UNIVERSIDAD NACIONAL AUTONOMA DE MÉXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA

FACULTAD DE INGENIERÍA

INVESTIGACIÓN Y DESARROLLO DE SISTEMAS DE CONTROL MEDIANTE VISIÓN TÉCNICA PARA MICROMÁQUINAS HERRAMIENTAS Y MICROMANIPULADORES

TESIS

QUE PARA OPTAR POR EL GRADO DE:

DOCTOR EN INGENIERÍA

MECÁNICA - DISEÑO MECÁNICO

P R E S E N T A:

GENGIS KANHG TOLEDO RAMÍREZ*

TUTORES:
ERNST KUSSUL
TATIANA BAIDYK

Ciudad Universitaria Agosto 2007

*Exbecario CONACYT

Jurado asignado:

Presidente:	Dr. López Parra Marcelo.
Secretario:	Dr. Dorador González Jesús Manuel.
1^{er} Vocal:	Dr. Kussul Ernst Mikhailovich.
1^{er} Suplente:	Dr. Santillán Gutiérrez Saúl Daniel.
2^{o} Suplente:	Dra. Baidyk Tatiana.

Lugar donde se realizó la tesis:

Laboratorio de Micromecánica y Mecatrónica (LMM) del
Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET)
de la Universidad Nacional Autonoma de México (UNAM).

Tutores de tesis:

Dr. Ernst Kussul.	**Dra. Tatiana Baidyk.**

Este trabajo se desarrolló en el Laboratorio de Micromecánica y Mecatrónica del Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET) de la Universidad Nacional Autónoma de México (UNAM) bajo la tutoria de los doctores Ernst Kussul y Tatiana Baidyk. Se contó con el apoyo de una beca para estudios doctorales del CONACYT y de una beca complemento DGEP - UNAM. Además se contó con el apoyo parcial de los siguientes proyectos: DGAPA PAPIIT IN118799, PAPIIT IN108606-3, PAPIIT IN116306-3, NSF-CONACYT 39395-U, DGAPA-PAPIIT IN112102 y CONACYT 33944-U.

Redacción y edición de tesis
con LaTeX 2$_\varepsilon$, *GNU-Emacs*
y sistema operativo libre
GNU/LINUX.

A Susy y Conenetl;

mi familia

♡

Vivimos en una sociedad altamente dependiente de la ciencia y de la tecnología, en la cual casi nadie sabe nada sobre ciencia y tecnología.

Carl Sagan

Se ha vuelto espantosamente obvio que nuestra tecnología ha excedido a nuestra humanidad.

Albert Einstein

Las máquinas y las computadoras deberán volverse una parte funcional en un sistema social orientado por la vida y no un cáncer que empieza por hacer estragos y acaba por matar al sistema.

Erich Fromm

Agradecimientos

En primer lugar deseo agradecer profundamente a mis tutores, los doctores Tatiana Baydik y Ernst Kussul por la gran confianza que me brindaron desde el primer momento, para la propuesta, el desarrollo y la conclusión de este trabajo. Gracias por su apoyo, sus sabios y pertinentes consejos, las múltiples enseñanzas que me dieron durante mi formación así como la libertad que me permitieron tomar para la elaboración de este trabajo de investigación. Algo fundamental para mi formación como investigador han sido sus innumerables guías que me han permitido no sólo cumplir con las publicaciones y participaciones requeridas para alcanzar el grado de doctor, sino que me han compartido de corazón la gran experiencia que a lo largo de varias décadas han acumulado en su trabajo de investigadores en varios países y en múltiples congresos internacionales. Muchas gracias, ***Spasibo!***.

En cada una de las tres etapas de este trabajo hubo personas a las cuales deseo agradecer especialmente. Durante la realización de este trabajo: el apoyo fundamental de mi novia, primero y luego esposa Susana Torres Sánchez "Susy", sin el cuál este trabajo no hubiese sido tan agradable y llevadero; a lo largo de todo el tiempo has sido un pilar de apoyo, emocional y de amor para mi ¡Gracias Peque! ¡Te amo! Agradesco a mi hijo Conenetl "Pablo Ilhuícatl", por todos los momentos de alegría y satisfacción que me ha brindado. Sin ti, este documento no hubiese llevado tu ejemplo, el de algo bien hecho y agradable.

Para iniciar mis estudios doctorales ha sido determinante la persuasión y consejos de mi gran amigo Adolfo de Unanué Tiscareño, el ha sido el primero que descubrió la vocación en mi para hacer una carrera como investigador y en ese sentido me animó para realizar mi doctorado. ¡Gracias Adolfo!

En el tramo final de mi tesis sobresale mi tutora, la Dra. Tatiana, ya que sin su guía, apoyo, consejos y su interés porque finalizara en buen término el documento, esta tesis no hubiese resultado lo que es ahora y en el tiempo previsto.

No quiero dejar de agradecer a mis padres Germán Toledo Orozco y Rosa Marina Ramírez Arano de Toledo su apoyo y sacrificio que luego de lustros de cuidados y educación han rendido frutos en mi formación profesional previa y en mi formación como ser humano, mismos que me han permitido sin lugar a dudas realizar mi doctorado. Así mismo, agradezco a mi hermano Rommel Toledo Ramírez por todo el apoyo que me ha brindado durante este trabajo, desde lo personal, ayudándome a mudarme, hasta los apoyos académicos, de traducción y puntos de vista que sin lugar a dudas han hecho de este trabajo algo mejor, al igual que agradezco a mi hermano Germán. Agradezco también a toda mi familia tanto natural como política así como a mis amigos del SIV por su cariño, aprecio y amistad desinteresada. Agradesco al Médico Félix Guillermo Márquez Celedonio que indirectamente influyó sobre este trabajo en aspectos personales, profesionales e intelectuales.

Agradezco mucho al estudiante de doctorado Oleksandr Makeyev "Sasha" quien me apoyó bastante en la traducción y redacción de mis trabajos, en especial mi primer artículo internacional. Agradezco también el apoyo y colaboración de mi amigo y compañero Josué Enríquez Zárate así como también a mi amigo Jesús Daniel Pérez Castañeda por su amistad y por haberme invitado a dar una plática sobre este trabajo en su institución.

Durante mi visita a Ucrania donde expuse los primeros resultados de mi trabajo, recibí la atención

y la hospitalidad muy amena y enriquecedora de los Drs. Taras K. Vintsiuk, Alexander V. Goltsev y Dmitri A. Rachkovskij, así como también de Danielok que tanto nos hizo conocer de su patria en tan poco tiempo, además, deseo agradecer a mis compañeros congresistas: Borbála Katalin Benkö de Hungría y Tomasz Rusc de Polonia.

During my visit to Ukraine in which I exposed the first results from my work, I received the attention and the very pleasant and enriching hospitality of Dr. Taras K. Vintsiuk, Dr. Alexander V. Goltsev and Dr. Dmitri A. Rachkovskij, as well as from Danielok who allowed to us to know his mother country in a short time, in addition, I want to thanks to my conference partners: Borbála Katalin Benkö from Hungary and Tomasz Rusc from Poland.

Durante la mayor parte de mi trabajo de investigación y desarrollo utilicé software libre, por lo que doy gracias a toda la comunidad internacional de software libre cuyo trabajo utilicé con éxito y de quienes aprendí bastante tanto en lo técnico como en lo humano. Gracias a todos ellos y a los proyectos en que participan, me he convencido mediante los hechos durante mi trabajo que el software libre creado comunitariamente es superior al comercial en gran medida, en especial para el trabajo de investigación. *Thanks very much to the international "Free Software Community".*

Mis más profundas gracias a todos mis compañeros del Laboratorio de Micromecánica y Mecatrónica (LMM), tanto doctores como académicos, trabajadores y estudiantes, con los cuales compartí el laboratorio y muchas cosas más. Son muchos los compañeros del LMM, sin pretender dejar de recordar a alguien, deseo agradecer en especial al Dr. Leopoldo Ruíz, al Dr. Alberto Caballero, a la Dra. Graciela y al Dr. Ascanio, así como al Maestro David Palomino y al Sr. Mario; y por supuesto, a todos mis compañeros estudiantes del laboratorio: Héctor Silva, Angélica Zamora, Mauricio Algalán, Gerardo Carrera, Eugenio Marín, Guillermo Saavedra, Paulo López, Paul Domínguez, Oscar Cuevas, Anabel Martín, Bogar Patiño, Germán Herrera, Octavio Ortigoza, Daniela Rovira, Tere, Oscar y todos los demás con los que compartí el laboratorio.

Gracias a mis sinodales, miembros de mi jurado de examen de candidatura y de grado doctores Marcelo López Parra, Jesús Manuel Dorador González y Saúl Daniel Santillán Gutiérrez por su tiempo, paciencia y comentarios que mejoraron este trabajo.

Agradezco también al personal administrativo, académico y de la base trabajadora, tanto de la Facultad de Ingeniería (FI) como del CCADET de la UNAM, por su apoyo brindado para este trabajo que ha ido desde el préstamo de material, dotación de insumos y apoyo en los trámites hasta mi aceptación en el programa de posgrado de la FI así como el apoyo para acudir a un congreso internacional y muchas cosas más.

No menos importante, sino fundamental es para mi agradecer al CONACYT que brindó el apoyo económico para mis estudios doctorales así como a la DGEP-UNAM, que además de brindarme un apoyo complementario ha hecho posible el programa de doctorado que he cursado junto con la FI. También el PAPIIT - UNAM me apoyo en parte para mi trabajo. Muchas gracias a todas estas instituciones y programas así como aquellos que las fundaron.

Agradezco a los siguientes proyectos que apoyaron en parte mi trabajo: DGAPA PAPIIT IN118799, PAPIIT IN108606-3 (Investigación y desarrollo de manipuladores de baja inercia), PAPIIT IN116306-3 (Investigación y desarrollo de un sistema de visión computacional para microequipo, NSF-CONACYT 39395-U (Investigación de sistemas de control basados en redes neuronales con aplicaciones en sistemas micromecánicos), DGAPA-PAPIIT IN112102 (Investigación y desarrollo de sistemas de control para micromáquinas herramientas y micromanipuladores) y CONACYT 33944-U (Desarrollo de microtecnología mecánica para aplicaciones de células de producción).

Por último agradezco a todos aquellos idealistas y activistas mexicanos y extranjeros que a lo largo de la historia de México han permitido que hoy se pueda gozar de una educación científica, tecnológica, pública, laica, gratuita y de calidad.

A todas las personas y entidades mencionadas ¡Muchísimas gracias! ¡Gracias a todos ustedes he logrado alcanzar esta importante meta!.

Índice general

Resumen xix

Abstract xxi

1. Introducción 1
- 1.1. Objetivos generales . . . 1
- 1.2. Marco teórico . . . 2
  - 1.2.1. Justificación. . . . 2
  - 1.2.2. Metodología. . . . 2
- 1.3. Objetivo de la tesis . . . 3
- 1.4. Contribución de este trabajo . . . 3
- 1.5. Organización del trabajo . . . 4

2. Antecedentes y estado del arte 5
- 2.1. Microtecnología . . . 5
  - 2.1.1. Aplicaciones de la microtecnología . . . 5
  - 2.1.2. Sistemas Micro Electromecánicos (MEMS) . . . 6
  - 2.1.3. Microfábricas . . . 6
  - 2.1.4. Tecnología de microequipo (MET) . . . 7
  - 2.1.5. Micromaquinado y microensamble . . . 7
  - 2.1.6. Reconocimiento de piezas en una microfábrica . . . 8
- 2.2. Visión por computadora . . . 8
  - 2.2.1. Visión por computadora en microfábricas . . . 8
  - 2.2.2. Reconocimiento de objetos . . . 9
  - 2.2.3. Estado del arte . . . 9
    - 2.2.3.1. Preprocesamiento de imágenes . . . 9
    - 2.2.3.2. Método de semejanza de patrones . . . 10
    - 2.2.3.3. Método de análisis de componentes principales . . . 10
    - 2.2.3.4. Método de semejanza gráfica . . . 10
    - 2.2.3.5. Transformación Hough generalizada . . . 10
    - 2.2.3.6. Técnica de supercuadráticos . . . 10
    - 2.2.3.7. Método de redes neuronales . . . 11
    - 2.2.3.8. Extracción de propiedades . . . 11
  - 2.2.4. Reconocimiento de posición . . . 11
    - 2.2.4.1. Métodos basados en redes neuronales . . . 11
    - 2.2.4.2. Métodos basados en seguimiento de movimiento . . . 11
    - 2.2.4.3. Técnicas de segmentación de imagen . . . 12
    - 2.2.4.4. Reconocimiento de posición de piezas en una microfábrica . . . 12

3. Sistema de visión por computadora para microensamble 13
- 3.1. Investigación en el Laboratorio de Micromecánica y Mecatrónica . . . 13
- 3.2. Sistema automático de manejo de piezas . . . 14
- 3.3. Subsistema de visión técnica (SVT) . . . 17

4. Diseño del SVT **19**
4.1. Principio de operación . . . 19
4.2. Piezas seleccionadas para experimentos . . . 19
4.3. Las imágenes del SVT . . . 20
4.4. Selección de clasificadores . . . 21
4.4.1. Resultados previos del clasificador LIRA . . . 21
4.4.2. Resultados previos del clasificador PCNC . . . 22
4.5. Bases de datos de imágenes utilizadas . . . 22
4.5.1. Imágenes normalizadas . . . 22
4.5.2. Descripción de las bases de datos . . . 23
4.6. Localización de piezas . . . 26

5. Clasificador neuronal LIRA **29**
5.1. Antecedentes . . . 29
5.1.1. Descriptor local aleatorio . . . 29
5.2. Estructura . . . 30
5.3. Operación . . . 32
5.4. Proceso de entrenamiento . . . 33
5.5. Distorsiones . . . 33
5.6. Discusión . . . 34

6. Clasificador Neuronal de Permutación de Códigos (PCNC) **35**
6.1. Estructura . . . 35
6.1.1. Extractor de propiedades . . . 36
6.1.2. Codificador . . . 38
6.1.2.1. Codificación de las propiedades . . . 39
6.1.2.2. Ejemplo de permutaciones . . . 40
6.1.2.3. Propiedades de las permutaciones . . . 40
6.1.2.4. Vector código **V** . . . 41
6.1.2.5. Adelgazador dependiente de contexto (CDT) . . . 42
6.1.3. Clasificador neuronal . . . 42
6.2. Proceso de entrenamiento . . . 43
6.3. Distorsiones . . . 43
6.4. Discusión . . . 44

7. Experimentos y resultados **47**
7.1. LIRA . . . 47
7.1.1. Experimentos preliminares . . . 48
7.1.2. Unicidad . . . 48
7.1.3. Sintonización de parámetros . . . 49
7.1.4. Distorsiones . . . 52
7.1.5. Ciclos de entrenamiento . . . 52
7.1.6. Aleatoriedad de los conjuntos para entrenamiento y prueba . . . 53
7.1.7. Mejores clasificadores LIRA . . . 53
7.1.7.1. Base de datos A . . . 53
7.1.7.2. Base de datos B . . . 54
7.1.7.3. Base de datos D . . . 54
7.1.8. Prueba con conjunto de entrenamiento ampliado . . . 55
7.2. PCNC . . . 57
7.2.1. Experimentos preliminares . . . 58
7.2.2. Unicidad . . . 58
7.2.3. Sintonización de parámetros . . . 59
7.2.4. Distorsiones . . . 62
7.2.5. Ciclos de entrenamiento . . . 63
7.2.6. Aleatoriedad de los conjuntos para entrenamiento y prueba . . . 63
7.2.7. Mejores clasificadores PCNC . . . 63
7.2.7.1. Base de datos A . . . 64
7.2.7.2. Base de datos B . . . 64
7.2.7.3. Base de datos D . . . 65

7.2.8. Prueba con conjunto de entrenamiento ampliado . . . 66
7.3. Comparación de clasificadores . . . 66
7.3.1. Tiempos . . . 66
7.3.2. Estabilidad en la creación de estructuras . . . 67
7.3.3. Parámetros . . . 68
7.3.4. Ciclos de entrenamiento . . . 68
7.3.5. Confiabilidad . . . 68
7.3.6. Resultados . . . 68
7.4. Búsqueda y reconocimiento de posición . . . 69

Conclusiones **73**

Trabajo futuro **75**

Apéndices 77

A. Software **79**
A.1. Módulo Optik . . . 79
A.2. Módulo RNA . . . 81
A.3. Módulo Localizador . . . 81
A.4. Interfaz . . . 82
A.5. Implementación de LIRA . . . 84
A.6. Software de línea de comandos . . . 85
A.6.1. lira2007 . . . 85
A.6.2. pcnc2007 . . . 86
A.6.3. Potencial para la experimentación . . . 87

B. Publicaciones **93**

Bibliografía **95**

Índice de figuras

3.1. Máquinas de la primera generación MET. 15
3.2. Sistema automático de manejo de piezas. 16
3.3. Sistema de Visión Técnica. 17
3.4. Esquema de localización de una pieza reconocida. 18
3.5. Diversas piezas utilizadas en el desarrollo del SVT. 18

4.1. Ventajas de la ventana circular aplicada a las imágenes. 23
4.2. Las cinco clases en la Base de Datos α. 24
4.3. Ejemplos de las ocho clases en la Base de Datos A. 24
4.4. Ejemplos de las siete clases de la Base de Datos B. 25
4.5. Ejemplos de las siete clases de la Base de Datos D. 25
4.6. Esquema del movimiento de la ventana de búsqueda. 27

5.1. Perceptrón de tres capas de Rosenblatt. 30
5.2. Estructura del clasificador neuronal LIRA. 31
5.3. Histogramas de distribución de entradas en dos construcciones LIRA distintas. 32
5.4. Distorsiones para ampliar el conjunto de entrenamiento. 34

6.1. Estructura del PCNC. 35
6.2. Procesos del extractor de propiedades. 36
6.3. Selección de puntos específicos. 36
6.4. Extracción de propiedades. 37
6.5. Procesos del codificador. 38
6.6. Permutación y su representación. 39
6.7. Permutaciones X. 41
6.8. Permutaciones Y. 42
6.9. Permutaciones XY. 43
6.10. Permutaciones Q. 44
6.11. Clasificador neuronal LIRA dividido. 45
6.12. Distorsiones para ampliar el conjunto de entrenamiento. 45

7.1. Imágenes reconocidas en la BD-A por el clasificador LIRA. 55
7.2. Imágenes mal reconocidas en la BD-A por el clasificador LIRA. 55
7.3. Imágenes reconocidas en la BD-B por el clasificador LIRA. 56
7.4. Imágenes mal reconocidas en la BD-B por el clasificador LIRA. 56
7.5. Imágenes reconocidas en la BD-D por el clasificador LIRA. 57
7.6. Imágenes mal reconocidas en la BD-D por el clasificador LIRA. 57
7.7. Imágenes reconocidas en la BD-A por el PCNC. 64
7.8. Imágenes mal reconocidas en la BD-A por el PCNC. 65
7.9. Imágenes reconocidas en la BD-B por el PCNC. 65
7.10. Imágenes mal reconocidas en la BD-B por el PCNC. 65
7.11. Imágenes reconocidas en la BD-D por el PCNC. 66
7.12. Imágenes mal reconocidas en la BD-D por el PCNC. 66
7.13. Localización y reconocimiento de piezas. 70
7.14. Localización y reconocimiento de piezas con redundancia. 70
7.15. Localización y reconocimiento de piezas que se tocan. 71

A.1. Diagrama a bloques del software OptikRNA. 80
A.2. Interfaz gráfica de usuario (IGU) de Optik. 81
A.3. Clases principales del módulo RNA y su relación de herencia. 82
A.4. Interfaz gráfica de usuario del software RNA. 83
A.5. Diagrama a bloques del software OptikRNA. 83
A.6. IGU Rna. Localizador de piezas. 84
A.7. Clase neurona y sus clases derivadas. 85

Índice de tablas

3.1. Generaciones de microequipo de acuerdo a su escala de reducción. 14

4.1. Características de las bases de datos utilizadas. 26
4.2. Las clases de las bases de datos. 26

7.1. Experimento preliminar para la sintonización de parámetros LIRA. 48
7.2. Experimento de unicidad con LIRA. 49
7.3. Experimento con el parámetro w. 49
7.4. Experimento con el parámetro η. 50
7.5. Experimento con el parámetro N. 50
7.6. Experimento con el parámetro p. 50
7.7. Experimento con el parámetro n. 50
7.8. Experimento con los parámetro p y n. 51
7.9. Mejor conjunto de parámetros LIRA. 51
7.10. Experimento con distorsiones con LIRA. 52
7.11. Experimento con diversos ciclos de entrenamiento. 53
7.12. Experimento con conjuntos aleatorios. 54
7.13. Resultado con los mejores clasificadores LIRA. 54
7.14. Experimento preliminar para la sintonización de parámetros PCNC. 59
7.15. Experimento de unicidad con el PCNC. 59
7.16. Experimento con el parámetro w. 59
7.17. Experimento con los parámetro p y n. 60
7.18. Experimento con el parámetro S. 60
7.19. Experimento con el parámetro N. 61
7.20. Experimento con el parámetro K. 61
7.21. Experimento con el parámetro D_c. 61
7.22. Experimento con el parámetro q. 62
7.23. Mejor conjunto de parámetros PCNC. 62
7.24. Experimento con distorsiones con el PCNC. 62
7.25. Experimento con diversos ciclos de entrenamiento. 63
7.26. Experimento con conjuntos aleatorios PCNC. 63
7.27. Resultado con los mejores clasificadores PCNC. 64
7.28. Tiempos empleados por los clasificadores. 67
7.29. Comparación de unicidad entre los clasificadores. 67
7.30. Confiabilidad de los clasificadores. 68
7.31. Resultados de los clasificadores sobre las bases de datos. 68

Investigación y desarrollo de un sistema de visión por computadora para aplicaciones en micromaquinado y microensamble

Tesis doctoral

Resumen

Gengis Kanhg Toledo Ramírez
Facultad de Ingeniería - CCADET
Universidad Nacional Autónoma de México

La presente tesis trata del desarrollo de un sistema de visión por computadora para reconocimiento y localización de piezas. El objetivo del sistema es reconocer y localizar una determinada pieza aleatoriamente ubicada en un área de trabajo para que pueda ser manipulada automáticamente. El trabajo contribuye a la automatización de los procesos de manipulación y ensamble de una microfábrica. Si bien el sistema surge como una necesidad de la microfábrica, éste puede ser utilizado en cualquier escala de trabajo.

El sistema desarrollado nace dentro del proyecto de creación de microfábricas totalmente automáticas y autoreproducibles. Proyecto iniciado en el Laboratorio de Micromecánica y Mecatrónica de la Universidad Nacional Autónoma de México. Una necesidad fundamental del proyecto es la de lograr buen desempeño y automatización de los diversos procesos. Una forma de satisfacer estas necesidades es mediante los métodos de visión por computadora. Para flexibilizar al máximo las tareas de micromanipulación y microensamble es fundamental contar con un sistema automático de reconocimiento y selección de piezas. Es por esta razón que las investigaciones presentadas tuvieron como objetivo desarrollar un sistema que resolviese tales necesidades.

El objetivo general de los estudios realizados fue desarrollar un sistema capaz de reconocer y localizar con precisión aceptable una pieza de cierto tipo ubicada en un área de trabajo para poder ser manipulada. Este objetivo general requiere del desarrollo de dos aspectos técnicos, uno es el reconocimiento de piezas y el otro es la localización de éstas.

El alcance de las investigaciones realizadas ha sido desarrollar un sistema fuera de línea en fase experimental. Para lograr desarrollar el sistema se estudió la problemática particular haciendo enfoque especial en las tareas críticas de clasificación y localización de piezas. Para tales tareas es indispensable la utilización de tecnologías y algoritmos de inteligencia artificial. Se seleccionaron, adaptaron y aplicaron dos algoritmos adaptivos basados en redes neuronales y en permutación de códigos. En tareas similares estos algoritmos demostraron un desempeño sobresaliente. Los algoritmos se denominan clasificador neuronal LIRA y Clasificador Neuronal de Permutación de Códigos (PCNC).

Para el análisis y pruebas del sistema se crearon tres bases de datos distintas con imágenes de piezas. Se utilizaron tornillos de diversas clases, tuercas, arandelas y otras piezas comunes en una línea de ensamble. Las imágenes han sido tomadas a partir de imágenes originales que contienen diversas piezas localizadas aleatoriamente. Las bases de datos se integraron con seis o siete tipos distintos de piezas. Las imágenes de las bases de datos presentan características reales de un ambiente industrial tales como iluminación no especial, brillos y sombras, fondo no uniforme y en algunas ocaciones oclusión parcial o amontonamiento. Dos bases de datos, llamadas A y B, se integraron con imágenes donde sólo un tipo de pieza aparecia a la vez mientras que la otra, llamada D, se integró con imágenes donde todos los tipos de piezas se encuentran revueltas y amontonadas.

Se realizaron múltiples experimentos de validación y prueba de los algoritmos propuestos así como experimentos de carácter práctico. Los porcentajes de reconocimiento alcanzados para las bases de datos A, B y D fueron, para el clasificador neuronal LIRA de 93.75 %, 94.14 % y de 90.47 % respectivamente, mientras que para el PCNC fueron de 96.87 %, 97.80 % y de 91.43 % respectivamente. El PCNC requirió el doble de tiempo para el reconocimiento de una imagen con respecto al clasificador neuronal LIRA.

El sistema logró, para ambos clasificadores utilizados, buen desempeño en reconocimiento de piezas. El mejor resultado de casi 98 % da buenas perspectivas al sistema desarrollado. El desempeño para la base de datos D, con la existencia de piezas amontonadas, revueltas y particahnente ocluidas es alentador.

En cuanto a la tarea de búsqueda y localización el sistema logró proporcionar las coordenadas de la pieza requerida. Sin embargo para mejorar la precisión es necesario emplear más tiempo de procesamiento lo que resta eficiencia al sistema propuesto por lo que el método de localización y búsqueda debe ser mejorado.

El trabajo marca una línea de investigación novedosa en visión por computadora aplicada a tareas de automatización en microensamble y micromaquinado; siendo la automatización de tales procesos una etapa fundamental para alcanzar la microfábrica totalmente automatizada.

Research and development of a computer vision system for micro machiney and micro assembly applications

Doctoral Thesis

Abstract

Gengis Kanhg Toledo Ramírez
Faculty of Engineering - CCADET
National Autonomous University of Mexico

The aim of this thesis is the development of a computer vision system for work pieces recognition and position location. The objective of the system is to recognize and to locate one certain work piece randomly located in a work area, so this work piece can be manipulated. The system contributes to the automatization of the machining and the assembly processes within a micro factory. Although the system had born as an automatization requeriment for a microfactory, the system can be used in any work scale.

The developed system was born from the project that consists in the creation of fully automated and shelf-reproducing microfactories. This project started at the Micromechanics and Mechatronics Laboratory of the National Autonomous University of Mexico. One fundamental requirement of this project is to achieve performance and automation for its different processes. One way to reach such requirements is to use computer vision methods. Computer vision methods allow high flexibility in the systems. For micro manipulation and micro assembly tasks, it is necessary to have a recognition and location system for work pieces. Therefore, the detailed researches in this thesis have the objetive of to develop such system allowing that these tasks will be done.

The main goal of the researches that were done was to develop one system able to accurately recognize and to locate with acceptable precision one work piece of certain type randomly located in a work area for manipulation. In order to achieve this goal two technical aspects require to be develop, one of them is the recognition of work pieces and the other one is the location of the position of such work pieces.

The main reach of the investigations has been to develop an off-line working system in experimental stage. In order to manage the system develop the particular problematic was studied with special approach in the critical tasks. These tasks are the classification and the localization of work pieces. For these tasks it is fundamental to use technologies and algorithms from artificial intelligence. Two algorithms were selected, adapted and applied. These algorithms are based in artificial neural networks and code permutation paradigms. Both algorithms allowed very good performance in similar tasks. The algorithms are called neural classifier LIRA and Permutative Code Neural Classifier (PCNC).

For the analysis and proves of the system, three different databases of images from work pieces were created. The used work pieces were screws of several types, nuts, washers and other common used work pieces found in an assembly line. The images were taken from source images that contain several ramdomly located work pieces. The data bases were made with six or seven types of work pieces. The images of the databases have real characteristics actually found in the industrial enviroment such as no special illumination condition, shines, shades and not uniform background. Two databases, which are called A and B, were made with images that have only one type of work piece at a time whereas the third one, which is called D, was made with images that contains all the types of work pieces turned around and heaped up.

For the selected algorithms several experiments for validation and testing were done as well as concrete experiments with practical characteristics.

The recognition rates reached for data bases A, B and D were, for the neuronal classifier LIRA 93.75 %, 94.14 % and 90.47 % respectively, whereas for the PCNC were 96.87 %, 97.80 % and 91.43 % respectively. The PCNC required double amount of time for image recognition with respect to the neuronal classifier LIRA.

For both used classifiers the system obtained good performance in the recognition task. The best result of almost 98 % gives us good perspective for the developed system. The performance for the data base D, with the existence of heaped up, turned round and partially occluded work pieces is encouraging.

About the work pieces location task the system is able to provide the coordinates of the required work piece. Nevertheless to improve its precision it is necessary to use larger processing time which reduces the efficience of the proposed system. Thus, the searching and location method must be improved.

This work marks a novel research line on the computer vision problem for automation of microassembly and micromachinery. The automation of such processes means a fundamental stage in order to achieve the fully automatic microfactory.

Capítulo 1

Introducción

El auge global por la miniaturización de productos y dispositivos gracias al desarrollo de nuevas tecnologías [1–3] hace necesario desarrollar objetos, maquinaria y equipo de pequeñas dimensiones. Desde 1998 el Laboratorio de Micromecánica y Mecatrónica (LMM) del Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET) de la UNAM dirigido por el Dr. Ernst Kussul tiene como objetivo crear microfábricas de escritorio totalmente automatizadas. El plan global incluye crear generaciones de microfábricas en las cuales los componentes de cada generación son producidos por la generación anterior, siendo cada vez estas generaciones, de menores dimensiones. Esta tecnología es llamada "Tecnología de Microequipo" o MET (por sus siglas en inglés) [4, 5].
Una tarea fundamental para lograr los objetivos mencionados es la automatización de los procesos de micromanufactura y microensamble. Mediante la automatización de estos procesos se alcanzará eficiencia y precisión en los microequipos a desarrollar [6, 7], tanto para su manufactura como para su operación. Los sistemas de visión por computadora son indispensables para lograr la automatización deseada [8–12].
Entre las tecnologías disponibles de visión por computadora, las que utilizan redes neuronales artificiales son de especial interés ya que han demostrado capacidad para tratar con los problemas que se presentan, i.e., reconocimiento automático de piezas, inspección de calidad, detección de posición, entre otros [13].
Las aplicaciones de visión por computadora para los procesos de ensamble de una microfábrica son diversas, la aplicación que desarrollamos en el presente trabajo tiene que ver con el reconocimiento de piezas aleatoriamente distribuidas así como la identificación de sus posiciones.
Dentro de las investigaciones del LMM se han desarrollado un par de clasificadores neuronales. El primero de ellos llamado LIRA o formalmente Clasificador Neuronal LIRA, está basado en una modificación de la red neuronal artificial Perceptrón [14]. Éste se probó primeramente con éxito en la tarea de reconocimiento óptico de caracteres [15], desde entonces también ha sido probado con buenos resultados en aplicaciones de visión por computadora aplicadas a problemas de micromecánica [16]. El segundo clasificador llamado PCNC, o formalmente Clasificador Neuronal de Permutación de Códigos, está basado en los principios de *redes neuronales de proyección asociativa* desarrollados desde los años 80's [17, 18]. Éste se ha probado también en la tarea de reconocimiento óptico de caracteres con excelentes resultados [19].
El presente trabajo trata sobre la investigación en torno a los clasificadores neuronales mencionados aplicándolos a la tarea de reconocimiento de piezas y de sus posiciones suponiéndolas en un ambiente real de trabajo. La investigación parte del análisis del problema a resolver así como de las características requeridas por un sistema que le de solución. Analiza el estado del arte de los sistemas de visión por computadora enfocándose a la problemática particular. Describe y analiza ampliamente los dos clasificadores propuestos y describe con detalle las bases de datos utilizadas para la experimentación con dichos clasificadores. Se realizan múltiples experimentos y los resultados son presentados y comparados entre sí. Además un método de localización de piezas es presentado y probado. Los resultados obtenidos dan cuenta de la eficiencia alcanzada en cada uno de los casos y para cada uno de los clasificadores utilizados.

1.1. Objetivos generales

Esta tesis está comprendida dentro de diversas líneas de investigación del LMM que tienen que ver con la aplicación y desarrollo de sistemas de visión por computadora. En lo general estas líneas de investigación buscan la automatización y el aumento de la precisión en el equipo micromecánico usando algoritmos adaptivos basados en visión por computadora.

Las metas a largo plazo de estas líneas de investigación son la investigación y desarrollo de sistemas de control automático para todos los niveles de una fábrica sobre mesa. Estos sistemas pueden usar redes neuronales, algoritmos genéticos u otros métodos de inteligencia artificial o control automático [20].
Las metas a mediano plazo contemplan todas las tareas encaminadas a desarrollar el sistema de manipulación automática de piezas, abarcando la investigación y desarrollo de cada uno de los subsistemas requeridos así como la integración y funcionamiento en línea de todo el sistema.
Las lineas de investigación que enmarcan el trabajo presentado son las siguientes:

- Investigación de redes neuronales y de clasificadores neuronales basados en estas redes para el reconocimiento de imágenes.
- Investigación y desarrollo de sistemas de control adaptivo para la automatización total de los procesos de producción en micromáquinas herramientas.
- Investigación de los algoritmos de reconocimiento de patrones basados en redes neuronales.

1.2. Marco teórico

Uno de los principales retos asociados con el desarrollo de sistemas de control de microfábricas se encuentra relacionado con la automatización apoyada en sistemas de visión por computadora . En este trabajo se describen los resultados de la investigación de un sistema de visión por computadora basado en clasificadores neuronales, probado en modo fuera de línea.

1.2.1. Justificación.

En el LMM se desarrolla microequipo para producir dispositivos micromecánicos. Investigaciones similares se han hecho en países desarrollados como Japón, Alemania, Suiza y Estados Unidos [1, 5]. La diferencia en los trabajos del LMM es que en éste laboratorio se usan componentes de bajo costo. Esta diferencia da la posibilidad de promover los resultados de investigaciones a la industria y lanzar los productos micromecánicos al mercado mas rápido. Sin embargo el uso de componentes de bajo costo implica decremento de precisión en el microequipo. Para obtener precisión comparable con el microequipo creado por países desarrollados es necesario usar algoritmos adaptivos, incluidos los algoritmos basados en visión por computadora. Los trabajos preliminares hechos en el LMM muestran que el empleo de algoritmos adaptivos dan por resultado un aumento muy grande de precisión. Por ejemplo, un algoritmo adaptivo permitió producir en la micromáquina-herramienta mexicana desarrollada en el LMM, una flecha de latón de 50 micrómetros de diámetro, resultado que ha sido producido con un microtorno japonés mucho más costoso. Los algoritmos de visión por computadora mostraron que es posible usarlos en procesos de microensamble [4]. Estas pruebas preliminares constituyen la base para estudios más avanzados de algoritmos de control basados en visión por computadora. Estas investigaciones forman la base de la presente tesis.
La parte esencial de este trabajo es un sistema de visión para manipulación de piezas en modo automatizado, el cual sea capaz de reconocer y localizar un determinado tipo de pieza para su manipulación en función a las tareas de procesado que se requieran ejecutar. Para lo anterior se diseñó un sistema de visión de micro piezas probando dos algoritmos, seleccionados entre los existentes, para automatizar el proceso de micromanipulación y microensamble buscando el mejor desempeño posible.

1.2.2. Metodología.

En primer lugar se hizo un análisis de la literatura relacionada y se detallaron las tareas de investigación, que para esta investigación han sido las que tienen como meta desarrollar un sistema de identificación y localización automático de piezas de trabajo para su utilización en la microfábrica proyectada en el LMM. Como base teórica para este objetivo se han investigado diversos algoritmos adaptivos basados en redes neuronales y en permutación de códigos. Con base en trabajos preliminares con procesos de microensamble usando visión por computadora se desarrollaron los algoritmos de control de microensamble usando un sistema de visión por computadora en modalidad fuera de línea (*off-line*). Hemos decidido el uso de los clasificadores neuronales LIRA y PCNC por los buenos resultados obtenidos en experimentos previos relacionados [16, 19]. Para la implementación de éstos algoritmos se realizó un sistema de cómputo, que además de contener todas las características y parámetros que el algoritmo requiere, posee módulos para la experimentación con diversas bases de datos de piezas. Para la creación de la base de datos para los experimentos se han utilizado piezas de trabajo reales bajo condiciones reales esperadas en una microfábrica.

Se han realizado múltiples experimentos de validación y prueba de los algoritmos propuestos así como experimentos concretos con tareas prácticas. Adicionalmente se ha desarrollado un algoritmo de localización de piezas que se ha construido en base a los resultados de los experimentos anteriores y a la validación de los algoritmos utilizados inicialmente.
Los sistemas clasificadores desarrollados y el algoritmo de búsqueda han sido probados mediante múltiples experimentos y los resultados obtenidos son mostrados en el capítulo correspondiente.

1.3. Objetivo de la tesis

Como objetivo propio de la presente tesis se buscó diseñar un sistema basado en las investigaciones de las tecnologías y métodos existentes así como en las desarrolladas recientemente en el LMM, adaptándolos para que sean útiles a las tareas propias de la clasificación y localización de piezas destinadas al microensamble y el micromaquinado. Se logró diseñar y probar un novedoso sistema de identificación y localización de piezas para la microfábrica basado en visión por computadora, el cuál es eficiente y automático, es un sistema capaz de reconocer diferentes piezas así como hallar sus posiciones, considerando las características reales y diversas en tamaño y forma que pueden tener las piezas a manipular.
Para alcanzar los objetivos mencionados fue necesario:

- Analizar las tareas requeridas.
- Elegir las tecnologías mas apropiadas (algoritmos genéticos, redes neuronales, etc.).
- Desarrollar e implementar el sistema utilizando un lenguaje de cómputo orientado a objetos.
- Probar el sistema con piezas de trabajo reales.
- Investigar, probar y sintonizar los clasificadores seleccionados.
- Corregir las estructuras y los algoritmos del sistema.
- Analizar los resultados obtenidos y realizar las correcciones pertinentes.

