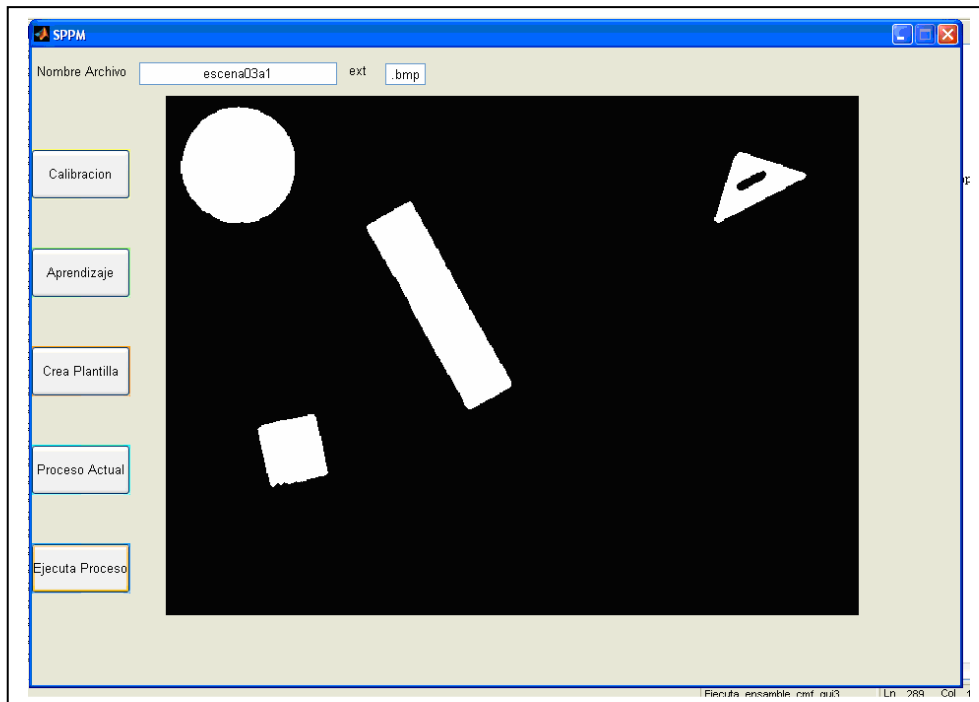


Figura 8.9 Escena en primera vuelta para ensamble de plantilla dos.

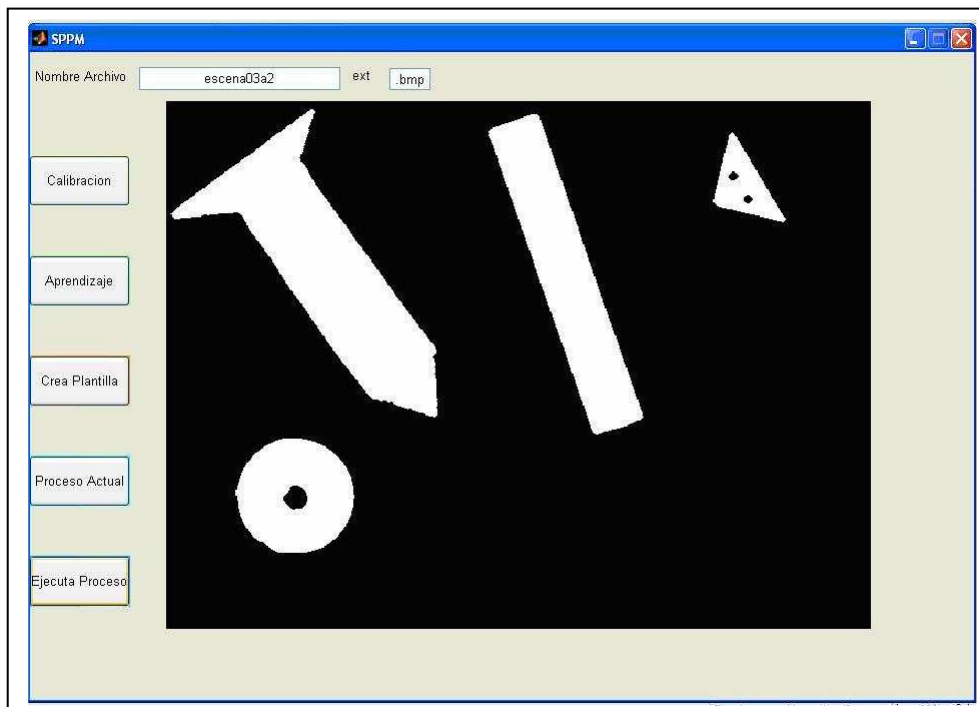
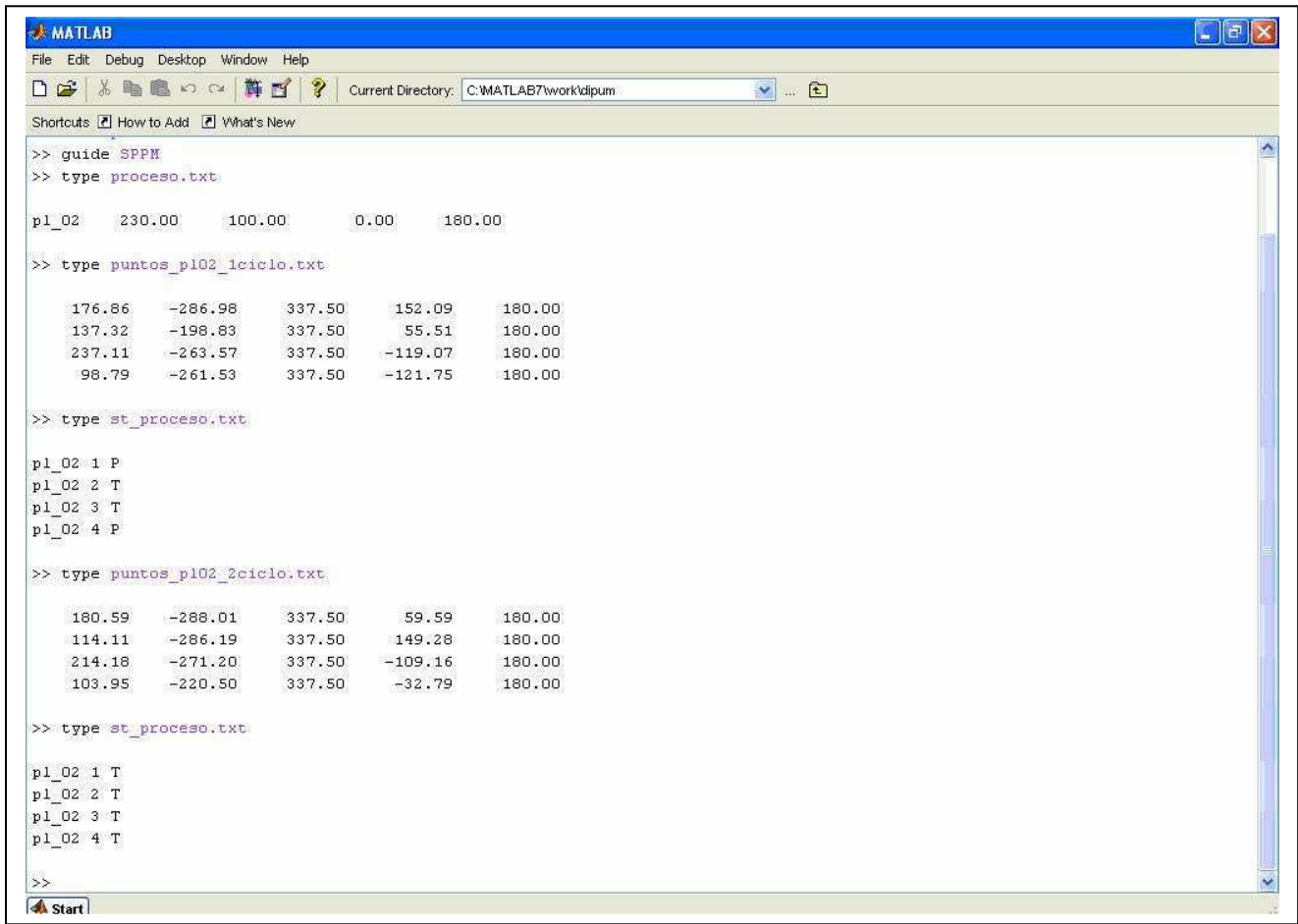


Figura 8.10 Escena en segunda vuelta para ensamble de plantilla dos.



```

MATLAB
File Edit Debug Desktop Window Help
Current Directory: C:\MATLAB7\work\kdlpum
Shortcuts How to Add What's New

>> guide SPPM
>> type proceso.txt

pl_02 230.00 100.00 0.00 180.00

>> type puntos_pl02_1ciclo.txt

176.86 -286.98 337.50 152.09 180.00
137.32 -198.83 337.50 55.51 180.00
237.11 -263.57 337.50 -119.07 180.00
98.79 -261.53 337.50 -121.75 180.00

>> type st_proceso.txt

pl_02 1 P
pl_02 2 T
pl_02 3 T
pl_02 4 P

>> type puntos_pl02_2ciclo.txt

180.59 -288.01 337.50 59.59 180.00
114.11 -286.19 337.50 149.28 180.00
214.18 -271.20 337.50 -109.16 180.00
103.95 -220.50 337.50 -32.79 180.00

>> type st_proceso.txt

pl_02 1 T
pl_02 2 T
pl_02 3 T
pl_02 4 T

>>
Start

```

Figura 8.11 Cálculo de los parámetros y el estado de las dos vueltas en el ensamble de la plantilla dos.