1.4. Contribución de este trabajo

Este trabajo marca una línea de investigación nueva en visión por computadora aplicada a tareas de automatización de ensamble y micromaquinado. Diversos sistemas particulares existen para reconocer objetos, sin embargo el trabajo aquí presentado desarrolla una investigación aplicada a una tarea específica. El sistema se ha probado con piezas reales con las que podría trabajar y en las condiciones reales de utilización. El sistema desarrollado ha logrado una eficiencia mínima de poco más del 90 % y una eficiencia máxima de poco más del 98 % en identificación de piezas; además de una eficiencia aceptable en localización de piezas. Una publicación internacional de prestigio ha merecido este trabajo [21] y otra más editada en México [22]. Los resultados logrados se han expuesto en diversos congresos tanto internacionales [13] como nacionales [23]. Una nueva publicación internacional está preparándose.
Específicamente, se desarrolló un sistema de visión por computadora para el reconocimiento de piezas dirigido a una microfábrica. Para el reconocimiento de piezas y de sus posiciones se adaptaron e implementaron el clasificador neuronal LIRA escala de grises [24] y el Clasificador Neuronal de Permutación de Códigos [19]. Los sistemas se calibraron y probaron con tres bases de datos formadas con piezas de trabajo bajo condiciones comunes. El primer prototipo de este sistema trabaja fuera de línea (*off-line*).
Este sistema de visión por computadora es flexible, genera un incremento en el nivel de automatización de los procesos de manipulación de piezas utilizado en los procesos de micromanufactura y microensamblado. Recordando la propuesta basada en generaciones sucesivas para el desarrollo de microequipo [25], podemos asegurar que este sistema de identificación y ubicación de piezas proporciona una etapa fundamental para lograr la automatización completa de una microfábrica, permitiendo así avanzar en uno de los objetivos primordiales del LMM que es producir micro piezas con buenas características de precisión a bajo costo [4].
Para la investigación y experimentación utilizando el Clasificador Neuronal LIRA y el PCNC se ha desarrollado software orientado a objetos en C++ estándar, este software asiste el proceso de creación de bases de datos para el entrenamiento y prueba de los clasificadores, implementa los clasificadores mismos, implementa el sistema de localización de piezas entre muchas otras funciones secundarias. El software desarrollado es flexible, fácil de utilizar y ha sido diseñado con capacidad de ser fácilmente ampliado y mejorado.

1.5. Organización del trabajo

La presente tesis se organiza de la siguiente forma. El Capítulo 1 da una introducción sobre el presente trabajo, enmarcándolo en las líneas de investigación a las que pertenece, brindando su marco teórico y sus objetivos para acabar explicando su contribución. El Capítulo 2 trata de los antecedentes básicos en microtecnología y visión por computadora. Estos antecedentes se enfocan en la relación que guardan entre sí ambos conceptos. Se revisa el estado del arte de la técnica de visión por computadora respecto a los problemas de reconocimiento de objetos y de localización de los mismos. El Capítulo 3 sitúa el sistema desarrollado dentro de las investigaciones que se llevan a cabo dentro del proyecto de microfábricas del LMM. El Capítulo 4 describe el sistema de visión técnica propuesto en este trabajo y presenta los algoritmos utilizados así como las bases de datos de imágenes creadas para probar el sistema. El Capítulo 5 está dedicado a la estructura, proceso de entrenamiento y discusión del clasificador neuronal LIRA mientras que el Capítulo 6 hace lo propio para el PCNC. El Capítulo 7 se ocupa de los experimentos realizados con los clasificadores LIRA y PCNC, se justifican y describen los experimentos realizados, se muestran los resultados obtenidos y la eficiencia de los métodos investigados. Por último se proporcionan las conclusiones del presente trabajo resumiendo los resultados obtenidos. En el Apéndice se describe el software desarrollado para realizar esta investigación y se listan las publicaciones realizadas con base en este trabajo.

Capítulo 2

Antecedentes y estado del arte

El presente capítulo se dedica a los antecedentes sobre el trabajo presentado. Se divide en dos partes: Microtecnología y Visión por computadora. Se dan los antecedentes y conceptos básicos sobre la microtecnología haciendo énfasis en la micromecánica y en las microfábricas. Se explican diversos problemas de las microfábricas que pueden resolverse mediante métodos de visión por computadora. Se orienta la discusión hacia el problema de reconocimiento de objetos, que es el área sobre la que trata este trabajo. La segunda parte se dedica a la visión por computadora en el tema de reconocimiento de piezas (objetos) y sus posiciones. El tema se enmarca en el ámbito de las microfábricas y se da un repaso breve al estado del arte sobre este tema. Se habla también del estado del arte para localización de objetos enmarcando esta tarea en el ámbito de una microfábrica.

2.1. Microtecnología

La microtecnología se refiere a aquellas tecnologías enfocadas a producir objetos, dispositivos y equipos de escala micrométrica. En 1959 el físico Richard Feynman advirtió por primera vez del enorme potencial de las tecnologías de escala reducida [26].
La microelectrónica fue la primera de las microtecnologías en ser desarrollada, se inició en la década de los 60's mediante la integración de cientos de transistores en una oblea de silicio de pocos centímetro cuadrados, desde entonces la popularidad de la microelectrónica ha venido en aumento no requiriendo mayor presentación. En la década de los 80's diversas investigaciones sugirieron la posibilidad de aplicar los métodos de la microelectrónica a la fabricación de dispositivos mecánicos naciendo con esto la micromecánica [27]. Desde entonces la microtecnología se ha basado mayoritariamente en los métodos de producción de semiconductores, dado que el silicio es el material más utilizado en la industria electrónica también lo ha venido siendo en la micromecánica[1]. Nuevas aplicaciones de la microtecnología han venido desarrollándose tales como las aplicaciones energéticas y químicas. Muchos campos microtecnológicos requieren de enfoques multidisciplinarios por lo que han surgido la micro mecatrónica y los sistemas micro opto-electromecánicos (MOEMS, por sus siglas en inglés) [28, 29]. Las aplicaciones energéticas, como las celdas de almacenamiento de energía, sufren calentamientos en el material que el silicio no soporta con eficiencia con respecto a otros materiales [30]. Igualmente en muchas aplicaciones químicas el silicio no es el mejor material. Lo anterior ha impulsado la investigación microtecnológica con materiales diversos como plásticos, cerámicas, cristales o metales como aluminio y acero inoxidable. Dado lo reciente de estas investigaciones en microtecnologías, existe la necesidad de utilizar nuevos materiales cuyos usos no están bien desarrollados con respecto al silicio. Materiales que requieren otras tecnologías de manufactura inclusive.

2.1.1. Aplicaciones de la microtecnología

Entre las áreas más importantes de aplicación de la microtecnología se encuentran la industria automotriz, la ingeniería médica, la electrónica, el control del ambiente, la tecnología aeroespacial, los sistemas de visión, la automatización industrial y la domótica entre otras menos importantes [31]. El mercado potencial para los microproductos fue estimado en más de 30 mil millones de euros para el año 2002 [32]. El impacto de la

[1] Dado que se tiene mucho conocimiento tecnológico sobre este elemento además de las instalaciones para su tratamiento.

microingeniería para el fin del siglo XX fue señalado mediante las cotas de participación en diversas áreas de la ingeniería. El área informática y de telecomunicaciones representará un 37 % del mercado mundial de microtecnologías, la industria automotriz un 31 %, la industria médica un 6 % y la de automatización un 2 %. El 24 % restante se repartirá entre las ingenierías restantes. Para el año 2004 la mayor parte de las aplicaciones microtecnológicas reportadas eran sensores siendo destinadas a computadoras o a sistemas micro electromecánicos (MEMS, por sus siglas en inglés) [30]. El resto de las aplicaciones microtecnológicas eran microactuadores y en menor cantidad dispositivos más elaborados de carácter experimental.
Existen diversas aplicaciones específicas para cada una de las principales áreas de aplicación de las microtecnologías [27, 33]. Entre las áreas de aplicación destacan las aplicaciones médicas como los sistemas de análisis de aire para respiración y sangre, componentes para sistemas de catéters y endoscópicos, instrumentos para microcirugía, microbombas y microválvulas, componentes para suministración gradual de fármacos, transductores ultrasónicos y electrodos para estimulación nerviosa. Para la industria automotriz destacan las aplicaciones mecatrónicas para los sistemas de inyección e ignición, inyección inteligente de combustible, sistemas de navegación y guía, sintetizadores de voz, seguridad y autodiagnóstico, control de temperatura, presión, distancia, etc. En el campo de las comunicaciones y la información destacan las aplicaciones de microconectores, equipo de prueba, cabezales para impresoras de inyección de tinta, guías de ondas electromagnéticas, dispositivos de microondas además de acopladores y multiplexores ópticos.

2.1.2. Sistemas Micro Electromecánicos (MEMS)

La microtecnología que ha experimentado más auge después de la microelectrónica ha sido la micromecánica. Sin embargo ésta se ha desarrollado acompañada muchas veces de microelectrónica. Dado lo anterior y a la diversidad de países participantes en su desarrollo ésta se conoce con diversos nombres. A esta tecnología en Japón se le conoce como "tecnología de micromáquinas" (MMT, por sus siglas en inglés), en Europa como "Tecnología de microsistemas" (MST, por sus siglas en inglés) y en los Estados Unidos como "Sistemas Micro Electromecánicos" (MEMS, por sus siglas en inglés). Los términos anteriores estrictamente significan conceptos similares pero no iguales, esto se debe a las variantes en líneas de investigación de cada uno de los países implicados. Sin embargo el objeto de estudio y aplicación de todas estas tecnologías es el mismo. En el futuro los criterios deberán generalizarse y una sola denominación tendrá que ser adoptada sobre las otras. En el presente trabajo se ha preferido utilizar MEMS por ser el término más empleado en nuestro país.
El término MEMS lo propuso el Profesor R. Howe en 1989 en Estados Unidos [34]. En lo general los MEMS están basados en las tecnologías de silicio y los procesos de fabricación son similares a los de generación de circuitos integrados. Las principales ventajas que ofrecen los MEMS son la capacidad de miniaturización de dispositivos existentes (e.g. sensores), el desarrollo de nuevos dispositivos bajo principios físicos que sólo funcionan en la pequeña escala micrométrica (e.g. biochip) y el desarrollo de nuevas herramientas para interactuar con los dispositivos MEMS creados (e.g. microscopio de efecto de túnel) [35].
Los MEMS se producen mayoritariamente con los métodos de producción de los semiconductores. Para el año 1997 más del 99 % de los MEMS comerciales utilizaban la litografía como tecnología de producción [36]. Por esto solo se permite fabricar piezas de dos dimensiones (2D) con cierto grosor y solo se permite el uso de materiales compatibles con la tecnología del silicio [4].
Existen tres ventajas sobresalientes por las que el desarrollo de MEMS es tan demandado [33, 37]. La primera es debida a que las deformaciones por calor en los materiales disminuyen con las dimensiones. La segunda tiene que ver con la reducción en el material necesario para producir, lo cual se logra principalmente por la reducción en el material de desecho. Por ésto se pueden utilizar materiales mas caros. La tercera ventaja se da en las vibraciones; dado que las amplitudes de las vibraciones se reducen mucho más a prisa que la reducción de las dimensiones del equipo. Esto último es debido a que las fuerzas inerciales se decrementan por un factor de cuarta potencia respecto a la dimensión, mientras que las fuerzas elásticas lo hacen a sólo un factor de potencia dos.

2.1.3. Microfábricas

El concepto de microfábrica fue propuesto por el Laboratorio de Ingeniería Mecánica de Japón en 1990 [38]. Las microfábricas tienen su base en el desarrollado de la micromecánica. El objetivo es crear una fábrica funcional de tamaño reducido capaz de producir microproductos. Una fábrica con estas características debe tener un alto grado de automatización o contar con interfaces para operarios humanos. La tendencia global en fábricas, sin embargo, es la reducción de las operaciones humanas. Las ventajas mas destacables de una microfábrica son su alta capacidad de ahorrar espacio y su reducido peso. Además se busca que una microfábrica sea flexible para que pueda producir gran diversidad de productos [39].

Una microfábrica automática debe tener diversos componentes. Como mínimo requiere de máquinas herramientas automáticas, dispositivos de ensamble automático, robots para alimentar las máquinas herramientas, dispositivos o sistemas de inspección de calidad, sistemas de eliminación de desechos, así como sistemas de inspección y reemplazo de herramental.
La idea de la creación de microfábricas está apoyada por investigadores de diversos países [3, 4, 40].
Un proyecto especial para el desarrollo de microfábricas basado en herramientas de micromáquinas miniatura fue iniciado en Japón [40], el Laboratorio de Ingeniería Mecánica desarrolló una microfábrica sobre mesa [7, 41]. Esta microfábrica consiste de máquinas herramienta, tales como un torno, una fresadora, una prensa; y máquinas de ensamble como lo son un brazo de transferencia y una mano con dos dedos. Esta microfábrica portable tiene dimensiones externas de $625\times490\times380$ mm^3 y un peso total de 34 Kg.
En Suiza, se han desarrollado los detalles del control de precisión de movimiento y los principios de micromanipulación para futuras microfábricas [3].
En Ucrania se inició el desarrollo de un microcentro de maquinado [25, 42, 43]. Este desarrollo continua en México donde se ha construido y caracterizado un microcentro de maquinado con dimensiones de $130\times160\times85mm^3$. Este centro de maquinado ha permitido manufacturar objetos de materiales diversos como acero y diversos plásticos de entre 50μ y 5mm [4]. La propuesta ucraniano-mexicana está basada en la tecnología de microequipo (MET[2]). Esta propuesta tiene como base ideológica el bajo costo comparada con los otros desarrollos que utilizan componente ultraprecisos que elevan el costo de los equipos considerablemente. La microfábrica mexicana pretende compensar la falta de precisión del equipo de bajo costo mediante técnicas de inteligencia artificial. El bajo costo de la MET no sólo implica ahorrar recursos, si no que tiene por objetivo hacer la tecnología para microequipo accesible a países, instituciones o grupos con recursos tecnológicos y económicos limitados.
Uno de los principales retos, común a todas las microfábricas mencionadas, es el de su total automatización. Ésto es con el fin de lograr producción masiva y flexible a costos reducidos.
Respecto a Estados Unidos, no se encontraron desarrollos importantes en micromáquinas ni existen trabajos similares referenciados en los trabajos consultados. Por lo anterior se piensa que tal desarrollo existe en ese país de forma secreta, ya que no es posible pensar que la actual potencia mundial no tenga interés en esta área. La alternativa es que toda su apuesta esté concentrada en la nanotecnología.

2.1.4. Tecnología de microequipo (MET)

Ante la necesidad de lograr desarrollos tecnológicos sin las limitaciones que tiene la tecnología de MEMS varios investigadores han propuesto utilizar tecnologías convencionales adaptándolas a pequeñas escalas [6, 33, 36, 44–46]. En 1996 el Dr. Ernst Kussul y sus colaboradores desarrollaron las bases de una nueva tecnología [25]. Propusieron desarrollar micromáquinas-herramientas y micropiezas con los métodos tradicionales agrupándolas en generaciones relacionadas entre sí por herencia; esto es, cada generación será una celda de manufactura con micromáquinas-herramientas, micromanipuladores y demás sistemas necesarios que funcionando en forma automática serán capaces de fabricar las piezas, y después las máquinas, que formen la segunda generación, igual que la anterior, pero de dimensiones más reducidas; y así sucesivamente [42, 43]. A esta propuesta se le ha llamado Tecnología de Micro Equipo o MET (Micro Equipment Tecnology) [4]. La tecnología MET requiere de micromáquinas herramientas y de dispositivos para microensamble que sean capaces de producir las piezas y luego las máquinas de una generación siguiente. La primera generación es desarrollada con tecnología mecánica convencional pero siguiendo un diseño que permita la subsecuente reducción de los componentes, es decir un diseño MET. La característica principal de dicho diseño es que no se incluyen piezas o partes con detalles relativamente pequeños de modo que el diseño general pueda ser reducido y fabricado sin grandes inconvenientes. Una característica fundamental de la MET es su bajo costo. Con respecto a tecnologías similares que usan métodos de mecánica convencional, la MET propone reducir considerablemente los costos. Tal reducción se logra mediante la sustitución de equipo de alta precisión (e.g. interferómetros y codificadores[3]) mediante el uso intensivo de técnicas de inteligencia artificial. Una de estas técnicas es la visión por computadora, misma que se aborda en este trabajo.

2.1.5. Micromaquinado y microensamble

Dos procesos fundamentales en una microfábrica son el micromaquinado y el microensamble [6, 7]. Estos procesos son responsables de la generación de micropiezas y de su posterior ensamble para concretar un

[2] Ésta se explica en la siguiente sección.
[3] Mejor conocidos por su nombre en inglés: encoders.

dispositivo o producto. Es por lo anterior que la mayoría de los trabajos sobre microfábricas se orientan sobre alguno de estos procesos.
Ambos procesos mencionados poseen características particulares de acuerdo a las dimensiones de la microfábrica. El micromaquinado deberá tomar en cuenta las características y dimensiones de las impurezas del material que se ocupe a medida que las dimensiones de trabajo cambien. El microensamble requerirá mayor precisión en los equipos encargados de la sujeción, manipulación, alineación y demás tareas propias de este proceso.
Las tecnologías de micromanufactura y microensamble requieren automatizar los sistemas de manipulación y de maquinado para que sean capaces de adaptarse a los requerimientos [47]. Un sistema para ensamble ha sido propuesto por Mardanov et al [48]. La estación de ensamble propuesta tiene módulos lógicos y rutas de información comunes de modo que pueda ser fácilmente integrada en un sistema completo de ensamble.

2.1.6. Reconocimiento de piezas en una microfábrica

En una microfábrica automática es imprescindible un sistema automático de manipulación de material. Se busca que este sistema al igual que el conjunto de la microfábrica sea flexible. La flexibilidad es una característica requerida de modo que las microfábricas no se dediquen a producir un solo producto, sino que éstas sean capaces de producir una amplia gama de productos o al menos variaciones de un mismo producto en forma masiva. Esta necesidad lleva a desarrollar un sistema de manipulación de piezas y materiales capaz de identificar un determinado tipo de pieza aleatoriamente localizado en alguna superficie de trabajo. Un sistema que sea capaz de identificar cierta pieza requerida y de localizarla de modo que la pieza deseada pueda ser sujetada para su manipulación. Este problema exige la aplicación de métodos de visión por computadora para reconocimiento de objetos. La resolución de esta tarea es fundamental para alcanzar los objetivos señalados para las microfábricas.

2.2. Visión por computadora

La visión por computadora, también llamada visión artificial o visión técnica, es una disciplina científica y técnica diversa y en proceso de maduración, se considera parte de la inteligencia artificial. La visión por computadora es una ciencia por que requiere de múltiples herramientas y conocimientos propios de las ciencias exactas y de las ciencias naturales (matemáticas, física, estadística, neurología, etc.). Es una técnica porque la implementación de sistemas prácticos requiere desarrollar técnicas, equipos y dispositivos que permitan experimentar con las propuestas teóricas y alcanzar fines prácticos [49].
Por lo general no existe una técnica o un método particular o genérico para los problemas de visión por computadora, más bien son aplicados múltiples conocimientos y técnicas de diversas áreas. Ha sido hasta los años 70's que la tecnología microelectrónica e informática ha permitido el despegue de este campo a través del incremento en la capacidad de cómputo, misma que posibilitó el trabajo con imágenes de forma acelerada y flexible.
En visión por computadora son escasos los avances genéricos y en general existen múltiples soluciones y propuestas para cada grupo de problemas planteados, siendo éstas frecuentemente muy particulares a la tarea en que se aplican. Desde la década de los 90's se han dado avances importantes con sistemas con autoaprendizaje y dedicados a los problemas de visión en 3D.

2.2.1. Visión por computadora en microfábricas

Dadas las características de una microfábrica se requiere que sean aplicadas diversas técnicas de automatización. La automatización de sistemas se sirve de diversas disciplinas científicas y tecnológicas las cuales se basan mayoritariamente en hardware, algoritmos y métodos computacionales. La automatización puede dividirse en blanda y dura[4]. La automatización dura se refiere a las técnicas de control automático de tareas específicas y bien determinadas como es el caso del control numérico, los controladores lógicos programables (PLC, por sus siglas en inglés) y los manipuladores básicos [50]. La automatización blanda se refiere al uso de métodos de inteligencia artificial (IA) para controlar y automatizar tareas que tienen un cierto grado de racionalidad o asemejan conductas humanas complejas como es el caso de algoritmos genéticos, sistemas expertos, lógica fuzzy y redes neuronales artificiales.

[4]No existe consenso en la clasificación para la automatización, sin embargo diversos autores se refieren siempre a dos grandes grupos que en buena medida son iguales, estos se denominan además de blanda/dura, procesal/racional, simple/compleja y algorítmica/racional.

Es por lo anterior que los sistemas de visión por computadora tienen importantes aplicaciones en las microfábricas. Estas aplicaciones están relacionadas con los procesos de ensamble, de inspección y de manipulación de piezas. En los procesos de ensamble, las tolerancias en la precisión de los cálculos generados por estas tareas son muy cerradas. Es por esto que para realizar la unión de dispositivos o piezas, es necesario un alto grado de exactitud del sistema. En los procesos de inspección se requiere medir el tamaño de micro piezas maquinadas y evaluar la calidad de sus superficies, detectar el nivel de contaminación por material cortado, inspeccionar el estado de las herramientas, determinar las posiciones relativas entre herramientas y materia prima, entre otras tareas. En cuanto a los procesos de manipulación, se requiere que sean lo más flexibles posibles, es decir, que puedan trabajar con una amplia gama de tareas, procesos y piezas, es por ello que el reconocimiento automático de piezas y de sus posiciones tiene funciones muy importantes.

Existen diversas propuestas para construir los sistemas de visión computacional para los propósitos mencionados: [51–54]. En los trabajos [16, 55, 56] se propone una alternativa para producir microdispositivos mediante ensamble automático basado en un sistema de visión por computadora.

Entre los diversos sistemas automáticos necesarios en una microfábrica orientados a los procesos de micromaquinado y microensamble se requiere un sistema automático de manipulación de material y productos. Un sistema de este tipo deberá tener las siguientes funciones: carga de material a los equipos de micromaquinado, extracción de las partes maquinadas, identificación espacial de las piezas requeridas para su sujeción, transporte, posicionamiento y liberación.

Con el objetivo de contribuir al desarrollo de una microfábrica automática, en el presente trabajo es desarrollado un sistema de reconocimiento y localización de piezas mediante visión por computadora. Este trabajo tiene como bases los resultados obtenidos y presentados en diversas publicaciones [13, 21]. Dicho sistema pensado para ser parte de una microfábrica es presentado en el Capítulo 3.

2.2.2. Reconocimiento de objetos

El problema del reconocimiento de objetos así como diversos métodos para su solución han sido presentados en [8, 9]. Existen muchos métodos y propuestas para resolver el problema del reconocimiento de objetos [11, 12, 54, 57–61]. A menudo la primera etapa de estos métodos es la extracción de ciertas características. Una de estas características ampliamente utilizada son los contornos de la imagen que incluye el objeto que se desea reconocer.

2.2.3. Estado del arte

En la presente sección se resume el estado del arte de la visión por computadora aplicada a reconocimiento de objetos. Diversas etapas son utilizadas para resolver estas tareas. En primer lugar se explica el preprocesamiento de imágenes que es por lo general la primera etapa utilizada en reconocimiento. Después se describen y discuten brevemente seis métodos reportados frecuentemente en la literatura sobre reconocimiento de objetos de dos dimensiones. Los métodos presentados son: semejanza de patrones (*pattern matching*), análisis de componentes principales (*principal components analisys*), semejanza gráfica (*graph matching*), transformada Hough generalizada (*generalised Houg transformation*) [60], técnica de supercuadráticos y técnica con redes neuronales. Los primeros cuatro requieren de una etapa de preprocesamiento en que se aplique algún algoritmo de extracción de contornos. Estos seis métodos pueden tener o no diversas variantes. En especial la técnica de redes neuronales es muy utilizada en reconocimiento de imágenes por lo que gran cantidad de investigaciones han sido realizadas con estas, siendo más apropiado hablar de técnicas basadas en redes neuronales debido a la diversidad existente. Al final de la sección se menciona la extracción de propiedades, que es una etapa comúnmente utilizada por métodos basados en redes neuronales y otros métodos similares.

2.2.3.1. Preprocesamiento de imágenes

Las etapas de preprocesamiento son ampliamente utilizadas para la preparación de las imágenes destinadas a reconocimiento de objetos. Diversas técnicas de preprocesamiento son utilizadas según la técnica específica a utilizar sobre ellas. Las técnicas más ampliamente utilizadas son: detección de bordes, segmentación de partes, extracción de propiedades y cálculo de parámetros geométricos. La etapa de preprocesamiento puede tomar un gran porcentaje de los recursos de cómputo y de tiempo con respecto al total de lo necesario para toda la etapa de reconocimiento.

2.2.3.2. Método de semejanza de patrones

La diferencia de cuadrados entre una imagen I y una imagen plantilla T es sumada respecto a sus píxeles:

$$r(x,y) = \sum_{i \in T} \sum_{j \in T} \left(I(x+i, y+j) - T(i,j)\right)^2 \tag{2.1}$$

Si la suma resultante es mayor que un cierto umbral obtenido experimentalmente, el objeto plantilla es encontrado en I [62]. Este método consume bastante memoria y tiempo de procesador además que no es capaz de trabajar adecuadamente con cambios de brillo o escalamiento de las imágenes.

2.2.3.3. Método de análisis de componentes principales

Este método es una técnica basada en correlación. Esta técnica tiene dos faces, una de aprendizaje y otra de aplicación. La imagen se almacena en un vector, para reducir el tiempo de cómputo este vector se transforma en otro de dimensión más pequeña mediante una transformación vectorial T. Este vector de transformación se escoge de tal manera que maximice la variación de las transformaciones de los vectores de imagen. En la etapa de aplicación, la imagen a ser procesada es transformada con el vector T y luego se busca la correlación con las imágenes de la base de datos [10]. Este método posee estabilidad en la detección y es capaz de realizar ciertas correcciones, sin embargo funciona sólo si el objeto puede ser diferenciado claramente respecto al fondo de la imagen.

2.2.3.4. Método de semejanza gráfica

Este método funciona sobre un modelo del contorno de un objeto en una imagen [60, 63]. La idea es analizar segmentos de recta ortogonales que crucen ciertos puntos del contorno del objeto a reconocer. La búsqueda de características de la imagen se hace sobre de estas líneas. Las características similares son utilizadas para comparar un modelo con una imagen dada para el reconocimiento de objetos. Cualquier característica puede ser implementada con este método (e.g. color, brillo, textura o contraste). Este método requiere bastante memoria, sin embargo el tiempo de cómputo es reducido. Con condiciones específicas este método puede trabajar con imágenes de objetos parcialmente ocultas, ampliadas, reducidas o giradas. Este método no funciona con imágenes que contengan varios objetos además de que el reconocimiento de objetos pequeños se torna complicado.

2.2.3.5. Transformación Hough generalizada

En este método los contornos son aproximados mediante un conjunto de puntos tomados del contorno del objeto en la imagen [64]. Cada punto sobre el contorno del objeto se describe en términos de un punto de referencia ubicado dentro del contorno. La imagen de contorno para este método puede ser obtenida aplicando un filtro Laplaciano de Gauss. Con este método pueden reconocerse imágenes de objetos parcialmente ocultos, con diversa escala o girados. Este método no consume demasiada memoria sin embargo el tiempo de cómputo requerido es elevado, además no funciona con objetos pequeños en las imágenes.

2.2.3.6. Técnica de supercuadráticos

La técnica de supercuadráticos es utilizada para el reconocimiento de objetos y para modelado [11]. Los supercuadráticos consisten en una familia de figuras descritas paramétricamente, que para el caso de dos dimensiones (2D) están definidas como:

$$\vec{X}(\theta) = \begin{pmatrix} a_1 \cos\theta^{\epsilon} \\ a_2 \sin\theta^{\epsilon} \end{pmatrix} \tag{2.2}$$

donde $\vec{X}$ es un vector 2D que describe el contorno de la imagen de un objeto; θ es el ángulo de orientación; ϵ es el parámetro de la figura; a_1 y a_2 son parámetros de tamaño [12]. Esta función describe un amplio grupo de figuras geométricas tales como círculos, rectángulos y rombos además de figuras más complejas como elipses. Mediante la combinación de estas figuras básicas la técnica de supercuadráticos puede usarse para modelar un sinnúmero de objetos más complejos. Los supercuadráticos son utilizados también en reconocimiento tridimensional (3D) [53]. Los supercuadráticos son invariables ante variaciones de tamaño y orientación de los objetos. La técnica de supercuadráticos sin embargo tiene diversos problemas para reconocer objetos de geometría compleja como el caso de objetos o seres naturales u objetos mecánicos con muchas aristas.

2.2.3.7. Método de redes neuronales

Para el reconocimiento de objetos el método de redes neuronales es utilizado [15, 54, 55, 57, 65]. Algunas características muy útiles de las redes neuronales son su capacidad de aprendizaje adaptivo, flexibilidad y capacidad para reconocimiento en tiempo real. Existen algunos trabajos de reconocimiento de objetos aplicando como técnica básica las redes neuronales [13, 54, 57]. Por ejemplo, para la tarea de reconocimiento de objetos circulares del tipo MEMS, los autores Radjenovic y Detter utilizan las imágenes de los objetos generadas por una etapa de preprocesamiento como entradas para un clasificador neuronal. El mejor porcentaje de reconocimiento obtenido por ellos fue 97.7 % para tres clases de objetos. El tiempo de la etapa de preprocesamiento fue de 1.2 s y de 0.4 s para la etapa de reconocimiento. Los autores Mitzias y Mertzios [57] utilizan en su trabajo un clasificador multineuronal, éste consiste en tres redes neuronales trabajando en paralelo. Una etapa distinta de preprocesamiento es utilizada para cada una de éstas redes, así que cada clasificador trabaja con diferentes características de la imagen a ser procesada tales como brillo, textura o contornos. Para probar el método utilizan nueve objetos delgados e irregulares, todos con un agujero al centro. Este método requiere demasiados recursos de cómputo para lograr reconocimiento en tiempo real.

2.2.3.8. Extracción de propiedades

Diversos métodos utilizan una etapa de extracción de propiedades sobre los objetos que se desean reconocer. Esta etapa consiste en la extracción de propiedades predefinidas en las imágenes de los objetos o en las imágenes preprocesadas, como en el caso de imágenes de contornos. Las propiedades a extraer pueden ser de naturaleza geométrica, de textura, de color u otra [57]. Las propiedades más largamente empleadas en la literatura son las geométricas que incluyen momentos, descriptores de Fourier, análisis sintácticos y aproximación de polígonos [66].

2.2.4. Reconocimiento de posición

Un sistema ideal de manipulación de piezas debe identificar una pieza determinada en una posición no predefinida dentro de una cierta área de trabajo. El sistema debe resolver la posición de la pieza en sus seis variables, tres coordenadas lineales (x,y y z) y tres coordenadas angulares (w_x,w_y,w_z). Algunas publicaciones dan cuenta de los esfuerzos en la solución de esta tarea ya sea parcial o completamente [57, 65, 67]. A continuación se resumen los métodos de localización de objetos más reportados en la literatura. Entendemos por localización de piezas a la determinación de la localización espacial de un objeto dado dentro de un área de trabajo utilizando métodos de visión por computadora.

2.2.4.1. Métodos basados en redes neuronales

Un grupo de métodos para la localización de objetos utiliza redes neuronales artificiales. El principio general en el que se basan estos métodos se resume a continuación. Se utiliza una red neuronal para la posición de cada eje coordenado en que se desea ubicar un objeto. El caso general es emplear dos redes neuronales para utilizarlas sobre un plano (caso de 2D). Cada red neuronal es entrenada en intervalos fijos sobre el plano para cada posición del objeto a localizar. De esta manera el objeto podrá ubicarse en cualquier sector del plano y ser reconocido mediante sus coordenadas con una aproximación similar al tamaño del intervalo utilizado. La precisión de este método depende del intervalo o los intervalos escogidos así como de la arquitectura de la red neuronal particular que se utilice. Este método es eficiente para aproximar la posición pero para obtener buena precisión requiere aumentar la disponibilidad de recursos, es decir, reducir el intervalo de entrenamiento aumenta el número de éstos por lo que los recursos de computo necesarios (memoria y hardware) aumentan así como el tiempo necesario para su procesamiento. Dos trabajos que hacen uso de este principio son [16, 65].
Existe un inconveniente para este principio de localización de objetos en aplicaciones para micromaquinado y microensamble. El sistema debe ser entrenado con cada objeto en una gran cantidad de posiciones distintas. Lo que hace que la tarea de entrenamiento sea sumamente demandante en recursos, sobre todo en tiempo. Esto resta flexibilidad al sistema cada vez que una nueva pieza de trabajo necesite ser reconocida por el sistema.

2.2.4.2. Métodos basados en seguimiento de movimiento

Otro método de localización de objetos mediante visión por computadora es reportado en [67]. Este método trabaja con objetos en movimiento en tiempo real. El método puede trabajar con objetos en 3D y parcialmente ocultos. Este método requiere de dos cámaras (estéreo-visión) y para cada objeto a localizar de un

premodelado CAD de éste y de una gran cantidad de memoria para almacenamiento de sus imágenes de perspectivas generadas.
Este método requiere objetos en movimiento no quedando clara su eficacia con objetos en reposo en la literatura. Además los recursos requeridos en memoria de cómputo son demasiado elevados.

2.2.4.3. Técnicas de segmentación de imagen

Una técnica para localizar un objeto dentro de una imagen tiene relación con extraer el contorno de ésta y aplicar algoritmos de separación mediante seguimiento de los contornos. Con esto se consigue partir una imagen que contenga varios objetos en varias subimágenes. Con ésto se puede reconocer el objeto en un segmento determinado y aproximar así su posición dentro de la imagen original.

2.2.4.4. Reconocimiento de posición de piezas en una microfábrica

Un sistema de localización de piezas debe ser congruente con los principios de flexibilidad de una microfábrica. Debe poderse adaptar a nuevas piezas por lo que debe buscarse la reducción en el tiempo de procesamiento previo. Debe también ser capaz de trabajar con diversos tipos de piezas sin que pueda hacerse con estas alguna generalización más allá de un tamaño límite congruente con las capacidades de la microfábrica. Como se ha visto existen pocas investigaciones y desarrollos sobre esta tarea. Es un área con mucho potencial pues existen diversas aplicaciones para un sistema práctico y funcional de localización de objetos.
En el siguiente capítulo se propone un sistema práctico de manipulación de piezas para una microfábrica el cual tiene como parte medular un sistema basado en visión por computadora.

Capítulo 3

Sistema de visión por computadora para microensamble

Este capítulo describe el sistema de visión por computadora para microensamble propuesto. Dicho sistema forma parte importante de la microfábrica en desarrollo en el laboratorio de investigación del autor. En el capítulo anterior se expuso la necesidad de que dicho sistema de visión sea desarrollado como parte de las investigaciones sobre microfábricas. Se explicó por qué un sistema de este tipo debe de estar basado en visión por computadora. El presente trabajo tiene como objetivo fundamental desarrollar el sistema automático de visión por computadora de reconocimiento de piezas para una microfábrica. Para explicar lo anterior, la siguiente Sección (3.1) introduce el laboratorio donde se ha desarrollado este trabajo dando cuenta de sus objetivos y líneas de investigación. En la Sección 3.2 se explica a detalle el sistema de manejo de piezas propuesto y su función dentro de la microfábrica en desarrollo. Por último, en la Sección 3.3 se describe el sistema de visión técnica para reconocimiento y localización de piezas que se propone.

3.1. Investigación en el Laboratorio de Micromecánica y Mecatrónica

Desde 1998, el Laboratorio de Micromecánica y Mecatrónica (LMM) del Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET) de la Universidad Nacional Autónoma de México (UNAM), encabezado por el Dr. Ernst Kussul, trabaja en el desarrollo de microfábricas y las tecnologías relacionadas. Diversos artículos especializados dan cuenta de la producción y los resultados alcanzados [4, 5, 20, 21, 55, 68]. El LMM desde su constitución y hasta el momento ha tenido diversas líneas de investigación. Las principales son las siguientes:

- Desarrollo de microtecnología mecánica para aplicaciones en células de producción.
- Investigación y desarrollo de sistemas de control para micromáquinas herramientas y micromanipuladores.
- Investigación de sistemas de control basados en redes neuronales con aplicaciones en sistemas micromecánicos.
- Investigación y desarrollo de un sistema de visión computacional para microequipo.
- Investigación y desarrollo de manipuladores de baja inercia.

Las bases de estas investigaciones y desarrollos fueron planteadas en [25]. La propuesta general se denomina Tecnología de Microequipo o MET (por sus siglas en inglés). Esta propuesta se ha introducido en la Sección 2.1.4 y se ha hecho de acuerdo a lo expuesto en [69]. Mediante el uso de generaciones sucesivas de microequipo capaz de reproducirse así mismo, pero con dimensiones más pequeñas, se puede llegar en pocas generaciones a equipo de dimensiones micrométricas partiendo de equipo miniatura de unos cuantos centímetros de tamaño, véase Tabla 3.1.
Además del objetivo principal de crear otra generación más pequeña para cada una de las generaciones MET, existen muchas otras aplicaciones diversas. Estas aplicaciones tienen que ver con la posibilidad de producción de piezas y equipos diversos de dimensiones correspondientes a cada generación MET.

Tabla 3.1: Generaciones de microequipo de acuerdo a su escala de reducción.

	Escala de reducción (ER)		
Generación	ER=2	ER=4	ER=8
1	*100 mm	*100 mm	*100 mm
2	50 mm	25 mm	12.5 mm
3	25 mm	6.25 mm	1.5 mm
4	12.5 mm	1.5 mm	0.2 mm
5	6 mm	0.4 mm	25 μm
6	3 mm	0.1 mm	3 μm
7	1.5 mm	25 μm	△
8	0.8 mm	6 μm	—
9	0.4 mm	1.5 μm	—
10	0.2 mm	△	—
* Valor inicial igual para todos los casos.			
△ Límite teórico de micromaquinado.			