8.4 Resultados.- En las dos pruebas descritas, el sistema ejecutó los ensambles programados satisfactoriamente. La exactitud en el cálculo de las posiciones es de $\pm 1\text{mm}$ (correspondiente a 0.33 mm para las dimensiones de trabajo). El proceso de reconocimiento de piezas mejoró mucho al incorporar, en el módulo de aprendizaje del SVA, la imagen binaria del patrón en su posición inicial. Esto permite reducir el error al considerar la forma de la pieza además de su tamaño y su topología. Lo cual establece como primer criterio de búsqueda del patrón los descriptores de diámetro equivalente y número de Euler (véase Apartado 5.1.1.4) y como segundo criterio su imagen. En el proceso de reconocimiento, el contar con la imagen de la pieza, brinda elementos suficientes para determinar con certeza a que patrón pertenece además para calcular la orientación que la pieza tiene y su sentido. Así para una pieza alargada además de reconocerla, se determina también su ángulo de orientación, que podrá estar entre 0° y 360° definiendo así un sentido inequívoco. El sistema falla en la determinación de éste ángulo cuando la pieza no tiene un único eje de menor inercia, como ocurre en el caso de un cuadrado, un disco o una rondana (véase Apartado 5.1.1.5). Para el caso de formas circulares, no existe problema al momento que el robot toma la pieza, no así para piezas con forma cuadrada.

Conclusiones

El sistema desarrollado trabaja satisfactoriamente para ensambles con piezas de forma libre que cumplan con las especificaciones establecidas. Entre estas especificaciones están ser de forma plana alargada de 1 a 8 cm con anchos entre 0.50 a 3 cm (dependiendo del tipo de efector final a utilizar, contamos con dos variantes), no deberán estar amontonadas ni tocarse entre si, no hay límite en cuanto al número de piezas que podrán aparecer en la escena. Estas piezas se ubican aleatoriamente en una mesa de trabajo montada sobre una charola que se desplaza sobre una banda transportadora. Al llegar a la estación de manipulación y ensamble, las piezas son tomadas de su posición actual y ubicadas en su posición final de ensamble. Las plantillas de ensamble se diseñan fácilmente mediante el sistema SPPM y se pueden modificar de manera simple editando el archivo de texto que las define. Entre las modificaciones que el usuario del sistema puede realizar directamente en el archivo de texto sin tener que volver a crear la plantilla tenemos; editar el nombre de una o varias piezas, modificar su valor de prioridad en el proceso de ensamble, cambiar cualquier valor ya sea de las coordenadas de su centro de masa o bien del ángulo de inclinación de la pieza, puede eliminar o añadir piezas al proceso, en una palabra cualquier cambio teniendo como límite solo su imaginación. Es posible modificar en línea estos parámetros. Las plantillas se ubican en cualquier parte de la mesa de trabajo o de cualquier otra superficie donde sean requeridas y se orientan según las necesidades del usuario siempre y cuando se localicen dentro del volumen de trabajo del robot. Es importante mencionar que el área destinada para el ensamble de la plantilla deberá estar libre de piezas u otros obstáculos que pudieran provocar alguna colisión al momento del ensamble. El proceso puede ejecutarse de forma continua y de manera no supervisada mientras se alimenta a la charola con piezas para ensamble. El sistema cumple con el requerimiento de modularidad, ya que fue integrado tomando como base tres de las nueve estaciones de trabajo que integran el centro de manufactura avanzada del Instituto Tecnológico Superior de San Luis Potosí, Capital. De igual forma puede crecer al incorporar nuevos módulos como podría ser por ejemplo, una estación de maquinado. Es un sistema fácil de operar en tanto que su aplicación cuenta con una interface gráfica de usuario que únicamente tiene una ventana en ella se presentan la totalidad de sus módulos. Cumple también con el requerimiento de ser objetivo y didáctico, ya que podemos utilizar un mismo conjunto de piezas y ensamblar varias plantillas totalmente diferentes a diferencia del sistema con el que actualmente cuenta la celda del Instituto con la que sólo podemos fabricar un solo producto que es un “cilindro de carrera corta”.

El presente trabajo se puede tomar como ejemplo para aquellos que desean implementar un sistema flexible de manufactura a partir de un sistema de fabricación rígido, como es el caso del sistema desarrollado para la CMA del ITSSLP,C .

Trabajo futuro

A partir del presente trabajo se proponen las siguientes mejoras a futuro:

- Incorporar en el SVA mejores características de reconocimiento al considerar también escenas donde sus elementos se junten o bien que estén traslapados. Mejorar la determinación del ángulo de inclinación para piezas cuadradas.
- Añadir un Subsistema de Estación de Maquinado, con al menos una máquina CNC, la cual realice operaciones de torno o fresado tales como torneado, barrenado, fileteado, redondeado y acabados en las piezas previo a su ensamblado.
- Añadir un Subsistema de Control de Calidad Asistido por Visión Artificial (SCCAV) con objeto de determinar si las operaciones realizadas por el SPPM requieren un reproceso o si cumplen con los estándares de calidad establecidos.
- Incorporar estaciones de trabajo adicionales. Éstas serían controladas directamente por el PLC del sistema de transporte.

Apéndice A

El siguiente ejemplo pretende mostrar la composición de transformaciones homogéneas bajo los criterios vistos en la Sección 4.4.1, se comprueba además que la composición de transformaciones homogéneas no es conmutativa y que el resultado es el mismo al resolver el problema siguiendo cualquiera de estos criterios.

Ejemplo 1:

En base a la Fig. A.1, obtener las coordenadas del punto B respecto al sistema fijo $X_0Y_0Z_0$, dadas las coordenadas del mismo punto respecto al sistema $X_1Y_1Z_1$. Es decir, encontrar la matriz de transformación que multiplique el vector $[10, 5, 0, 1]$ y cuyo resultado deba ser el vector $[50, -60, 5, 1]$. Resolver el problema mediante el criterio descrito en los puntos 2 y 3 del Apartado 4.4.1.

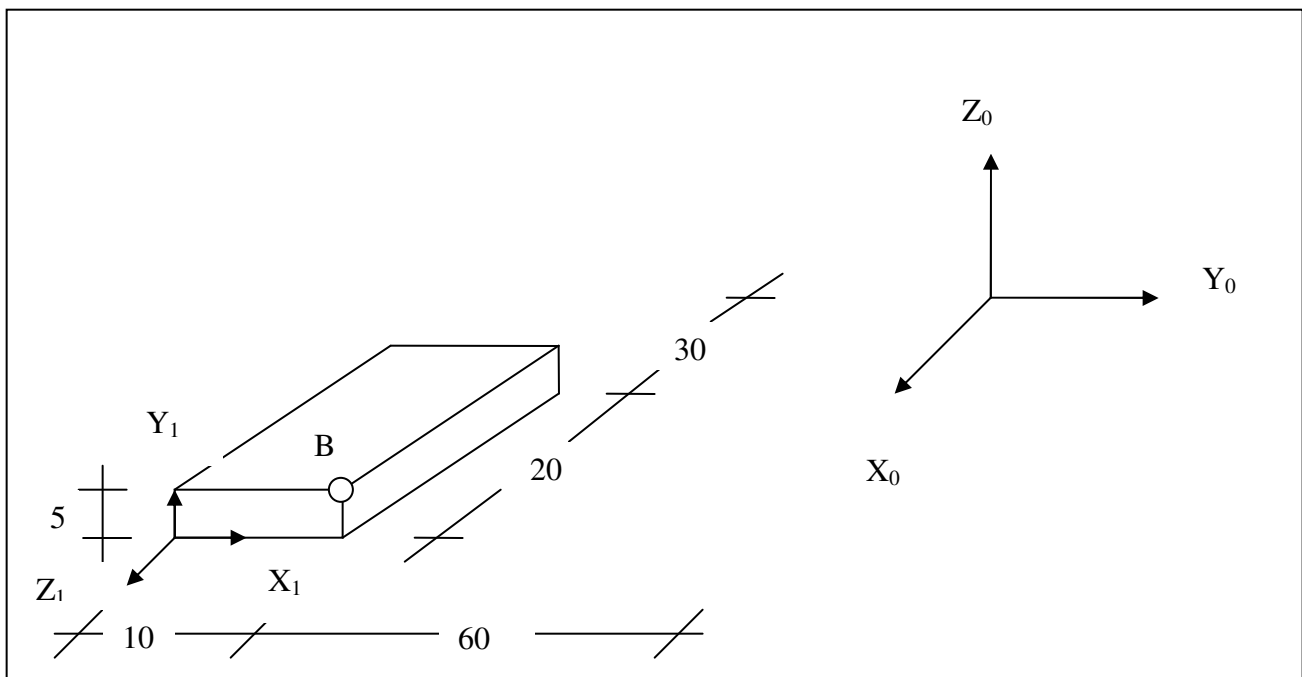


Figura A.1 Determinar las coordenadas del punto B respecto a $X_0Y_0Z_0$ en base las coordenadas del mismo punto pero respecto al sistema $X_1Y_1Z_1$.

1ª. Solución.- De acuerdo al criterio 3, el sistema móvil $X_0Y_0Z_0$ se traslada al origen del sistema fijo $X_1Y_1Z_1$ y se aplican rotaciones respecto al sistema trasladado para alinearlo y hacerlo coincidir con el sistema $X_1Y_1Z_1$. Se aplica una traslación por $[50, -70, 0, 1]$, seguida de una rotación respecto al eje OZ de 90° , seguida de una rotación respecto al eje OX de 90° como lo indica la Figura A.2, dando como resultado la siguiente expresión:

$$T = \text{Tr} \begin{bmatrix} 1 & 0 & 0 & 50 \\ 0 & 1 & 0 & -70 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C90^\circ & -S90^\circ & 0 & 0 \\ S90^\circ & C90^\circ & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C90^\circ & -S90^\circ & 0 \\ 0 & S90^\circ & C90^\circ & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -1 & 0 & 50 \\ 1 & 0 & 0 & -70 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T = \begin{bmatrix} 0 & 0 & 1 & 50 \\ 1 & 0 & 0 & -70 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

De tal forma que las coordenadas del punto B referidas al sistema $X_0Y_0Z_0$ serán:

$$B = T \times [10 \ 5 \ 0 \ 1]^T$$

$$= \begin{bmatrix} 0 & 0 & 1 & 50 \\ 1 & 0 & 0 & -70 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 10 \\ 5 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 50 \\ -60 \\ 5 \\ 1 \end{bmatrix}$$

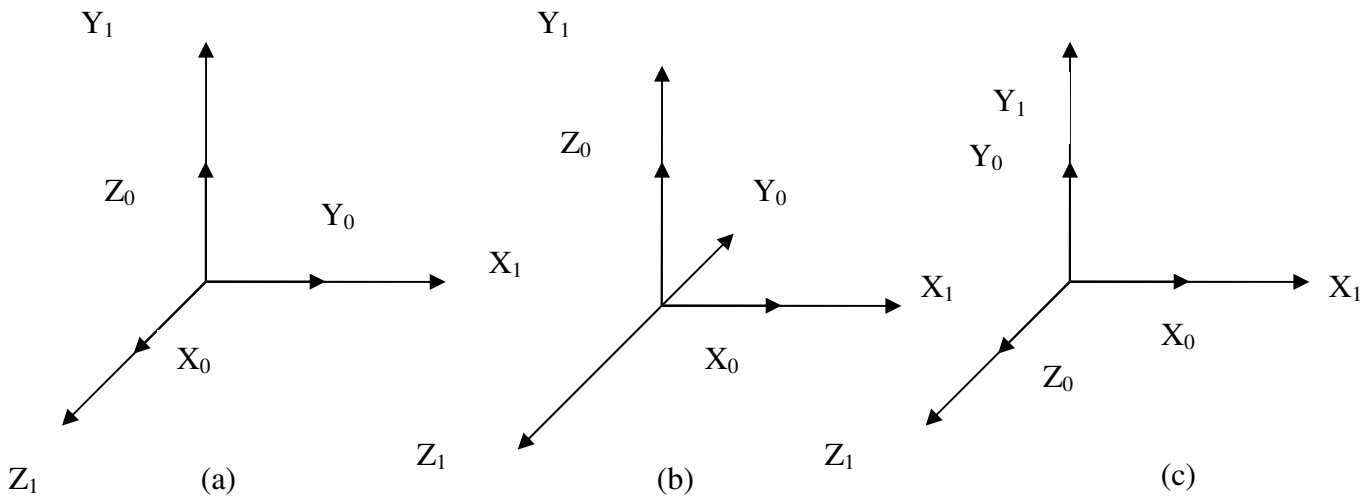


Figura A.2 (a) Sistemas coincidentes en el origen después de la traslación. (b) Después de rotar 90° respecto a OZ . (c) Después de rotar 90° respecto OX , aquí se observa que los sistemas están alineados y coinciden.

2ª. Solución.- De acuerdo al criterio 2, el sistema móvil $X_1Y_1Z_1$ se considera primeramente coincidente y alineado con el sistema fijo $X_0Y_0Z_0$ aplicamos una rotación de 90° respecto al eje OZ , seguida de una rotación respecto al eje OY de 90° , de esta manera logramos que el sistema $X_1Y_1Z_1$ tome su orientación definitiva, nos resta aplicar una traslación respecto al

sistema fijo $X_0Y_0Z_0$ por $[50, -70, 0, 1]$ como lo indica la Figura A.3, dando por resultado la siguiente expresión:

$$T = Tr R(y, 90) R(z, 90) = \begin{bmatrix} 1 & 0 & 0 & 50 \\ 0 & 1 & 0 & -70 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C90^\circ & 0 & S90^\circ & 0 \\ 0 & 1 & 0 & 0 \\ -S90^\circ & 0 & C90^\circ & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ C90^\circ & -S90^\circ & 0 & 0 \\ S90^\circ & C90^\circ & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T = \begin{bmatrix} 1 & 0 & 0 & 50 \\ 0 & 1 & 0 & -70 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T = \begin{bmatrix} 0 & 0 & 1 & 50 \\ 1 & 0 & 0 & -70 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

De tal forma que las coordenadas del punto B referidas al sistema $X_0Y_0Z_0$ serán:

$$B = T \times [10 \ 5 \ 0 \ 1]^T$$

$$= \begin{bmatrix} 0 & 0 & 1 & 50 \\ 1 & 0 & 0 & -70 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 10 \\ 5 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 50 \\ -60 \\ 5 \\ 1 \end{bmatrix}$$

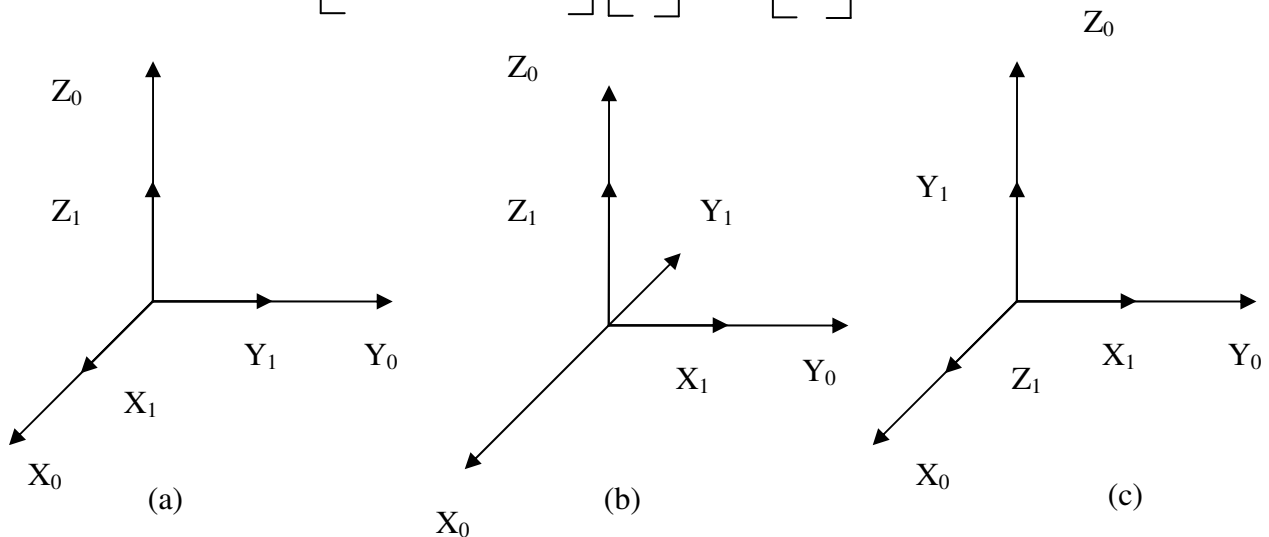


Figura A.3 (a) Sistemas alineados y coincidentes en el origen. (b) Después de rotar 90° respecto a OZ . (c) Después de rotar 90° respecto OY , por último, se aplica la traslación respecto al sistema fijo $X_0Y_0Z_0$ por $[50, -70, 0, 1]$

Apéndice B

Código fuente de las funciones del SVA y del SPPM. Se codifican en lenguaje C y corren en el ambiente MatLab Ver. 7 sobre el sistema operativo MS-Windows XP.

Índice de funciones

Adquisición de imagen.	100
Analiza escena.	100
Aprendizaje.	102
Compara imágenes.	103
Crea plantilla para ensamble.	104
Crea proceso de ensamble.	105
Descripción de imagen.	105
Ejecuta ensamble en línea.	106
Ejecuta ensamble.	107
Lee patrones.	113
Lee plantilla.	114
Preproceso de imagen.	115

function [f] = Adq_imagen(archivo, ext, ri, rf, ci, cf)

```
%ADQ_IMAGEN Obtiene una imagen f en tonos de gris, recortada y en formato
%.jpg, dada una imagen original de nombre archivo, a color y en cualquier formato
%.ext La imagen de salida tendrá un tamaño: (rf-ri) renglones por (cf-ci)
%columnas. Las coordenadas de la esquina superior
%izquierda del área recortada respecto a la imagen original son (ri, ci)
filename=sprintf('%s%s', archivo, ext);
f=imread(filename);
f=f(ri:rf, ci:cf);
filename=sprintf('%s.jpg', archivo);
imwrite(f, filename);
f=imread(filename);
```