Las principales áreas tecnológicas de aplicación son la medicina, la microelectrónica y la industria espacial. Las posibilidades son muy alentadoras ya que no sólo se tendrá la capacidad de producir piezas y equipos de dimensiones micrométricas, sino de cualquier dimensión intermedia entre lo micro y lo meso.
Sin embargo, para crear la primera generación MET existen diversos problemas de ingeniería que es necesario resolver. A medida que nuevas generaciones sean creadas nuevos problemas asociados a sus dimensiones deberán enfrentarse debido a la miniaturización progresiva. Con el objetivo general de desarrollar el microequipo en el LMM se llevan a cabo diversas investigaciones. Estas investigaciones tienen que ver con diseño mecánico, precisión mecánica, ingeniería térmica, resistencia de materiales, interfaces, sistemas mecatrónicos e inteligencia artificial entre otros. Todas estas investigaciones son necesarias para alcanzar los objetivos mencionados para el LMM.
Como avance concreto en el objetivo general del LMM se ha desarrollado y caracterizado un microcentro de maquinado con dimensiones de $130 \times 160 \times 85 mm^3$. Este centro de maquinado ha permitido manufacturar objetos de entre $50\mu m$ y $5mm$. Estas piezas tienen geometrías tridimensionales complejas, como es el caso de tornillos, engranes y ejes graduados que han sido producidos. Estas piezas producidas son de materiales diversos como acero, plásticos varios o latón [5]. Adicionalmente se han diseñado y construido otros prototipos de la primera generación MET como es el caso de un par de manipuladores. El centro de maquinado y los manipuladores se muestran en la Figura 3.1.
Los conceptos MET y microfábrica están ampliamente relacionadas en la perspectiva del LMM ya que la viabilidad de esta tecnología depende de la capacidad de construir una microfábrica altamente automatizada de primera generación. Esta microfábrica deberá ser capaz de producir las piezas y luego las máquinas de una siguiente generación más pequeña como se expuso en la Sec. 2.1.4. Con el objetivo de lograr la automatización y la flexibilidad de la microfábrica se ha argumentado que una función muy importante es la de un sistema automático de manejo de piezas (Sec. 2.2.1). En la siguiente sección se expone la propuesta de dicho sistema enmarcando sobre éste los objetivos del presente trabajo.

3.2. Sistema automático de manejo de piezas

En una fábrica convencional los materiales y las piezas producidas pertenecen a un proceso masivo. Por lo general este proceso es poco o nada flexible durante el tiempo de producción. Esta forma de trabajo genera gran cantidad de productos de un mismo tipo siendo la forma dominante en el mundo desarrollado al día de hoy. El concepto de microfábrica no es solamente una fábrica de tamaño reducido, sino un nuevo concepto de producción [39]. En el concepto de producción de la microfábrica se busca hacer de la producción algo flexible. Lo anterior significa que los productos pueden variar en diseños y geometrías entre muchos otros aspectos en un mismo lote de producción. Lo anterior hace que el material suministrado así como los productos intermedios y terminados del proceso no sean fácilmente manejables mediante equipos rígidos especialmente adaptados a un sólo producto. Es por ello que un sistema automático de manejo de materiales y piezas debe ser desarrollado para la microfábrica. Independientemente de ello, un sistema así podrá ser utilizado además en las fábricas convencionales.

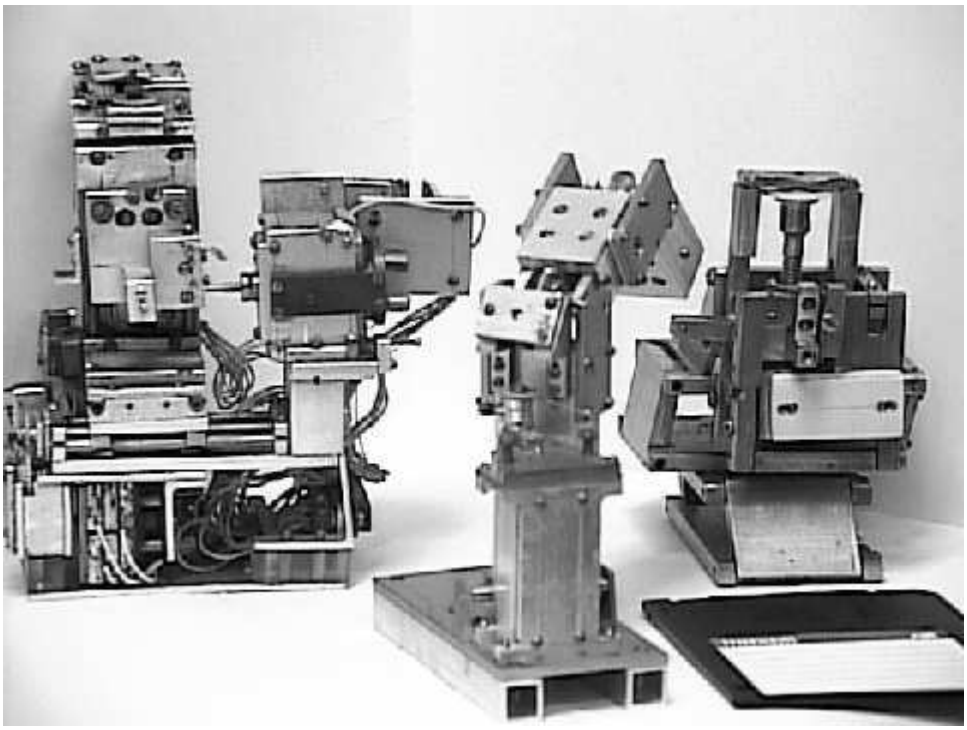


Figura 3.1: Máquinas de la primera generación MET. Izquierda, microcentro de maquinado. Centro, manipulador de topología común. Derecha, manipulador basado en paralelogramos. Un disco magnético de $3\frac{1}{4}$" se ubica en la imagen para dar cuenta de las dimensiones.

Diversas operaciones de una microfábrica requieren un sistema automático de manejo de materiales y piezas, entre las que sobresalen:

- El suministro y manejo de materia prima de estado sólido.
- La manipulación de los productos intermedios y finales.
- La manipulación de piezas y partes para el microensamble.
- El remplazo y cambio de herramientas.
- La manipulación y reemplazo de partes para labores de mantenimiento.

Dada la amplia variedad de materiales y objetos a ser manejados y a las funciones requeridas, un sistema de tales características deberá ser altamente flexible y deberá contar con algunas capacidades atribuibles a la inteligencia, como es el caso de la identificación de texturas para diferenciar el material o la determinación de la posición de ciertos objetos aleatoriamente ordenados con miras en su manipulación.
El problema descrito se aborda en el presente trabajo proponiendo un sistema automático de manejo de piezas enfocado a la tarea de reconocimiento de piezas aleatoriamente distribuidas. Un sistema capaz de resolver esta compleja tarea será fundamental para el desarrollo de los diversos sistemas de manipulación de materiales y partes para una microfábrica.
La meta de un sistema automático de manejo de piezas es que sea capaz de reconocer una determinada pieza aleatoriamente ubicada en una cierta área de trabajo además de su posición y orientación respectiva, de tal forma que pueda ser tomada con un manipulador para ser movida a alguna otra posición en el espacio según los requerimientos del proceso respectivo. Un sistema que cumpla con las expectativas mencionadas debe ser flexible, es decir, debe ser capaz de manejar piezas de diversas formas y con múltiples dimensiones.
Como solución al problema planteado en el presente trabajo se tiene un Sistema Automático de Manipulación de Piezas (SAMP). Este sistema se ha propuesta para formar parte en una celda de ensamble en una microfábrica. El sistema propuesto se muestra en la Figura 3.2. El SAMP está compuesto por dos cámaras, un manipulador, un sistema de control del manipulador (SCM), un sistema inteligente de manipulación (SIM) y un sistema de visión técnica (SVT).

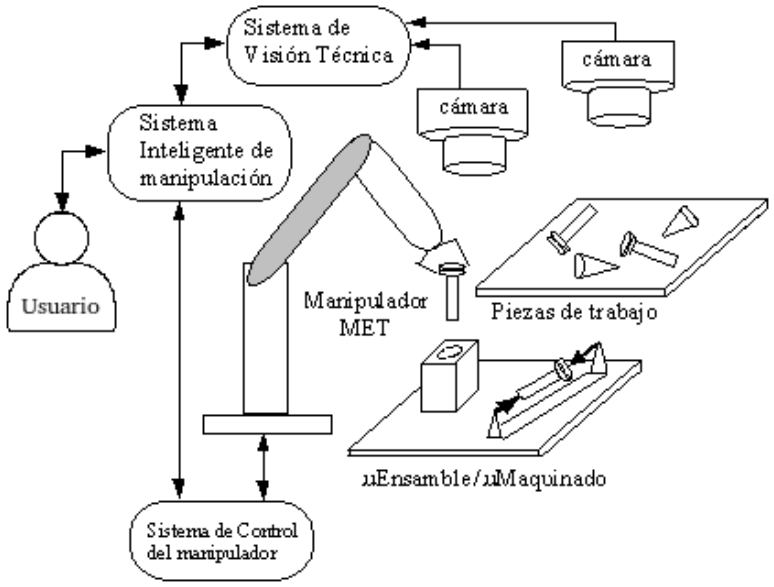


Figura 3.2: Sistema automático de manejo de piezas.

En congruencia con la MET, el diseño del SAMP utiliza componentes de bajo costo y de manufactura simple. La parte mecánica del sistema es el manipulador, el cuál debe ser diseñado con MET para permitir su consecuente decremento en dimensión, un diseño se ha propuesto en [4]. Las dos cámaras del diseño son cámaras *web* de bajo costo y baja resolución. Para la segunda y siguientes generaciones MET, estas cámaras deberán emplear lentes para reducir su área de enfoque sin deteriorar la calidad de imagen. El resto de los componentes, los tres subsistemas propuestos, son sistemas lógicos implementados mediante software y hardware. Estos subsistemas no requieren ser escalados en las sucesivas generaciones MET.

Cada uno de los tres subsistemas del SAMP tiene una función específica. El SIM es el control central del sistema de manipulación. Es el encargado de comunicarse con el exterior para recibir tareas y devolver información del estado de éstas así como del estado general del SAMP, también intercomunica los otros dos subsistemas. El SCM es la parte lógica del manipulador, es el responsable de convertir los requerimientos dados para el SIM en términos de coordenadas espaciales a el espacio de estados de la topología particular del manipulador así como realizar la planeación de trayectorias de éste. Convirtiendo los requerimientos en señales eléctricas para mover los actuadores del manipulador. El SCM también devuelve el estado del manipulador, tanto espacial (posición, orientación y estado del efector) como lógica (banderas de error, de éxito u otros estados). El SVT es la parte inteligente del SAMP, es el responsable de devolver una posición espacial ante el requerimiento de localizar cierto tipo de pieza en el área de trabajo. Esto es, dada la necesidad de manipular una o más piezas, el SIM pide al SVT buscar éstas en el área de trabajo y devolver la posición de cada una de ellas para que éstas puedan ser requeridas físicamente al manipulador mediante el SCM. El usuario del SAMP puede ser un humano u otro sistema, e.g. sistema de manufactura o centro de control de la microfábrica.

Existen dos tareas comunes a ser resueltas por el SAMP. Una es colocar un cierto tipo de pieza ubicada en el área de trabajo en una micromáquina herramienta, la otra tarea es realizar un ensamble sencillo utilizando dos piezas del área de trabajo. Para la primera tarea el SIM pedirá al SVT localizar la pieza solicitada, en caso de que exista alguna de estas piezas en el área de trabajo, el SVT deberá devolver al SIM la ubicación espacial de la misma. El SIM enviará entonces esta posición y orientación al SCM junto con los datos de destino para la pieza. La posición destino puede ser conocida por el SIM o no, según ésta sea de una máquina previamente ubicada o no. La segunda tarea es más complicada ya que requiere que el manipulador tenga capacidad para ensamble, y de no conocerse la geometría exacta de las piezas a ensamblar, se requerirá de un sistema de visión por computadora basado en la propuesta dada en [16]. Esta segunda tarea requerirá además que la tarea particular de ensamble este previamente programada de modo que el manipulador pueda colocar la primera pieza en la posición y lugar correctos para posteriormente ensamblarle la segunda pieza. Una vez que se logre desarrollar un sistema capaz de realizar la segunda

tarea, nada impide que éste pueda realizar ensambles con más piezas.
Dadas las características y requerimientos funcionales de cada uno de los sistemas que componen el SMAP, en el presente trabajo se ha desarrollado el SVT, ya que es el sistema que propone más retos y mayor dificultad, además de constituir una tarea que no está resuelta satisfactoriamente según se revisó en la Sección 2.2.3.
Dados los grandes retos que presenta este sistema, para realizar las investigaciones orientadas al desarrollo del SVT se procuró aislar el problema de los demás subsistemas y componentes del SAMP sin perder la problemática fundamental que se presenta. Por esto, el trabajo se hace sobre la parte del SAMP que contiene el SVT y una sola cámara además de su área de trabajo. Es decir, este trabajo está basado en la tarea de reconocimiento de piezas y sus posiciones en ambiente de microfábricas. En la siguiente sección final de este capítulo se aborda en detalle el SVT.

3.3. Subsistema de visión técnica (SVT)

En las Secciones 2.1.6 y 2.2.4.4 se han descrito los requerimientos generales para el Sistema de Visión Técnica (SVT) para una microfábrica. Se tiene que un SVT para localización de piezas debe ser congruente con los principios de flexibilidad de una microfábrica. Para lo cual debe ser capaz de trabajar con diversos tipos de piezas, además adaptarse a nuevas piezas en poco tiempo por lo que debe minimizarse el tiempo de procesamiento previo requerido para el reconocimiento de nuevas piezas.
Como se explicó en la sección anterior, el propósito del SVT es reconocer objetos y la posición de éstos. Este sistema está compuesto de tres elementos además de la cámara. Estos elementos son una interfase, un localizador de objetos y un controlador para la cámara (véase Fig. 3.3). La Interfase es la unidad que se comunica con el exterior, el SAMP (Véase Fig. 3.2) además de intercomunicarse con los otros dos elementos del sistema. A través de la Interfase el SVT recibe ordenes y envía información. El localizador de objetos es el elemento principal del SVT, su función es identificar y localizar un objeto requerido en el campo de visión de la cámara, devolviendo sus coordenadas. El controlador de la cámara se encarga de capturar imágenes del área de trabajo cuándo así lo requiera el localizador de objetos.

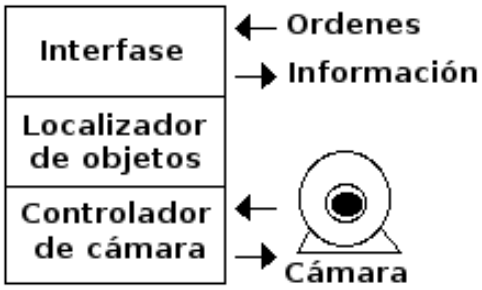


Figura 3.3: Sistema de Visión Técnica.

El objetivo es encontrar las coordenadas de las piezas con una precisión aceptable para su posterior manipulación. Esta tarea debe ser eficiente en cuanto al uso del tiempo y el uso de recursos (equipo, memoria y tiempo de procesador, etc.), de tal manera que el sistema pueda ser utilizado en línea de producción (*on-line*) además de ser económico. En la Fig. 3.4 se muestra el resultado esperado por el SVT ante la solicitud de localizar un tornillo dentro del área de trabajo. Se busca que el SVT localice el primer objeto del tipo requerido y devuelva su posición y su orientación. Lo ideal es que esto se haga respecto al centro de gravedad de la pieza. La posición y orientación deben tener precisión tal, de forma que permitan la manipulación de la pieza mediante un robot.
Para el desarrollo del SVT han sido hechas las siguientes consideraciones sobre las piezas a reconocer y a ubicar.

- Se encuentran en un área de trabajo igual al campo visual de la cámara.
- Se localizan en un mismo plano[1].

[1]Esta consideración no es del todo cierta para una de las bases de datos utilizadas, ya que las piezas están amontonadas y por ende, en distintos planos; sin embargo, la distancia entre éstos es relativamente pequeña.

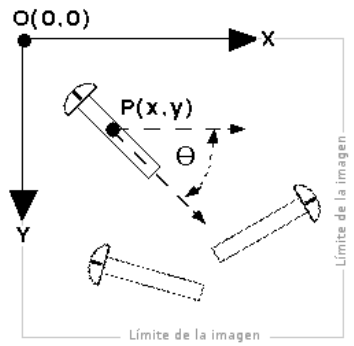


Figura 3.4: Ejemplo de un tornillo reconocido. El sistema devuelve las coordenadas (x, y, θ) con respecto del origen de la imagen.

- Están aleatoriamente distribuidas y orientadas.
- Pueden estar juntas e incluso una sobre otra, este caso no es general pero fue considerado para los experimentos.
- Tienen sección transversal predominantemente redonda, plana o una combinación de ambas características (en un caso), esto facilita la tarea de reconocimiento ya que piezas así dan por lo común una vista superior igual sin importar su rotación respecto de su eje principal. La vista superior es importante porque es la perspectiva de la cámara del SVT.

El SVT se probó con diversos conjuntos de piezas reales sin ningún tipo de tratamiento especial. Las piezas escogidas son predominantemente mecánicas las cuales se encuentran comúnmente en una línea de producción. Se utilizaron piezas tales como tornillos de diversas clases, tuercas, arandelas y otras piezas maquinadas. En la Fig. 3.5 se muestran algunas de las piezas utilizadas. En congruencia con un ambiente de ensamble y producción, estas piezas en ocasiones presentan superficies sucias, tienen colores oscuros, brillo heterogéneo y a veces tienen sombras. Todas las anteriores son características que complican la tarea de reconocimiento.

Figura 3.5: Algunas de las diversas piezas que se utilizaron en las investigaciones y experimentos al desarrollar el SVT.

El siguiente capítulo se dedica a la parte medular del SVT, que es el sistema de localización de piezas. Se describirán las bases de datos de imágenes utilizadas para la experimentación y se explicarán los métodos utilizados para la investigación y desarrollo del sistema identificador de piezas además de explicar el método particular de localización propuesto.

Capítulo 4

Diseño del SVT

En los capítulos anteriores se han dado los antecedentes sobre microtecnología y microfábricas. Se ha explicado la importancia de desarrollar y aplicar tecnologías de visión por computadora para la automatización de estas microfábricas. Se ha propuesto también un sistema automático de manejo de piezas cuyo elemento principal es un SVT, este sistema es el objetivo principal de este trabajo. Este capítulo aborda el diseño del SVT y está ordenado de la siguiente manera. En la siguiente sección se discute el principio de operación del SVT. Luego se justifican y describen las piezas utilizadas para los experimentos además de las características de adquisición de imágenes del sistema. Con los anteriores elementos se pasa a justificar los clasificadores seleccionados para los experimentos dando resultados previos y concretos de éstos. Después se describen las bases de datos utilizadas para las pruebas con los clasificadores. Por último se aborda el método de localización de objetos que se ha propuesto y aplicado en este trabajo.

4.1. Principio de operación

Como se ha explicado, la tarea primordial del SVT es el reconocimiento de piezas ubicadas en una cierta área de trabajo. El principio para resolver esta tarea es el uso de técnicas de visión por computadora (Sec. 2.2.2). Estas técnicas consisten por lo general en el uso de clasificadores lineales o no lineales dependiendo de la complejidad de la tarea. Un clasificador es un sistema capaz de agrupar o catalogar las entradas que recibe a través de ciertos patrones comunes pertenecientes en los elementos del conjunto a catalogar [70]. A cada grupo distinto se le llama clase. Para nuestra tarea particular el clasificador tendrá como objetivo reconocer piezas ubicadas en un área de trabajo determinada.
Como se ha visto en la Sección 3.3, el reconocimiento de piezas en el área mencionada se hace mediante el procesamiento de una imagen que del área de trabajo captura la cámara situada sobre ella. Esto significa que el reconocimiento sobre el área de trabajo y las piezas que allí se encuentren se realizará realmente sobre una imagen que la cámara capture de dicha área. De esta forma los procesos requeridos se harán sobre una imagen 2D de una escena 3D, específicamente de una toma en vista superior de la escena. La cámara utilizada se ubica a una distancia fija sobre el área de trabajo de tal manera que su campo de visión corresponda al área de trabajo. El objetivo del SVT se compone de dos tareas: el reconocimiento de piezas y la localización de éstas.
De los dos párrafos anteriores se deduce la necesidad de seleccionar un clasificador para la tarea de reconocimiento de piezas a través de imágenes además de elegir algún método para la localización de estas piezas. Dado que no se puede localizar algo sin saber lo que se busca, la primer y más importante tarea a resolver es la de reconocimiento de piezas. La segunda tarea que se plantea es la de la localización de piezas con base en la capacidad de reconocimiento previo de éstas.
La siguiente sección describe las piezas utilizadas y las características de su representación, es decir, de las imágenes que de ellas captura el SVT.

4.2. Piezas seleccionadas para experimentos

Para el desarrollo y experimentación con el SVT se utilizaron diversos ejemplares de varios tipos de piezas mecánicas. La mayoría de las piezas seleccionadas son piezas convencionales utilizadas para sujeción, e.g. tornillos. Las características generales de las piezas seleccionadas se describen a continuación:

Materiales y colores similares. Las piezas seleccionadas son metálicas excepto una que es de plástico gris. Todas tienen tonalidades metálicas oscuras que van del gris claro al negro. Varias piezas tienen colores similares entre si. Si bien ninguna pieza está pulida, todas presentan en su superficie algún grado de brillantez. Estas características implican que el trabajo de reconocimiento de piezas se ayude de las diversas tonalidades pero ésto no pueda ser un factor determinante como sería el caso de piezas de colores distintos entre si.

Tamaño diverso. El tamaño de las piezas seleccionadas está entre 28.8 mm y 4.2 mm. Se cuidó que algunas piezas del conjunto tuvieran tamaños muy similares entre si para conservar la complejidad del reconocimiento. El uso de piezas de diverso tamaño es normal en un ambiente de trabajo real sin embargo en la bibliografía consultada para reconocimiento de objetos mediante imágenes se utilizan por lo común objetos de tamaño muy similar (Sec. 2.2.3). Si bien este hecho complica la tarea de reconocimiento, se aleja de una aplicación práctica para microfábricas, restando flexibilidad al sistema.

Diversa forma. Esta característica es la más común en los experimentos reportados en la literatura sobre el problema de reconocimiento; de otra forma la clasificación debiera hacerse en base a texturas o color, lo cuál facilitaría la tarea de reconocer piezas de igual forma pero diversa superficie. En la selección hecha las piezas varían de forma igual que lo hacen en tamaño y color, es decir varían bastante en lo general pero existen casos de similitud.

Piezas de sección circular o con partes planas. Todas las piezas seleccionadas tienen sección transversal circular o algún grado de planitud[1]. Esto hace que la posición aleatoria de las piezas sobre una superficie plana (el área de trabajo del SVT) adquieran siempre una vista superior similar. Esta característica lleva la tarea de reconocimiento a un problema bidimensional (2D). Si bien esta propiedad es limitante en cuanto a piezas con estructuras complejas en 3D, la amplia gama de piezas mecánicas con estas características es suficiente para dar trascendencia teórica y práctica al trabajo desarrollado. Además considérese que la tecnología MEMS es capaz de producir mayoritariamente estructuras planas [27].

4.3. Las imágenes del SVT

En atención a que el trabajo de reconocimiento (clasificación) y localización se realiza en realidad sobre las imágenes de las piezas y no directamente sobre éstas, y una vez que se han presentado las características de las piezas utilizadas, se presentan en esta sección las características de la adquisición de estas imágenes. Como se mencionó en la Sección 3.2, el SVT está planteado para seguir los principios de la tecnología MET. Dos de tales principios son el bajo costo y la capacidad de implementación a escalas reducidas. Es por ésto que el sistema de digitalización del área de trabajo se hace con una cámara de bajo costo de tecnología CCD[2]. A continuación se listan las características de las imágenes con que trabaja el SVT.

Escala de grises. Todas las imágenes capturadas son convertidas a escala de grises para reducir el espacio de almacenamiento de éstas y por ende el tiempo y complejidad del procesamiento. Si bien el color puede ser una ventaja para la clasificación, se ha explicado que las piezas tienen tonalidades similares por lo que utilizar color daría muy pocas ventajas contra la gran cantidad de recursos adicionales que esto requeriría.

Iluminación no especial. La iluminación utilizada por el SVT no es especial ni regulada. La iluminación empleada ha sido la adecuada para trabajo humano con distinción moderada de detalle (aproximadamente 300 luxes)[3]. El único requerimiento del SVT es que la misma iluminación sea utilizada en todo momento por el sistema.

Brillos y sombras. Debido al tipo de materiales utilizados por las piezas seleccionadas éstas presentan brillo sobre sus superficies además de sombras. Estas características complican la tarea de reconocimiento pero a su vez plantean un problema práctico en el cuál el preprocesamiento de extracción de bordes encuentra algunos problemas.

Fondo no uniforme La superficie utilizada en el área de trabajo del SVT se ha buscado de un color contrastante con el de las piezas utilizadas y al mismo tiempo que su color y superficie no sean

[1] Si bien esta palabra no está en el diccionario es usada comúnmente en el lenguaje técnico como una medida del grado en que algo es plano.

[2] Siglas de Couple-Charged Device (Dispositivo de cargas (eléctricas) acopladas).

[3] Norma Oficial Mexicana NOM-025-STPS-1999, Condiciones de iluminación en los centros de trabajo.

reflectantes. Sin embargo las características de iluminación y de sombras de las piezas generan en los hechos una superficie no uniforme. Esto se traduce en una imagen cuyo fondo no es uniforme.

Las condiciones anteriores existen en los ambientes reales de trabajo en mecánica y micromecánica. Sin embargo estas condiciones reales complican la tarea de reconocimiento en el SVT lo que constituyen un reto en el desarrollo del sistema, sobre todo en lo concerniente a clasificación de imágenes.
Con los elementos discutidos hasta aquí se está en condiciones para discutir y justificar los clasificadores utilizados para los experimentos hechos con el fin de resolver esta tarea. La siguiente sección aborda este asunto.

4.4. Selección de clasificadores

Las características y potencial de las redes neuronales en tareas de reconocimiento de imágenes han sido ampliamente señaladas [71, 72]. Las redes neuronales son sistemas altamente paralelos, adaptivos, poseen estructuras no lineales y tienen buenas capacidades de generalización. Las redes neuronales han sido utilizadas en muchas aplicaciones de análisis de señales como sensado remoto, visión robótica, reconocimiento de voz, tareas de sensado de fusión y muchas otras [49, 73, 74].
En la investigación desarrollada en este trabajo con miras al desarrollo del SVT se han utilizado dos métodos basados en redes neuronales ya que estos métodos han sido aplicados con éxito a diversos problemas de reconocimiento de imágenes. Los métodos utilizados se llaman LIRA y PCNC. Estos métodos son producto del trabajo de los investigadores Ernst Kussul y Tatiana Baidyk [15, 18, 75–79]. Estos métodos han sido desarrollados con la idea de resolver diversas tareas de reconocimiento de imágenes (e.g. reconocimiento de caracteres escritos, de rostros, de texturas y de formas de objetos). Ambos métodos están basados en el perceptrón de una capa de Rossenblatt [14] y en un extractor de propiedades de imágenes de propósito general, este último está basado en descriptores locales existentes en la corteza cerebral de animales [80]. Ambos clasificadores utilizados en el presente trabajo han demostrado su fiabilidad en las tareas de reconocimiento de caracteres escritos y de rostros humanos. Por esto a continuación se describen dos bases de datos populares para luego presentar los resultados obtenidos al aplicar los clasificadores LIRA y PCNC sobre éstas.
La base de datos MNIST [81] fue creada para comparar diversos métodos de reconocimiento de caracteres manuscritos [82]. Ésta contiene 60 000 muestras para entrenamiento y 10 000 para prueba. Desde su publicación han sido probados muchos algoritmos de reconocimiento y los errores de estos varían entre 1.0 % y 0.4 % [15, 83].
La base de datos de rostros del Laboratorio de Investigación Olivetti (ORL, por sus siglas en inglés) [84] contiene 400 imágenes de 40 personas distintas, esto es 10 imágenes por persona. Las diferencias entre las imágenes de una misma persona consisten en desplazamientos de la cabeza o inclinación de esta, expresiones faciales distintas y de presencia o ausencia de lentes. Como regla cinco imágenes de cada persona son usadas para entrenamiento y las otras cinco para prueba, debiéndose seleccionar aleatoriamente para que los resultados sean confiables estadísticamente.

4.4.1. Resultados previos del clasificador LIRA

El clasificador LIRA mostró su capacidad inicial en la tarea de reconocimiento de caracteres manuscritos. Ha sido reportada una tasa de reconocimiento de 99.6 % para la base de datos MNIST, en su momento ese resultado fue el segundo mejor conocido para tal base de datos. Un año después el clasificador LIRA fue mejorado y sobre la misma base de datos MNIST obtuvo un resultado de 99.45 % [15]. Además, el clasificador neuronal LIRA se probó en un problema de microensamble [16, 55]. Este clasificador se aplicó al reconocimiento de la posición relativa en un par agujero-aguja. Para este propósito fue creada una base de datos con 441 imágenes. Se agruparon las imágenes con diferentes posiciones de la aguja con intervalos de 0.1mm sobre los ejes X y Y. Se utilizó una cámara web y cuatro fuentes de luz de modo que éstas proporcionasen cuatro sombras de la aguja. Estas cuatro sombras junto con el par aguja-agujero permitieron deducir la posición tridimensional entre el par aguja-agujero. La salida del clasificador LIRA en este caso fueron las coordenadas (x, y). Se obtuvo una tasa de reconocimiento de 99.5 % para las abscisas y de 89.9 % para las ordenadas con una precisión de 0.1 mm. Este resultado significa una muy buena tasa de reconocimiento [16].
Luego de haber obtenido estos resultados alentadores se decidió probar y aplicar el clasificador neuronal LIRA al problema de reconocimiento de objetos, que para la tarea particular de este trabajo son piezas mecánicas.

4.4.2. Resultados previos del clasificador PCNC

El clasificador PCNC ha sido propuesto en [85, 86]. Este clasificador se probó en la tarea de reconocimiento de caracteres escritos con la base de datos MNIST dando una tasa de error de 0.37 % [76]. Se probó también en la tarea de reconocimiento de rostros con la base de datos ORL obteniéndose una tasa promedio de error de 0.1 % para 10 experimentos.
El PCNC ha sido probado también en reconocimiento de microobjetos [77]. Para ello se produjeron 40 tornillos de 3mm de diámetro con un torno CNC. Diez tornillos fueron producidos correctamente y 30 con errores deliberados mediante el cambio en la posición del cortador del torno. Esta posición fue de 0.1mm menos que lo correcto para diez tornillos, 0.1 mm más que lo correcto para otros diez tornillos y los restantes diez se produjeron con una distancia de 0.2 mm. Las imágenes de las cuerdas de estos se capturaron con una cámara web y se creó con estas una base de datos. De cada uno de los cuatro grupos de tornillos (1 correcto y 3 con errores), se tomaron en forma aleatoria cinco imágenes para entrenamiento del clasificador y 5 para prueba. Diversos experimentos se hicieron con diferentes parámetros del clasificador. La mejor tasa de reconocimiento de estos experimentos fue de 92.5 % [86].
Dado los resultados anteriores del PCNC en tareas de micromecánica se ha aplicado este en el presente trabajo a la tarea de reconocimiento de piezas.
Los detalles de los clasificadores LIRA y PCNC así como los experimentos con ellos se dan en los siguientes capítulos.
A continuación se pasa a describir las características de las bases de datos de imágenes de piezas utilizadas para la experimentación y prueba de los clasificadores seleccionados.

4.5. Bases de datos de imágenes utilizadas

Con miras a lograr el objetivo propuesto de reconocimiento de piezas, los clasificadores desarrollados requieren trabajar con gran cantidad de ejemplos de imágenes de los objetos a reconocer. Se requieren dos grupos de estas imágenes. Un grupo es necesario para el entrenamiento del clasificador y el otro grupo es necesario para probarlo. Dado lo anterior y las piezas seleccionadas se crearon bases de datos de imágenes. De cada base de datos se forma un grupo de imágenes para entrenamiento y otro para prueba.

4.5.1. Imágenes normalizadas

Cada imagen que compone una base de datos, ya sea para entrenamiento o para prueba, ha sido normalizada de manera que el clasificador pueda trabajar con ella. Por normalización se quiere dar a entender una serie de características iguales para todas las imágenes. Estas características se dan a continuación con la justificación correspondiente.

Características de las imágenes del SVT. Las imágenes de las bases de datos han sido extraídas de imágenes tomadas por el SVT, de tal forma que las imágenes de entrenamiento y prueba coincidan con las imágenes de trabajo. Por esto las características de las imágenes del SVT descritas en la Sección 4.3 son las mismas que para las imágenes de las bases de datos. Dichas características son: imágenes en escala de grises, imágenes sin iluminación especial, imágenes con brillos y sombras e imágenes cuyo fondo no es del todo uniforme.

Pertenencia. Cada imagen de la base de datos contiene una pieza centrada, esto con independencia de lo que exista adicionalmente, como lo pueden ser otras piezas. La excepción a esta característica se da para un grupo de imágenes, premeditadamente creadas, que no contienen ninguna pieza en el centro como se explicará más adelante.

Imágenes centradas y con orientación fija. Dado el eje mayor de simetría de cada pieza[4], su centro se define como el punto medio de este eje. El centro de la imagen coincide entonces con el centro de la pieza correspondiente. Además, el eje de simetría de la pieza se orienta a cero grados con respecto a la imagen. Para diferenciar entre 0° y 180° el eje de simetría de las piezas debe definirse como vector y no como segmento de recta, esto es, el punto inicial deberá tomarse como "base" de la pieza y el punto final como "punta o cabeza" de la pieza (piénsese en un tornillo, un poste o un machuelo por ejemplo). En realidad, el centrado y orientación de las imágenes no es exacto porque lo realiza un software con base en marcas hechas por el usuario (véase Sec. A.1).

[4]Todas las piezas utilizadas en este trabajo son simétricas en la vista de trabajo (vista superior).

Tamaño fijo. Las imágenes tienen un tamaño fijo y cuadrado de 100 o 150 píxeles por lado, según la base de datos particular. Dado esto, las imágenes deben poder contener a la pieza más grande existente con que se desee trabajar (reconocer y localizar). Además, a mayor dimensión de la imagen mayor es la resolución y el detalle de las piezas, pero mayores son también los requerimientos de memoria de almacenamiento y tiempo de procesamiento que necesitan estas imágenes. Luego de lo anterior y de probar varios valores para la primer base de datos se eligió el de 150 píxeles por lado (22 500 píxeles por imagen). Después se utilizo con éxito el tamaño de 100 píxeles por lado (10 000 píxeles por imagen) para las demás bases de datos. Las imágenes son cuadradas por que esto está de acuerdo con la forma natural de almacenamiento de imágenes que es rectangular y porque dicha forma facilita la rotación de las imágenes, como se explicará adelante en esta misma sección.

Baja resolución. De acuerdo a la naturaleza de las piezas empleadas que tienen detalles finos (e.g. cuerdas en los tornillos) y de lo escrito en el punto anterior, las imágenes tienen baja resolución. Esto facilita el procesamiento y no afecta el reconocimiento del SVT el cuál no se hace en base a detalles, sino a las características contrastantes de las diversas piezas a reconocer (forma, tonos, etc.).

Ventana circular. Un algoritmo de ventana circular se ha aplicado a las imágenes cuadradas. Esta ventana se centra en la imagen y tiene por diámetro la dimensión lateral de la imagen. Las orillas que quedan fuera de la ventana se hacen negras. Este procedimiento es simple en la implementación pero consume algunos recursos computacionales por lo que debe ser justificado. Al estar la pieza correspondiente centrada en la imagen, la aplicación del algoritmo no afectará a la imagen de la pieza y en cambio facilitará que esta imagen sea rotada cualquier ángulo sin cambio en los límites de la imagen. La aplicación de esta ventana contribuye a la uniformidad de las imágenes, eliminando así el análisis de información cambiante pero inútil en los bordes y esquinas. Este párrafo se resumen en la Fig. 4.1.

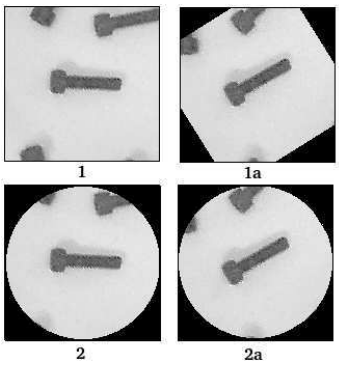


Figura 4.1: Ventajas de aplicar una ventana circular a las imágenes de piezas. 1) Imagen de pieza centrada sin ventana. 1a) Resultado de rotar 32° la imagen 1. 2) Imagen de pieza centrada con ventada circular. 2a) Resultado de rotar 32° la imagen 2.

Piezas no aisladas. Las piezas en las imágenes no siempre están aisladas. De acuerdo a los requerimientos del SVT esta característica es indispensable para poder trabajar con escenarios reales con piezas aleatoriamente localizadas en donde comúnmente existen piezas juntas o amontonadas.

4.5.2. Descripción de las bases de datos

Una vez descritas y explicadas las características que tienen las imágenes, se describen las bases de datos empleadas para los experimentos con los clasificadores elegidos. Cuatro bases de datos de imágenes fueron creadas. Una con cuatro tipos de piezas, otra con siete y dos con seis. Cada base de datos contiene un tipo adicional llamado "no pieza", el cuál en lugar de contener una pieza centrada, contiene fondo de imagen. Esto se hace con el fin de que el clasificador sea capaz de diferenciar entre alguna de las piezas y el fondo, es decir, que el clasificador tome en cuenta la posibilidad de que ninguna pieza exista en el centro de la imagen. Con la finalidad de hacer referencia posterior a las bases de datos mencionadas cada una de ellas se nombra con una letra. La primer base de datos se nombra con la primera letra griega mientras que las demás se harán con letras comunes mayúsculas, la razón de esto que es la primera base de datos sólo se utilizó para

experimentos preliminares del clasificador LIRA, mientras que las otras se han utilizado ampliamente en los experimentos del Cap. 7. A continuación se da la descripción de cada una de ellas.

La base de datos α está formada con cuatro tipos de piezas, utiliza cinco clases, cuatro correspondiente a los tipos de pieza y la quinta clase es de tipo "no pieza". Esta base de datos contiene 30 imágenes para cada una de las cinco clases distintas (150 imágenes), 15 para el conjunto de entrenamiento y 15 para el conjunto de prueba. En la Fig. 4.2 se muestran ejemplos de las imágenes de esta base de datos.