function [piezas, npz]=Analiza_escena3(imagen, ext, ri, rf, ci, cf, CVv, CVh)

```
%ANALIZA_ESCENA Función que toma como entrada una imagen con las piezas por
%manipular colocadas de manera libre sobre la mesa de trabajo. La función
%llama primeramente a trabajar a la función que toma la imagen y la recorta
%de tal modo que solo aparezca la región de interés, posteriormente se llama
%a operar a la función que preprocesa la imagen y la regresa binarizada y
%lista para ser segmentada y de esa manera llamar a la función que caracteriza y
%describe los objetos segmentados. Finalmente se llama a trabajar a la
%función que reconoce las piezas de la escena respecto a la base de
%conocimiento previamente almacenada y que contiene información de las
%clases correspondientes a los patrones.
M_Analisis=[];
f=Adq_imagen(imagen, ext, ri, rf, ci, cf); % Toma la imagen
fb=Pre_imagen(f); % Preprocesa la imagen
imshow(fb); % Muestra la imagen
[L, n]=bwlabel(fb); % Segmenta la imagen
[areas, ed, sd, ex, ec, eu, phi, cm, ang] = Desc_imagen(L, n, 1, rf-ri+1, 1, cf-ci+1); % Describe la imagen
Fv=CVv/(rf-ri+1); % factor de escala vertical
Fh=CVh/(cf-ci+1); %factor de escala horizontal
j=0;
for i=1:2:2*n-1
    j=j+1;
    cmx(j)=cm(i);
    cmy(j)=cm(i+1);
end;
HU1=phi(:, 1);
for i=1:n % Almacena en memoria descriptores de región
    M_Analisis=[M_Analisis; ed(i) sd(i) ex(i) ec(i) eu(i) HU1(i) cmx(i) cmy(i) ang(i)];
end;
M_Analisis(:, 7)=M_Analisis(:, 7)*Fh; % convierte centro de masa de valores de pixel a milímetros
M_Analisis(:, 8)=M_Analisis(:, 8)*Fv;
[Bp, nom_pat, npat]=lee_patrones('patrones'); % Lee base de datos de patrones
%Aplicamos una matriz de transformación al centro de masa referido respecto
%al sistema coordenado de la cámara con el propósito de referirlo ahora
%respecto a un sistema coordenado derecho ubicado en el mismo origen y con los ejes x y invertidos y
%con eje z saliendo de la mesa de trabajo
A=180;
%Giro (grados) del sistema base respecto al eje z
C=90;
A=(A*pi)/180;
C=(C*pi)/180;
Rx=[1 0 0 0; 0 cos(A) -sin(A) 0; 0 sin(A) cos(A) 0; 0 0 0 1];
Rz=[cos(C) -sin(C) 0 0; sin(C) cos(C) 0 0; 0 0 1 0; 0 0 0 1];
```

```

Transf=Rz*Rx;
for i=1:n % Para cada pieza en la escena
    desc=M_Analisis(i, 1:6); % copiar los descriptores, centro de masa y ángulo de
    inclinación
    piezas(i).cm=M_Analisis(i, 7:8);
    piezas(i).ang=M_Analisis(i, 9);
    fc=false(rf-ri+1, cf-ci+1); % crear una imagen binaria con fondo negro del tamaño de la región
de interés
    [r, c]=find(L==i); % obtiene las coordenadas (renglón, columna) de los puntos que
    pertenecen al objeto (i)
    a=[r, c];
    ls=length(a);
    hold on;
    for k=1:ls % asigna (1) al elemento (renglón, columna) de la imagen binaria que
    corresponde
        fc(a(k,1), a(k,2))=1; % a los índices almacenados en el arreglo (a)
    end;
    hold off;
    theta=(piezas(i).ang*pi)/180; % ángulo en radianes que el eje mas alargado de la pieza forma con la
horizontal
    T=[cos(theta) sin(theta) 0; -sin(theta) cos(theta) 0; 0 0 1];
    tform=maketform('affine', T); % obtiene transformación afín de rotación
    fd=imtransform(fc, tform); % aplica transformación afín que regresa la pieza a su posición
horizontal
    B=bwlabel(fd); % segmenta la imagen transformada
    E=regionprops(B,'Image'); % obtiene propiedades de región
    fe=E.Image;
    fg=false(rf-ri+1, cf-ci+1); % crea e inicializa dos imágenes binarias
    fg2=false(rf-ri+1, cf-ci+1);
    H=bwlabel(fe);
    [ren, col]=find(H==1); % aísla la imagen
    b=[ren, col];
    ls=length(b);
    hold on;
    for k=1:ls
        fg(b(k,1), b(k,2))=1; % almacena imagen en fg
    end;
    hold off;
    comp=0;
    for j=1:npat % busca en la base de conocimiento de patrones
        if (Bp(j,1)>desc(1)*0.95)&&(Bp(j,1)<desc(1)*1.05)&&(Bp(j,5)==desc(5))% si el tamaño es similar ( $\pm 5\%$ )
            nombre=sprintf('%s.bmp', nom_pat(j,:)); % y tiene el mismo número de Euler
            fp=imread(nombre); % leer la imagen del patrón
            H=bwlabel(fp);
            Hr=regionprops(H, 'BoundingBox', 'Image'); % se segmenta y se obtienen propiedades de región
            caja=[Hr.BoundingBox];
            [ren, col]=find(H==1); % se asigna a la imagen binaria inicializada fg1
            b=[ren, col];
            ls=length(b);
            fg1=false(rf-ri+1, cf-ci+1);
            hold on;
            for k=1:ls
                fg1(b(k,1), b(k,2))=1;
            end;
            hold off;
            ed1=M_Analisis(i, 1);
            comp=0;
            sentido=0;
            [comp, sentido]=Compara_imagenes2(fg, fg1, fg2, ed1, caja);
            % se compara con el objeto a reconocer con el patrón
        end
    end
end

```

```

    if sentido == 2
        piezas(i).ang=piezas(i).ang+180;           % si el sentido =2, la pieza debe estar girada 180°
    end;
    if comp == 1
        piezas(i).nombre=nom_pat(j,:);           % si comparó exitosamente con el patrón
        piezas(i).snt=sentido;
        piezas(i).altura=Bp(j,7);               % se almacena también el sentido y su altura
        break;
    end;
end;                                           % si no comparó exitosamente, se busca otro patrón
end;
if comp == 0
    piezas(i).nombre='Desconocido';             % si no existe patrón que compare, se etiqueta como
end;                                           % "Desconocido"
x=piezas(i).cm(1);
y=piezas(i).cm(2);
z=0;
pto=[x; y; z; 1];
pto_t=Transf*pto;                             % coordenadas del centro de masa son referidas a un
piezas(i).cm(1)=pto_t(1);                     % sistema coordinado sobre la mesa de trabajo con
piezas(i).cm(2)=pto_t(2);                     % el mismo origen pero con ejes x,y invertidos
end;                                           % respecto a la imagen tomada sobre la escena.
npz=n;

```

function Aprendizaje_gui3(nombre_patron, base_nombre, sec_ini, sec_inc, sec_fin, ext, ri, rf, ci, cf)

```

%APRENDIZAJE Es la fase en que el sistema puede dar de alta la información
%correspondiente a las clases que integran la base de conocimiento de los
%patrones. Para cada imagen se aplican las fases en las que se divide el análisis
%de una escena y que corresponden a :
%Adquisición de imagen, Preproceso de imagen, Segmentación de la escena,
%y Descripción, esta última fase es donde se obtienen los descriptores
%seleccionados para el presente trabajo y de los cuales se obtiene el valor
%medio de cada uno de ellos almacenándolos en la base de conocimiento de
%clases, para su posterior uso en tareas de reconocimiento de piezas.
filename=sprintf('%s', base_nombre);
f=Adq_imagen(filename, ext, ri, rf, ci, cf);   % Toma de imagen
fb=Pre_imagen(f);                             % Preproceso de imagen
figure, imshow(fb);                           % Muestra la imagen binarizada
[L, n]=bwlabel(fb);                           % Segmenta la escena y describe los objetos
segmentados
[areas, ed, sd, ex, ec, eu, phi, cm, ang] = Desc_imagen(L, n, 1, rf-ri+1, 1, cf-ci+1);
for i=1:n
    fc=false(rf-ri+1, cf-ci+1);               % Inicializa una imagen binaria
    [r, c]=find(L==i);                       % Obtiene relación de valores ( renglón,columna) de
píxeles
    a=[r, c];                                 % que forman parte de la pieza (i)
    ls=length(a);
    hold on;
    for k=1:ls
        fc(a(k,1), a(k,2))=1;               % Obtiene imagen binaria de la pieza aislada (i)
    end;
    hold off;
    theta=(ang(i)*pi)/180;                   % Obtiene el ángulo de inclinación de la pieza
    T=[cos(theta) sin(theta) 0; -sin(theta) cos(theta) 0; 0 0 1];
    tform=maketform('affine', T);           % Obtiene y aplica transformación afin que regresa la pieza
    fc=imtransform(fc, tform);               % a su posición horizontal (0°)
end;

```

```

B=bwlabel(fc); % segmenta imagen
E=regionprops(B,'Image'); % obtiene la imagen delimitada por un rectángulo mínimo
fd=E.Image;
figure, imshow(fd); % muestra la imagen
nombre_patron=input('¿Nombre patrón ? ', 's'); % preguntamos por el nombre del patrón
% Preguntamos por la altura de la pieza y la almacenamos junto a sus
% descriptores y al nombre del patrón
altura_pieza=input('¿Altura de la pieza (mm) ? ');
if altura_pieza>0
    fi=fopen('patrones.txt', 'a+');
    fprintf(fi, '%s', nombre_patron);
    fprintf(fi, '%10.2f %10.4f %10.6f %10.6f %10.6f %10.6f %10.6f\r\n', ed(i), sd(i), ex(i), ec(i), eu(i),
phi(i, 1), altura_pieza);
    nombre=sprintf('%s.bmp', nombre_patron);
    imwrite(fd, nombre);
    st=fclose('all');
end;
end;

```

function [cmp, sentido]=Compara_imagenes2(original, patron, patron_inv, ed, caja)

```

%COMPARA_IMAGENES.- Función que recibe como datos de entrada la imagen a
%comparar y la imagen patrón, los valores que retorna son el resultado de
%la comparación (1 imágenes coincidentes, 0 no coincidentes) y el sentido
%de la imagen a comparar respecto al patrón (1 coincide con el patrón, 2
%sentido opuesto al que presenta el patrón).
fr=imabsdiff(original, patron); % obtiene la diferencia absoluta entre las imágenes original y patrón
fr=bwmorph(fr, 'thin', 4); % aplica adelgazamiento en 4 pasadas a la imagen resultante del
proceso anterior
fr=bwlabel(fr); % convierte a matriz de etiquetas la imagen resultante del paso
anterior
fr_pr=regionprops(fr, 'Area'); % obtiene el área
w=[fr_pr.Area];
puntos1=sum(w); % suma los puntos de la imagen diferencia
pts_max_perm=0.1*(0.25*pi*ed^2); % obtiene el valor máximo de puntos para una imagen diferencia para ser
considerada como posible pieza reconocida. Umbral establecido como el 10% del área del patrón
% Se invierte la imagen del patrón vertical y horizontal simulando giro de 180° para considerar la otra
posibilidad.
piv=patron(caja(4):-1:1, 1:caja(3));
pih=piv(1:caja(4), caja(3):-1:1);
[pat, n1]=bwlabel(pih);
[ren, col]=find(pat==n1);
b=[ren, col];
ls=length(b);
hold on;
for k=1:ls
    patron_inv(b(k,1), b(k,2))=1;
end;
fr=imabsdiff(original, patron_inv);
fr=bwmorph(fr, 'thin', 2);
fr=bwlabel(fr);
fr_pr=regionprops(fr, 'Area');
w=[fr_pr.Area];

```

```

puntos2=sum(w);          % obtiene la suma de puntos de la imagen diferencia del patrón y la imagen
girada 180 grados
if (puntos1<pts_max_perm)&&(puntos1<puntos2)
    cmp=1;
    sentido=1;          % compara exitosamente y su sentido coincide con el del patrón
elseif (puntos2<pts_max_perm)&&(puntos2>puntos1)
    cmp=1;
    sentido=2;          % compara exitosamente y su sentido es opuesto al del patrón
elseif (puntos1>pts_max_perm)&&(puntos2>pts_max_perm)
    cmp=0;
    sentido=0;          % la pieza no coincide con el patrón
end;

```

function Crea_plantilla_para_ensamble_gui3(nombre_plantilla, imagen, ext, ri, rf, ci, cf, CVv, CVh)

```

%CREA_PLANTILLA_PARA_ENSAMBLE Genera el archivo de texto con información de
%las piezas que participan en el proceso de ensamble, la posición y
%orientación de cada una de ellas respecto al sistema coordenado 2D de la
%mesa de trabajo, cuyo origen se localiza en la esquina superior izquierda
%de la misma. Además posee información de la prioridad que tiene la pieza
%en el proceso mismo.
nombre_archivo=sprintf('%s.txt', nombre_plantilla);
[piezas, npz]=Analiza_escena3(imagen, ext, ri, rf, ci, cf, CVv, CVh);
piezas(1).prd=1;
for i=1:npz
    if ~isequal(piezas(i).nombre, 'Desconocido')
        fprintf('La pieza: %s, ', piezas(i).nombre);
        piezas(i).prd=input('Será ensamblada en el orden ? ');
    else piezas(i).prd=0;
    end;
end;
%Ordenamos respecto al valor de prioridad de forma ascendente.
for i=1:npz-1
    for j=i+1:npz
        if piezas(i).prd > piezas(j).prd
            temp=piezas(i);
            piezas(i)=piezas(j);
            piezas(j)=temp;
        end;
    end;
end;
%Almacenamos en disco la información de la plantilla
fi=fopen(nombre_archivo, 'a');
for i=1:npz
    fprintf(fi, '%s', piezas(i).nombre);
    fprintf(fi, '%10.2f %10.2f %10.2f %2d\n', piezas(i).cm(1), piezas(i).cm(2), piezas(i).ang, piezas(i).prd);
end;
st=fopen('all');

```

Function Crea_proceso ()

```
%Solicita el nombre de la plantilla
nombre_plantilla=input('Nombre de la plantilla ? ', 's');
% Coordenadas x,y,z del origen del sistema plantilla de ensamble, referido respecto al
% sistema de la mesa de trabajo.
dxp=input('Coordenada x del origen del sistema plantilla respecto al sistema de la mesa de trabajo ? ');
dyp=input('Coordenada y del origen del sistema plantilla respecto al sistema de la mesa de trabajo ? ');
dzp=input('Coordenada z del origen del sistema plantilla respecto al sistema de la mesa de trabajo? ');
%Con esto, trasladamos el origen de la mesa de trabajo al origen del
% sistema coordinado de la plantilla en su posición final de ensamblado.
%A continuación rotamos respecto al eje z del sistema coordinado de la mesa de trabajo para alinearlos
con
%el sistema coordinado de la plantilla en su posición final.
Cpg=input('Giro (en grados)del sistema mesa de trabajo respecto a su eje z, para alinearse al sistema de
plantilla ? ');
%Almacenamos en disco la información de la plantilla con su ubicación final
%para su ensamble
fi=fopen('proceso.txt', 'w');
fprintf(fi, '%s', nombre_plantilla);
fprintf(fi, '%10.2f %10.2f %10.2f %10.2f\n', dxp, dyp, dzp, Cpg);
st=fclose('all');
```

function [areas, ed, sd, ex, ec, eu, phi, cm, ang] = Desc_imagen(L, n, ri, rf, ci, cf)

```
%DESC_IMAGEN Obtiene los cuatro descriptores que utilizaremos además del
%centro de masa y orientación de cada pieza en la escena.
%La siguiente instrucción obtiene los descriptores listados en base a la
%matriz de etiquetas (L) obtenida en la fase de segmentación, el resultado
%lo asigna a la variable de estructura (D).
D=regionprops(L, 'Area', 'Centroid', 'EquivDiameter', 'Solidity', 'Orientation', 'Extent', 'Eccentricity', 'Euler');
%Las areas de las (n) regiones en píxeles las obtenemos mediante:
areas=[D.Area];
%El descriptor Diametro Equivalente (Diametro en píxeles de un círculo cuya area es igual a la de la
región
%en estudio), lo obtenemos mediante:
ed=[D.EquivDiameter];
%El descriptor Solidity, Escalar calculada como la razón de el área de la
%región al área de la concha convexa, se obtiene por:
sd=[D.Solidity];
%El descriptor Extent, Escalar calculada como la razón de el área de la
%región de interes al área de la caja envolvente , se obtiene por:
ex=[D.Extent];
%El descriptor Eccentricity, Escalar calculada como la razón de la
%distancia focal a la longitud del eje mayor de la elipse equivalente que
%tiene los mismos momentos de segundo orden que la región de interes, se
%determina mediante:
ec=[D.Eccentricity];
%El descriptor EulerNumber, Escalar igual al número de objetos en la región
%menos el número de huecos, se obtiene por:
eu=[D.EulerNumber];
%El centro de masa de cada región se obtiene mediante:
cm=[D.Centroid];
%El ángulo de inclinación del eje de mayor longitud de la pieza respecto al
%eje horizontal en el sistema coordinado de la mesa de trabajo, en grados
```



```

CVh=parametros(6);
dx=parametros(7);
dy=parametros(8);
dz=parametros(9);
A=parametros(10);
B=parametros(11);
C=parametros(12);
dxi=parametros(13);
dyi=parametros(14);
dzi=parametros(15);
Cig=parametros(16);
st=fclose(fi);
%Lee los datos de plantilla a ensamblar: nombre y ubicación final.
fi=fopen('proceso.txt', 'r');
nombre_plantilla=fscanf(fi, '%s', 1);
ubicacion=fscanf(fi, '%f %f %f %f', 4);
st=fclose('all');
dxp=ubicacion(1);
dyp=ubicacion(2);
dzp=ubicacion(3);
Cpg=ubicacion(4);
Ejecuta_ensamble_cmf_gui3(nombre_plantilla, '\fotos\escena001', '.bmp', ri, rf, ci, cf, CVv, CVh, dx,
dy, dz, A, B, C, dxi, dyi, dzi, Cig, dxp, dyp, dzp, Cpg)

```

function Ejecuta_ensamble_cmf_gui3(nombre_archivo, imagen, ext, ri, rf, ci, cf, CVv, CVh, dx, dy, dz, A, B, C, dxi, dyi, dzi, Cig, dxp, dyp, dzp, Cpg)

```

%EJECUTA_ENSAMBLE Función que ensambla piezas que se localizan en la mesa de trabajo,
%para ello, toma la imagen de la escena en formato especificado por la extensión,
%analiza la escena para determinar que piezas aparecen y en dónde se localiza.
%En el proceso de ensamble programado para ejecutarse se especifican las piezas
%que intervienen en el proceso, la posición final respecto al origen del sistema
%coordinado de la mesa de trabajo y el orden o prioridad en la ejecución del proceso de
%ensamble. Previo al inicio de la operación, debe especificarse los
%componentes de giro y traslación que debe aplicarse al sistema coordinado
%de la mesa de trabajo para definir la posición final de la plantilla de
%ensamble. Esto es debido a que la información de posición y orientación de
%las piezas en el archivo de la plantilla es sólo respecto al sistema
%coordinado local a la mesa de trabajo (origen en la esquina superior
%izquierda de la charola rectangular que funciona como mesa de trabajo).
h_efector_final=0;
%Lee el archivo que contiene información de plantilla
[pzs_pl, npl]=lee_plantilla(nombre_archivo);
% Coordenadas x,y,z del origen del sistema mesa de trabajo, referido al
% sistema base del robot.
Tr=[1 0 0 dx; 0 1 0 dy; 0 0 1 dz; 0 0 0 1];
%De esta manera trasladamos el sistema base del robot al origen del
%sistema coordinado de la mesa de trabajo.
%A continuación rotamos el sistema base para alinearlo con el sistema de la
%mesa de trabajo.
%A.-Giro (grados) del sistema base respecto al eje x
%B.-Giro (grados) del sistema base respecto al eje y
%C.-Giro (grados) del sistema base respecto al eje z
A=(A*pi)/180;
B=(B*pi)/180;