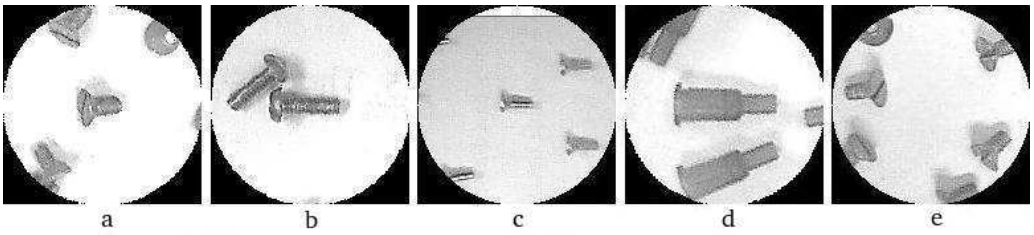


Figura 4.2: Ejemplos de imágenes para las cinco clases de la base de datos α: a) tornillo I, b) tornillo II, c) base de tubo, d) cono y e) ausencia de pieza en el centro de la imagen.

La base de datos A está formada con siete tipos de piezas, contiene 40 imágenes para cada una de las ocho clases diferentes utilizadas (320 imágenes), 20 para el conjunto de entrenamiento y 20 para el conjunto de prueba. En la Fig. 4.3 se muestran ejemplos de los elementos de esta base de datos.

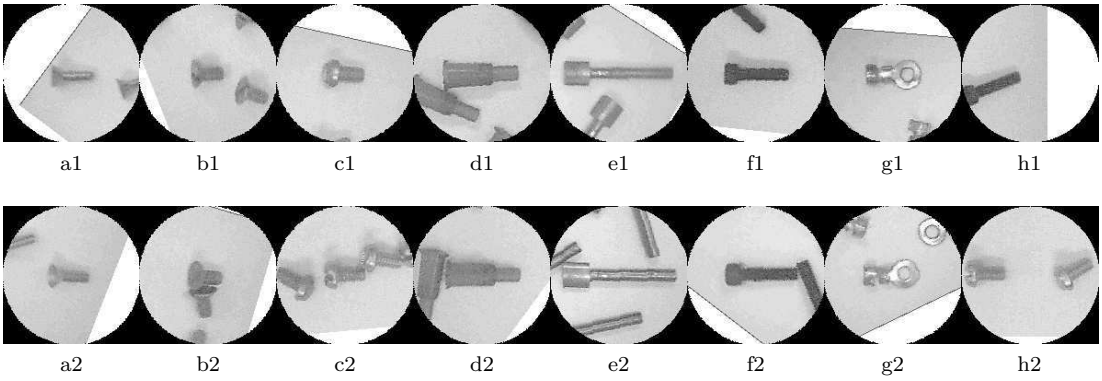


Figura 4.3: Ejemplos de las ocho distintas clases en la base de datos A. Se dan dos ejemplos para cada clase: a) base de tubito; b) tornillo de cabeza cónica; c) tornillo de cabeza redonda; d) cono; e) eje de estator; f) tornillo allen; g) terminal de cable; h) ausencia de pieza en el centro de la imagen.

La base de datos B está formada con seis tipos de piezas, contiene 78 imágenes para cada una de las siete clases diferentes utilizadas (546 imágenes en total), 39 imágenes para el conjunto de entrenamiento y 39 para el conjunto de prueba. En la Fig. 4.4 se muestran dos ejemplos de cada clase contenida en esta base de datos.

La base de datos C fue experimental, no se reporta por no haberse conseguido ningún aporte al trabajo con ella.

Una base de datos D con características especiales respecto a las anteriores fue creada. Esta base de datos utiliza las mismas piezas que la base de datos B, estando formada con seis tipos de piezas. Las imágenes normalizadas de la base de datos D contienen piezas amontonadas, en algunos casos ligeramente ocluidas, ligeramente inclinadas respecto al plano de la imagen y la característica más importante, las piezas están revueltas. Esta base de datos constituirá un reto para los clasificadores que en ella sean probados. Esta base de datos contiene 55 imágenes para cada una de las siete clases diferentes utilizadas (385 imágenes en total). Debido a que el número de imágenes por clase no es muy grande respecto a los otros ejemplos descritos, se utilizó un total de 70 % del total de imágenes para entrenamiento y el resto, 30 % para prueba. En la Fig. 4.5 se muestran dos ejemplos de cada clase contenida en esta base de datos. Los ejemplos de la

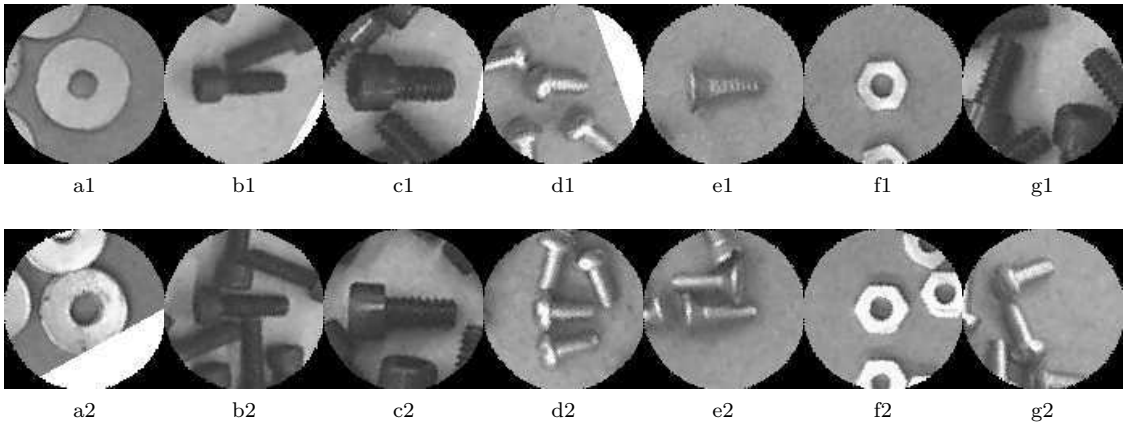


Figura 4.4: Ejemplos de las siete distintas clases de la base de datos B. Se dan dos ejemplos para cada clase: a) arandela; b) tornillo allen chico; c) tornillo allen grande; d) tornillo de cabeza de gota; e) tornillo de cabeza plana; f) tuerca; g) ausencia de pieza en el centro de la imagen.

primera fila se han escogido de entre las muestras que para el ojo humano se perciben como más sencillas, mientras que los de la segunda fila se eligieron de entre aquellas muestras que se percibieron como más complicadas.

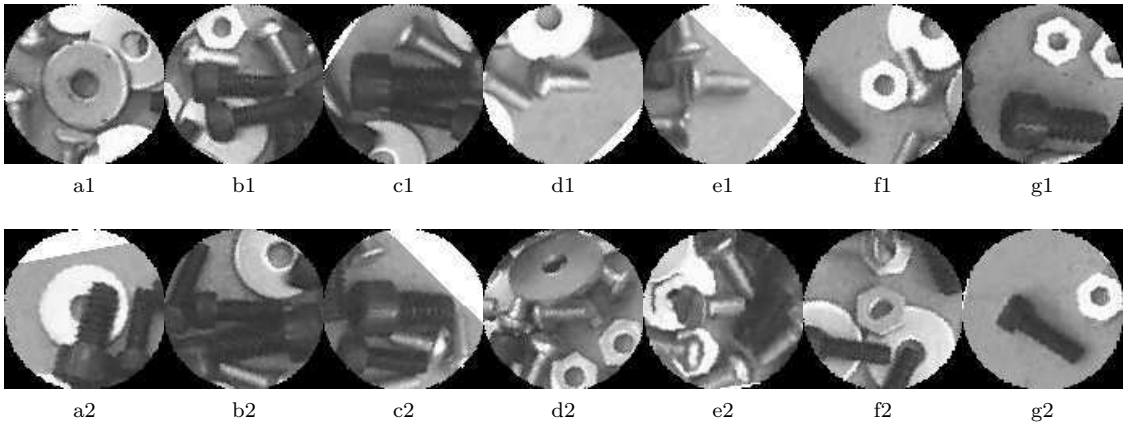


Figura 4.5: Ejemplos de las siete distintas clases de la base de datos D. Se dan dos ejemplos para cada clase: a) arandela; b) tornillo allen chico; c) tornillo allen grande; d) tornillo de cabeza de gota; e) tornillo de cabeza plana; f) tuerca; g) ausencia de pieza en el centro de la imagen.

Las imágenes para los grupos de entrenamiento y prueba fueron seleccionadas aleatoriamente de entre toda la base de datos, para lo cuál se utilizó un algoritmo pseudoaleatorio basado en C++. Tanto el grupo destinado a entrenamiento como el destinado a prueba tienen el mismo número de elementos para todas las bases de datos, excepto para la D, de tal forma que la probabilidad de que una imagen de la base de datos pertenezca a uno u otro grupo es la misma. Con motivo de esta aleatoriedad en la formación de los grupos para entrenamiento y prueba los resultados que sean obtenidos por los clasificadores serán estadísticamente confiables.

En la Tabla 4.1 se presenta un resumen de las principales características de las bases de datos utilizadas y en la Tabla 4.2 se enumeran las clases de cada una de las bases de datos empleadas.

Con base en la existencia de un clasificador previamente probado y entrenado en la tarea de reconocimiento de ciertas piezas de interés, se explica en la siguiente y última sección de este capítulo el método utilizado para la solución del problema de localización de piezas.

Tabla 4.1: Resumen de características de las bases de datos utilizadas.

Base de datos:	No. de clases	Imágenes por clase	Total de imágenes	Conjunto de entrenamiento		Conjunto de prueba	
				Imágenes	%	Imágenes	%
BD-α	5	30	150	75	50	75	50
BD-A	8	40	320	160	50	160	50
BD-B	7	78	546	273	50	273	50
BD-D	7	55	385	280	72.72	105	27.28

Tabla 4.2: Las clases de las bases de datos utilizadas (orden alfabético).

	Base de datos		
Clase no.	α	A	B y D
1	base de tubo	base de tubito	arandela
2	cono	cono	no pieza central
3	no pieza central	eje de rotor	tornillo allen chico
4	tornillo I	no pieza central	tornillo allen grande
5	tornillo II	terminal de cable	tornillo gota
6	-	tornillo allen	tornillo plano
7	-	tornillo cabeza cónica	tuerca
8	-	tornillo cabeza redonda	-

4.6. Localización de piezas

La literatura especializada en reconocimiento de imágenes en el tema localización de objetos sobre una imagen da cuenta de que esta tarea se encuentra poco desarrollada (Sec. 2.2.4). En base a las técnicas existentes y a la tarea particular a resolver se propone un método de localización descrito a continuación.
El método utilizado en el presente trabajo presupone que una técnica de reconocimiento de piezas existe y ha sido probada con buenos resultados. Por reconocimiento de piezas se quiere decir que éstas pueden ser reconocidas a través de una imagen normalizada de la pieza correspondiente (Sec. 4.5.1). Dado lo anterior, el sistema de localización tiene como base un clasificador previamente entrenado con las piezas con que se desee trabajar. La localización se hace sobre una imagen del área de trabajo del SVT tomada por éste, a esta imagen se le ha llamado *escena*. Para lograr el reconocimiento de la posición de una pieza determinada sobre la escena se ha desarrollado un algoritmo de búsqueda. Este algoritmo aplica una ventana sobre la escena de idénticas dimensiones que las piezas normalizadas utilizadas por el clasificador (Fig 4.6). El cuadro pequeñito de color blanco dentro de la ventana representa su centro. Lo ideal de la localización es que este cuadrito central se haga corresponder al centro previamente definido de la pieza a localizar (Pag. 22).
Las operaciones del algoritmo se detallan a continuación:

1. El algoritmo inicia con la ventana moviéndose a través de toda la escena empezando por el centro.
2. Este movimiento es hecho en forma espiral y sirve para encontrar una pieza determinada. Los pasos horizontales y verticales de este movimiento son respectivamente Δx, Δy.
3. En cada posición, la ventana es rotada un paso angular $\Delta\theta$.
4. Para cada posición angular el sistema busca la pieza solicitada. Esto es, el sistema obtiene la imagen correspondiente a la ventana y la procesa mediante el clasificador.
5. La ventana continua su rotación hasta que algo sea reconocido o se completada una revolución.
6. El centro de la ventana se traslada hasta que algo es reconocido o son alcanzados los límites de la escena. En este último caso el reconocimiento es nulo.
7. Si la pieza a localizar es reconocida, el sistema almacena las coordenadas x, y, θ y finaliza la tarea de reconocimiento o continua otra búsqueda según lo que se le haya ordenado. Esto último se discute adelante.

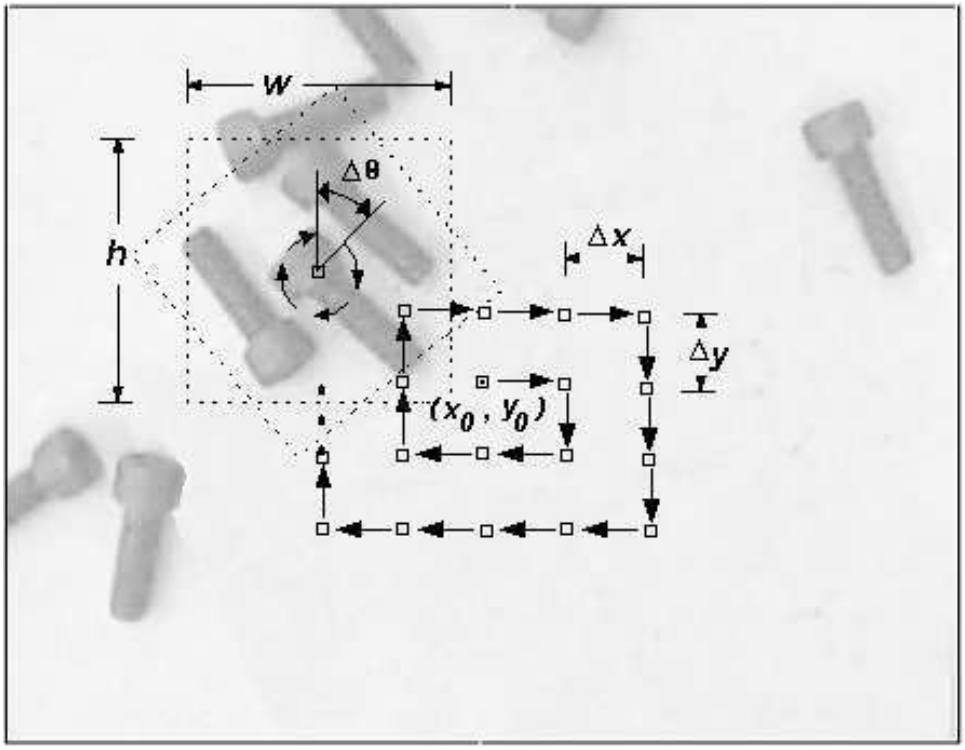


Figura 4.6: Esquema del movimiento de la ventana de búsqueda (trayectoria espiral).

El algoritmo se inicia desde el centro de la escena porque es allí donde existe mayor probabilidad de encontrar algo. Esto es debido a que el clasificador es flexible y puede localizar una pieza aunque esta no coincida exactamente con la ventana de búsqueda, así que cerca de los bordes las posibilidades de encontrar algo se reducen por que el límite de la escena es alcanzado.
Dado que los conjuntos de entrenamiento incluyen imágenes normalizadas que provienen de los límites de una escena, el sistema de localización será capaz de identificar una pieza cerca de estos límites. Sin embargo con el método propuesto esta posibilidad haría el sistema mucho más lento. Decidir entonces incluir imágenes normalizadas cuyo centro este en los límites de la imagen o imágenes que no puedan salir de la escena debe ser sujeto de configuración.
El punto crucial de este algoritmo es su exactitud y el tiempo requerido. En caso de que una pieza sea reconocida el sistema devolverá como la localización y orientación de ésta los valores x, y y θ; estos valores son los correspondientes a la posición y orientación del centro de la ventana de búsqueda. La flexibilidad del clasificador empleado puede hacer que la pieza buscada sea encontrada sin existir esta coincidencia en forma exacta. Esto, por un lado permite no hacer demasiado pequeños los desplazamientos Δx y Δy, pero por otro, impacta en la exactitud de la posición. Por otra parte, si la posición del centro de la pieza no es muy exacta, el algoritmo de búsqueda devolverá una orientación errónea. Con respecto al tiempo, se tiene que si la pieza a localizar está cerca del centro de la escena, el algoritmo será rápido, pero si está sobre un borde el algoritmo será muy lento.
Una adecuación al algoritmo presentado es utilizar, una vez que se ha localizado una pieza, una búsqueda fina con valores mas pequeños para los parámetros Δx, Δy, $\Delta\theta$. Una reducción de orden dos o cuatro sobre los valores iniciales dará un resultado en la localización de la pieza más preciso.
En la Sec. 7.4 se presentan los resultados de aplicar el método de localización descrito usando los dos clasificadores propuestos en este trabajo.
En los siguientes dos capítulos se describirán los clasificadores utilizados, el clasificador LIRA en el siguiente capítulo y el PCNC en el subsiguiente.

Capítulo 5

Clasificador neuronal LIRA

El presente capítulo está dedicado al primer clasificador utilizado en los experimentos de desarrollo del SVT. Este clasificador es llamado LIRA y se describe a profundidad en el presente capítulo. En primer lugar se ofrecen los antecedentes y el porque de la elección de LIRA. Después se detalla su estructura, su operación así como su proceso de entrenamiento. La penúltima sección se dedica al uso de distorsiones para ampliar la base de datos de entrenamiento y en la última sección se discute brevemente sobre este clasificador.

5.1. Antecedentes

Hace poco más de 16 años los investigadores Ernst Kussul, Tatiana Baidyk y Dimitri Rachkovskij iniciaron una serie de investigaciones con el propósito de desarrollar un sistema genérico de reconocimiento de imágenes [17, 18, 79, 87]. El sistema desarrollado utiliza como clasificador el popular perceptrón de una capa [14]. El punto medular de estas investigaciones ha sido la creación de un extractor de propiedades de propósito general. Lo anterior debido a que la tarea fundamental de clasificación en imágenes pasa necesariamente por la extracción de propiedades de interés sobre éstas. Por ello el desarrollo de un extractor de propiedades para imágenes es muy importante. El extractor de propiedades desarrollado está basado en el concepto de "descriptores locales aleatorios" (RLD[1]). Para propósitos de este desarrollo los términos *descriptor* y *propiedad* se consideran sinónimos. El objetivo de este clasificador de propósito general ha sido su aplicación a tareas de reconocimiento de caracteres manuscritos, de rostros y para tareas mecánicas como ensamble, control de calidad e identificación de piezas como se ejemplificó en el capítulo anterior.

5.1.1. Descriptor local aleatorio

El descriptor local aleatorio (RLD) ha sido presentado en [76] y se basa en dos ideas recientes. La primera de estas ideas es la de descriptores aleatorios de imágenes la cuál fue propuesta por Frank Rosenblatt [14]. En el perceptrón de tres capas de Rosenblatt (Fig. 5.1) cada una de las neuronas de la capa asociativa hace la función de un descriptor aleatorio de la imagen. Cada una de estas neuronas es conectada a un punto aleatoriamente seleccionado de la *retina* o imagen de entrada[2] y se calcula una función que tiene por entradas a los brillos de estos puntos. Un aspecto importante es que las conexiones de las neuronas de la capa asociativa con la entrada no pueden ser modificadas una vez que se han establecido, en especial durante el proceso de entrenamiento.
La segunda idea es el descubrimiento de que en la corteza cerebral de los animales, en el área de las funciones de visión, existen descriptores locales que se corresponden a elementos de contorno locales tales como orientación y movimientos [80]. Es probable que el conjunto de descriptores locales descubiertos esté incompleto porque en los experimentos de Wiesel y Hubel sólo esos descriptores referidos fueron detectados, los cuales fueron preparados para serles presentados a los animales con que se realizó el experimento. Muy probablemente no fueron investigados todos los descriptores (propiedades) que pueden ser extraídos por el cortex visual de los animales.
El descriptor aleatorio de Rosenblatt puede superar la desventaja anterior, sin embargo la aplicación de tales descriptores a una imagen completa decrementa considerablemente la efectividad de su aplicación. Con

[1] Siglas en inglés de Random Local Descriptor, como es conocido.
[2] Realmente se conecta a la capa de entrada que es una representación de la imagen y no necesariamente la imagen en si misma.

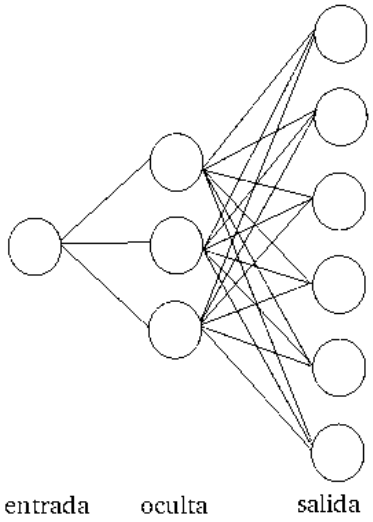


Figura 5.1: Perceptrón de tres capas de Rosenblatt.

estas ideas es que el Dr. Kussul y sus colaboradores han propuesto el RLD el cual es similar al propuesto por Rosenblatt pero con la diferencia de que los descriptores no son aplicados a toda la imagen en proceso sino a una cierta área de la misma.

5.2. Estructura

Con base en el RLD presentado en la sección anterior, se ha desarrollado un clasificador neuronal en el cual cada detector aleatorio ha sido aplicado a su propia área local correspondiente la cuál ha sido seleccionada aleatoriamente en la imagen. Llamamos a este clasificador neuronal como clasificador de área de recepción limitada o simplemente LIRA[3]. Dos variantes principales de este clasificador han sido desarrolladas. Estas variantes son similares en estructura pero con modificaciones de acuerdo a como es presentada una imagen en la entrada del clasificador. Si la imagen es presentada en formato binario, es decir, únicamente con valores de entrada *negro* o *blanco* es utilizado el clasificador neuronal LIRA binario; si la imagen de entrada se presenta con niveles de gris se utiliza el clasificador neuronal LIRA escala de grises. El *clasificador neuronal de área de recepción limitada, escala de grises* o simplemente *LIRA escala de grises* es el que se ha adaptado para el presente trabajo, de acuerdo a los requerimientos propios del SVT que desarrollamos y a las imágenes que se requieren utilizar.
A continuación se describe la estructura del clasificador neuronal LIRA escala de grises (Fig. 5.2) el cuál es una red neuronal artificial con cuatro capas:

- Capa de entrada (S) $S = s_1, s_2, ..., s_k$
- Capa de grupos (I) $Grupo = grupo_1, ..., grupo_N$
- Capa asociativa (A) $A = a_1, a_2, ..., a_N$
- Capa de salida (R) $Y = y_1, y_2, ..., y_m$

La capa S corresponde a la imagen de entrada que se desea clasificar. En esta capa llamada también *retina*, cada neurona corresponde al brillo de cada uno de los píxeles de la imagen a ser procesada, de esta forma el rango de salida de estas neuronas es $[0, B]$, donde 0 es igual a brillo nulo (negro) y B es igual al máximo brillo posible. Esta capa tiene $W \cdot H$ neuronas, donde W y H son respectivamente el ancho y el alto de la imagen a ser clasificada. La capa de grupos o capa I contiene N grupos de neuronas, cada $grupo_i$ tiene p neuronas ON y q neuronas OFF. Cada neurona ON se activa si $x_{ij} > T_{ON_{ij}}$, cada neurona OFF se activa si $x_{ij} < T_{OFF_{ij}}$, donde x_{ij} es la entrada a la neurona correspondiente, $T_{ON_{ij}}$ es el umbral de la neurona ON y $T_{OFF_{ij}}$ es el umbral de la neurona OFF. Cada umbral de cada neurona es

[3]LIRA es el acrónimo en inglés de "LImited Receptive Area".

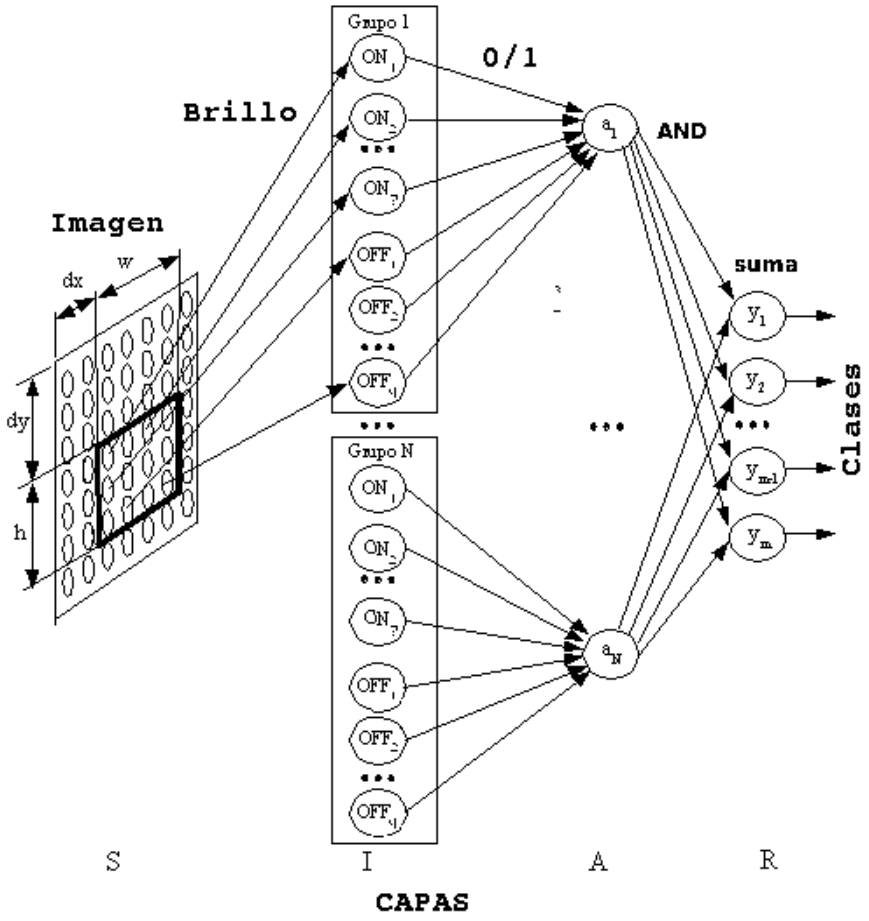


Figura 5.2: Estructura del clasificador neuronal LIRA.

seleccionado aleatoriamente del intervalo $[0, B \cdot \eta]$, donde η es una constante experimental del clasificador a ser seleccionada del intervalo $(0, 1]$. Las neuronas de cada grupo están conectadas aleatoriamente a las neuronas de la capa S localizadas en una ventana rectangular $w \cdot h$ definida aleatoriamente sobre la capa S. Los valores dx y dy son seleccionados aleatoriamente del intervalo $[0, W - w]$ y $[0, H - h]$ respectivamente. Los parámetros w y h son muy importantes para el desempeño del clasificador neuronal, éstos deben ser seleccionados experimentalmente. Las conexiones entre las capas S y la capa I no cambian durante el proceso de entrenamiento y tampoco lo hacen los umbrales asociados a cada una de las neuronas de I. La capa A contiene N neuronas, cada una tiene $p+q$ entradas conectadas a la salida del grupo correspondiente. Una neurona de la capa A se activa si, y solo si todas sus entradas están activas, la salida de la neurona es 1 si está activa e igual a 0 si no está activa. Las conexiones entre las capas I y A no pueden ser modificadas durante el proceso de entrenamiento. Cada una de las neuronas de la capa A corresponde a un RLD (Sec. 5.1.1). Todas las neuronas de la capa A están conectadas a todas y cada una de las neuronas de la capa R. El peso de estas conexiones si puede ser modificado durante el proceso de entrenamiento. Es precisamente en el valor de todos los pesos de estas conexiones donde se ubica la memoria del clasificador. La salida de la neurona i de la capa R es calculada con la siguiente ecuación:

$$y_i = \sum_{j=0}^{N} w_{ji} \cdot a_j, \tag{5.1}$$

donde w_{ji} es el peso de la conexión entre la neurona j de la capa A y la neurona i de la capa R y a_j es la salida de la neurona j de la capa A.

El número N se escoge experimentalmente de forma que pueda asegurar que el proceso de reconocimiento sea estadísticamente estable. También el número de neuronas ON y neuronas OFF por grupo en la capa I es seleccionado experimentalmente de manera que para:

$$PA = 100 \cdot \frac{K}{N} \tag{5.2}$$

se tenga K dentro de los límites dados por:

$$K = c \cdot \sqrt{N} \qquad |c \in [1, 5] \tag{5.3}$$

donde K es el número de neuronas activas en la capa A, N el número total de neuronas y PA el porcentaje de neuronas activas respecto al total, mientras que c se escoge experimentalmente dentro del rango dado. La ecuación (5.2) se deriva de datos neurofisiológicos, se tiene que el número de neuronas activas en la corteza cerebral de los mamíferos es cientos de veces menor que el total de neuronas existentes en él.

5.3. Operación

Atendiendo a la estructura presentada, cuando una imagen es presentada a través de la entrada del clasificador neuronal LIRA, se calcula la respuesta de la capa A, para esto primero se deberá calcular la respuesta de la capa I y luego la propia de la capa A. Una vez calculada la respuesta de la capa A para una imagen presentada, se representa esta actividad como un vector binario $\vec{A}$, ya que las salidas posibles de las neuronas de la capa A son binarias. A este cálculo del vector $\vec{A}$ lo llamamos *codificación de la imagen* y al vector $\vec{A}$ le llamamos *código de la imagen*. Esto es posible ya que la estructura del clasificador LIRA a la salida de la capa A no sufre alteración alguna desde su creación, por lo cuál cada imagen tendrá un vector $\vec{A}$ asociado y único. Es por esto que guardar los códigos de imagen de todo el grupo de imágenes destinadas para entrenamiento en alguna memoria masiva (disco duro) permite ahorrar mucho tiempo de cómputo al no tener que recalcular los códigos de la imagen cada vez que una misma requiera volver a ser procesada, como es en el caso del proceso de entrenamiento el cual se explica a continuación.

Como una prueba de la adecuada distribución aleatoria en la construcción del clasificador neuronal LIRA se tiene la posibilidad de consultar su histograma de distribución de entradas (Fig. 5.3). En estos histogramas, pertenecientes a dos construcciones LIRA distintas, se presenta en cada caso la distribución espacial de todas las entradas de la capa I sobre la capa S. La intensidad representa el número de conexiones para cada uno de los píxeles correspondientes, donde el color blanco es el máximo de densidad posible y el negro la densidad mínima o cero. La comparación entre los histogramas da prueba de que no hay dos construcciones idénticas LIRA.

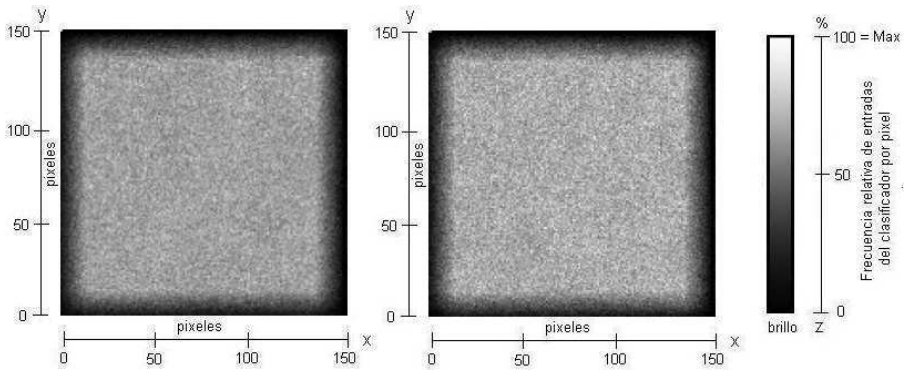


Figura 5.3: Histogramas de distribución de entradas en dos construcciones LIRA distintas.

5.4. Proceso de entrenamiento

Para llevar a cabo el proceso de entrenamiento se requiere contar con un conjunto de imágenes que representen las clases de interés con que se desee entrenar el clasificador. Cada una de estas imágenes debe tener asociada una etiqueta con la clase a la cuál representa, de modo que el clasificador pueda aprender de ellas, es decir, ser entrenado.
El proceso de entrenamiento consta de varios ciclos, en cada ciclo cada imagen del conjunto de entrenamiento se presenta una vez al clasificador junto con su etiqueta correspondiente.
Al inicio de todo el proceso de entrenamiento se borra la memoria, esto es se hace que todos los pesos de las conexiones entre las capas A y R sean igual a cero. Esto simboliza el hecho de que no existan conexiones entre estas capas. Este primer paso debe omitirse si se desean aplicar ciclos de entrenamiento adicionales a un clasificador ya entrenado.
Inmediatamente después se inician los ciclos de entrenamiento, cada uno de los cuales consiste en los siguientes pasos:

1. Una imagen es presentada como entrada del clasificador. Se calcula entonces el *código de la imagen* y se procesa con el clasificador para que la salida del clasificador sea calculada. El código de la imagen se almacenará para que todos los sucesivos ciclos procesen este código y no la imagen desde la entrada, ahorrando importantes recursos de cómputo y tiempo.
2. Después de calcular la salida del clasificador (salida de la capa R), la clase correcta que corresponde a la imagen de entrada es leída a través de la etiqueta asociada a ésta. Para hacer el entrenamiento robusto, la salida correspondiente a la clase correcta es modificada conforme a la ecuación:

$$y_c(t+1) = y_c(t)\dot{(}1 - T_E) \tag{5.4}$$

   donde $y_c(t)$ y $y_c(t+1)$ es la salida de la clase correcta antes y después de ser modificada y T_E es una constante llamada excitación adicional de la neurona ganadora.
3. Después de esto, se detecta la neurona de la capa R con el más alto valor de salida, llamada neurona ganadora. Esta neurona representa la clase reconocida para la imagen dada como entrada.
4. Sea y_w la salida de la neurona ganadora y y_c la salida de la neurona que realmente representa la clase correcta. Si $y_w = y_c$ no se hace nada. Si $y_w \neq y_c$, entonces

$$w_{jc}(t+1) = w_{jc}(t) + a_j, \forall j \tag{5.5}$$
$$w_{jw}(t+1) = w_{jw}(t) - a_j, \forall j \tag{5.6}$$
$$w_{jw}(t+1) < 0 \Longrightarrow w_{jw}(t+1) = 0, \forall j \tag{5.7}$$

   donde $w_{jc}(t)$ es el peso correspondiente de la conexión entre la neurona j de la Capa A y la neurona correcta c de la Capa R antes de ser modificada y $w_{jc}(t+1)$ es el mismo peso después de ser modificado; $w_{jw}(t)$ es el peso correspondiente de la conexión de la neurona j de la capa A con la neurona ganadora de la capa R antes de la modificación y $w_{jw}(t+1)$ una vez modificado; a_j es el valor de salida de la neurona j de la capa A.

El proceso completo de entrenamiento para el clasificador neuronal es un proceso iterativo de varias decenas de ciclos. El proceso de entrenamiento termina después de un número determinado de ciclos o cuando la cantidad de errores del último ciclo de entrenamiento es menor que un valor preestablecido. Para el primer caso se alcanza un valor experimental dado por el hecho de que el clasificador se ve saturado por el proceso de entrenamiento, pues para un cierto número de ciclos adicionales ya no corresponde ningún incremento en el porcentaje de aciertos realizados sobre un conjunto de prueba. Ese conjunto de prueba tiene idénticas características que el conjunto de entrenamiento, sólo que los elementos de éste no tienen ninguna etiqueta asociada que de cuenta de la clase a la que pertenecen; y por supuesto, contiene imágenes distintas.

5.5. Distorsiones

Con el objetivo de ampliar el conjunto de imágenes destinadas para entrenamiento del clasificador neuronal LIRA se consideró la aplicación de tres tipos de distorsiones sobre las imágenes originales. Estas distorsiones son de tipo desplazamiento en sentido horizontal, vertical y angular (Fig. 5.4). Se han seleccionado estos tipos de distorsiones ya que representan a las diversas variaciones que pueden encontrarse en las imágenes

a reconocer por el clasificador. Otras variaciones como el esquileo[4] aplicadas para caracteres manuscritos [15] no se esperan en el SVT por lo que no se aplican.

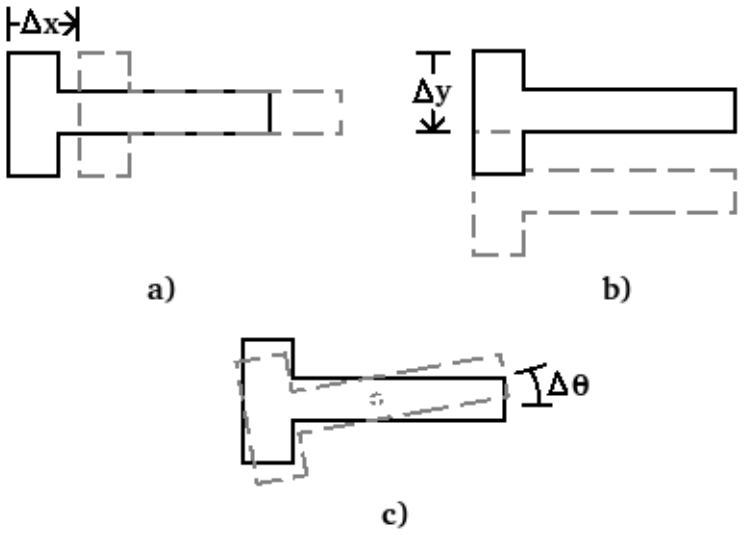


Figura 5.4: Los tres tipos de distorsiones consideradas para su aplicación sobre las imágenes del conjunto de entrenamiento del clasificador LIRA. a) Distorsión de desplazamiento horizontal, b) distorsión de desplazamiento vertical y c) distorsión angular.

Se ha desarrollado un programa de cómputo capaz de aplicar estas distorsiones sobre las imágenes del conjunto de entrenamiento. Estas distorsiones pueden ser aplicadas individualmente o creando combinaciones. Las distorsiones han sido aplicadas por pares simétricos teniendo como unidad de desplazamiento el píxel. Para una determinada distorsión se da el número de pares de distorsiones y el valor correspondiente de desplazamiento en píxeles. Por ejemplo, al requerir dos pares de distorsiones en el sentido horizontal con un desplazamiento de un píxel se tendrán 4 nuevas imágenes distorsionadas por cada una de las existentes en el conjunto de entrenamiento. Estas imágenes estarán desplazadas horizontalmente de su original ± 1 y ± 2 píxeles. La aplicación de estas distorsiones y experimentos con las mismas se expondrán en la Sección 7.1.4.

5.6. Discusión

Una desventaja del Clasificador neuronal LIRA es su sensibilidad a los pequeños desplazamientos en la imagen. Este detalle se trata de compensar mediante la generación de imágenes distorsionadas ligeramente a partir del conjunto de imágenes para entrenamiento. Sin embargo esta forma de compensación no sirve para imágenes cuyos objetos incluidos sufran mayor desplazamiento, y es por esta razón principal que se ha desarrollado y probado también el Clasificador PCNC, el cuál se abordará en el siguiente capítulo.

[4]Se utiliza "esquileo" como traducción del término ampliamente conocido en inglés "skewing" que se refiere, para este caso, a una transformación lineal vectorial del espacio de imagen.

Capítulo 6

Clasificador Neuronal de Permutación de Códigos (PCNC)

El capítulo que inicia está dedicado al segundo clasificador neuronal utilizado para la implementación del SVT. Este clasificador llamado PCNC ha sido introducido en la Sección 4.4 teniendo diferencias sustanciales con el clasificador LIRA pero también algunas similitudes. A continuación en la siguiente sección se describe detenidamente la estructura del PCNC y los procesos que lleva a cabo. En la sección posterior se aborda el proceso de entrenamiento del mismo. En la Sección 6.3 se explican las distorsiones probadas con las base de datos utilizadas con el PCNC mientras que la Sección 6.4 y última dedica unas líneas a la discusión sobre este clasificador respecto a la tarea de la que se ocupa.

6.1. Estructura

El PCNC está basado en la estructura genérica del paradigma *Associative Projective Neural Networks* (APNN) descrita en [17, 18]. Dicho paradigma incluye al *clasificador de umbral aleatorio* [79, 87], al *clasificador neuronal de subespacio aleatorio* [78], al clasificador LIRA [24] y al *clasificador neuronal de permutación de códigos* (PCNC[1]) [85]. El PCNC al igual que el clasificador neuronal LIRA trabaja con imágenes en escala de grises.
La estructura del PCNC consta de tres partes que trabajan en forma seriada, estas son un extractor de propiedades, un codificador y un clasificador neuronal (Fig. 6.1).

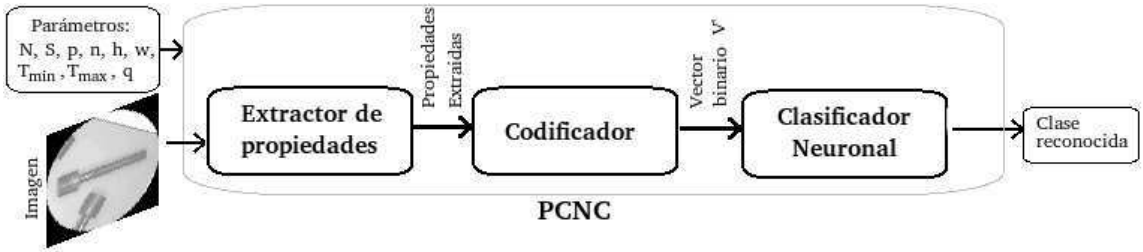


Figura 6.1: Diagrama a bloques de la estructura del PCNC mostrando los elementos principales de intercambio entre ellos.

De forma muy general el PCNC inicia su trabajo cuando una imagen en escala de grises es presentada a la entrada del extractor de propiedades, las propiedades extraídas por este último son presentadas al codificador que las transforma en un vector binario de gran dimensión, vector que por último es dado al clasificador neuronal de una capa para ser procesado por éste, ya sea para propósitos de entrenamiento, de prueba o para reconocimiento de alguna clase previamente entrenada.

[1]PCNC son las siglas de permutative code neural classifier.

6.1.1. Extractor de propiedades

El extractor de propiedades (Fig. 6.2) inicia su trabajo con una imagen en escala de grises. Éste selecciona sobre la imagen una serie de puntos específicos (Fig. 6.3). Muchas formas de selección de estos puntos específicos pueden ser utilizadas, lo importante es que estos puntos representen las propiedades de la imagen que intervengan en su clasificación o diferenciación entre otras imágenes distintas a ser reconocidas. Dos métodos para definir los puntos específicos son por umbral de brillo y por extracción de contornos. Ambos métodos requieren de la selección de un umbral de brillo determinado B. Para el primer método, los puntos específicos de la imagen serán todos aquellos píxeles cuyo brillo b_{ij} sea mayor que B. Para el método de contorno se seleccionan todos aquellos puntos en donde el gradiente del brillo, es decir los contornos, sean mayores que el umbral B; con esta última selección por umbral se logra eliminar considerablemente el ruido de la imagen de contorno.

Figura 6.2: Procesos del extractor de propiedades, su interrelación, así como sus entradas y salidas.

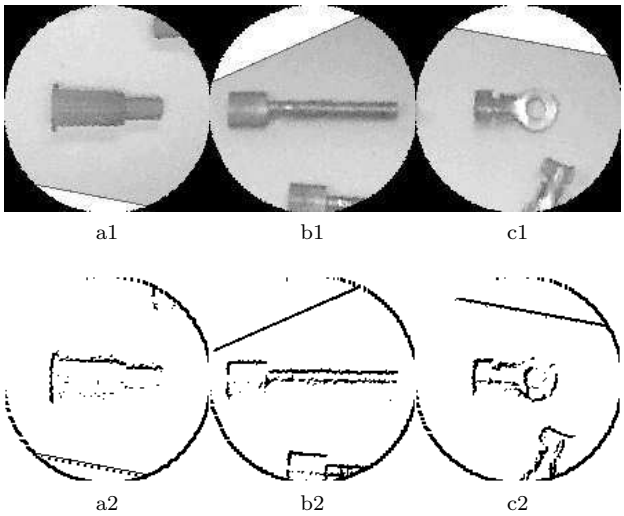


Figura 6.3: Selección de puntos específicos con el método de umbral de bordes. Se parte de una imagen normalizada y se aplica un operador de extracción de bordes Sobel. Se seleccionan aquellos puntos resultantes que son mayores que el umbral predefinido B. Fila superior. Imágenes normalizada originales. Fila inferior. Resultado de la selección de los puntos específicos sobre las imágenes respectivas de arriba, todos los píxeles negros en la imagen son puntos específicos seleccionados.

Para el presente trabajo, de acuerdo a la naturaleza de las imágenes y las piezas que éstas contienen descritas en el Capítulo 4, se ha utilizado el método de extracción de contornos mediante un operador Sobel [88]. El método de umbral de brillo aplicado en imágenes relativamente grandes como las utilizadas en este trabajo

(100×100 o 150×150) resulta en grandes cantidades de puntos específicos que implican el incremento masivo de recursos necesarios para el procesamiento de éstos y para las posteriores etapas del PCNC.
Para cada punto específico se define un rectángulo de dimensión $w \times h$ en cuyo centro está precisamente este punto (Fig. 6.4). Desde dentro de éste rectángulo se extraen múltiples propiedades de la imagen mediante el procedimiento explicado a continuación. Un conjunto de puntos positivos p y negativos n determinan cada una de las propiedades dentro del rectángulo. Estos puntos se distribuyen aleatoriamente dentro del rectángulo y su número es fijo para toda la estructura. Cada punto P_{rs} tiene asociado un umbral T_{rs} el cuál se define aleatoriamente en el rango:

$$T_{min} \leq T_{rs} \leq T_{max} \tag{6.1}$$

Los puntos positivos serán activos siempre que su brillo sea:

$$b_{rs} \geq T_{rs}. \tag{6.2}$$

Los puntos negativos serán activos siempre que su brillo sea:

$$b_{rs} \leq T_{rs}. \tag{6.3}$$

Una determinada propiedad existirá en el rectángulo si todos los puntos tanto positivos como negativos se encuentran activos.

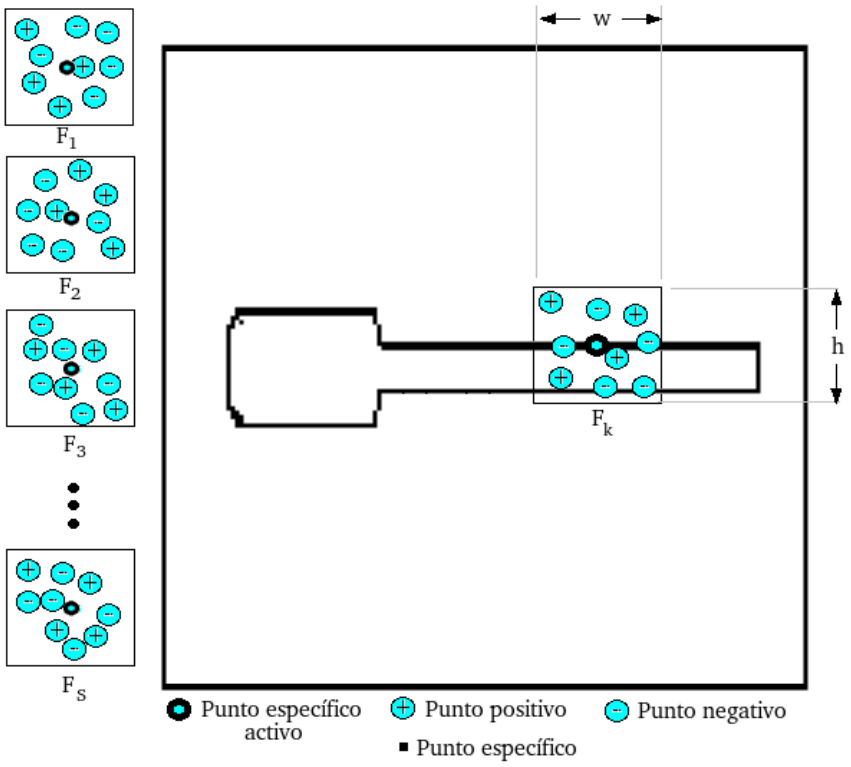


Figura 6.4: Extracción de propiedades en la imagen. Primero se definen S propiedades. Cada una mediante p puntos positivos y n puntos negativos aleatoriamente distribuidos sobre un rectángulo de $w \times h$ píxeles. A la izquierda se ilustran ejemplos de estas propiedades con $p = 4$ y $n = 5$. En el cuadro principal se muestran los puntos específicos de una imagen. Para cada uno de estos puntos se prueba la existencia de las S propiedades. En la imagen se muestra el proceso de búsqueda de la propiedad F_k.

Se procura que ninguno de estos rectángulos de búsqueda de propiedades en la imagen salga de la misma. Debe recordarse que una característica de las imágenes que se busca reconocer en este trabajo (imágenes normalizadas) no tienen puntos de interés en las orillas. Sin embargo si se elijen relativamente grandes

los parámetros de ventana w y h con respecto a las dimensiones de la imagen W y H se tendrá más probabilidad de que existan rectángulos que salgan de los límites de la imagen, esto se hace más probable si existen puntos específicos cerca de las orillas. Considerando lo anterior se expande la imagen $w/2$ píxeles a cada lado y $h/2$ píxeles arriba y abajo con color blanco, es decir, significando ausencia de todo punto específico posible y evitando que cualquiera de estos rectángulos salga de la imagen ampliada.

Se utilizan muchas propiedades distintas $F_i \mid i \in [1, S]$. Donde S es por lo general del orden de unidades de millar. El extractor de propiedades examina las S propiedades para cada uno de los puntos específicos definidos. Todas las propiedades extraídas de todos los puntos específicos definidos son entregados al codificador.

6.1.2. Codificador

El codificador (Fig. 6.5) transforma las propiedades dadas por el extractor de propiedades a un vector binario:

$$V = \{v_i \mid v_i = \{0, 1\},\ i \in (1, N)\}\,, \tag{6.4}$$

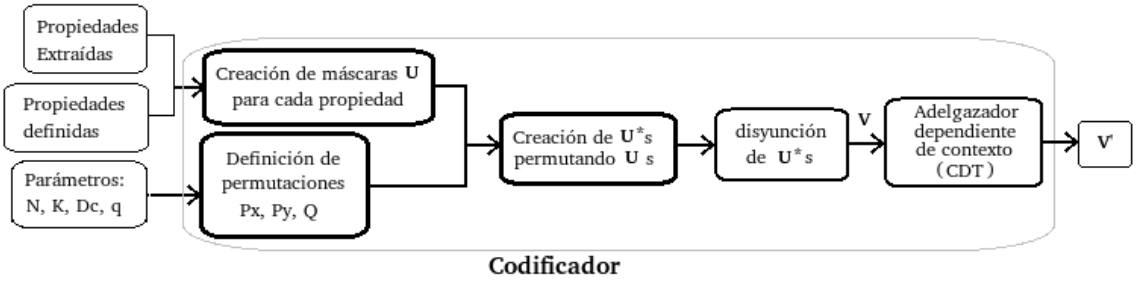


Figura 6.5: Principales procesos que lleva a cabo el codificador así como sus entradas y salidas.

Para cada propiedad extraída F_k el codificador crea un vector adicional binario:

$$U_k = \{u_i \mid u_i = \{0, 1\},\ i \in (1, N)\}\,, \tag{6.5}$$

Este vector contiene K 1's, donde $K \ll N$, al menos mil veces menor. Un procedimiento aleatorio que se explica más tarde es utilizado para elegir las posiciones de los unos en el vector U para cada propiedad F_k. Este procedimiento genera la lista de posiciones de unos para cada característica y salva todas estas listas en memoria no volátil. El vector U_k es llamado *máscara* de la propiedad F_k.

En la siguiente etapa del proceso de codificación se hace necesario transformar el vector auxiliar U al nuevo vector U^* el cual corresponde a la propiedad localizada en la imagen. Esta transformación se hace mediante permutaciones de los componente del vector U. El número de permutaciones depende de la localización de la propiedad en la imagen. Las permutaciones correspondientes a las direcciones horizontal (X) y vertical (Y) son permutaciones diferentes.

Una permutación de m elementos P^m puede ser representada como un vector m-dimensional. Para aplicar esta permutación a un vector éste debe ser de dimensión m. En términos formales:

$$P^m(V) = V' \mid V' = \{v'_i\}\ ,\ v'_{p_i} = v_i\ , \quad P, V, V' \in \Re^m \tag{6.6}$$

Lo cual significa que el resultado de aplicar la permutación P^m a un vector V resulta en un nuevo vector V' cuyos componentes se definen tomando cada componente v_i de V y colocándolo en la posición p_i de V', es decir, a la que apunta el índice correspondiente del vector de permutación P^m. En la Fig. 6.6 se ilustra un ejemplo gráfico simple a este respecto.

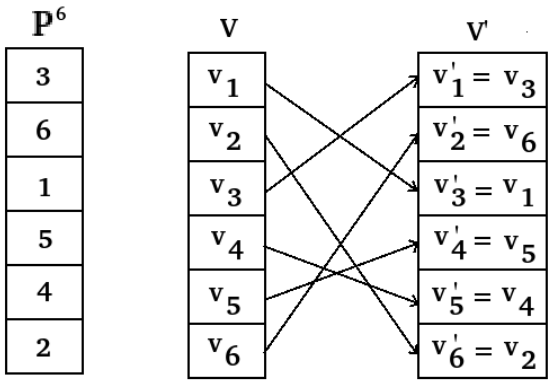


Figura 6.6: Permutación y su representación sobre un vector. Se ilustra una permutación P^6 como un vector. El resultado de aplicar esta permutación sobre un vector V se ilustra a la derecha. Cada elemento v_i de V es reordenado en la posición señalada por la componente correspondiente de P^6, dando como resultado un nuevo vector V' con los mismos elementos de V pero en orden distinto.

6.1.2.1. Codificación de las propiedades

El problema a resolver es obtener códigos binarios de las propiedades extraídas los cuales tengan correlación fuerte si la distancia entre las localizaciones de las propiedades es pequeña y tengan correlación débil o no tengan ninguna si esta distancia es grande. Por ejemplo, si una misma propiedad F_k se extrae tanto en un extremo de la pieza en la imagen como en el otro extremo entonces éstas deben ser codificadas como vectores binarios distintos U^*_{k1} y U^*_{k2}, existiendo entre estos correlación débil o ausencia alguna de correlación. En el caso de que la misma propiedad sea encontrada en puntos vecinos entonces estas deben ser codificadas con los mismos vectores U^*_{k3} y U^*_{k4}. Esta propiedad que se acaba de describir permite que el sistema de reconocimiento sea insensible a pequeños desplazamientos de los objetos en la imagen.
Para codificar la localización de la propiedad F_k en la imagen es necesario definir y dar valor a la distancia de correlación D_c. Sea la propiedad F_k detectada en dos puntos distintos $P_1(x_1, y_1)$ y $P_2(x_2, y_2)$, sean U^*_{P1} y U^*_{P2} los vectores binarios que codifican a F_k para estos puntos respectivamente y sea d la distancia euclidiana entre estos puntos dada por:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{6.7}$$

Se requiere que:

$$d < D_c \Rightarrow U^*_{P1} \text{ y } U^*_{P2} \quad \text{estén correlacionados y,} \tag{6.8}$$

$$d \geq D_c \Rightarrow U^*_{P1} \text{ y } U^*_{P2} \quad \text{no estén correlacionados.} \tag{6.9}$$

Para buscar cumplir con estas propiedades se calculan los siguientes valores para una propiedad detectada F_k en un punto $P(i, j)$:

$$\begin{aligned} X &= i/D_c, \\ E(X) &= int\,(X)\,, \\ R(X) &= i - E(X) \cdot D_c, \end{aligned} \tag{6.10}$$

$$\begin{aligned} Y &= j/D_c, \\ E(Y) &= int\,(Y)\,, \\ R(Y) &= j - E(Y) \cdot D_c, \end{aligned} \tag{6.11}$$

$$p_x = int\left(\frac{R(X)}{D_c} N\right), \tag{6.12}$$

$$p_y = int\left(\frac{R(Y)}{D_c}N\right), \tag{6.13}$$

donde $int()$ es la función entero. Por lo tanto $E(X)$ y $E(Y)$ son las partes enteras de X y Y respectivamente y $R(X)$ y $R(Y)$ son las partes fraccionarias de X y Y respectivamente.
La máscara (vector U_k) de la propiedad F_k se considera como un código de esta propiedad localizado en el punto origen de la imagen: $O(0,0)$. Se definen las permutaciones $\mathbf{P_x}$ y $\mathbf{P_y}$ como requisito para obtener los códigos correspondientes a las propiedades existentes en cualquier punto de la imagen fuera del origen. En general, para obtener el código de la propiedad F_k perteneciente al punto $P(i,j)$ se procede de la siguiente forma:

1. Primero se trata el desplazamiento horizontal, para lo cual se aplica $E(X)$ veces la permutación $\mathbf{P_x}$ al vector U_k, después se aplica la misma permutación una vez más pero solamente a los primeros p_x componentes del vector U_k.
2. Segundo, se trata el desplazamiento vertical de forma análoga. Se aplica la permutación $\mathbf{P_y}$ $E(Y)$ veces con una permutación adicional para los primeros p_y componentes de U_k.

6.1.2.2. Ejemplo de permutaciones

El procedimiento anterior se ilustra con un ejemplo para facilitar su comprensión. La dimensión del vector U y los valores x y y del punto P se eligen pequeños con el propósito de dar claridad al proceso. Tomemos como parámetros del PCNC $D_c = 6$ y $N = 8$ y sean $\mathbf{P_x}$ y $\mathbf{P_y}$ dos permutaciones de dimensión N.
Supóngase detectada la propiedad F en el punto $P(10,14)$ y sea el vector U la máscara correspondiente a esta propiedad. La tarea consiste en codificar U en un nuevo vector U^* atendiendo a las coordenadas del punto P. Como primer paso se aplican (6.10), (6.11), (6.12) y (6.13) resultando: $E(X) = 1$, $E(Y) = 2$, $p_x = 5$ y $p_y = 2$. Se aplica a continuación $E(X)$=1 permutación $\mathbf{P_x}$ al vector U (Fig. 6.7) y luego al resultado (U_1) se aplica una permutación adicional únicamente a los primeros $p_x = 5$ componentes obteniendo con esto el vector U', todas las componentes que no sean definidas mediante la permutación parcial se copian del vector anterior correspondiente no importante si fueron permutados o no. Con el proceso anterior tenemos las siguientes trayectorias de ejemplo: $u_1 \rightarrow u_{1,3} \rightarrow u'_4$, $u_2 \rightarrow u_{1,7} \rightarrow$ se elimina y $u_8 \rightarrow u_{1,5} \rightarrow u'_8, u'_5$. Tenemos que para el primer ejemplo el valor de u_1 es permutado dos veces y termina en u'_4. Para el segundo caso u_2 se permuta una sola vez a $u_{1,7}$ y luego se ignora porque a la componente 7 es menor que $p_x = 5$ y por que la componente correspondiente en U' ya ha sido ocupada por u_7. Para el tercer caso u_8 se permuta las dos veces pero adicionalmente es copiado a la componente u'_5 pues luego de permutar los primeros 5 elementos de U' queda vacía.
En este ejemplo se han dado los tres casos posibles que pueden ocurrir, sin embargo debe tenerse en cuenta que de acuerdo a la naturaleza práctica de los vectores máscaras (U) que tienen dimensión N muy grande y sus componentes son mayoritariamente ceros, los casos especiales de eliminación ocurren muy rara vez como se explicará.
Una vez habiendo realizado las permutaciones X, realizamos las permutaciones Y con el mismo procedimiento, sólo que ahora lo haremos con el vector U' y aplicaremos la permutación $\mathbf{P_y}$ y los parámetros $E(Y)$ y p_y. En la Fig. 6.8 se muestra gráficamente este procedimiento. El resultado de esta permutación es el resultado de ambas permutaciones aplicadas sobre el vector máscara U de la propiedad F y le llamamos vector U^*. Este vector codifica la propiedad F en la ubicación del punto $P(10,14)$ de la imagen correspondiente.

6.1.2.3. Propiedades de las permutaciones

Consideremos ahora las propiedades de las permutaciones descritas. Supóngase que la propiedad F_k ha sido detectada en dos puntos $P_1(x_1, y_1)$ y $P_2(x_2, y_2)$tales que $x_1 \neq x_2$ y $y_1 \neq y_2$. Sea U_k la máscara correspondiente para esta propiedad. Se definen d_x y d_y como:

$$d_x = |x_2 - x_1|, \tag{6.14}$$

$$d_y = |y_2 - y_1| \tag{6.15}$$

Suponiendo que $dx \neq 0$, luego de realizar las permutaciones horizontales ($\mathbf{P_x}$) los vectores correspondientes U_1 y U_2 serán distintos. Sea Δn la diferencia en el número de 1's entre los vectores. Puede mostrarse que el valor promedio de Δn puede calcularse de forma aproximada con:

$$\Delta n \approx \frac{K}{D_c} d_x \tag{6.16}$$

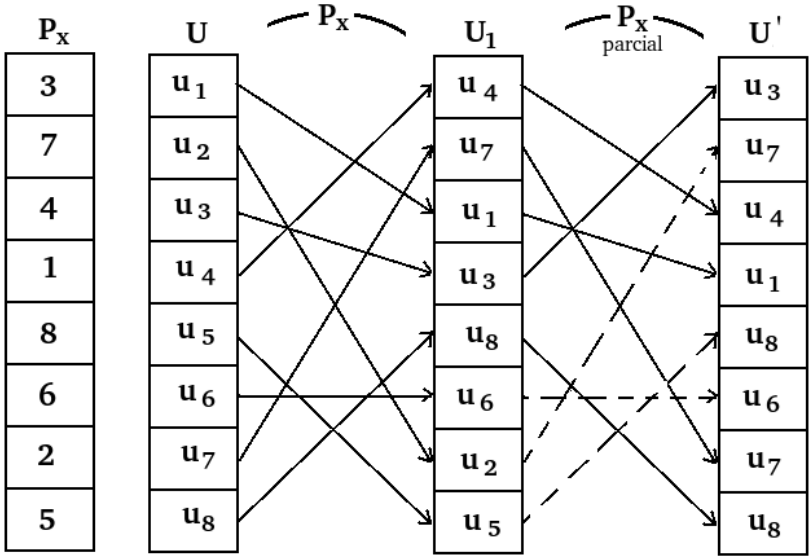


Figura 6.7: Ejemplo de permutaciones $\mathbf{P_X}$ sobre el vector U con $E(X) = 1$ y $p_x = 5$. Se aplica una vez la permutación a todo el vector y una vez más sólo a los primeros cinco componentes del mismo. En este sentido las líneas punteadas representan cambios que no deben realizarse.

donde K es el número de unos del vector auxiliar binario U de la propiedad F_k. Luego de las permutaciones verticales ($\mathbf{P_y}$) los vectores resultantes correspondientes U_1^* y U_2^* tendrán diferencias que pueden ser estimadas por:

$$\overline{\Delta n} \approx K \left(1 - \left(1 - \frac{d_x}{D_c}\right)\left(1 - \frac{d_y}{D_c}\right)\right),\ 1 > \overline{\Delta n} > 0. \tag{6.17}$$

Por lo anterior los vectores U_1^* y U_2^* estarán correlacionados solamente si se cumple $d_x < D_c$ y $d_y < D_c$. La correlación se incrementará si d_x y d_y se decrementan.
Puede verse en la Fig. 6.9 que distintos componentes del vector U pueden pretender quedar en la misma posición luego de realizar las permutaciones. Por ejemplo, luego de las permutaciones $\mathbf{P_x}$ el componente u_8 ocupa dos posiciones en U' o luego de aplicar ambas permutaciones el componente u_6 termina en dos posiciones y el u_8 en tres. Estos eventos son indeseables y no son un problema para el PCNC pues la probabilidad de que tales eventos indeseables sucedan está relacionada inversamente a la dimensión de N y a la relación N/K y atendiendo a los valores grandes de N y pequeños para K la probabilidad de estos eventos es menor de 0.01 % [75].

6.1.2.4. Vector código V

Una vez calculados todos los vectores U_r^* de todas las propiedades detectadas en la imagen se crea el vector código definido como:

$$V = \left\{ v_i \ \mid v_i = \bigwedge u_{ri}^*\ ,\ i \in (1, N) \right\} \tag{6.18}$$

donde $\bigwedge$ es el símbolo de la disyunción, u_{ri}^* es el i-ésimo componente del vector U_r^*, vector que es el corresponde a la propiedad detectada F_r.
Al utilizar números aleatorios independientes para la generación de las máscaras se logra que este proceso de codificación produzca representaciones mayoritariamente independientes para todas las propiedades. La única pero débil influencia entre las distintas propiedades aparece cuando se absorbe algún *1* en la disyunción, Eq. (6.18).

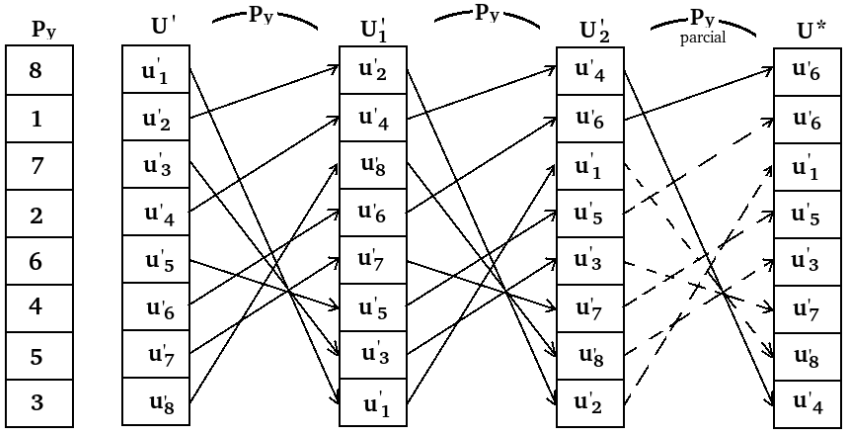


Figura 6.8: Ejemplo de permutaciones $\mathbf{P_Y}$ aplicado sobre el vector resultante de las permutaciones $\mathbf{P_X}$ con $E(Y) = 2$ y $p_y = 2$. La permutación se aplica dos veces a todo el vector y una vez más solamente a los primeros dos elementos del mismo. Las líneas punteadas de la última permutación indican cambios que no deben realizarse.

6.1.2.5. Adelgazador dependiente de contexto (CDT)

Para poder reconocer alguna pieza en una imagen se hace necesario utilizar combinación de propiedades, es decir, combinar la existencia de ciertas propiedades con la ausencia de otras. Para lograr este propósito de combinación de propiedades se ha utilizado con éxito el CDT[2] o *Adelgazador dependiente de contexto.* [68]. El CDT ha sido desarrollado en base al proceso de normalización de vectores [89]. Si bien existen diversos procedimientos de implementación del CDT en este trabajo se utiliza el procedimiento ilustrado en la Fig. 6.10, el cuál requiere de un número entero q como parámetro de entrada y consiste en lo siguiente:

1. Se genera una nueva permutación $\mathbf{Q}$ de dimensión N la cuál es independiente de $\mathbf{P_x}$ y $\mathbf{P_y}$.
2. Se prueba cada componente v_i del vector V, si $v_i = 0$ no se hace nada si $v_i = 1$ se considera la trayectoria de este componente individual durante q permutaciones $\mathbf{Q}$. Si esta trayectoria pasa por al menos un elemento *1* del vector V, el valor de v_i se hace 0.
3. Al vector resultante se le llama V'.

Un ejemplo gráfico de lo anterior se tiene en la Fig. 6.10 donde $q = 4$. La permutación $\mathbf{Q}$ se representa por las flechas. Para el elemento v_1 se sigue la trayectoria indicada por la permutación la cuál es: $v_1 \rightarrow v_7 \rightarrow v_8 \rightarrow v_5$, si cualquiera de las componentes v_7, v_8 o v_5 es igual a 1 entonces $v_1 = 0$. Lo mismo se hace para todos los elementos de V, construyéndose un nuevo vector sin alterar el vector original.
El número q es un parámetro de reconocimiento del sistema. Una vez que se aplica el CDT al vector V se ha cumplido con el proceso correspondiente del codificador. El vector V' de dimensión N representa el código de la imagen presentada originalmente al extractor de propiedades. Este resultado está listo para ser pasado al clasificador neuronal.

6.1.3. Clasificador neuronal

En la Sección 4.4 se ha mencionado y hecho referencia al perceptrón de una capa de Rosenblatt. Este perceptrón posee muy buena convergencia sin embargo requiere que en el espacio paramétrico las clases tengan separabilidad lineal. Para obtener esta separabilidad lineal, las etapas anteriores del PCNC, el extractor de propiedades y el codificador, convierten el espacio paramétrico de una imagen que es representado por el brillo de todos los píxeles de ésta, a un espacio paramétrico de mayor dimensión. En general se tiene un espacio paramétrico de dimensión $W \cdot H$ convertido a otro de dimensión N, donde W y H son el ancho y el alto en píxeles de las imágenes a procesar por el PCNC y N es el parámetro del PCNC igual a la dimensión del vector V'. Este procedimiento descrito mejora considerablemente la separabilidad lineal del espacio paramétrico que representa la imagen y el objeto que esta contiene.

[2]Llamado así por ser las siglas en inglés de Context Dependent Thinning.

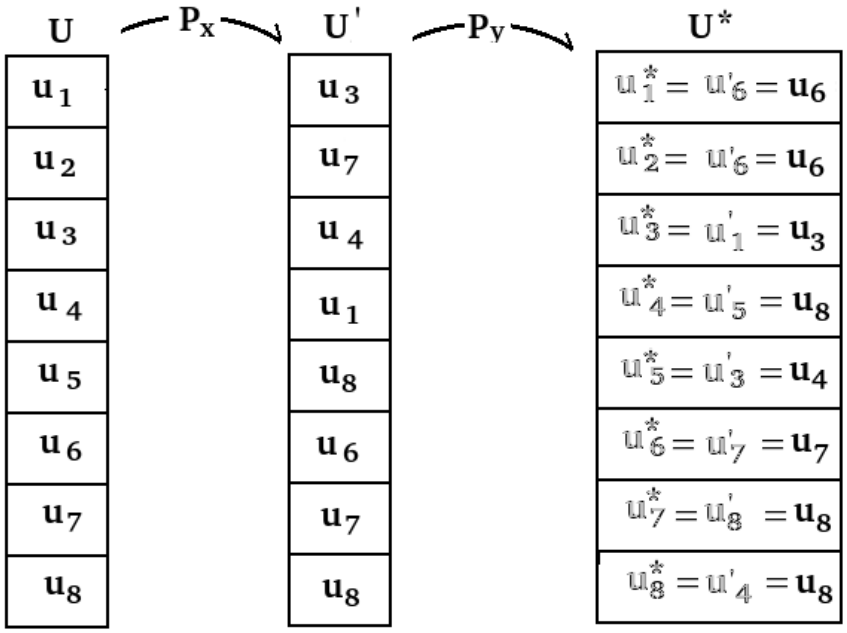


Figura 6.9: Resultados obtenidos al aplicar las permutaciones $\mathbf{P_x}$ sobre el vector U y luego la permutación $\mathbf{P_y}$ a ese resultado U'. En la figura se muestra el orden final de las componentes del vector U dentro del vector final U^*.

En la Fig. 6.11 se presenta de nuevo la estructura del clasificador neuronal LIRA pero dividiéndolo. Obsérvese que las primeras tres capas S, I y A cumplen la función de un extractor de propiedades al ser las entradas respectivas de cada grupo de la capa I seleccionadas aleatoriamente dentro de un rectángulo posicionado aleatoriamente en la imagen (Sec. 5.2). En cambio, las salidas de la capa A y la capa R en su totalidad, cumplen la función propia del clasificador neuronal cuya estructura está basada como ya se explicó en el perceptrón. De lo que se trata es de utilizar esta segunda parte de la estructura junto con el extractor de propiedades y el codificador (Fig. 6.1). Es decir, respecto del clasificador LIRA, se sustituyen las capas S, I y A por los dos bloques previamente descritos del PCNC. Debe notarse que con esta sustitución, la capa A del clasificador neuronal pasa a contener exactamente al vector V' por lo cuál debe tener N elementos. Una vez que se ha descrito con todo detalle la estructura del PCNC se explicará su proceso de entrenamiento.

6.2. Proceso de entrenamiento

De acuerdo a las similitudes entre las estructuras del PCNC y del clasificador neuronal LIRA se tiene que la única parte de ambos que varia durante el proceso de entrenamiento son las conexiones entre las capas A y R, por lo tanto el proceso de entrenamiento descrito para el clasificador LIRA (Sec. 5.4) funciona igual para el PCNC y por lo tanto es aplicado. El proceso de codificación de las imágenes aplicado a LIRA es válido para el PCNC por lo que se aplica también. Esto se debe a que cada imagen tendrá un vector V' que codifica sus propiedades extraídas por lo cuál pueden usarse éstas a través de V' en lugar de la imagen con propósitos de entrenamiento ahorrando importantes recursos de computo y de tiempo en el proceso.

6.3. Distorsiones

De forma similar que con el clasificador neuronal LIRA y con el objetivo de ampliar el conjunto de imágenes destinadas para el entrenamiento del PCNC, se consideró la aplicación de distorsiones sobre las imágenes normalizadas originales. Atendiendo a que el PCNC no es sensible a desplazamientos cartesianos de la imagen, sólo se aplicaron distorsiones angulares (Fig. 6.12). Estas distorsiones aplicadas sobre las imágenes, representan pequeñas variaciones sobre la exacta alineación de las piezas con respecto al eje horizontal que

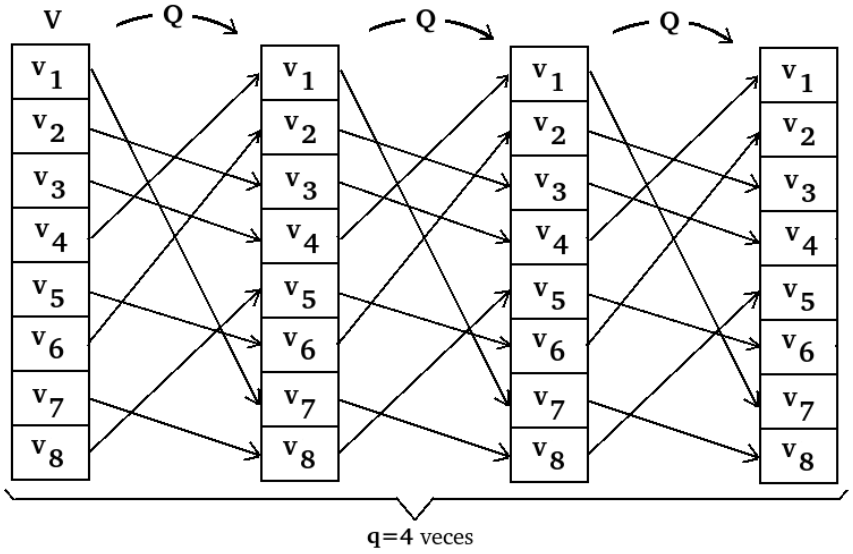


Figura 6.10: Aplicación de q permutaciones **Q** sobre el vector V.

pasa por el centro de la imagen. Por esto se realizaron experimentos con diversas distorsiones angulares. Estos experimentos se explican en el siguiente capítulo, en la Sección 7.2.4.
Para realizar estas distorsiones se ha utilizado el mismo método de creación de imágenes distorsionadas que para el clasificador LIRA. Se consideró la creación de tales distorsiones en pares, con un mismo valor absoluto angular, pero con signos opuestos.