```

```

C=(C*pi)/180;
Rx=[1 0 0 0; 0 cos(A) -sin(A) 0; 0 sin(A) cos(A) 0; 0 0 0 1];
Ry=[cos(B) 0 sin(B) 0; 0 1 0 0; -sin(B) 0 cos(B) 0; 0 0 0 1];
Rz=[cos(C) -sin(C) 0 0; sin(C) cos(C) 0 0; 0 0 1 0; 0 0 0 1];
T=((Tr*Rx)*Ry)*Rz;
% Para referir coordenadas de la escena en la mesa de trabajo tomadas por la cámara
% respecto al sistema coordenado de la mesa de trabajo, trasladamos el
% origen del sistema coordenado de la mesa de trabajo al origen del sistema
% coordenado de la imagen tomada por la cámara.
Tri=[1 0 0 dxi; 0 1 0 dyi; 0 0 1 dzi; 0 0 0 1];
% Giramos el sistema coordenado de la mesa de trabajo ya trasladada
% respecto a su eje z un ángulo de -90° para alinearlo con el sistema coordenado de la escena
% tomada por la cámara.
Ci=(Cig*pi)/180;
Rzi=[cos(Ci) -sin(Ci) 0 0; sin(Ci) cos(Ci) 0 0; 0 0 1 0; 0 0 0 1];
Ti=Tri*Rzi;
%Para referir coordenadas del sistema de plantilla respecto al sistema
%coordenado de la mesa de trabajo:
Trp=[1 0 0 dxp; 0 1 0 dyp; 0 0 1 dzp; 0 0 0 1];
%Con esto, trasladamos el origen de la mesa de trabajo al origen del
%sistema coordenado de la plantilla en su posición final de ensamblado.
%A continuación rotamos respecto al eje z del sistema coordenado de la mesa de trabajo para alinearlo
con
%el sistema coordenado de la plantilla en su posición final.
Cp=(Cpg*pi)/180;
Rzp=[cos(Cp) -sin(Cp) 0 0; sin(Cp) cos(Cp) 0 0; 0 0 1 0; 0 0 0 1];
Tp=Trp*Rzp;
%Analizamos la escena para obtener las piezas que la integran, así como su
%posición y orientación respecto a la mesa de trabajo.
[piezas, npz]=Analiza_escena3(imagen, ext, ri, rf, ci, cf, CVv, CVh);
% Establece como "Disponibles" las piezas localizadas en la escena.
for i=1:npz
    piezas(i).disp='d';
end;
%Lee información del proceso anterior, nombre y cantidad de piezas.
fi=fopen('st_proceso.txt', 'r');
s='a';
ind=0;
while ~isempty(s)
    s=fscanf(fi, '%s', 1);
    posicion=fscanf(fi, '%d ', 1);
    status=fscanf(fi, '%c\n\r', 1);
    if isempty(s)
        continue;
    else
        ind=ind+1;
    end;
end;
st=fclose('all');
% Inicializamos el arreglo en memoria pzs_pl
fi=fopen('st_proceso.txt', 'r');
num_pend=0;
for i=1:npl
    s=fscanf(fi, '%s', 1);
    posicion=fscanf(fi, '%d ', 1);
    status=fscanf(fi, '%c\n\r', 1);
    if isequal(nombre_archivo, s) && (ind==npl)
        pzs_pl(i).st=status;
    else
        pzs_pl(i).st='P';
    end;
end;

```

```

end;
if pzs_pl(i).st=='P'
    num_pend=num_pend+1;
end;
end;
st=fclose('all');
if num_pend==0
    for i=1:npl
        pzs_pl(i).st='P'
    end;
end;
%Recorreremos la plantilla de ensamble y en cada pieza buscamos en la escena
%dicha pieza siempre y cuando esté aún disponible, que no sea una pieza
%desconocida y que el status de la pieza indique que está pendiente de
%ensamblarse.
k=0;
for i=1:npl
    for j=1:npz
        if isequal(pzs_pl(i).nombre, piezas(j).nombre) && isequal(piezas(j).disp, 'd') && isequal(pzs_pl(i).st, 'P')
            x2=piezas(j).cm(1);
            y2=piezas(j).cm(2);
            % La coordenada z del efector final se corrige con la altura de
            % de la superficie de trabajo al origen del sistema coordenado
            % del efector final (altura vertical) más la altura de la pieza
            % por manipular.
            h_pieza=piezas(j).altura;
            z2=h_efector_final+h_pieza-5;
            p2=[x2; y2; z2; 1];
            p1=Ti*p2;
            p0=T*p1;
            %Ahora vamos a calcular otro punto(p12)en base al centro de
            %masa(p1), con la característica que se localiza en el eje
            %central y principal de mayor longitud (mínimo momento de
            %inerencia) de la pieza , a una distancia en sentido horizontal de
            %10 mm respecto al punto p1, de tal manera que la distancia
            %dirigida p1 a p12 indica el sentido correcto de la pieza sobre
            %la mesa de trabajo.
            %Caso 1) Cuando el ángulo de inclinación de la pieza está entre
            %90° y 270° respecto a una paralela al eje x en el sistema coordenado de la mesa de trabajo
            %y que pasa por el c.m. de la pieza.
            ang_inc=piezas(j).ang;
            if ((ang_inc > 90) && (ang_inc < 270))
                x3 = p1(1)-10;
            else
                x3 = p1(1)+10;
            end;
            end;
            x4=x3;
            y4=p1(2);
            m=tan(piezas(j).ang*pi/180);
            y3=m*(x3-p1(1))+p1(2);
            %Se calcula el ángulo más pequeño entre los segmentos de recta
            %p1p4 al p1p3
            m1=(y4-p1(2))/(x4-p1(1)); %pendiente de la recta p1p4
            m2=(y3-p1(2))/(x3-p1(1)); %pendiente de la recta p1p3
            ang_fi=abs(atan((m2-m1)/(1+m1*m2)));

            %Caso 1a. Cuando el ángulo de inclinación se encuentra entre
            %90° y 180°.
            if (ang_inc>90) && (ang_inc<180)
                ang_giro_ef=ang_fi*180/pi; %Gira el efector final a la derecha el ángulo en grados.
            end;
        end;
    end;
end;

```

```

end;
if ((ang_inc>180)&&(ang_inc<270))|((ang_inc<-90)&&(ang_inc>-180))
    %Caso 1b. Cuando el ángulo de inclinación se encuentra entre
    %180° y 270°
    ang_giro_ef=-(ang_fi*180/pi); %cambio de sentido (gira el efector final a la izquierda)
end;
if (ang_inc == 90)
    %Caso 1c. Si el ang_inc es exactamente 90°
    ang_giro_ef=90; % El efector final gira a la derecha 90°.
end;
if ang_inc==180
    %Caso 1d. Si el ang_inc es exactamente 180
    ang_giro_ef=0; % El efector final no gira.
end;
%Caso 1e. Si el ang_inc es exactamente 270°
if ang_inc == 270
    ang_giro_ef=-90; % El efector final gira a la izquierda 90°.
end;
%Caso 2a. El ángulo de inclinación está entre 0° y 90°
if (ang_inc>0)&&(ang_inc<90)
    ang_giro_ef=180-(ang_fi*180/pi); % Giro a la derecha el ángulo suplementario a fi
end;
%Caso 2b. El ángulo de inclinación está entre 270° y 360°
if((ang_inc>270)&&(ang_inc<360))|((ang_inc<0)&&(ang_inc>-90))
    ang_giro_ef=-(180-(ang_fi*180/pi)); % Giro a la izquierda el ángulo suplementario a fi
end;
%Caso 2c. El ángulo de inclinación es exactamente 0°
if(ang_inc==0)|((ang_inc==360))
    ang_giro_ef=180; % Podría ser también -180°, es indistinto.
end;
piezas(j).disp='u'; % Ponemos el estado de la pieza en Utilizado.
% Ya podemos almacenar las coordenadas del centro de masa de la
% pieza en la escena, así como el ángulo y sentido de giro del
% efector final para que el robot pueda acceder a tomar la
% pieza que se localiza en la escena, es decir, almacenaremos
% los valores fuente donde se realizará la operación de tomar
% pieza (operación de "pick")
% Ajustes de operación.
ang_giro_cmp=180; % Angulo de giro B,(Giro respecto al eje x local al efector final "guiñada")
k=k+1;
puntos(k).x=p0(1);
puntos(k).y=p0(2);
puntos(k).z=p0(3);
puntos(k).A=(ang_giro_ef*pi)/180;
puntos(k).B=(ang_giro_cmp*pi)/180;
fi=fopen('puntos.txt', 'a');
fprintf(fi, '%10.2f %10.2f %10.2f %10.2f %10.2f\n', p0(1), p0(2), p0(3), (ang_giro_ef*pi)/180,
(ang_giro_cmp*pi)/180);
st=fopen('all');
% PARA LA POSICION DE DESTINO DE LA PIEZA, ES DECIR, SOBRE LA
% PLANTILLA DE ENSAMBLADO TENEMOS:
x1p=pzs_pl(i).param(1);
y1p=pzs_pl(i).param(2);
% La coordenada z del efector final se corrige con la altura de
% de la superficie de plantilla al origen del sistema coordenado
% del efector final (altura vertical) más la altura de la pieza
% por manipular.
z1p=h_efector_final+h_pieza-5;
p1p=[x1p; y1p; z1p; 1];
% Referimos el punto sobre la plantilla respecto al sistema