6.4. Discusión

Se ha visto que el clasificador PCNC es igual que el LIRA en cuando a su método de clasificación, variando en ambos la forma en que se crean o extraen las propiedades de la imagen a clasificar. Es entonces en los procesos de extracción de propiedades y de codificación donde está la diferencia entre el PCNC y LIRA. Esta diferencia tiene dos características de importancia en el PCNC, una de éstas es la capacidad de aplicar y probar diversos métodos de extracción de puntos específicos y no limitarse únicamente a niveles de brillo a través de un umbral, abriendo paso de esta manera a las posibilidades que ofrece el preprocesamiento de las imágenes a clasificar; la otra característica medular es la consideración explícita que lleva a cabo el codificador sobre la posición de las propiedades encontradas en la imagen.
La principal desventaja del PCNC sobre el clasificador LIRA está en sus mayores requerimientos de cómputo debido a la necesidad de realizar gran cantidad de cálculos con vectores durante el proceso de codificación de cada imagen. Comparando la etapa de extracción de propiedades de LIRA y del PCNC se observa que ambos ubican ventanas aleatoriamente en la imagen de entrada, el número de puntos de cada una de ellas puede ser similar, sin embargo, mientras la cantidad total de ventanas aleatorias en LIRA será igual al número de grupos en la capa I, que es un parámetro de este clasificador del orden de 100000 para la tarea de reconocimiento que nos ocupa [13], se tiene que en el PCNC el número de tales ventanas será el número de puntos específicos extraídos de la imagen multiplicado por el parámetro S, y dado que este parámetro está entre 1000 y 10000 [75], tenemos que para el caso de $S = 1000$ con 100 puntos específicos igualamos los requerimientos del clasificador LIRA y en el orden en que este último número sea rebasado lo serán los requerimientos del PCNC respecto de LIRA en términos generales. Si consideramos que las imágenes normalizadas con que se ha trabajado (Sec. 4.5.1) tienen dimensión de 100×100 como mínimo, igual a 10000 píxeles se tiene que el número de puntos específicos para igualar los requerimientos de LIRA deberán ser de aproximadamente el 1 %. Esto lleva a intentar mejorar el método de selección de puntos específicos o a tener que trabajar con las imágenes normalizadas reducidas cierta escala. Por lo anterior, de usarse el tamaño original de las imágenes, el método de selección de puntos específicos por umbral arrojará gran porcentaje de puntos respecto al total de píxeles de la imagen por lo que los requerimientos computacionales y de tiempo crecerán tanto que se violarán los principios de economía establecidos para las microfábricas y

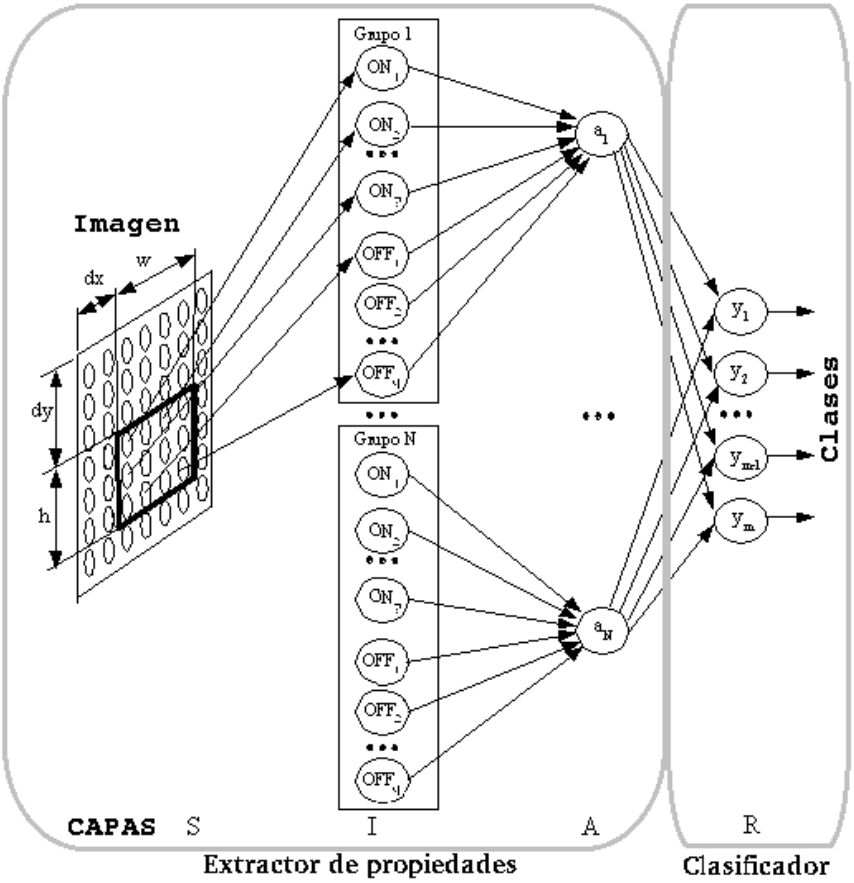


Figura 6.11: Clasificador neuronal LIRA dividido en dos partes según su función partículas: extractor de propiedades y clasificador.

Figura 6.12: Ejemplo de una distorsión angular para las imágenes del conjunto de entrenamiento destinadas al PCNC.

el SVT (Sec. 2.1.3 y 3.3). El inconveniente anterior se resolvió utilizando el umbral de gradiente de brillo (extracción de contornos), lo cual ha reducido el número de puntos específicos significativamente sin que se haya traducido en reducción de las propiedades que posibilitan la adecuada clasificación de las imágenes. Prueba de lo anterior se presenta en el capítulo siguiente y último.

Capítulo 7

Experimentos y resultados

Este capítulo final se dedica a los experimentos realizados con los clasificadores LIRA y PCNC sobre las bases de datos descritas en la Sección 4.5.2. La primera sección de este capítulo se dedica a los experimentos con el clasificador neuronal LIRA y la segunda a los experimentos con el PCNC. La tercer sección compara los resultados de ambos clasificadores y discute sobre de éstos. La cuarta y última sección se dedica a la tarea de búsqueda y localización de piezas.

En este trabajo un experimento significa una serie de pruebas con el clasificador LIRA o PCNC encaminadas a probar o concluir una propiedad u objetivo particular del mismo.

Para ambos clasificadores neuronales se realizaron múltiples experimentos con las bases de datos descritas. Los experimentos llevados a cabo han sido objetivos, sistemáticos y estadísticamente convincentes. Los experimentos son objetivos por que parten de un plan predeterminado y bien definido para su realización el cuál se explica en breve. Son sistemáticos por que se han hecho de forma ordenada y organizados en etapas, utilizando cuando es necesario los resultados obtenidos en los experimentos previos en los subsecuentes. Y por último los experimentos son estadísticamente convincentes por que se ha tenido cuidado de ejecutar múltiples pruebas agrupadas en experimentos con metodología idéntica.

Todos los experimentos descritos en este trabajo se ejecutaron en un computador con procesador Intel© Pentium© 4 a 2.80 GHz con 512 KB de memoria caché y 512 MB de memoria RAM con sistema operativo GNU/Linux kernel 2.6.17-11-generic.

7.1. LIRA

Primero que nada es importante señalar que dada la estructura del clasificador LIRA (Sec. 5.2), el orden de magnitud en los valores prácticos de algunos de sus parámetros ($W \times H$, $w \times h$ y N), y por el carácter aleatorio de su construcción es sumamente improbable que dos estructuras idénticas LIRA sean creadas, esto considerando idénticos parámetros de creación. Por lo anterior es importante mencionar cuando un experimento utilizó un mismo clasificador LIRA (misma creación) y cuando se utilizaron distintos clasificadores con idénticos parámetros. Adicionalmente debido al proceso de entrenamiento del clasificador LIRA (Sec. 5.4), se tiene que un mismo clasificador puede estar entrenado con diversos conjuntos de entrenamiento y diversos ciclos de entrenamiento, por lo cuál aún copias idénticas de un mismo clasificador pueden presentar respuestas distintas. Tener presente lo anterior ha sido importante para la planeación de los experimentos realizados y comprende varios resultados que se mostrarán más adelante.

Cada una de las bases de datos con imágenes empleadas consiste en dos conjuntos fijos de imágenes, uno para entrenamiento y otro para prueba. Estos conjuntos se seleccionaron aleatoriamente de la base de datos respectiva. Adicionalmente, las bases de datos descritas pueden tener imágenes para propósitos especiales, las cuales se explican en los experimentos que las ocupan. Para ciertos experimentos realizados se utilizaron los conjuntos fijos de entrenamiento, mientras que para otros ambos conjuntos se seleccionaron aleatoriamente de entre toda la base de datos respectiva. Todos los experimentos realizados con el clasificador LIRA utilizaron las bases de datos A, B y D con excepción de los experimento sobre distorsiones y sobre conjunto ampliado de entrenamiento.

Los experimentos realizados, su justificación, metodología y resultados se abordan en las secciones siguientes.

7.1.1. Experimentos preliminares

Las primeras pruebas con el clasificador LIRA han sido reportadas en [13]. Estas pruebas utilizaron la base de datos α descrita en la Sección 4.5.2. Se probaron varias combinaciones de parámetros para el clasificador considerando los mejores parámetros reportados en [15]. En este grupo de pruebas no se utilizaron distorsiones para el conjunto de imágenes de entrenamiento. En la Tabla 7.1 se muestran los resultados de este experimento. El mejor resultado obtenido en el experimento fue con una ventana LIRA de 15×15 píxeles, 175 000 neuronas en la capa A, cuatro neuronas ON y tres neuronas OFF en cada grupo, el parámetro η del clasificador igual a 1.0 y el parámetro de entrenamiento T_E igual a 0.15. Para estos parámetros el porcentaje de neuronas activas en la capa A fue 0.164 %. Para esta prueba el porcentaje de reconocimiento correcto fue de 94 %. El mejor desempeño obtenido para múltiples combinaciones de parámetros fue alcanzado con 40 ciclos de entrenamiento por lo que los mismos se utilizaron para todas las pruebas del experimento descrito.

Tabla 7.1: Experimento preliminar con nueve pruebas para la sintonización de parámetros LIRA con la base de datos α.

No. de experimentos	1	2	**3**	4	5	6	8	9
Tamaño de la ventana ($w \cdot h$)	$12 \cdot 12$	$15 \cdot 15$	$\mathbf{15 \cdot 15}$	$10 \cdot 10$	$13 \cdot 13$	$17 \cdot 17$	$15 \cdot 15$	$10 \cdot 10$
Constante LIRA (η)	0.8	1.0	**1.0**	1.0	1.0	1.0	1.0	1.0
Neuronas capa A (N) (miles)	175	175	**175**	175	175	175	200	200
Neuronas ON por grupo (p)	3	3	**4**	4	4	5	4	4
Neuronas OFF por grupo (n)	4	4	**3**	3	3	3	3	3
Neuronas activas (%)	0.06	0.09	**0.16**	0.12	0.13	0.08	0.17	0.15
Porcentaje de reconocimiento (%)	68	89	**94**	93	85	88	89	89

Luego de múltiples pruebas preliminares y otras adicionales se determinó que el parámetro de entrenamiento T_E igual a 0.15 es el que mejor rendimiento ofreció para todos los casos y bases de datos por lo que se fijo al valor dado.

Los experimentos de las secciones siguientes se realizaron teniendo como base el mejor conjunto de parámetros obtenido en este experimento preliminar, es por ello que en lo sucesivo se les hará referencia a este conjunto de parámetros como parámetros base.

7.1.2. Unicidad

Al inicio de esta sección se ha hecho mención de la característica única de cada construcción de un determinado clasificador LIRA, aún con los mismos parámetros. Para estudiar cuál es el comportamiento de tales clasificadores construidos así se realizó el experimento descrito a continuación. Este experimento consistió en una serie de pruebas con distintas construcciones LIRA utilizando los mismos parámetros y utilizando los mismos conjuntos fijos de entrenamiento y prueba, siendo entrenadas con el mismo número de ciclos de entrenamiento. Estas consideraciones llevan a que la única variable del experimento es la estructura única de cada construcción LIRA. Este experimento se describe en primer lugar ya que para experimentos posteriores donde se utilizan distintas construcciones del clasificador LIRA además de la variable que se pretenda estudiar se tendrá inevitablemente la variabilidad de la estructura particular de cada construcción LIRA. Por esta razón a este experimento se le llama de unicidad, derivado del hecho de que cada clasificador LIRA es único.

En la Tabla 7.2 se muestran los resultados obtenidos para 10 pruebas realizadas con los parámetros base[1]: $w = 15$, $\eta = 1{,}0$, $N = 175000$, $p = 4$ y $n = 3$ y 40 ciclos de entrenamiento. Adicionalmente se presenta el promedio de reconocimiento obtenido y la desviación estándar para mejor comprensión de estos datos.

De los resultados mostrados se tiene que la estructura única de cada clasificador particular puede arrojar resultados considerablemente distintos por lo cuál es importante considerar esto para obtener un clasificador LIRA particular que arroje los mejores resultados para una base de datos de trabajo determinada. El experimento de aleatoriedad de los conjuntos de entrenamiento y prueba será importante para ser considerado junto con el de unicidad.

[1]Tanto por el clasificador LIRA como para el PCNC la ventana respectiva es cuadrada, por lo que el valor del parámetro h es igual al de w. Por esta razón de aquí en adelante se da sólo el valor para w.

Tabla 7.2: Resultados del experimento con 10 construcciones LIRA para cada una de las tres bases de datos. Cada una de las 10 corridas se hizo con idénticos parámetros, mismos conjuntos fijos de entrenamiento y prueba, y mismo número de ciclos de entrenamiento. Todos los resultados se dan en unidades porcentuales que indican el reconocimiento logrado.

LIRA:	1	2	3	4	5	6	7	8	9	10	$\bar{x}$	σ
BD-A	87	86	91	89	86	90	81	90	93	84	87.7	3.14
BD-B	93	90	90	92	91	91	92	91	93	91	91.4	1.02
BD-D	87	89	88	89	84	88	89	88	89	89	88.0	1.48

7.1.3. Sintonización de parámetros

El segundo experimento estuvo encaminado a obtener el mejor conjunto de parámetros del clasificador LIRA para cada una de las bases de datos empleadas. Por mejor conjunto de parámetros debe entenderse aquellos que permitan un máximo porcentaje de reconocimiento del clasificador al ser entrenado y probado sobre los conjuntos de entrenamiento y prueba.
Con los parámetros base previamente mencionados se creó un clasificador LIRA y se entrenó y probó con los conjuntos fijos para entrenamiento y prueba. Se entrenó con 40 ciclos. Estos parámetros dieron un porcentaje de reconocimiento de 88.6 % promedio para cinco pruebas.
Partiendo de este conjunto de parámetros base se realizaron diversos experimentos para estudiar el comportamiento que cada uno de los parámetros del clasificador LIRA tenia sobre su resultado (porcentaje de reconocimiento). Para esto se realizaron diversas pruebas para cada uno de los parámetros del clasificador por separado. Para el estudio del comportamiento de cada parámetro se modificó únicamente el valor del mismo con respecto al conjunto base. Cada una de las pruebas de cada uno de los parámetros se realizó con un clasificador distinto ya que no es posible modificar los parámetros de un clasificador LIRA ya creado. La excepción a esto es el número de ciclos de entrenamiento, por lo tanto se le deja a la siguiente sección el análisis de éste parámetro que a demás no es propiamente un parámetro del clasificador sino de su entrenamiento. Un experimento adicional fue el realizado conjuntamente para los parámetros p y n, ya que estos están intrínsecamente relacionados, se estudiaron también en conjunto realizando distintas pruebas para varios valores fijos de $p + n$. La razón de utilizar en todos estos experimentos los conjuntos fijos de entrenamiento y prueba ha sido no introducir una variable más que repercutiera en los experimentos. Luego entonces las variables que afectaron estos experimentos fueron el o los parámetros modificados y el hecho de que cada prueba requirió una construcción nueva de un clasificador, lo cuál genera resultados considerablemente variables como se vio en el experimento de unicidad. Los resultados obtenidos se muestran a continuación. El mejor resultado para cada caso se presenta en negritas.
En la Tabla 7.3 tenemos el experimento con la variación del parámetro de ventana w para cada una de las bases de datos utilizadas. Para las bases de datos A y B los mejores resultados fueron para $w = 15$ mientras que para la base de datos D $w = 17$ logró el mejor resultado.

Tabla 7.3: Resultados de pruebas del clasificador LIRA con variación del parámetro de dimensión de la ventana LIRA w sobre las bases de datos A, B y D.

Parámetro	w	9	11	13	15	17	19	21
BD-A	%	81	81	80	**89**	86	85	85
BD-B	%	82	78	78	**94**	82	80	88
BD-D	%	83	85	87	85	**89**	88	86

En la Tabla 7.4 se presenta el experimento con el parámetro η, aquí los resultados no son uniformes pues mientras para la base de datos A el mejor valor fue $\eta = 0{,}90$ para la base de datos B fue de $\eta = 1{,}00$ y para la base de datos D hubo un empate entre $\eta = 0{,}80$ y $\eta = 0{,}90$.
Las pruebas para el parámetro N de LIRA se presentan resumidas en la Tabla 7.5. Se tiene que para todas las bases de datos los mejores resultados están en valores de N entre 175 000 y 200 000. Esta variable es muy importante ya que el valor de N es directamente proporcional a los tiempos y recursos necesarios para correr un determinado clasificador LIRA. Es por esta razón que ante dos resultados iguales se debe tomar el menor valor del parámetro correspondiente.
Los experimentos con los parámetros p y n se hicieron por separado y luego en conjunto, ya que como se ha explicado éstos están íntimamente ligados en la obtención de características de la imagen. En las

Tabla 7.4: Resultados de pruebas del clasificador LIRA con variación del parámetro η sobre las bases de datos A, B y D.

Parámetro	η	0.80	0.90	0.95	1.00
BD-A	%	90	**92**	90	89
BD-B	%	86	83	84	**94**
BD-D	%	**88**	**88**	86	85

Tabla 7.5: Resultados de pruebas del clasificador LIRA con variación del parámetro N sobre las bases de datos A, B y D.

Parámetro	N	130 000	150 000	175 000	200 000	220 000
BD-A	%	81	89	89	**90**	88
BD-B	%	76	83	**94**	80	83
BD-D	%	84	85	85	**88**	88

Tablas 7.6, 7.7 y 7.8 se presentan los resultados de las pruebas estos tres experimentos. Puede verse que estos parámetros tienen uniformidad en sus respuestas ya que para los experimentos individuales, p tuvo los mejores valores entre 4 y 5 y n entre 2 y 3. Sin embargo al analizar dichos parámetros en conjunto se tiene que para las bases de datos A y B los mejores resultados se alcanzaron para $p = 3$, $n = 4$ y $p = 4$, $n = 3$ respectivamente. Mientras que para la base de datos D hubo un empate entre los parámetros con $p = 4$, $n = 2$ y $p = 5$, $n = 3$.

Tabla 7.6: Resultados de pruebas del clasificador LIRA con variación del parámetro p sobre las bases de datos A, B y D.

Parámetro	p	2	3	4	5	6
BD-A	%	85	88	**89**	83	86
BD-B	%	79	69	**94**	82	85
BD-D	%	86	85	85	**89**	86

Tabla 7.7: Resultados de pruebas del clasificador LIRA con variación del parámetro n sobre las bases de datos A, B y D.

Parámetro	n	1	2	3	4	5	6
BD-A	%	35	78	**89**	86	83	64
BD-B	%	73	82	**94**	83	88	80
BD-D	%	85	**86**	85	80	83	78

El análisis de los datos obtenidos da muestra de la relación de cada uno de los parámetros del clasificador LIRA sobre su resultado. Se ha visto que los parámetros w y N influyen en el resultado para cada base de datos en forma mas o menos uniforme mientras que p, n y η no lo hacen.
Ahora bien, debido a la interdependencia que los parámetros juegan en el desempeño del clasificador LIRA, debe dudarse que los mejores parámetros obtenidos de forma individual garanticen que un conjunto formado por los mismos sea el que mejor resultados proporcione. Para esto es necesario realizar otro experimento en donde se prueben diversos conjuntos de parámetros que tengan como base los resultados obtenidos tanto previamente como en esta sección. Por lo tanto se han probado tres conjuntos de parámetros sobre la base de datos A, el primer conjunto es el de los parámetros obtenidos en la Sección 7.1.1, el segundo es el de los parámetros que dieron el mejor resultado en el experimento de parámetros independientes y el tercer conjunto es el construido con los parámetros que fueron mejores en los experimentos individuales. Para este experimento se han hecho cinco distintas construcciones LIRA para cada conjunto de parámetros, para obtener el promedio de reconocimiento correcto para cada uno en lugar de conformarse con un sólo resultado, pues como se explicó en la Sección 7.1.2 diversas construcciones LIRA con idénticos parámetros

Tabla 7.8: Resultados de pruebas del clasificador LIRA con variación de los parámetros p y n sobre las bases de datos A, B y D para distintos valores de $p+n$.

Parámetros $p/n,\ p+n=6$	1/5	2/4	3/3	4/2	5/1	-	-
BD-A (%):	63	83	82	78	33	-	-
BD-B (%):	82	88	91	92	84	-	-
BD-D (%):	77	85	85	**87**	80	-	-
Parámetros $p/n,\ p+n=7$	1/6	2/5	3/4	4/3	5/2	6/1	-
BD-A (%):	33	73	**90**	89	66	43	-
BD-B (%):	80	89	88	**94**	91	87	-
BD-D (%):	74	81	84	85	83	81	-
Parámetros $p/n,\ p+n=8$	1/7	2/6	3/5	4/4	5/3	6/2	7/1
BD-A (%):	43	58	80	83	86	72	55
BD-B (%):	78	80	87	83	86	85	82
BD-D (%):	72	82	81	85	**87**	78	80

dan resultados con cierta variación. Nótese que en la Tabla 7.4 se reporta el mejor resultado obtenido en estos experimentos de sintonización. Este resultado es de 92 % de reconocimiento correcto. En la Tabla 7.9 se resumen las pruebas realizadas para este experimento con las cinco corridas para cada uno de los 3 conjuntos de parámetros descritos, los resultados obtenidos y el promedio obtenido para cada caso.

Tabla 7.9: Resumen de resultados de pruebas para obtener el mejor conjunto de parámetros LIRA para la base de datos A. $\bar{x}$ es el promedio y σ es la desviación estándar.

Parámetros	Valores					Construcciones/(%)					$\bar{x}$	σ
	w	η	N	p	n	1	2	3	4	5	(%)	(%)
Base	15	1.0	175 000	4	3	81	89	91	92	90	88.6	3.93
Mejor en pruebas	15	0.9	175 000	4	3	85	91	87	88	91	88.4	2.33
Mejores aislados	15	0.9	200 000	3	4	78	85	89	85	90	85.4	4.22

De estas pruebas queda claro que el conjunto de parámetros formado por los mejores parámetros obtenidos en los experimentos con variación de parámetros aislados no dan los mejores resultados, en cambio, los otros dos conjuntos de parámetros han dado resultados muy similares, con sólo 0.2 % de diferencia entre los resultados. Para seleccionar uno de ellos se ha calculado la desviación estándar, lo que permite conocer en que medida varían los datos con respecto a su promedio. Atendiendo a este criterio puede seleccionarse el segundo conjunto de parámetros de la Tabla 7.9 llamado *mejor en pruebas* como aquel con mayor confianza en crear un clasificador neuronal LIRA destinado a la base de datos A. Sin embargo el clasificador creado que alcanzó un mejor resultado individual se obtuvo con los parámetros base por lo éste ha sido usado para los experimentos siguientes que requieren un clasificador LIRA ya creado.

Respecto a la base de datos B se aplicaron pruebas similares resultando que el mismo conjunto de parámetros que alcanzó el mejor desempeño en la base de datos A lo hizo también para esta base de datos. Este resultado es muy importante, pues señala que un conjunto determinado de parámetros es capaz de alcanzar los mejores resultados en al menos dos bases de datos distintas.

Para la base de datos D los resultados fueron un tanto distintos, los parámetros que alcanzaron un mejor porcentaje de reconocimiento fueron $w = 17$, $\eta = 0{,}90$, $N = 200000$, $p = 4$ y $n = 2$ con 40 ciclos de entrenamiento. Cabe destacar las diferencias entre esta base de datos y las anteriores, por un lado las imágenes se presentan más complicadas ya que están amontonadas y todos los tipos de piezas están revueltas, además debido a la menor cantidad de imágenes de esta base de datos el porcentaje utilizado para el conjunto de entrenamiento respecto al total no fue de 50 % sino de 72.72 % como se explicó en la sección respectiva.

7.1.4. Distorsiones

Con el objetivo de aumentar el porcentaje de reconocimiento del clasificador LIRA se realizó el siguiente experimento que consistió en aumentar el conjunto de entrenamiento añadiendo imágenes distorsionadas generadas a partir de las imágenes originales de este conjunto. Las distorsiones aplicadas se han explicado en forma general en la Sección 5.5. La idea de incluir tales distorsiones es, además de aumentar el número de imágenes en el conjunto de entrenamiento, el de dar más capacidad al clasificador LIRA para reconocer mayor porcentaje de muestras que estén ligeramente desplazadas o rotadas con respecto al centro de la imagen, como es la característica de las imágenes empleadas para todas las bases de datos. Sin embargo para la mayoría de las distorsiones aplicadas resulta un porcentaje de reconocimiento malo, inferior al 70 % o menos. Sólo dos pruebas lograron resultados aceptables pero ninguna hizo posible mejorar el desempeño del clasificador. Estas pruebas se realizaron con el mejor clasificador LIRA creado para la base de datos A cuyos parámetros son los parámetros base ya descritos. La primera prueba consistió en aplicar sólo dos distorsiones de cada tipo (horizontal, vertical y angular) para cada una de las imágenes del conjunto de entrenamiento. Estas distorsiones fueron las mínimas posibles, de sólo un píxel o un grado en cada uno de ambos sentidos posibles. La segunda prueba se realizó con cuatro distorsiones para cada uno de los tipos posibles. En la Tabla 7.10 se resumen estas pruebas.

Tabla 7.10: Experimento usando distorsiones para aumentar el conjunto de entrenamiento del clasificador LIRA aplicado sobre la base de datos A.

	Número de prueba	
	1	2
Número de distorsiones	6	12
distorsiones horizontales	+1, -1	+2, +1, -1, -2
distorsiones verticales	+1, -1	+2, +1, -1, -2
distorsiones angulares	+1°, -1°	+2°, +1°, -1°, -2°
Ciclos de entrenamiento	50	40
Porcentaje de reconocimiento (%)	91	91

Estos resultados obtenidos se deben al carácter normalizado de las imágenes del conjunto de entrenamiento, si bien las imágenes de las bases de datos utilizadas no están perfectamente alineadas, si están en lo general bien orientadas por lo que añadir imágenes con desplazamientos mayores induce confusión durante el entrenamiento del clasificador LIRA. En términos generales la respuesta del clasificador para esta tarea de reconocimiento de piezas es mejor sin distorsiones.

7.1.5. Ciclos de entrenamiento

El siguiente experimento descrito tuvo por objetivo encontrar el mejor número de ciclos de entrenamiento necesarios para que un determinado conjunto de parámetros LIRA logre el mejor resultado con una base de datos determinada. Para la realización de este experimento se realizaron cuatro pruebas con igual número de clasificadores construidos a partir de los mejores parámetros obtenidos en la sección previa. Es importante repetir que el proceso de entrenamiento cambia la memoria de un clasificador LIRA, sin embargo ni el entrenamiento ni el borrado total de esta memoria alteran la estructura de un mismo clasificador [2]. De las cuatro pruebas hechas, dos ocuparon los conjuntos fijos de entrenamiento y prueba de forma de no introducir una variable adicional al experimento. Dos más utilizaron distintas distorsiones, las cuales se describieron en la sección anterior. Para cada clasificador se ejecutaron paulatinamente diversos ciclos de entrenamiento empezando por 10 e incrementándolos por la misma cantidad hasta 100, Los resultados obtenidos dan cuenta de que un valor asintótico es alcanzado desde antes de los 70 ciclos por lo cuál el resto no es tomado en cuenta. Estos resultados obtenidos se presentan en la Tabla 7.11.
Se observa que para dos clasificadores distintos con idénticos parámetros, idéntico número de ciclos de entrenamiento e idénticos conjuntos de entrenamiento y prueba, como es el caso de los pares de pruebas presentados (pruebas 1 y 2, 3 y 4, etc.) se alcanzan resultados distintos como se vio en la Sección 7.1.2. Sin

[2]Debe tenerse en cuenta que la memoria de un clasificador neuronal LIRA está en los pesos entre las conexiones de dos de sus capas los cuales se representan por números. De acuerdo a su implementación cambiar estos valores no altera su estructura, sin embargo, en una red neuronal natural estas conexiones realmente se forman mediante nuevas conexiones físicas de las dendritas, por lo que su estructura si cambia.

Tabla 7.11: Experimento con diversos ciclos de entrenamiento sobre ocho clasificadores LIRA. Para cada base de datos los parámetros son los mismos. Las pruebas se realizaron con las bases de datos A, B y D.

		No. de ciclos de entrenamiento						
Prueba	BD	10	20	30	40	50	60	70
1	A	33	67	78	**93**	93	93	93
2	A	45	60	75	84	**90**	90	90
3	A (Distorsiones 1)	73	76	73	89	**91**	91	91
4	A (Distorsiones 2)	68	84	89	**91**	90	90	90
5	B	63	75	82	77	86	90	**93**
6	B	57	67	73	81	**91**	89	**91**
7	D	70	85	**89**	89	89	89	89
8	D	71	87	**89**	89	89	89	89

embargo lo que aquí nos ocupa es conocer la relación asintótica de la respuesta con respecto a los ciclos de entrenamiento. Puede verse en la tabla presentada que las ocho pruebas convergieron en distinto número de ciclos. Este número de ciclos varió drásticamente con cada base de datos distinta. Así para la base de datos A tanto para las pruebas normales como para las que usaron distorsiones alcanzaron su máxima respuesta entre los 40 y 50 ciclos de entrenamiento. Para la base de datos B ambas pruebas alcanzaron estabilidad en su respuesta hasta los 70 ciclos. Para la base de datos D el reconocimiento máximo se alcanzó desde los 30 ciclos de entrenamiento. Como casos especiales se tiene la prueba 6 donde el resultado cayó dos puntos para 60 ciclos para luego recuperarse.
Conocer este comportamiento y poder predecirlo es importante ya que cada ciclo adicional de entrenamiento consume recursos y tiempo, lo cuál es un factor determinante para el desempeño del sistema de reconocimiento.

7.1.6. Aleatoriedad de los conjuntos para entrenamiento y prueba

Para mostrar la confiabilidad y el comportamiento del clasificador LIRA sobre una base de datos determinada se realizó este experimento. El objetivo ha sido mostrar los resultados con un mismo clasificador entrenado y probado con distintos conjuntos de entrenamiento y prueba. Estos conjuntos fueron tomados aleatoriamente de la base de datos a 50 % para cada conjunto para las bases de datos A y B y para la base de datos D se tomaron 72 % de la imágenes para entrenamiento y el resto para prueba. En la Tabla 7.12 tenemos el resultado de efectuar 20 corridas distintas además del promedio obtenido, el mejor y peor resultado así como la desviación estándar de los datos.

7.1.7. Mejores clasificadores LIRA

Enfocándose en el uso práctico que se pretende dar al sistema de reconocimiento el propósito de esta sección es explicar las características concretas de los mejores clasificadores LIRA para cada una de las bases de datos utilizadas. Un resumen de estos resultados se presenta en la Tabla 7.13. Para su adecuada interpretación consúltese las tablas 4.1 y 4.2. En este resumen se muestran los parámetros empleados en cada caso así como el resultado porcentual exacto obtenido en el reconocimiento, un desglose del número de errores por cada una de las clases utilizadas así como la contribución porcentual de cada clase en el error total de reconocimiento. Los percentiles totales no suman 100 % debido al uso de aproximación a dos dígitos decimales.

7.1.7.1. Base de datos A

Para la prueba de reconocimiento sobre la base de datos A con el mejor clasificador LIRA obtenido se alcanzaron los siguientes resultados. Las imágenes de cinco clases (cono, eje de rotor, no pieza central, terminal de cable y tornillo allen) fueron reconocidas con un porcentaje de aciertos de 100 %. Las imágenes de las otras tres clases (base de tubito, tornillo de cabeza cónica y tornillo cabeza redonda) fueron reconocidas con seis, uno y tres errores respectivamente. Estas últimas piezas son muy similares en tamaño y forma.

Tabla 7.12: Resultados de pruebas con el clasificador LIRA con conjuntos de entrenamiento y prueba seleccionados aleatoriamente para las bases de datos A, B y D. Para cada base de datos se utilizó un mismo clasificador.

Prueba No.:	1	2	3	4	5	6	7	8	9	10
BD-A:	88	88	81	88	90	88	90	88	93	86
BD-B:	90	90	91	93	93	89	90	90	92	93
BD-D:	87	89	90	88	90	79	91	92	83	86
Prueba No.:	11	12	13	14	15	16	17	18	19	20
BD-A:	90	83	88	80	86	90	84	91	86	91
BD-B:	95	93	89	93	90	94	95	89	87	91
BD-D:	89	85	90	85	88	91	81	85	87	85
BD-A										
Mejor: 93 %	Peor: 80 %									
Promedio: 87.4 %	Desviación estándar: 3.4 %									
BD-B										
Mejor: 95 %	Peor: 87 %									
Promedio: 91.4 %	Desviación estándar: 2.15 %									
BD-D										
Mejor: 92 %	Peor: 79 %									
Promedio: 87.1 %	Desviación estándar: 3.37 %									

Tabla 7.13: Resumen de resultados de los mejores clasificadores LIRA para las bases de datos utilizadas. T: ciclos de entrenamiento. R: porcentaje de reconocimiento.

Base de datos:	Parámetros					T	R	Número de errores por clase/Contribución del total (%)							
	w	η	N	p	n		(%)	1	2	3	4	5	6	7	8
A	15	1.0	175 000	4	3	40	93.75	6/3.75	0	0	0	0	0	1/0.63	3/1.88
B	15	1.0	175 000	4	3	70	94.14	3/1.10	1/0.37	1/0.37	0	3/1.10	3/1.10	5/1.83	-
D	17	0.9	200 000	4	2	70	90.47	2/1.90	2/1.90	0	0	2/1.90	4/3.81	0	-

Dos ejemplos de cada clase de las imágenes reconocidas para esta base de datos se muestran en la Fig. 7.1. Las 10 imágenes mal reconocidas se presentan en la Fig. 7.2 para poder evaluar su similitud.

7.1.7.2. Base de datos B

Con la base de datos B se tuvo un resultado de solo 0.39 % por arriba del respectivo para la base de datos A. Los parámetros que alcanzaron tal resultado son los mismos que para la base de datos A. Estos parámetros son para ambas bases de datos los mejores encontrados. Una diferencia particular es que esta base de datos requirió de 70 ciclos de entrenamiento para alcanzar su máxima eficiencia mientras que la base de datos A lo hizo con sólo 50. Esta base de datos contiene 70 % más imágenes que la A. Ejemplos de las imágenes reconocidas correctamente para esta base de datos se presentan en la Fig. 7.3. Las 16 imágenes que no fueron reconocidas correctamente se presentan en la Fig. 7.4. En el reconocimiento de LIRA sobre la base de datos B existió más uniformidad en los errores ya que seis de las siete clases tuvieron algún error, siendo la única clase que no tuvo errores la número 4 (tornillo allen grande).

7.1.7.3. Base de datos D

Para la base de datos D con 105 imágenes en el conjunto de prueba 10 imágenes no se reconocieron correctamente. Algunos ejemplos de imágenes que se reconocieron bien se presentan en la Fig. 7.5. En estos resultados se muestra la capacidad de LIRA de reconocer piezas parcialmente ocluidas además de su flexibilidad para hacerlo con ejemplos ligeramente girados con respecto al plano de vista. Las imágenes que no se reconocieron correctamente se presentan en la Fig. 7.6. Estas imágenes mal reconocidas se distribuyen de la siguiente forma entre las clases: dos imágenes de arandela, dos de no pieza central, dos de tornillo

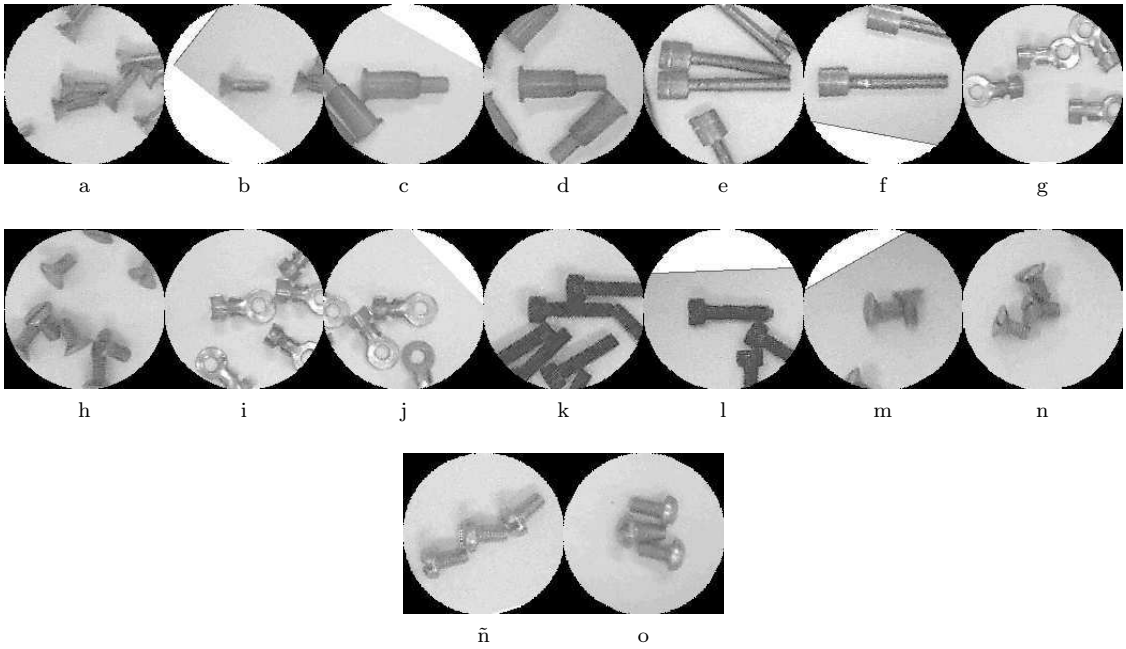


Figura 7.1: Ejemplos de imágenes correctamente reconocidas con el mejor clasificador LIRA para la base de datos A. Se muestran dos ejemplos por cada clase.

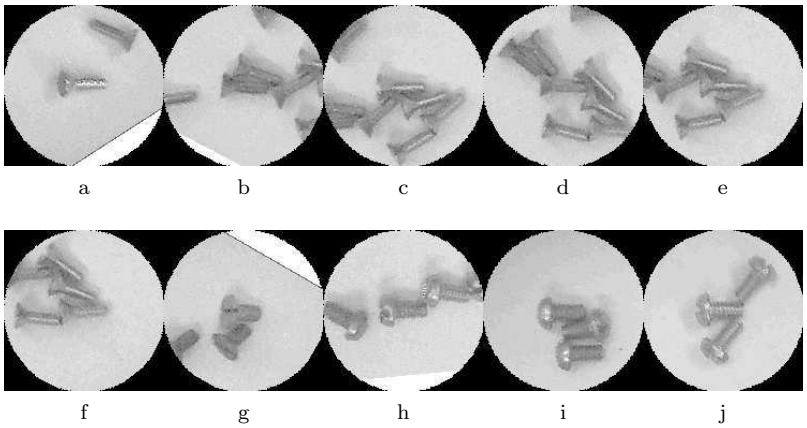


Figura 7.2: Las 10 imágenes mal reconocidas con el mejor clasificador LIRA para la base de datos A.

gota y cuatro de tornillo plano. El resto de las clases, tornillo allen chico, tornillo allen grande y tuerca se reconocieron con 100 % de efectividad.

7.1.8. Prueba con conjunto de entrenamiento ampliado

Una prueba adicional se realizó ampliando el conjunto de entrenamiento de la base de datos D con imágenes extras que se tenían disponibles las cuales no fueron incluidas originalmente ya que todas las bases de datos se construyeron con un número uniforme de imágenes para todas las clases utilizadas. Se agregaron 537 imágenes a las 280 existentes en el conjunto de entrenamiento. Estas imágenes adicionales se distribuían entre las clases de la siguiente forma: 91 imágenes de arandelas, 35 de tornillos allen chico, 0 para tornillo

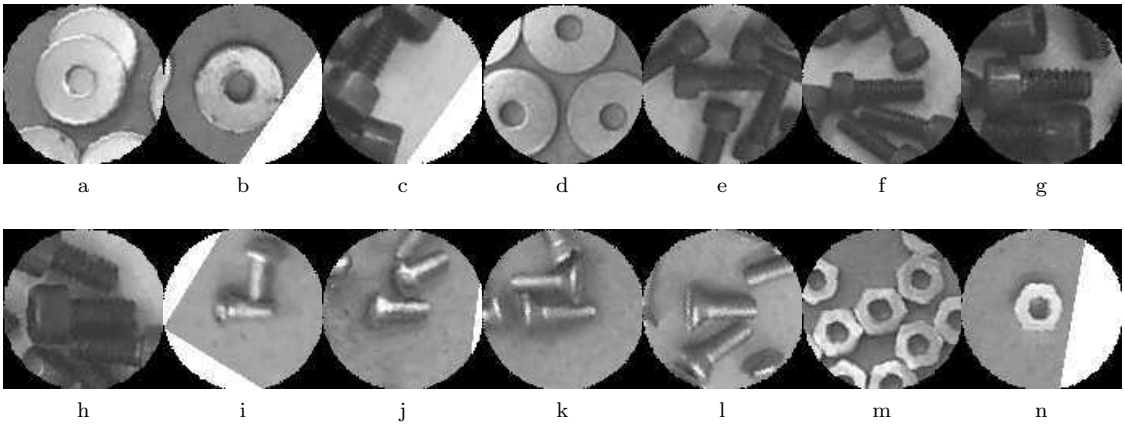


Figura 7.3: Ejemplos de imágenes correctamente reconocidas con el mejor clasificador LIRA para la base de datos B. Se muestran dos ejemplos por cada clase.

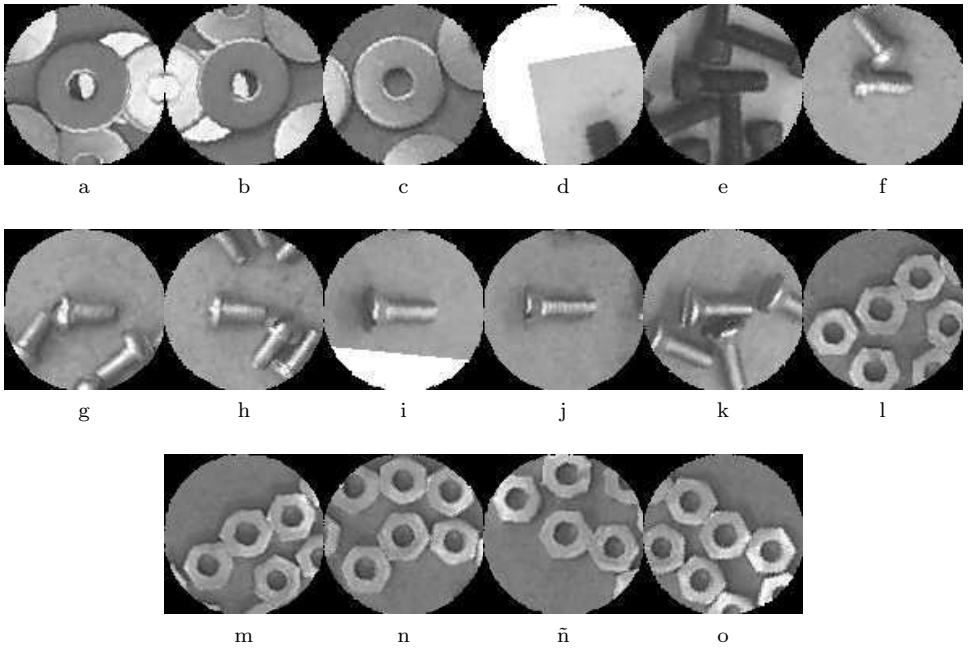


Figura 7.4: Las 16 imágenes mal reconocidas con el mejor clasificador LIRA para la base de datos B.

allen grande[3], 22 de tornillos de cabeza de gota, 18 de tornillos de cabeza plana, 79 de tuercas y 292 imágenes para ausencia de pieza en el centro de la imagen. El resultado de esta prueba fue muy satisfactorio para esta base de datos que notablemente es la más representativa de un caso real de aplicación. Es por ello que a esta prueba única se le ha dado una sección propia. El porcentaje de reconocimiento obtenido fue de 94 %.

[3]El hecho de no tener más imágenes de tornillos allen grande fue el impedimento para que la base de datos D tuviese mayor cantidad de imágenes.

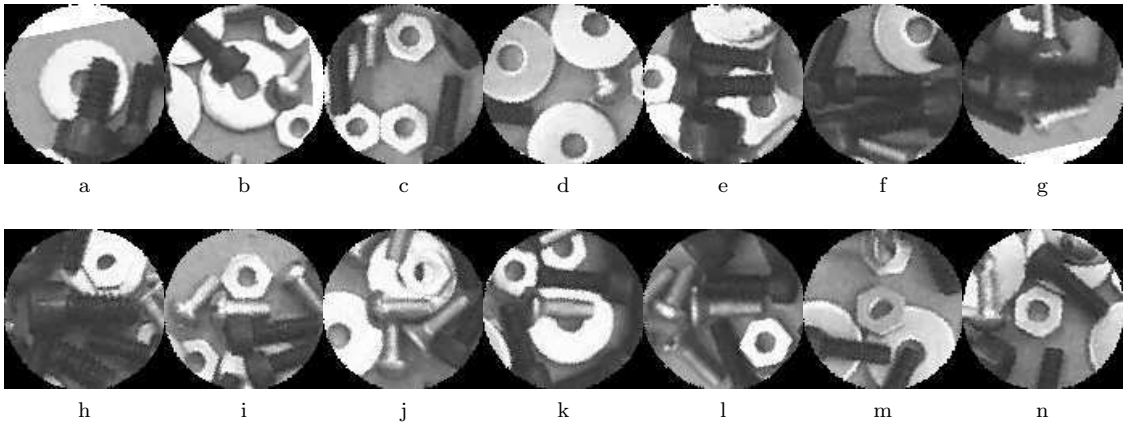


Figura 7.5: Ejemplos de imágenes correctamente reconocidas con el mejor clasificador LIRA para la base de datos D. Se muestran dos ejemplos por cada clase.

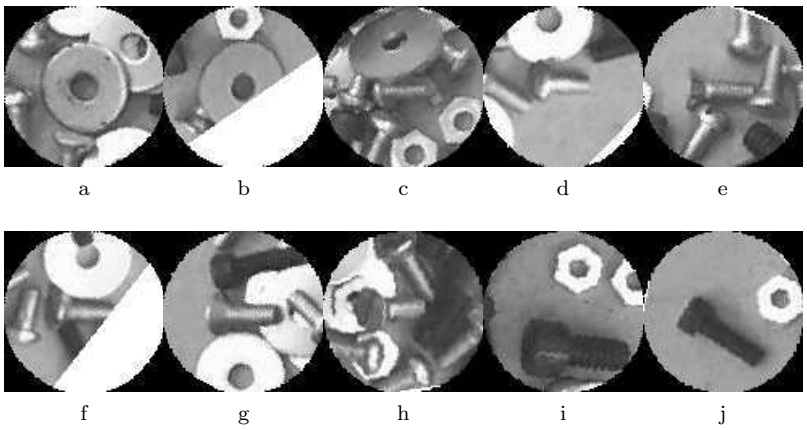


Figura 7.6: Las 10 imágenes mal reconocidas con el mejor clasificador LIRA para la base de datos D. Dos arandelas, dos tornillos de cabeza de gota, cuatro tornillos de cabeza plana y dos imágenes con ausencia de pieza en el centro de la imagen.

7.2. PCNC

De forma similar a lo que se hizo con el clasificador LIRA, se aplicaron una serie de experimentos con el PCNC sobre las bases de datos de imágenes creadas para este trabajo. Además, de acuerdo a la estructura del PCNC (Sec. 6.1) tenemos una característica a destacar que comparte con el clasificador LIRA. El primer bloque del PCNC es un extractor de propiedades cuya estructura tiene un carácter único debido a la aleatoreidad con que son definidas las propiedades a utilizar. Este carácter único existe para distintas creaciones de un PCNC aún con idénticos parámetros. Respecto al último bloque de la estructura PCNC se ha descrito (Sec. 6.1.3) que es idéntica a la estructura respectiva del clasificador LIRA así como el proceso de entrenamiento. Por esta razón existen similitudes con LIRA, sin embargo la clave en la diferencia como ya se ha dicho está en la extracción de las propiedades que pasan al clasificador neuronal en cada caso.

Cabe destacar que los experimentos realizados con el PCNC se aplicaron sobre las mismas bases de datos que para LIRA con las mismas características. A continuación se describen los experimentos realizados y se presentan los resultados obtenidos.

7.2.1. Experimentos preliminares

Para iniciar los experimentos con el PCNC se tomó como punto de partida los parámetros reportados para la tarea de reconocimiento de caracteres manuscritos [77]. Las principales diferencias entre esta tarea y la de reconocimiento de piezas que ocupa este trabajo son tres. La primera es que para los caracteres se tienen imágenes aisladas mientras que para las piezas éstas frecuentemente están junto a otras en las imágenes[4]. La segunda tiene que ver con la gama de brillos de las imágenes originales, mientras que para los caracteres estos se generan con un solo color de tinta lo que se traduce en una gama de brillos pequeña, las piezas tienen una amplia gama de tonalidades lo que se traduce en una gamma completa de brillos. La tercer diferencia es el tamaño de las imágenes utilizadas, mientras que para los caracteres no se requieren mas que imágenes de unas pocas decenas de píxeles por lado, para las piezas se utilizaron al menos un centenar. Lo anterior es de tomarse en cuenta ya que los parámetros utilizados en aquella tarea seguramente no serán los mejores para la que ocupa este trabajo.
Atendiendo a lo anterior se inició el experimento preliminar con los parámetros mencionados para la tarea de reconocimiento de caracteres sobre un subconjunto pequeño de la base de datos A. Este subconjunto se escogió al azar y a partir del cuál también al azar, se formaron los conjuntos de entrenamiento y prueba preliminares los cuales consistieron respectivamente en 32 y 8 imágenes solamente, es decir, que para las ocho clases de la base de datos, sólo cuatro muestras por clase para entrenamiento y una para prueba.
Se realizaron varias pruebas respecto al método de extracción de puntos específicos, dando mejores resultados el método de contornos (Sec. 6.1.1). Este método no sólo dio mejores valores porcentuales de reconocimiento, sino que resulta en tiempos reducidos con respecto al método de umbral de brillo debido a que el número de puntos específicos se reduce considerablemente. Si se tiene en cuenta que los puntos específicos deben permitir la clasificación de la pieza en cuestión, se entiende que con los bordes de la imagen basta para esta tarea. Luego de otras pruebas preliminares se determinó utilizar $T_{max} = Brillo_{max}$, donde $Brillo_{max}$ es el valor máximo de brillo posible correspondiente al color blanco en escala de grises. No se da un valor explícito ya que éste depende del formato y compresión de las imágenes que se utilicen. La variable T_{max} junto con T_{min} tiene el propósito de acotar el intervalo de brillos posible para la definición de umbrales de los puntos de cada una de las propiedades definidas (Sec. 6.1.1). Tanto para la tarea de reconocimiento de piezas como para las características de las bases de datos de imágenes se utilizó todo el intervalo posible de brillos. Así mismo, se encontró que el parámetro B igual a la mitad del brillo máximo es el que mejor resultado proporcionaba. La selección de los parámetros T_{max}, T_{min} y B se explica dadas las características de las imágenes utilizadas que poseen brillos en todo el rango permitido. Así se tienen en la base de datos piezas negras y otras de colores metálicos claros aunado al brillo que algunas poseen. Para todos los experimentos se trabajó con una ventana del extractor de propiedades cuadrada por lo que los parámetros w y h se hicieron iguales. Es por esta razón que sólo se hará referencia al primer parámetro w.
Las primeras pruebas con los parámetros reportados para la tarea de reconocimiento de caracteres arrojó porcentajes de reconocimiento muy pobres de entre 37 % y 50 %. Por lo cuál luego de varias pruebas variando diversos parámetros se alcanzó un mejor resultado con 75 % de reconocimiento. En la Tabla 7.14 se presentan algunas de las pruebas preliminares realizadas para el ajuste de parámetros. El resultado máximo alcanzado en este experimento no se debió a un único conjunto de parámetros, sin embargo se escogió entre estos empates al que representa en términos de eficiencia ventajas sobre los otros (prueba 4), por ejemplo, ante dos pruebas con parámetros idénticos a excepción de uno que dan el mismo resultado, se escogió el parámetro menor pues esto representa en todos los casos menor tiempo de procesamiento del PCNC, lo anterior es especialmente cierto para los parámetros N y más aún en el parámetro S.
El mejor conjunto de parámetros seleccionado se utilizó como base para los experimentos siguientes, del mismo modo que se hizo con el clasificador LIRA. Estos parámetros base son: $B = Brillo_{max}/2$, $T_{max} = Brillo_{max}$, $T_{min} = 0$, $w = h = 10$, $p = 5$, $n = 4$, $S = 1000$, $N = 300000$, $K = 20$, $D_c = 5$, $q = 2$. El PCNC construido con estos parámetros aplicado sobre la base de datos A alcanzó en una prueba un porcentaje de reconocimiento de 92 %.

7.2.2. Unicidad

De la misma forma que con LIRA se realizaron 10 corridas con los parámetros base para cada una de las bases de datos para conocer que tanto varían construcciones distintas del PCNC con idénticos parámetros, entrenadas con el mismo número de ciclos y aplicadas sobre idénticos conjuntos de entrenamiento y prueba. Los resultados de este experimento se muestran en la Tabla 7.15.

[4]En los trabajos preliminares para el PCNC referenciados en el Cap. 6 se han usado caracteres manuscritos aislados.

Tabla 7.14: Experimento preliminar para la sintonización de parámetros del PCNC con la base de datos A con nueve pruebas.

No. de prueba	1	2	3	**4**	5	6	8	9
Tamaño de la ventana ($w = h$)	11	11	10	**10**	10	10	10	10
Puntos positivos y negativos (p/n)	3/6	5/4	5/4	**5/4**	5/4	5/4	4/5	5/4
Número de propiedades definidas (S)	6400	1000	1000	**1000**	1500	1000	1000	1000
Tamaño del vector de codificación (N) (miles)	128	300	300	**300**	300	400	400	400
Parámetro de encoder (K)	16	20	20	**20**	20	20	20	20
Distancia de correlación (D_c)	8	5	5	**5**	5	5	5	7
Parámetro del CDT (q)	5	1	1	**2**	2	2	2	2
Reconocimiento (%)	50	62	62	**75**	50	62	62	62

Tabla 7.15: Resultados del experimento con 10 construcciones para cada uno de las clasificadores PCNC de cada base de datos. Cada una de las 10 corridas se hizo con idénticos parámetros, mismos conjuntos fijos de entrenamiento y prueba, y mismo número de ciclos de entrenamiento. Todos los resultados se dan en unidades porcentuales.

PCNC	1	2	3	4	5	6	7	8	9	10	$\bar{x}$	σ
BD-A:	91	92	91	85	93	92	91	93	92	93	91.3	2.24
BD-B:	93	95	96	94	91	94	95	96	92	95	94.1	1.58
BD-D:	84	81	87	78	81	87	78	86	76	81	81.9	3.75

Tenemos aquí que el PCNC obtuvo mejores resultados que LIRA para las bases de datos A y B, sin embargo perdió ante LIRA en la base de datos D. Sin embargo en este experimento no se está calificando aún la máxima capacidad de reconocimiento, sino la variación entre construcciones con idénticos parámetros y al respecto puede verse que para la base de datos A el PCNC tuvo una menor desviación estándar que LIRA, sin embargo para la base de datos B esta desviación estándar para el PCNC fue mayor que para LIRA y en el caso de la base de datos D, fue de más del doble.

7.2.3. Sintonización de parámetros

Este experimento tiene por objetivo obtener el mejor conjunto de parámetros para el PCNC aplicado sobre las bases de datos utilizadas. Siguiendo el mismo procedimiento que para LIRA se analizarán como repercute las variaciones en cada uno de los parámetros en el resultado de reconocimiento sobre cada una de las bases de datos utilizadas. Los parámetros p y n se tratan juntos ya que tienen naturaleza similar al caso de LIRA.

Para estos experimentos se utilizaron los parámetros base descritos en la Sección 7.2.1. A continuación se presentan los resultados obtenidos.

Al igual que para LIRA, los parámetros de ventana w y h se hicieron iguales y por ello se hace referencia únicamente a w. El experimento con el parámetro w se resumen en la Tabla 7.16. Se tiene que para la base de datos A un valor de $w = 12$ alcanzó el mejor resultado, mientras que para la base de datos B hubo un empate múltiple para w igual a 9, 10, 11 y 13, siendo el resultado para $w = 12$ de sólo un punto porcentual menor. Para la base de datos D el mejor resultado fue para $w = 11$ y $w = 20$. Lo que puede concluirse aquí es que los valores correctos de w deben estar entre 9 y 13, teniendo mejores resultados los valores para $w = 11$ o $w = 12$ según la base de datos que se utilice.

Tabla 7.16: Resultados de pruebas del PCNC con variación del parámetro de dimensión de la ventana PCNC w sobre las bases de datos A, B y D.

Parámetro	w	5	8	9	10	11	12	13	14	15	20
BD-A	%	91	90	93	93	92	**95**	91	91	93	89
BD-B	%	89	90	**94**	**94**	**94**	93	**94**	91	90	89
BD-D	%	83	83	83	82	**86**	80	81	79	83	**86**

Dados los resultados obtenidos en LIRA para p y n y dada la naturaleza similar de estos parámetros en ambos clasificadores, para el PCNC sólo se probaron los parámetros en forma conjunta. Los resultados de este experimento se muestran en la Tabla 7.17. Tenemos que los valores que permitieron una mejor respuesta en el reconocimiento fueron bastante uniformes y se concentraron para las tres base de datos utilizadas es el caso de que $p + n = 7$ y específicamente para las tres base de datos (A, B y D) los mejores valores resultaron para p y n igual a 4 y 3 respectivamente. Además para las bases de datos B y D hubo empates de la respuesta máxima alcanzada, siendo para la base de datos B con los valores $p = 5$ y $n = 2$ y para la base de datos D $p = 2$ y $n = 5$.

Tabla 7.17: Resultados de pruebas del PCNC con variación de los parámetros p y n sobre las bases de datos A, B y D para distintos valores de $p + n$.

Parámetros $p/n,\ p+n=7$	2/5	3/4	4/3	5/2	6/1	-
BD-A (%):	49	76	**95**	93	88	-
BD-B (%):	93	93	**96**	**96**	90	-
BD-D (%):	**88**	86	**88**	87	79	-
Parámetros $p/n,\ p+n=9$	3/6	4/5	5/4	6/3	7/2	-
BD-A (%):	70	81	93	93	90	-
BD-B (%):	87	95	94	93	91	-
BD-D (%):	82	87	82	82	76	-
Parámetros $p/n,\ p+n=11$	4/7	5/6	6/5	7/4	8/3	9/2
BD-A (%):	85	41	79	88	89	89
BD-B (%):	88	89	84	92	91	81
BD-D (%):	55	70	79	75	76	72

El parámetro S es el número de propiedades definidas para el extractor de propiedades del PCNC. S está relacionado en forma directamente proporcional al tiempo de ejecución del PCNC. Así tenemos que para doble valor de S tendremos doble consumo de tiempo de ejecución de un PCNC particular. Esta relación es importante para tomar una decisión adecuada. En la Tabla 7.18 se presentan los resultados obtenidos para el experimento con el parámetro S. Se tiene que para la base de datos A el mejor resultado se alcanzó para S igual a 2500 y 4000, por lo que se toma el primero. Para la base de datos B el mejor resultado fue para los valores de S iguales a 5000 y 6000, sin embargo considerando la contribución de S al tiempo de ejecución del PCNC es importante ubicar también los valores obtenidos para S igual a 1500, 2500 y 10 000 dado que el resultado para éstos sólo es un punto porcentual menor que el reconocimiento máximo. Tomando en cuenta entonces que $S = 5000$ obtuvo un reconocimiento de 96 % y que $S = 1500$ obtuvo un resultado de 95 % se elige entonces este último, pues el costo de un punto porcentual más significa más del triple de tiempo de ejecución del PCNC respectivo. Para la base de datos D un caso similar se tiene con S igual a 2000 y 6000, por lo que se toma el primer valor debido a su costo tres veces menor.

Tabla 7.18: Resultados de pruebas del PCNC con variación del parámetro S sobre las bases de datos A, B y D.

Parámetro	S	400	600	800	1000	1200	1500	2000	2500	3000	4000	5000	6000	10000
BD-A	%	85	83	92	93	91	87	93	**95**	91	**95**	90	93	88
BD-B	%	91	93	91	94	94	**95**	91	95	94	91	**96**	**96**	95
BD-D	%	83	84	80	82	80	83	**86**	82	85	84	81	**87**	84

El parámetro N está ligado a la eficiencia del PCNC de forma similar pero no igual que para LIRA. Mientras que para ambos N es un parámetro de la red neuronal, para el PCNC es además un parámetro del codificador el cuál utiliza este valor para crear tres permutaciones aleatorias. Sin embargo este valor no influye como dimensión del vector de codificación por que se trabaja con vectores comprimidos (Sec. 6.1.2). En la Tabla 7.19 se resumen los resultados obtenidos para las pruebas con el parámetro N. Tenemos dos empates para las pruebas con las base de datos A y B con valores de N de 200 000, 300 000 y 300 000 y 400 000 respectivamente. Para la base de datos D el mejor resultado se tiene para $N = 400000$, lo que indica que en general el mejor desempeño puede encontrarse en valores de N entre 300 000 y 400 000. Sin

embargo, dada la relación proporcional de N con el consumo de recursos del PCNC, debe tomarse siempre el menor valor.

Tabla 7.19: Resultados de pruebas del PCNC con variación del parámetro N sobre las bases de datos A, B y D.

Parámetro	N	100 000	200 000	250 000	300 000	400 000	500 000
BD-A	%	91	**93**	89	**93**	89	91
BD-B	%	90	93	93	**94**	**94**	**94**
BD-D	%	83	77	82	82	**84**	83

Para el experimento con el parámetro K se obtuvieron los resultados dados en la Tabla 7.20. Se tiene un resultado variable para cada base de datos. Para la base de datos A hubo un triple empate con K igual a 15, 20 y 25; para la base de datos B el mejor resultado fue para $K = 5$ y para la base de datos D hubo un empate con K igual a 15 y 30. Se comprueba la teoría para este parámetro, que representa el número de unos de los vectores máscara generados para cada propiedad, en el sentido de que son valores pequeños los que dan mejores resultados. Se tiene entonces que este parámetro K no debe ser menor de 5 o mayor de 30 para alcanzar buenos resultados.

Tabla 7.20: Resultados de pruebas del PCNC con variación del parámetro K sobre las bases de datos A, B y D.

Parámetro	K	5	10	15	20	25	30
BD-A	%	75	92	**93**	**93**	**93**	90
BD-B	%	**95**	94	94	94	93	90
BD-D	%	80	81	**83**	82	82	**83**

Para el parámetro D_c que es la distancia de correlación del codificador del PCNC los resultados obtenidos se muestran en la Tabla 7.21. Se tiene que para la base de datos A hubo un empate con D_c igual a 2 y 5, mientras que para la base de datos B el mejor resultado fue con $D_c = 6$ y para la base de datos D existió también un empate para D_c igual a 4 y 8. Estos resultados nos dicen, para la tarea particular que se trata, que la distancia de correlación debe tener un valor pequeño menor que 10 pero mayor que uno.

Tabla 7.21: Resultados de pruebas del PCNC con variación del parámetro D_c sobre las bases de datos A, B y D.

Parámetro	D_c	1	2	4	5	6	8	10	15	20	25
BD-A	%	70	**93**	91	**93**	92	90	90	88	87	74
BD-B	%	84	91	93	94	**95**	92	89	91	82	82
BD-D	%	77	72	**85**	82	83	**85**	78	82	72	76

El parámetro q es un factor del CDT que combina las distintas propiedades encontradas para una imagen determinada (Sec. 6.1.2.5). El resultado de las pruebas con este parámetro se muestran en la Tabla 7.22. El resultado de variar este parámetro sobre el resultado del PCNC es bastante irregular por lo que sólo puede entenderse tomando en cuenta las posibles variaciones por el concepto de unicidad (Sec. 7.2.2). Así se tiene que para la base de datos A el mejor resultado se tuvo para $q = 5$, para la base de datos B para $q = 10$ mientras que para la base de datos D el mejor resultado se obtuvo al no aplicar el CDT, esto es $q = 0$.
Dados los anteriores resultados tenemos que el mejor conjunto de parámetros para la base de datos A se dio para los parámetros base pero con $q = 5$ (Tabla 7.22). Considérese también el conjunto de parámetros construido tomando los mejores parámetros que aisladamente dieron mejores resultados en los experimentos considerando además su eficiencia. Así, tomando estos dos conjuntos de parámetros junto con los parámetros base se realizó el siguiente experimento con cinco creaciones distintas para cada conjunto de parámetros, para de esta forma obtener resultados estadísticamente confiables. En la Tabla 7.23 se muestran estos resultados donde se ve que los mejores parámetros obtenidos aisladamente también constituyeron el mejor conjunto de parámetros. Adicionalmente es de notar que este conjunto de parámetros logró la menor desviación estándar en sus respuestas.

Tabla 7.22: Resultados de pruebas del PCNC con variación del parámetro q sobre las bases de datos A, B y D.

Parámetro	q	0	1	2	3	4	5	6	7	8	9	10	11	12	13
BD-A	%	90	92	93	93	92	**96**	93	94	87	88	86	91	93	93
BD-B	%	94	93	94	90	94	94	92	92	89	90	**97**	95	93	93
BD-D	%	**85**	80	82	83	80	84	80	82	78	80	80	79	83	82

Tabla 7.23: Resumen de resultados de pruebas para obtener el mejor conjunto de parámetros PCNC para base de datos A. $\bar{x}$ es el promedio y σ es la desviación estándar.

Parámetros:	Parámetros								Construcciones/(%)					$\bar{x}$	σ
	w	p	n	S	N	K	D_c	q	1	2	3	4	5	(%)	(%)
Base	10	5	4	1000	300000	20	5	2	93	85	93	93	92	91.2	3.12
Mejor en pruebas	10	5	4	1000	300000	20	5	5	95	90	86	90	88	89.8	2.99
Mejores aislados	12	4	3	2500	200000	20	5	5	93	93	94	88	94	92.5	2.24

El mismo procedimiento descrito se utilizó para obtener los mejores parámetros para el clasificador PCNC aplicado a las bases de datos B y D obteniendo un porcentaje de reconocimiento de 97 % para la base de datos B con los parámetros $w = 10$, $p = 4$, $n = 3$, $S = 1500$, $N = 300000$, $K = 20$, $D_c = 6$ y $q = 10$. Para la base de datos D el mejor resultado obtenido fue de 91 % de reconocimiento con los parámetros $w = 11$, $p = 4$, $n = 3$, $S = 2000$, $N = 400000$, $K = 15$, $D_c = 4$ y $q = 0$. Este último resultado con $q = 0$ implica que este PCNC no utilizó el CDT para lograr el mejor desempeño.

7.2.4. Distorsiones

Con el objetivo de mejorar la capacidad de reconocimiento del clasificador PCNC se realizó un par de pruebas empleando distorsiones de las imágenes originales del conjunto de entrenamiento para ampliarlo. La prueba se hizo de la misma forma que para el clasificador LIRA (Sec. 7.1.4). En la Tabla 7.24 se resumen estas pruebas.

Tabla 7.24: Experimento usando distorsiones para aumentar el conjunto de entrenamiento del PCNC aplicado sobre la base de datos A.

	Número de prueba	
	1	2
Número de distorsiones	6	12
distorsiones horizontales	+1, -1	+2, +1, -1, -2
distorsiones verticales	+1, -1	+2, +1, -1, -2
distorsiones angulares	+1°, -1°	+2°, +1°, -1°, -2°
Ciclos de entrenamiento	30	30
Porcentaje de reconocimiento (%)	93	91

Los resultados obtenidos con las distorsiones son similares a los obtenidos para el clasificador LIRA, sin embargo considerando el mejor porcentaje de reconocimiento del PCNC las distorsiones no contribuyeron a mejorar el comportamiento del PCNC y si se considera además el tiempo adicional requerido para crear las distorsiones y para el entrenamiento de éstas la conclusión es que su aplicación no es conveniente para la tarea que nos ocupa. La explicación a esto se da por el hecho que los clasificadores son entrenados para el reconocimiento de imágenes normalizadas y el agregar distorsiones sólo distrae la memoria del clasificador hacia casos que se apartan de tal normalización. A medida que los clasificadores reconozcan una pieza ligeramente desviada de su estado normalizado (Sec. 4.5.1) con respuesta similar a su estado normalizado entonces no podrá aplicarse tal clasificador para la correcta identificación del ángulo de orientación o posición cartesiana de tal imagen como se verá más adelante en la Sec. 7.4.

7.2.5. Ciclos de entrenamiento

Se realizó la misma prueba que para LIRA con los ciclos de entrenamiento. Se crearon dos PCNCs con idénticos parámetros para cada una de las base de datos utilizadas. En este experimento se tuvo una respuesta más rápida para alcanzar el reconocimiento máximo respecto a LIRA. Por esta razón las pruebas se hicieron con pasos de 5 en lugar de 10 ciclos de entrenamiento. En la Tabla 7.25 se muestran los resultados obtenidos. Para la base de datos A el reconocimiento máximo se alcanzó desde los 15 ciclos de entrenamiento, sin embargo se dio un resultado notable a solo 10 ciclos para la segunda prueba, pues se alcanzó un resultado de 93 % pero luego con ciclos adicionales cayó esta respuesta y no volvió a repetirse. Para la base de datos B la respuesta fue uniforme y se tuvo a los 15 ciclos para la prueba 3 y a los 25 ciclos para la prueba 4. Para la base de datos D sucedió algo similar pero con 25 y 30 ciclos para la primera y segunda de sus pruebas (pruebas 5 y 6 respectivamente). Se deduce que con 30 ciclos de entrenamiento se obtienen resultados correctos.

Tabla 7.25: Experimento con diversos ciclos de entrenamiento sobre seis clasificadores PCNC. Para cada base de datos los parámetros son los mismos. Las pruebas se realizaron en las bases de datos A, B y D.

	No. de ciclos de entrenamiento					
Número de prueba:	5	10	15	20	25	30
BD-A 1:	66	88	**94**	94	94	94
BD-A 2:	76	93	**90**	90	90	90
BD-B 3:	69	90	**96**	96	96	96
BD-B 4:	67	89	94	**97**	97	97
BD-D 5:	39	59	69	81	**88**	88
BD-D 6:	51	61	64	79	84	**86**

7.2.6. Aleatoriedad de los conjuntos para entrenamiento y prueba

Una prueba muy importante es la de poder entrenar y probar el clasificador sobre conjuntos de entrenamiento y prueba variables. Por ello se realizó un experimento con estos conjuntos tomados aleatoriamente de toda la base de datos en la misma proporción que se hizo originalmente, esto es, 50 % para las base de datos A y B y 72 % para la base de datos D para los conjuntos respectivos de entrenamiento y el resto para los conjuntos de prueba respectivamente. Para cada base de datos distinta se utilizó un PCNC distinto, pero para cada una de las diez pruebas de cada base de datos se utilizó el mismo PCNC. Los resultados obtenidos mostrados en la Tabla 7.26 dan cuenta de la estabilidad del PCNC en cuanto a una misma estructura particular. Los porcentajes para cada base de datos variaron poco, lo que se ve en la desviación estándar obtenida para cada caso.

Tabla 7.26: Resultados de pruebas con el PCNC con conjuntos de entrenamiento y prueba seleccionados aleatoriamente para las bases de datos A, B y D. Para cada base de datos se utilizó un mismo clasificador.

Prueba No.:	1	2	3	4	5	6	7	8	9	10	$\bar{x}$	σ
BD-A:	94	92	93	93	90	90	91	93	91	94	92.1	1.45
BD-B:	96	96	97	94	95	96	95	97	95	95	95.6	0.92
BD-D:	90	90	89	91	95	88	87	92	89	90	90.1	2.12

7.2.7. Mejores clasificadores PCNC

En esta sección se describen a detalle los resultados obtenidos con los mejores PCNC obtenidos para las bases de datos utilizadas. Primero que nada se muestra en la Tabla 7.27 un resumen de los parámetros utilizados para estos clasificadores así como la respuesta obtenida por ellos a detalle. Para todos ellos se utilizaron 30 ciclos de entrenamiento. En la Tabla se muestra además de los parámetros empleados en cada

base de datos y el resultado porcentual exacto obtenido en el reconocimiento, un desglose del número de errores en cada caso por cada una de las clases utilizadas así como la contribución porcentual de cada clase en el error total de reconocimiento. Consúltense las Tablas 4.1 y 4.2 para interpretar las clases para cada base de datos. Tómese en cuenta que los percentiles totales no suman 100 % debido a la aproximación a dos dígitos decimales.

Tabla 7.27: Resumen de resultados de los mejores clasificadores PCNC para las bases de datos utilizadas. T, ciclos de entrenamiento. R, porcentaje de reconocimiento.

Base de datos:	Parámetros								T	R	Errores por clase/(%)							
	w	p	n	S	N	K	D_c	q			1	2	3	4	5	6	7	8
A	10	5	4	1000	300000	20	5	5	30	96.87	4/2.50	0	0	0	0	0	1/0.63	0
B	10	4	3	1500	300000	20	6	10	30	97.80	1/0.37	1/0.37	0	3/1.10	0	1/0.37	0	-
C	11	4	3	2000	400000	15	4	0	30	91.43	6/5.71	0	1/0.95	1/0.95	1/0.95	0	0	-

7.2.7.1. Base de datos A

El PCNC logró obtener un porcentaje de error para la base de datos A mayor a 96 %. Solo 5 imágenes de un total de 160 no fueron reconocidas correctamente. Estas imágenes se repartieron en sólo dos clases (base de tubito y tornillo cabeza cónica) mientras que para las otras seis clases el reconocimiento fue del 100 % (cono, eje de rotor, no pieza central, terminal de cable, tornillo allen y tornillo cabeza redonda). En la Fig. 7.7 se presentan dos ejemplos de cada clase de entre las imágenes reconocidas y en la Fig. 7.8 se presentan todas las imágenes que no se reconocieron correctamente.

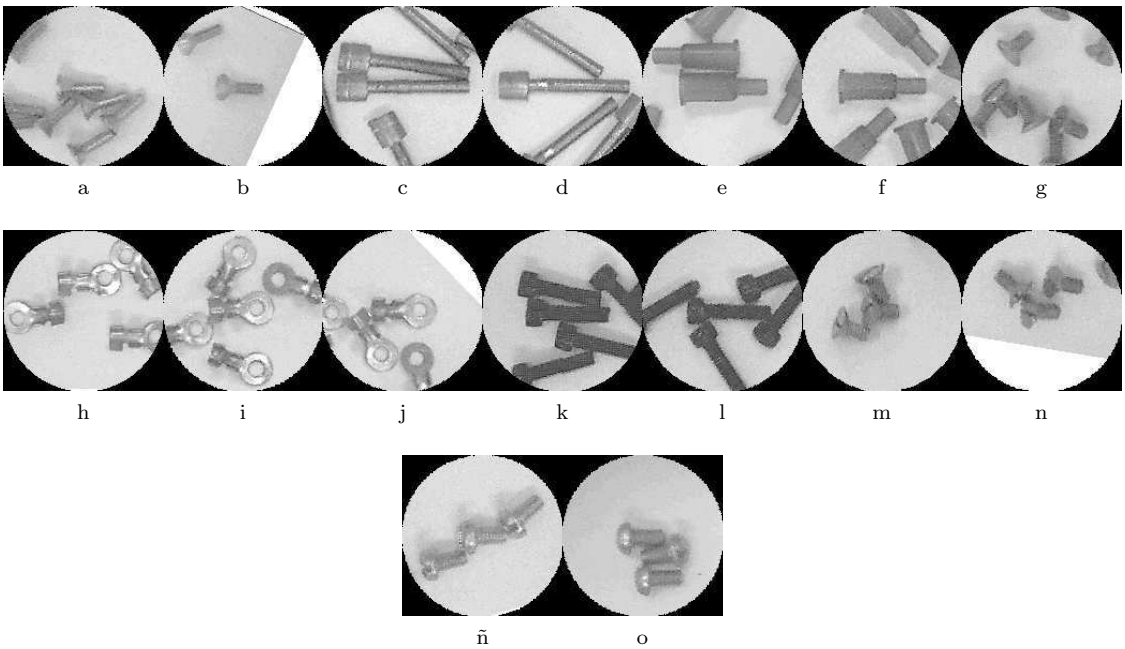
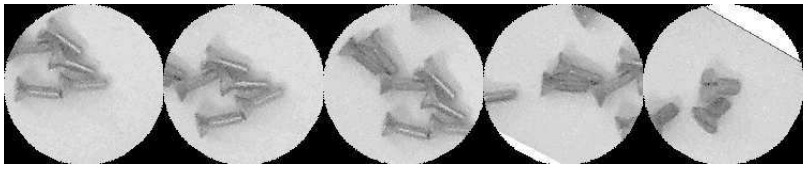


Figura 7.7: Ejemplos de imágenes correctamente reconocidas con el mejor PCNC para la base de datos A. Se muestran dos ejemplos por cada clase.