```

```

% de la mesa de trabajo
p2p=Tp*p1p;
ang_inc=pzs_pl(i).param(3)+90;
if ((ang_inc > 90) && (ang_inc < 270))
    x3 = p2p(1)-10;
else
    x3 = p2p(1)+10;
end;
x4=x3;
y4=p2p(2);
m=tan(ang_inc*pi/180);
y3=m*(x3-p2p(1))+p2p(2);
%Se calcula el ángulo más pequeño entre los segmentos de recta
%p2p4 al p2p3
m1=(y4-p2p(2))/(x4-p2p(1)); %pendiente de la recta p2p4
m2=(y3-p2p(2))/(x3-p2p(1)); %pendiente de la recta p2p3
ang_fi=abs(atan((m2-m1)/(1+m1*m2)));

%Caso 1a. Cuando el ángulo de inclinación se encuentra entre
%90° y 180°.
if (ang_inc>90) && (ang_inc<180)
    ang_giro_ef=ang_fi*180/pi; %Gira el efector final a la derecha el ángulo en grados.
end;
if ((ang_inc>180)&&(ang_inc<270))|((ang_inc<-90)&&(ang_inc>-180))
    %Caso 1b. Cuando el ángulo de inclinación se encuentra entre
    %180° y 270°
    ang_giro_ef=-(ang_fi*180/pi); %cambio de sentido (gira el efector final a la izquierda)
end;
if (ang_inc == 90)
    %Caso 1c. Si el ang_inc es exactamente 90°
    ang_giro_ef=90; % El efector final gira a la derecha 90°.
end;
if ang_inc==180
    %Caso 1d. Si el ang_inc es exactamente 180
    ang_giro_ef=0; % El efector final no gira.
end;
%Caso 1e. Si el ang_inc es exactamente 270°
if ang_inc == 270
    ang_giro_ef=-90; % El efector final gira a la izquierda 90°.
end;
%Caso 2a. El ángulo de inclinación está entre 0° y 90°
if (ang_inc>0)&&(ang_inc<90)
    ang_giro_ef=180-(ang_fi*180/pi); % Giro a la derecha el ángulo suplementario a fi
end;
%Caso 2b. El ángulo de inclinación está entre 270° y 360°
if((ang_inc>270)&&(ang_inc<360))|((ang_inc<0)&&(ang_inc>-90))
    ang_giro_ef=-(180-(ang_fi*180/pi)); % Giro a la izquierda el ángulo suplementario a fi
end;
%Caso 2c. El ángulo de inclinación es exactamente 0°
if(ang_inc==0)|(ang_inc==360)
    ang_giro_ef=180; % Podría ser también -180°, es indistinto.
end;
% Aplicamos la transformación para referir finalmente respecto
% al sistema base del robot
p2b=T*p2p;
% Sin embargo, debemos corregir el ángulo (ang_giro_ef) con el
% ángulo girado por la mesa de trabajo al momento de alinearse
% con la plantilla en su ubicación final.
ang_giro_ef=ang_giro_ef - Cpg;
if (ang_giro_ef>180)

```

```

        ang_giro_ef=-(360 - ang_giro_ef);
    end;
    if (ang_giro_ef<(-180))
        ang_giro_ef=360 + ang_giro_ef;
    end;
    k=k+1;
    puntos(k).x=p2b(1);
    puntos(k).y=p2b(2);
    puntos(k).z=p2b(3);
    puntos(k).A=(ang_giro_ef*pi)/180;
    puntos(k).B=(ang_giro_cmp*pi)/180;
    fi=fopen('puntos.txt', 'a');
    fprintf(fi, '%10.2f %10.2f %10.2f %10.2f %10.2f\n', p2b(1), p2b(2), p2b(3), (ang_giro_ef*pi)/180,
(ang_giro_cmp*pi)/180);
    st=fclose('all');
    % Asignamos status de "T" (terminado) a la pieza ensamblada de
    % la plantilla actual
    pzs_pl(i).st='T';
    break;
end
end;
end;
% Guardar información del status del proceso de ensamble
fi=fopen('st_proceso.txt', 'w');
for i = 1:ind
    fprintf(fi, '%s', nombre_archivo);
    fprintf(fi, '%2d %c\r\n', pzs_pl(i).prd, pzs_pl(i).st);
end;
st=fclose('all');
% SE ESTABLECE COMUNICACION CON EL CONTROL DEL ROBOT
fclose('all');
s=serial('com1', 'BaudRate', 9600, 'DataBits', 8, 'Parity', 'Even', 'StopBits', 2);
fopen(s);
resp(1)='L';
for ciclo=1:2:k-1
%ENVÍO DE POSICIÓN DE "TOMAR" AL ROBOT MEDIANTE PUERTO SERIAL
while resp(1)=='L'
    resp(1)='N';
    valor=puntos(ciclo).x;
    cadena=num2str(valor);
    paq = ['PRN ', cadena, char(13)];
    fwrite(s, paq);
    valor=puntos(ciclo).y;
    cadena=num2str(valor);
    paq = ['PRN ', cadena, char(13)];
    fwrite(s, paq);
    valor=puntos(ciclo).z;
    cadena=num2str(valor);
    paq = ['PRN ', cadena, char(13)];
    fwrite(s, paq);
    valor=puntos(ciclo).A;
    cadena=num2str(valor);
    paq = ['PRN ', cadena, char(13)];
    fwrite(s, paq);
    valor=puntos(ciclo).B;
    cadena=num2str(valor);
    paq = ['PRN ', cadena, char(13)];
    fwrite(s, paq);
end;
%ENVÍO DE POSICIÓN DE "DEJAR" AL ROBOT MEDIANTE PUERTO SERIAL

```

```

while resp(1)~='L'
    resp=char(fread(s, 2, 'char'));
end;
while resp(1)=='L'
    resp(1)='N';
    valor=puntos(ciclo+1).x;
    cadena=num2str(valor);
    paq = ['PRN ', cadena, char(13)];
    fwrite(s, paq);
    valor=puntos(ciclo+1).y;
    cadena=num2str(valor);
    paq = ['PRN ', cadena, char(13)];
    fwrite(s, paq);
    valor=puntos(ciclo+1).z;
    cadena=num2str(valor);
    paq = ['PRN ', cadena, char(13)];
    fwrite(s, paq);
    valor=puntos(ciclo+1).A;
    cadena=num2str(valor);
    paq = ['PRN ', cadena, char(13)];
    fwrite(s, paq);
    valor=puntos(ciclo+1).B;
    cadena=num2str(valor);
    paq = ['PRN ', cadena, char(13)];
    fwrite(s, paq);
end;
while resp(1)~='L'
    resp=char(fread(s, 2, 'char'));
end;
end;
%Envío de caracter 'F' indicando el fin de la transmisión.
cadena='F';
paq = ['PRN ', cadena, char(13)];
fwrite(s, paq);
fclose(s);

```

function [Base_patrones, nombres_pat, npat]=lee_patrones(nom_archivo_pat)

```

%LEE_PATRONES Función que toma de un archivo de texto la información
%almacenada para cada una de las clases que integran la base de
%conocimiento de patrones a utilizar en la aplicación. Requiere como dato
%de entrada el nombre del archivo de texto sin extensión, y como valores de
%salida de la función se obtienen el número de patrones que forman la base
%de conocimiento y una matriz con los valores promedio de los descriptores
%que se utilizan en el proyecto y que son: Área en píxeles, Diámetro
%equivalente, Solidez, primer descriptor invariante de HU, número de Euler
% y la altura de la pieza. Otro dato a la salida es un vector con los nombres de cada uno de los patrones.
Base_patrones=[];
filename=sprintf('%s.txt', nom_archivo_pat);
fi=fopen(filename, 'r+');
s='a';
npat=0;
while ~isempty(s)
    s=fscanf(fi, '%s', 1);

```



```

desc=fscanf(fi, '%f %f %f %f %f %f %f\n\r', 7);
if npat == 0
    nombres_pat=s;
else
    nombres_pat=char(nombres_pat, s);
end;
Base_patrones=[Base_patrones; desc'];
npat=npat+1;
end;
st=fclose('all');
npat=npat-1;
for i = 1:npat-1
    for j = i+1:npat
        if (Base_patrones(j, 1)<Base_patrones(i,1))
            temp_val=Base_patrones(i,:);
            temp_nom=nombres_pat(i,:);
            Base_patrones(i,:)=Base_patrones(j,:);
            nombres_pat(i,:)=nombres_pat(j,:);
            Base_patrones(j,:)=temp_val;
            nombres_pat(j,:)=temp_nom;
        end;
    end;
end;
end;

```

function [pzs_pl, nppl]=lee_plantilla(nom_archivo_pl)

```

%LEE_PLANTILLA Función que toma de un archivo de texto la información
%almacenada para cada una de las plantillas de ensamble. Requiere como dato
%de entrada el nombre del archivo de texto sin extensión, y como valores de
%salida de la función se obtienen el número de piezas de la plantilla, los valores de las coordenadas x,y
%de los centros de masa de las piezas que participan en el proceso de ensamblado de la plantilla, los
ángulos
%que forman el eje de mayor elongación de cada pieza con eje horizontal del sistema coordenado de la
mesa de trabajo
%en grados, el valor de prioridad en la ejecución del proceso y el status para el proceso de ensamblado.
Otro dato a
%la salida es un vector con los nombres de cada uno de los patrones que
%corresponden a las piezas.
filename=sprintf('%s.txt', nom_archivo_pl);
fi=fopen(filename, 'r+');
s='a';
nppl=0;
while ~isempty(s)
    s=fscanf(fi, '%s', 1);
    if isempty(s)
        continue;
    end;
    param=fscanf(fi, '%f %f %f', 3);
    prd=fscanf(fi, '%d\n\r', 1);
    nppl=nppl+1;
    pzs_pl(nppl).nombre=s;
    pzs_pl(nppl).param=param';
    pzs_pl(nppl).prd=prd;
    pzs_pl(nppl).st='P';
end;

```

```
end;  
st=fclose('all');
```

function[fb] = Pre_imagen(f)

```
%PRE_IMAGEN Preprocesa la imagen (f) y obtiene como salida una imagen  
%binarizada (fb). Aplica un corrimiento en la distribución de su histograma  
%para considerar la serie de niveles de gris distribuidas entre el 35 y el  
%100 por ciento de su representación inicial. Aplica además filtro espacial  
%Top Hat para un suavizado y mejora de la imagen. Posteriormente aplica  
%binarización de la imagen mediante umbral, obteniendo así una imagen con  
%dos tonos, fondo blanco y objetos negros, finalmente invierte la imagen  
%binaria.
```

```
%Aplica el corrimiento  
fb=gscale(f, 'minmax', 0.35, 1.0);  
%Aplica el filtro TopHat  
se=strel('disk', 2);  
fb = imclose(imopen(fb, se), se);  
%Binarización de la imagen  
fb = im2bw(fb, graythresh(fb));  
%Obtiene el complemento de la imagen  
fb=imcomplement(fb);
```


Apéndice C

Código fuente del programa almacenado en el control del robot, programado en lenguaje MELFA-BASIC IV. Este es llamado por el PLC cuando la mesa de trabajo se detiene en la estación de manipulación y ensamble.

```
10 DEF CHAR DATO_X           % declara variables caracter
20 DEF CHAR DATO_Y
30 DEF CHAR DATO_Z
40 DEF CHAR DATO_A
50 DEF CHAR DATO_B
66 DEF IO LIBERA = BIT,7     % declara bit 7 del modulo I/O como LIBERA
70 MOV P1                    % mueve el robot a la posición inicio
80 OPEN "COM1:" AS #1       % abre el Puerto de comunicación RS-232
110 *INICIO
120 HOPEN 1                  % abre la pinza 1
130 GOSUB *POSICION         % trasfiere el control a la subrutina POSICION
140 GOSUB *PICK             % trasfiere el control a la subrutina TOMAR
180 GOSUB *POSICION         % trasfiere el control a la subrutina POSICION
190 GOSUB *PLACE           % trasfiere el control a la subrutina LIBERAR
230 GOSUB *INICIO          % trasfiere el control a la subrutina INICIO
250 *POSICION
255 OVRD 50                  % baja la velocidad al 50%
280 INPUT #1,DATO_X$        % lee por el puerto 1 la coordenada X ,
                             % si el dato es la letra F, termina el programa, si continua
282 IF DATO_X$="F" THEN GOTO *FINAL ELSE 290
290 M1=VAL(DATO_X$)         % convierte a valor numérico el dato leído
300 P2.X=M1                 % lo asigna a la variable de posición P2
330 INPUT #1,DATO_Y$        % de forma análoga para las coordenadas Y, Z, Alabeo y
                             % Guiñada
340 M2=VAL(DATO_Y$)
350 P2.Y=M2
380 INPUT #1,DATO_Z$
390 M3=VAL(DATO_Z$)
400 P2.Z=M3
430 INPUT #1,DATO_A$
440 M4=VAL(DATO_A$)
450 P2.A=M4
480 INPUT #1,DATO_B$
490 M5=VAL(DATO_B$)
500 P2.B=M5
520 RETURN
550 *PICK                    % Inicia subrutina TOMAR
560 MOV P1                  % mueve el robot a la posición inicio
562 DLY 0.5                 % retardo de 1/2 segundo
570 MOV P2,-9              % mueve el efector final 9 mm por encima de P2
571 OVRD 10                 % disminuye la velocidad al 10%
572 DLY 0.5                 % retardo de 1/2 segundo
580 MOV P2                  % mueve el efector final al punto P2
585 DLY 0.5                 % retardo de 1/2 segundo
590 HCLOSE 1                % cierra la pinza
592 DLY 0.5                 % retardo de 1/2 segundo
600 MOV P2,-9              % mueve el efector final 9 mm por encima de P2
607 PRINT #1,"L"          % transmite por el puerto RS-232 al programa
610 RETURN                  % Ejecuta_ensamble_en_linea( ) que ha recibido de
```

640 *PLACE	% manera satisfactoria los datos de posición TOMAR
660 MOV P1	% Inicia subrutina DEJAR
662 DLY 0.5	% de forma semejante ejecuta la rutina que libera la pieza
670 MOV P2,-9	
671 OVRD 10	
672 DLY 0.5	
680 MOV P2	
685 DLY 0.5	
690 HOPEN 1	
692 DLY 0.5	
700 MOV P2,-9	
707 PRINT #1,"L"	
710 RETURN	
800 *FINAL	% termina el proceso
810 MOV P1	% mueve el robot a la posición inicio
820 LIBERA=1	% inicializa el estado de la variable controlada por PLC
830 DLY 3	% retardo de 3 segundos
840 LIBERA=0	% cambia el estado de la variable controlada por PLC
850 GOTO *INICIO	% transfiere el control a INICIO para la siguiente pieza
900 END	% fin del programa

Bibliografía

- [1] Festo Didactic, *Manual de instrucciones de COSIROP para el lenguaje MELFA-BASIC IV.*, 2005.
- [2] Gómez-Andrade Armando Carlos, *Manual tutorial para brazo robótico Mitsubishi RV-2AJ*. CMA del ITSSLP,C., 2009.
- [3] Mompin P, *Sistemas CAD/CAM/CAE*, Mundo Electrónico Marcombo, Barcelona, 1988.
- [4] Vallejo R. Erick, *Sistemas flexibles de manufactura*. Ingeniería & Desarrollo. Universidad del Norte. 3-4: 43-49, 1998.
- [5] Chiang S. Luciano, *Diseño conceptual de productos mecatrónicos*, Universidad Católica de Chile, 2003.
- [6] H. Golnabi, A. Asadpour, *Design and application of industrial machine vision systems*, Robotics and computer-integrated manufacturing 630-637, 2007.
- [7] Nist, Integration definition for function Modeling (IDEF0), 1993, <http://www.idef.com>
- [8] Toledo-Ramírez Gengis K. *Investigación y desarrollo de sistemas de control mediante visión técnica para micromáquinas herramientas y micromanipuladores*. Tesis doctoral, Facultad de Ingeniería, UNAM, agosto 2007.
- [9] Cognex., *Manual para sistema de visión Legend 530 DVT Machine Vision*, 2002.
- [10] González, R.C.I, Woods, R.E. and Steven L.E., *Digital image processing using MatLab*, PEARSON Prentice, 2004.
- [11] Joerg Wolf Paul Robinson, *Mitsubishi RV-2AJ Industrial robot programming and calibration Lab Notes*, School of Computing, Communication and Electronics The University of Plymouth, 2001.
- [12] Maravall Gómez-Allende Darío, *Reconocimiento de formas y visión artificial*, Addison-Wesley Iberoamericana, 1988.
- [13] Schalkoff R.J, *Digital image processing and computer vision*, Wyley & Sons, 1989.
- [14] Fu K.S, Gonzalez R.C., Lee C.S.G. *Robótica: control, detección, visión e inteligencia*, Mc.Graw Hill, 1989.
- [15] N. Otsu, *A threshold selection method from gray level histograms*, IEEE Trans. on syst. man and cyber. Vol SMC-9(1), pp. 62-66, jan 1979.
- [16] I. Pitas, *Digital image processing algorithms and applications*, Wiley, 2000.
- [17] Horn, LISP *Examples involving arrays and binary images* Cap. 10, pp. 151-167, 1988.

- [18] Hu, M.K., *Visual patern recongition by moment invariant*, IEEE Transactions.inform theory, vol 8, pags. 179-187, 1962.
- [19] Haralick R.M. and Shapiro L.G, *Computer and robot vision*, Addison-Wesley, 1992.
- [20] Shapiro L.G. and Rosenfeld A, *Computer vision and image processing*. Academic-Press. Inc, 1992.
- [21] Gonzalez R.C. and Woods R.E, *Digital image processing*, Addison-Wesley, 1992.
- [22] Iglesias Arturo, *Implementación de un sistema de visión la detección, reconocimiento y manipulación de piezas de ensamble por medio de robot de uso industrial*. Tesis de maestría, CENIDET Cuernavaca, Mor., dic 1991.

Glosario

AGV.- Vehículo Guiado Automáticamente (del inglés: Automatic Guided Vehicles).

ASI.- Interfece de Sensores y Actuadores (del inglés: Actuators and Sensors Interface).

CCADET.- Centro de Ciencias Aplicadas y Desarrollo Tecnológico.

CMA.- Centro de Manufactura Avanzada.

CNC.- Control Numérico Computarizado (del inglés: Control Numeric Computerized)

COSIMIR.-Simulador de robots industrials orientado a celdas (del inglés: Cell Oriented Simulation of Industrial Robots)

DIPUM.- Uso de MatLab en el Procesamiento Digital de Imágenes (del inglés: Digital Image Processing Using Matlab).

GUI.- Interface Gráfica del Usuario (del inglés: Graphical User Interface)

IDEF0.- Definición de la Integración para la Modelización de las Funciones (del inglés: Integration Definition for Function Modeling).

IPT.- Librería de funciones para Procesamiento de Imágenes (del inglés: Image Processing Toolbox).

ITSSLP,C.- Instituto Tecnológico Superior de San Luis Potosí, Capital.

MatLab.-Laboratorio de Matemáticas (del inglés: Matrix Laboratory).

PLC.- Controlador Lógico Programable (del inglés: Programmable Logic Controler).

FMS.- Sistemas de Manufactura Flexibles (del inglés: Flexible Manufacturing Systems).

SEM.- Sistema Estación de Maquinado.

SPPM.- Sistema de Programación de Procesos de Manufactura.

SMP.- Sistema de Manipulación de Piezas.

SVA.- Sistema de Visión Artificial.

SCCAV.- Subsistema de Control de Calidad Asistido por Visión Artificial.

UNAM.- Universidad Nacional Autónoma de México.