7.2.7.2. Base de datos B

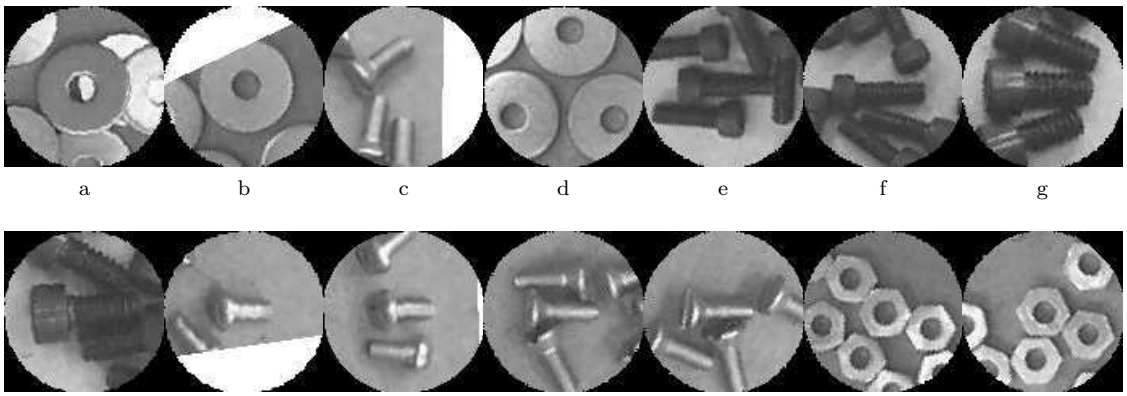
En la base de datos B existió el mejor resultado obtenido en el presente trabajo. Este resultado fue de 97.80 % de reconocimiento correcto. Dos ejemplos de cada una de las clases tomadas del grupo de 267 imágenes bien reconocidas se muestran en la Fig. 7.9. Las imágenes mal reconocidas se distribuyeron en cuatro de las siete clases de la base de datos B de la siguiente forma: un tornillo plano, tres tornillos



a b c d e

Figura 7.8: Las cinco imágenes mal reconocidas con el mejor clasificador PCNC para la base de datos A.

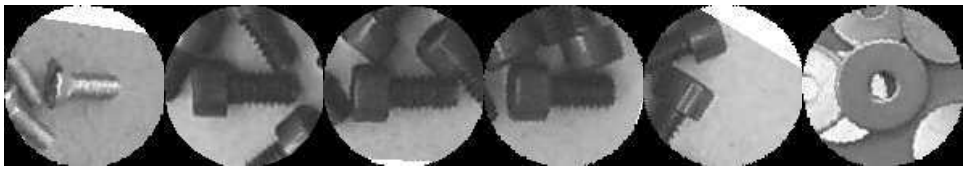
allen grandes, una no pieza central y una arandela. Las clases tornillo allen chico, tornillo gota y tuerca se reconocieron sin errores. En la Fig. 7.10 se muestran las seis imágenes mal reconocidas.



a b c d e f g

h i j k l m n

Figura 7.9: Ejemplos de imágenes correctamente reconocidas con el mejor PCNC para la base de datos B. Se muestran dos ejemplos por cada clase.



a b c d e f

Figura 7.10: Las seis imágenes mal reconocidas con el mejor clasificador PCNC para la base de datos B.

7.2.7.3. Base de datos D

Para la base de datos D con imágenes revueltas y amontonadas el resultado del PCNC ha sido satisfactorio con 96 aciertos y nueve errores. Dos ejemplos de cada clase del total de imágenes correctamente reconocidas se muestran en la Fig. 7.11. Los nueve errores se repartieron entre cuatro clases, uno para cada una de las siguientes clases: tornillo plano, tornillo gota y tornillo allen grande y seis errores para la clase no pieza central. Todas las imágenes mal reconocidas se muestran en la Fig. 7.12.

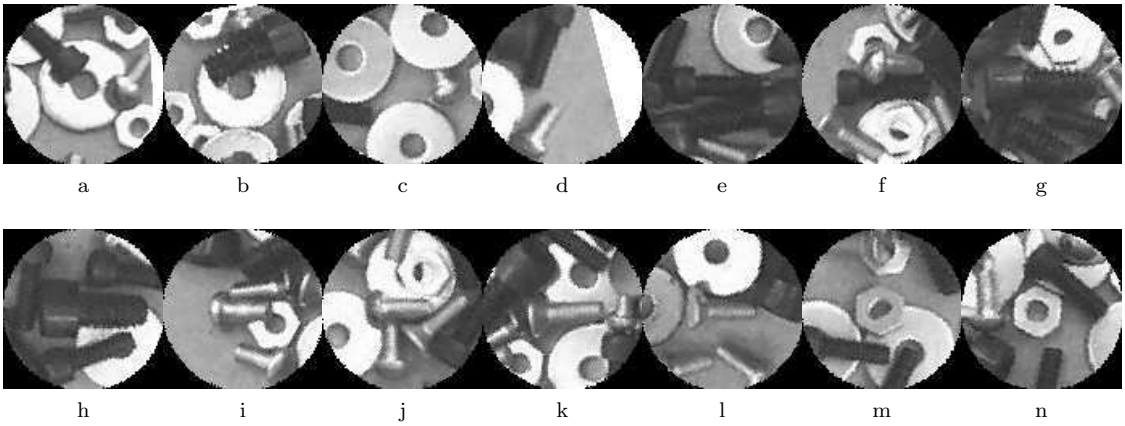


Figura 7.11: Ejemplos de imágenes correctamente reconocidas con el mejor PCNC para la base de datos D. Se muestran dos ejemplos por cada clase.

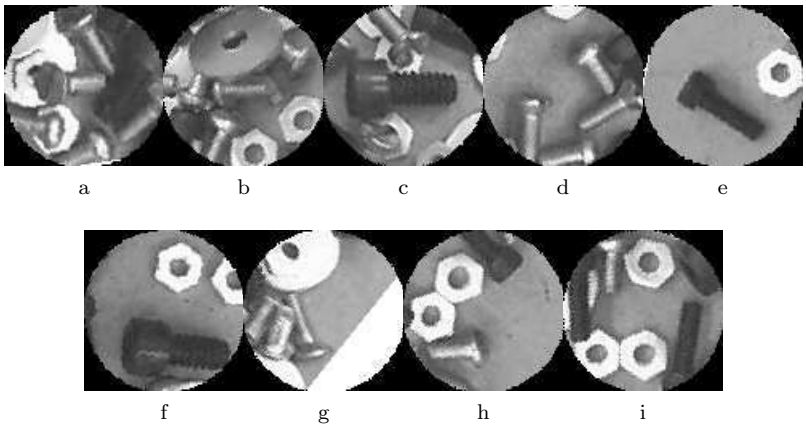


Figura 7.12: Las nueve imágenes mal reconocidas con el mejor clasificador PCNC para la base de datos D.

7.2.8. Prueba con conjunto de entrenamiento ampliado

La misma prueba que se aplicó para LIRA con un conjunto de entrenamiento ampliado de la base de datos D, se aplicó para el PCNC. Esta ampliación consiste en añadir las imágenes extras descritas en la Sección 7.1.8. El resultado de entrenar al PCNC con 537 imágenes agregadas al conjunto de entrenamiento resultó en un porcentaje de reconocimiento sobre el conjunto de prueba de 93 %. Los errores fueron cinco, distribuidos en una no pieza central, dos tornillos allen grandes, un tornillo gota y un tornillo plano.

7.3. Comparación de clasificadores

El propósito de esta sección es comparar los resultados obtenidos de ambos clasificadores en las distintas pruebas realizadas con el objetivo de facilitar la elección de uno u otro atendiendo a la aplicación que se requiera.

7.3.1. Tiempos

El porcentaje de reconocimiento máximo que sobre una base de datos de imágenes logre un clasificador es el parámetro principal para medir su eficiencia. Sin embargo el consumo de recursos que se tenga es un factor muy importante también y en especial el tiempo. Es por ello que se muestran y comparan los tiempos

empleados por ambos clasificadores utilizados para una misma base de datos y en un mismo equipo de cómputo. Con esto se tiene una clara y objetiva comparación entre los clasificadores, considerando además el resultado obtenido en cuanto al porcentaje de reconocimiento. Estos datos son presentados en la Tabla 7.28.

Tabla 7.28: Tiempos empleados por los clasificadores LIRA y PCNC sobre la base de datos A.

		Tiempos por tarea					
		Creación	Entrenamiento (160 imágenes)		Prueba	Guardar	Reconocer
Clasificador	Reconocimiento (%)	o carga	codificación	entrenamiento	(160 imágenes)	en disco	una imagen
LIRA	93 %	3.08s	70s	154s	110s	0.65s	0.687s
PCNC	97 %	1.25s	198s	148s	222s	0.67s	1.387s

Todos estos tiempos dependen de los parámetros de los clasificadores, como se explicó el parámetro N para el clasificador LIRA y los parámetros S y N para el PCNC son los que más impactan en los tiempos de ejecución de las diversas tareas de estos clasificadores. Los resultados presentados en la tabla son para el mejor clasificador para cada uno de ellos para la base de datos A. Tenemos que los tiempos de creación, carga y salvado de estos clasificadores es pequeño y considerando que en un trabajo continuo de reconocimiento sólo serán cargados una vez entonces este tiempo es mínimo y no impacta el desempeño de ninguno de los clasificadores. Lo mismo aplica para el tiempo de guardado. Para el caso del entrenamiento dividido en el tiempo de codificación y de entrenamiento de los códigos, se tiene que para el PCNC el tiempo de codificación es casi de tres veces el respectivo para LIRA, mientras que los tiempos de entrenamiento de códigos es muy similar. Si se considera que el entrenamiento de una determinada base de datos a utilizar se hace una sola vez, entonces, al menos para este caso, la ventaja de cuatro puntos porcentuales más de reconocimiento que tiene el PCNC pagará el costo en el tiempo del entrenamiento. Respecto al tiempo de prueba, el PCNC requiere un tiempo doble que LIRA. El mismo argumento que para el tiempo de entrenamiento puede esgrimirse para los tiempos de prueba. Sin embargo, los tiempos de reconocimiento de una imagen particular tienen la misma proporción entre los clasificadores. Es este tiempo el factor clave que dependiendo de la aplicación y uso particular que se requiera debe ser tomado en cuenta junto con el porcentaje de reconocimiento de cada clasificador para escoger uno de ellos. En general si la tarea no es crítica en tiempo debiera emplearse el PCNC debido a su poder superior de reconocimiento, mientras que si la tarea particular exige tiempo críticos y a su vez puede tolerar más errores entonces el clasificador LIRA debe ser utilizado.

7.3.2. Estabilidad en la creación de estructuras

En los experimentos de unicidad de LIRA y del PCNC se crearon y probaron diez clasificadores para cada uno con idénticos parámetros, ciclos de entrenamiento y conjuntos de entrenamiento y prueba. De estos experimentos la variable resultante de importancia es la desviación estándar. Téngase en cuenta que la desviación estándar es una medida de la variación de los resultados respecto a su promedio, por lo que a menor valor mejor confiabilidad se tiene en los resultados para un conjunto de parámetros dada a la hora de construir un clasificador con ellos. Los resultados presentados en las Tablas 7.2 y 7.15 se resumen y comparan en la Tabla 7.29. Se tiene que LIRA fue superior en estabilidad para las bases de datos B y D, mientras que para la base de datos A tuvo una estabilidad menor que el PCNC. En base a estos resultados no puede declararse ningún ganador entre los dos tipos de clasificadores probados en cuanto a esta característica.

Tabla 7.29: Comparación en la desviación estándar obtenida entre los clasificadores LIRA y PCNC sobre las tres bases de datos utilizadas.

	Clasificador	
Base de datos:	LIRA	PCNC
A	3.14 %	2.24 %
B	1.02 %	1.58 %
D	1.48 %	3.75 %

7.3.3. Parámetros

Para ambos clasificadores se hicieron una serie de experimentos variando un sólo parámetro a la vez para estudiar la forma en que varían los resultados obtenidos y para encontrar el mejor clasificador para cada base de datos. Sin embargo para cada una de estas pruebas es necesario construir un nuevo clasificador por lo que siempre se tiene añadido al resultado obtenido la incertidumbre dada por la unicidad. Dado el argumento anterior y el estudio de las estructuras LIRA y PCNC se concluye que la búsqueda de parámetros para ambos clasificadores utilizados requiere de múltiples pruebas con cada base de datos a utilizar.

7.3.4. Ciclos de entrenamiento

De los experimentos hechos con los clasificadores para encontrar el mejor número de ciclos de entrenamiento se tiene que el clasificador LIRA requirió entre 30 y 70 ciclos de entrenamiento para alcanzar un valor estable mientras que el PCNC sólo requirió entre 15 y 30 ciclos. Esta diferencia no influye significativamente en el rendimiento total de los clasificadores pues ambos utilizan codificación del conjunto de entrenamiento para evitar codificar cada nuevo ciclo de entrenamiento. Siendo el tiempo de entrenar cada ciclo adicional para ambos clasificadores muy similar como puede verse en la tabla 7.28

7.3.5. Confiabilidad

Una vez construido un clasificador debe conocerse que tan confiable y estable es éste para reconocer objetos. Para analizar esta propiedad sobre las bases de datos y conocer en qué medida varia una construcción única de un determinado clasificador se realizaron los experimentos con conjuntos aleatorios de entrenamiento y prueba. El resumen y la comparación de estos experimentos se tiene en la Tabla 7.30
Se tiene que el PCNC fue en todos los casos más confiable que el clasificador LIRA. Para las base de datos A y B superó por más del doble a LIRA. Este resultado es de lo más importante por que da cuenta de la capacidad de los clasificadores de ser entrenados y probados con imágenes aleatoriamente seleccionadas.

Tabla 7.30: Resumen de los experimentos de confiabilidad para los clasificadores LIRA y PCNC sobre las tres bases de datos utilizadas. Los resultados se expresan como la desviación estándar obtenida sobre las diez pruebas realizadas para cada base de datos y cada clasificador.

	Clasificador	
Base de datos:	LIRA	PCNC
A	3.40 %	1.45 %
B	2.15 %	0.92 %
D	3.37 %	2.12 %

7.3.6. Resultados

Luego de diversos experimentos se encontraron los mejores parámetros para cada clasificador y para cada base de datos. El desempeño del PCNC fue superior al clasificador LIRA para todas las base de datos utilizadas. Sólo en la base de datos D este resultado fue casi igual. Un resumen de estos resultados se presenta en la Tabla 7.31.

Tabla 7.31: Resultados obtenidos por los clasificadores LIRA y PCNC sobre las bases de datos A, B y D. Los resultados expresan el porcentaje de reconocimiento sobre la base de datos respectiva.

	Clasificador	
Base de datos:	LIRA	PCNC
A	93.75 %	96.87 %
B	94.14 %	97.80 %
D	90.47 %	91.43 %

Del análisis y comparación de los resultados obtenidos por los mejores clasificadores para las bases de datos tenemos que para la base de datos A ambos clasificadores reconocieron el 100 % de las clases 2 a 6 (Tabla

7.13), es decir, las clases cono, eje de rotor, no pieza central, terminal de cable y tornillo allen. Para la base de datos D las clases 3 y 7 (tornillo allen chico y tuerca) fueron también reconocidas sin errores por ambos clasificadores. En cuanto a las base de datos B y D que están construidas con el mismo tipo de piezas y el mismo número de clases pero de complejidad distinta, el clasificador LIRA no tuvo errores para la clase 4 (tornillo allen grande) mientras que el PCNC no tuvo errores para ninguna de estas dos bases de datos para la clase 3 (tornillo allen chico).
Comparando los errores obtenidos por cada clasificador para cada una de las base de datos utilizadas tenemos que las cinco imágenes mal reconocidas con el PCNC para la base de datos A se encuentran dentro de las diez mal reconocidas por LIRA para esta misma base de datos. Es decir, estas cinco imágenes no pudieron ser reconocidas por ninguno de los clasificadores (Fig. 7.8).
Para la base de datos B la comparación da resultados distintos ya que sólo una imagen no pudo ser reconocida por ambos clasificadores Fig. 7.4 b.
Por último, para la base de datos D cuatro imágenes no pudieron ser reconocidas por ninguno de las clasificadores. Estas imágenes son las mostradas en las Figs.:7.6 c, h, i, j que corresponden a las Figs: 7.12 b, a, f y e respectivamente.
Por otra parte ambos clasificadores no tuvieron problema en reconocer las piezas que ubicándose cerca del extremo de la escena[5] poseen una parte blanca que rellena la falta de imagen original. Ejemplos de estas imágenes bien reconocidas se tienen en la Fig. 7.1 b, Fig. 7.3 n, Fig. 7.7 n y Fig. 7.9 i.
Otra cualidad muy importante de ambos clasificadores ha sido poder reconocer piezas parcialmente obstruidas como puede verse en las Figs. 7.5 a, b y Fig: 7.11 b.

7.4. Búsqueda y reconocimiento de posición

Uno de los objetivos del presente trabajo es la creación de un sistema automático de localización de piezas o SVT (Sec. 3.3). En este punto del trabajo ya se han expuesto dos métodos de identificación de piezas constituidos por los clasificadores LIRA y PCNC. Cualquiera de estos clasificadores ha demostrado, en los experimentos expuestos en las secciones anteriores, resultados suficientemente buenos por lo que cualquiera de ellos puede ser usado como base en el método de localización de piezas. Una vez que el clasificador seleccionado es entrenado con una base de datos particular que contiene imágenes de las piezas con la que se desea trabajar, puede aplicarse el método de localización de piezas sobre una imagen o escena particular. Dicho método se ha expuesto en la Sección 4.6.
Como ejemplo se tomó el mejor clasificador LIRA para la base de datos A entrenado con las siete clases de esta base de datos y se realizaron distintas búsquedas para localizar una determinada pieza en alguna imagen dada.
En la Fig. 7.13 mostramos dos ejemplos de imágenes reconocidas. Una de las imágenes contiene dos conos reconocidos (conos 1, 2 en la Fig. 7.13(a). En la Fig. 7.13 (b) la imagen contiene tres terminales de cable reconocidas (1, 2, 3).
El pequeño cuadro blanco en la figura representa el lugar donde la pieza solicitada ha sido reconocida y el cuadro grande representa la ventana correspondiente de búsqueda asociada a esta posición.
Al pie de las imágenes puede leerse las coordenadas de la posición y la orientación encontradas por el sistema para cada una de las piezas reconocidas.
El clasificador neuronal es un método flexible, esto significa que el sistema reconoce una pieza fuera de su centro, pero suficientemente cerca del cuerpo de la pieza.
Algunas veces el sistema puede reconocer una misma pieza varias veces. En la Fig. 7.14(a) se muestra un reconocimiento múltiple de una pieza (casos 1 y 3).
Esto sucede por que el sistema puede encontrar una pieza lejos de su centro, por ejemplo las marcas 2 y 3 en la Fig 7.14(a) y 1, 7 en la Figura 7.14(b).
En otros casos, esto pasa porque los parámetros de búsqueda (Δx, Δy, $\Delta\theta$) tienen valores muy grandes, por ejemplo la marca 1 en la Fig 7.14(a) y las marcas 2, 4, 8 y 9 en la Figura 7.14(b).
La marca 4 de la Fig. 7.14(a) y la marca 6 de la Fig. 7.14(b) tienen suficiente precisión para el trabajo de manipulación.
Dos ejemplos de imágenes con piezas que se tocan una a otra se presentan en la Fig. 7.15. En la Fig. 7.15(a) el sistema encontró tres piezas (obsérvese que el punto de localización está lejos del centro de estas piezas). En la Fig. 7.15(b) el sistema presenta redundancia. En este caso el sistema puede filtrar los resultados considerando la mejor respuesta obtenida en la salida del clasificador LIRA, lo cual significa que la pieza

[5]Se entiende por escena la imagen original de la cuál se recortaron las imágenes para formar las bases de datos o sobre la cuál se pretende localizar una pieza determinada.

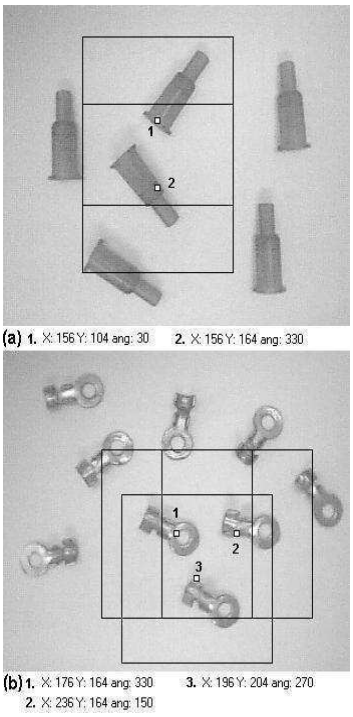


Figura 7.13: Localización y reconocimiento de piezas.

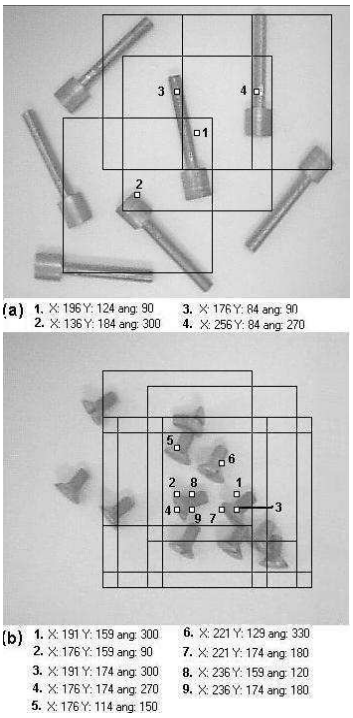


Figura 7.14: Localización y reconocimiento de piezas con redundancia.

con las marcas 2 y 3 se considerará reconocida en la marca número 2 que está mucho mejor posicionada con respecto al centro de la pieza.

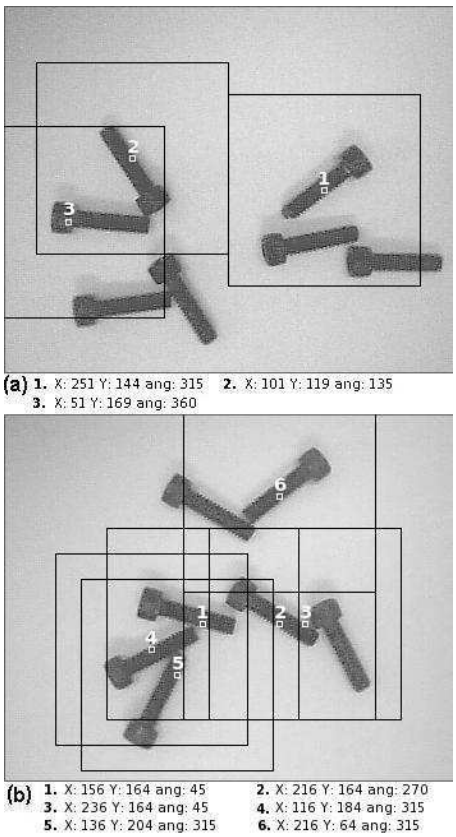


Figura 7.15: Localización y reconocimiento de piezas que se tocan.

Los resultados obtenidos son buenos para permitir la manipulación, sin embargo el método de búsqueda debe ser mejorado para incrementar la precisión de la localización y del ángulo de la pieza a ser reconocida. De los resultados presentados se tiene que en el sistema desarrollado existen dos problemas específicos de reconocimiento que son:

1. La redundancia en el reconocimiento y
2. La precisión del localizador de piezas.

Respecto a la redundancia en el reconocimiento se tiene que esto sucede debido a que los clasificadores utilizados tienen suficiente flexibilidad como para reconocer una pieza aún lejos de su centro. Esto en primera instancia al momento de la localización representa una ventaja, sin embargo cuando se quiere obtener la localización exacta de la pieza entonces la misma característica representa un problema. Una forma de resolver este problema es comparando las respuestas neuronales de los clasificadores de la salida ganadora y seleccionando la que mayor valor tenga, la cuál será a su vez el punto más cercano al centro de la pieza. Sin embargo esta situación resta eficiencia al método de localización por lo que más estudios deben hacerse al respecto para reducir este problema.
Con respecto a la precisión del localizador, ésta depende tanto de la estructura particular del clasificador que se utilice, como de su entrenamiento así como de los parámetros propios del localizador. Si los parámetros que definen el avance de la ventana de búsqueda se reducen la precisión de búsqueda aumentará sin embargo mayor tiempo de cómputo será necesario; por el contrario, si estos parámetros se aumentan la precisión baja y el tiempo aumenta. Dado lo anterior es importante encontrar los valores óptimos para los parámetros del localizador. Además de lo anterior pueden introducirse mejoras al método de localización como la realización de una búsqueda fina una vez que se ha localizada una región donde exista una pieza de interés.

Considerando los resultados de los experimentos presentados con el localizador así como los dos problemas que en ocasiones se presentan tenemos que el sistema es un excelente paso inicial para la localización e identificación de piezas; sin embargo más investigación y desarrollo debe de hacerse en esta área particular para mejorar las prestaciones del sistema de localización.

Conclusiones

En el Laboratorio de Micromecánica y Mecatrónica se desarrolla microequipo de bajo costo para producir dispositivos micromecánicas. Una característica que permite el bajo costo es la utilización de algoritmos adaptivos para compensar la mayor precisión respecto a los equipos de alto costo. Entre los algoritmos adaptivos destacan los basados en visión por computadora.

Este trabajo marca una línea de investigación nueva en visión por computadora aplicada a tareas de automatización de ensamble y micromaquinado. Diversos sistemas particulares existen para reconocer objetos, sin embargo este trabajo desarrolla una investigación aplicada a una tarea práctica y específica. El sistema se ha probado con piezas reales con las que podría trabajar y en las condiciones prácticas de utilización. El sistema desarrollado ha logrado una eficiencia de más del 97 % en identificación y una eficiencia aceptable en localización de piezas.

Específicamente, se desarrolló un sistema de visión por computadora para el reconocimiento de piezas dirigido a una microfábrica. Se adaptaron e implementaron dos clasificadores neuronales, el clasificador neuronal de área de recepción limitada (LIRA) y el Clasificador Neuronal de Permutación de Códigos (PCNC) para el reconocimiento de piezas y de sus posiciones. Ambos clasificadores fueron implementados a través de programas de cómputo en C++ estándar. Los sistemas se calibraron y probaron con tres bases de datos de imágenes formadas con piezas de trabajo reales.

Este sistema de visión por computadora es flexible, genera un incremento en el nivel de automatización de los procesos de manipulación de piezas utilizado en los procesos de micromanufactura y de microensamble. Este sistema de identificación y ubicación de piezas proporciona una etapa fundamental para lograr la automatización completa de una microfábrica, permitiendo así avanzar en el objetivos general de producir micro piezas con buenas características de precisión a bajo costo.

Se realizó un análisis de la literatura relacionada al tema. Como base teórica para este objetivo se investigaron dos algoritmos adaptivos basados en redes neuronales y en permutación de códigos. Se decidió el uso de los clasificadores neuronales LIRA y PCNC por los buenos resultados obtenidos en experimentos previos relacionados. Para la creación de la base de datos para los experimentos se han utilizado piezas de trabajo reales bajo condiciones reales esperadas en una microfábrica. Se realizaron múltiples experimentos de validación y prueba de los algoritmos propuestos así como experimentos concretos con tareas prácticas. Adicionalmente se ha desarrollado un algoritmo de localización de piezas que se ha construido en base a los buenos resultados obtenidos con los clasificadores utilizados.

El sistema de visión por computadora desarrollado tuvo como tarea primordial encontrar las coordenadas de una pieza determinada ubicado en el campo visual de una cámara acoplada al sistema. El objetivo fue encontrar estas coordenadas con una precisión aceptable para su posterior manipulación. El sistema se probó con diversos conjuntos de piezas reales sin ningún tipo de tratamiento especial. Las piezas escogidas son predominantemente mecánicas las cuales se encuentran comúnmente en una línea de producción. Se utilizaron piezas tales como tornillos de diversas clases, tuercas, arandelas y otras piezas maquinadas. En congruencia con el ambiente de ensamble y producción estas piezas en ocasiones presentan colores oscuros, brillo heterogéneo, sombras y superficies sucias. Todas las anteriores características complican la tarea de reconocimiento.

Para resolver el problema de reconocimiento de piezas se seleccionaron dos clasificadores basados en redes neuronales, el clasificador neuronal LIRA y el PCNC. Ambos clasificadores fueron adaptados para la tarea de reconocimiento de piezas.

Para el análisis e implementación del sistema se crearon tres bases de datos distintas con imágenes de las piezas de características descritas. La base de datos A con ocho ciases y 40 imágenes por clase (320 imágenes en total). La base de datos B con siete clases y 78 imágenes por clase (546 imágenes en total) y la base de datos D con 55 imágenes por cada una de siete clases (385 imágenes en total).

Para ambos clasificadores se realizaron múltiples experimentos con las tres bases de datos distintas constituidas. Los experimentos llevados a cabo fueron objetivos, sistemáticos y estadísticamente válidos. Para

ambos clasificadores se aplicaron los mismos experimentos con idéntica metodología a cada una de las tres bases de datos utilizadas. El primer experimento tuvo por objetivo medir la unicidad de la estructura ante construcciones distintas que usan los mismos parámetros. El segundo se encaminó a obtener el conjunto de parámetros que permitieran obtener el mejor porcentaje de reconocimiento. El tercer experimento utilizó aumento del conjunto de entrenamiento con distorsiones. El cuarto experimento tuvo como fin encontrar el mejor número de ciclos de entrenamiento a utilizar. El quinto experimento tuvo por objetivo medir la confiabilidad del reconocimiento, para esto se realizando diez corridas tomando conjuntos de entrenamiento y prueba aleatorios. Los experimentos descritos se ejecutaron en un computador con procesador *Intel Pentium*© 4 a 2.80 GHz y 512 MB de memoria RAM con sistema operativo GNU/Linux kernel 2.6.17-11-generic.

En las pruebas de unicidad ninguno de los clasificadores fue superior al otro ya que el más estable dependió de la base de datos particular. El clasificador PCNC fue más estable para la base de datos A mientras que para las bases de datos B y D LIRA fue superior.

En las pruebas con distorsiones los resultados fueron similares en ambos clasificadores. El aumento del conjunto de entrenamiento con distorsiones no logró igualar los resultados respectivos obtenidos sin el uso de distorsiones.

Para las pruebas de confiabilidad el PCNC fue superior para todas las bases de datos respecto del clasificador LIRA. Variando este resultado desde 50 % más con la base de datos D hasta más de 100 % con la base de datos A.

El clasificador neuronal LIRA aplicado para la base de datos A tuvo un porcentaje de reconocimiento de 93.75 %, misma que se obtuvo sin utilizar distorsiones. Utilizando distorsiones la tasa de reconocimiento fue de 91 %. Para la base de datos B el mejor reconocimiento obtenido fue de 94.14 % y de 90.47 % para la base de datos D.

El clasificador neuronal PCNC aplicado para la base de datos A tuvo un porcentaje de reconocimiento de 96.87 %, misma que se obtuvo sin utilizar distorsiones. Utilizando distorsiones la tasa de reconocimiento fue de 93 %. Para la base de datos B el mejor reconocimiento obtenido fue de 97.80 % y de 91.43 % para la base de datos D.

Respecto a los tiempos de procesamiento el PCNC requirió el doble de tiempo en reconocimiento de una imagen que el clasificador LIRA. En cuanto al tiempo de codificación utilizado como primera fase en el entrenamiento el PCNC requirió casi el triple del tiempo que LIRA y el doble de tiempo para el proceso de prueba.

El porcentaje de reconocimiento del PCNC para la base de datos A fue superior en 3 % respecto a LIRA, para la base de datos B fue superior en 3.5 % y para la base de datos D superior en casi 1 %.

La superioridad global tomando en cuenta el porcentaje de reconocimiento y el tiempo empleado es sólo discutible para la base de datos D en que LIRA obtuvo respecto al PCNC menos de un punto porcentual de reconocimiento pero empleando la mitad de tiempo que éste.

Ambos clasificadores utilizados demostraron tener buen desempeño en reconocimiento de piezas siendo las imágenes de los mismos de baja calidad y con características que complican el reconocimiento como sombras y brillos. Además el desempeño para la base de datos D, significativamente más compleja por la existencia de piezas amontonadas, revueltas y parcialmente ocultas en algunos casos, también ha sido sobresaliente en ambos clasificadores.

En cuanto a la tarea de búsqueda y localización los resultados obtenidos son aceptables sin embargo para mejorar la precisión es necesario emplear más tiempo de procesamiento lo que resta eficiencia al sistema de visión por computadora propuesto por lo que el método de localización y búsqueda debe ser mejorado.

Para la implementación de los dos clasificadores neuronales, para el localizador y para la creación de las bases de datos de las imágenes se desarrollaron diversos módulos de software. Todo este software constituyó una herramienta indispensable para la investigación y los experimentos realizados. El software se desarrolló mediante un lenguaje de Programación Orientado a Objetos, C++ estándar, desde el sistema operativo GNU/Linux.

Trabajo futuro

Según se expuso en los primeros capítulos de este trabajo, él mismo forma parte de un proyecto de mediano plazo que consiste en un sistema automático de manipulación de piezas. Este trabajo ofrece respuestas a la parte medular del sistema que es el sistema de visión por computadora, sin embargo diversos trabajos adicionales quedan por hacer tanto en mejoras al propio sistema de visión como en completar los sistemas complementarios; estos trabajos van desde la necesidad de más investigación de vanguardia hasta la simple adaptación de componentes. Por lo tanto el trabajo a futuro requiere y ofrece a su vez, formación de recursos humanos desde el servicio social hasta el doctorado.

En cuanto a las mejoras al sistema de visión por computadora propuesto en este trabajo, se tiene, en primer lugar, la mejora del sistema de localización de piezas, este tema requiere de investigación de vanguardia ya que el tema está poco desarrollado en el ámbito mundial. En segundo lugar se requiere mejorar el tiempo empleado para el reconocimiento de piezas por el sistema, esto se consigue principalmente a medida que nuevos sistemas de cómputo de mejor tecnología y capacidades se hacen accesibles; también es posible revisar el software y la implementación de los clasificadores empleados para aumentar su eficiencia así como mejorar la propuesta misma de los clasificadores o estar abiertos a la utilización de otros que lograsen a futuro mayor eficiencia. Un sistema que permita elegir entre varios clasificadores o agregar fácilmente otros nuevos se hace deseable. Otra mejora, a ser introducida en los clasificadores, es la capacidad de reconocimiento con iluminación variable así como a diferentes escalas e incluso con variaciones de posición no sólo en el plano sino en el espacio.

En cuanto a los demás subsistemas del sistema automático de manejo de piezas, del que el (sub)sistema de visión por computadora es cerebro y parte sustancial, tenemos el requerimiento de desarrollar o adaptar un manipulador así como su subsistema de control. Además es necesario desarrollar el subsistema inteligente de manipulación así como integrar todo a modo de producto terminado.

Por último es deseable mejorar la documentación existente sobre el sistema, tanto a nivel usuario, como de servicio y técnico.

Sirva el presente trabajo para mostrar la tesis propuesta, la cuál consistió en aumentar la automatización de los sistemas artificiales mediante visión por computadora. Se deja entonces un proyecto consolidado el cuál es gran base para continuar con la investigación aplicada a la microtecnología y los sistemas de visión por computadora en este flamante campo tecnológico.

Apéndices

Apéndice A

Software

Con el objetivo de contar con una herramienta para la investigación y los experimentos sobre los clasificadores neuronales LIRA y PCNC, así como para el localizador y la creación de las bases de datos, se crearon varios paquetes de software. El software se desarrolló mediante un lenguaje de Programación Orientado a Objetos (POO). Los lenguajes orientados a objetos permiten contar con módulos de software reutilizables, flexibles y de fácil mantenimiento. Esto hace posible que la investigación y desarrollo sean más eficientes en lo que a software se refieren y pone a disposición inmediata aquellos módulos de software que han sido probados con buenos resultados ya sea para un sistema completo, un sistema alternativo o para realizar otro tipo de experimentos.
El software desarrollado en primera instancia se compone de cuatro módulos: Optik, RNA, Localizador e Interfaz. El módulo Optik se encarga de la creación de bases de datos de imágenes, el módulo RNA es la implementación del clasificador neuronal LIRA, el módulo Localizador busca una pieza determinada en una imagen y la Interfaz es el módulo encargado de la intercomunicación entre los demás módulos así como con el usuario. En la Fig. A.1 se muestra un diagrama a bloques de estos módulos así como las interconexiones entre éstos y los archivos que se manejan, también se listan las funciones disponibles para el usuario. Se describen a continuación con más detalle cada uno de los módulos en los se hará referencia continua a la figura mencionada.
El software desarrollado posee muchas ventajas de la programación orientada a objetos (POO). Se utilizó el lenguaje de programación C++. En un inicio se usó el Ambiente integral de desarrollo Borland^{MR} y con ello algunos objetos predefinidos por Borland^{MR} fueron utilizados para la manipulación de imágenes y la creación de las interfaces gráficas.
Más tarde se decidió trasladar el código existente a C++ estándar por lo que se abandonó el uso de dicho ambiente de desarrollo. El uso del C++ estándar posibilita que el código pueda ser trasladado fácilmente a cualquier plataforma y a sistemas embebidos, lo anterior es especialmente importante para nuestro proyecto. Además, los costos de desarrollo se reducen al no depender de software comercial haciendo que el sistema sea más económico, característica que es una de las pautas más importantes en todo el trabajo desarrollado. Bajo esta nueva pauta de trabajo se desarrolló también el software PCNC.
La aplicación Optik está siendo desarrollado para GNU/Linux [REF] y las demás aplicaciones para el clasificador LIRA y el PCNC se desarrollaron en C++ estándar en este mismo sistema operativo. También el localizador de piezas. Si bien no se les desarrolló interfaz gráfica, esto se hizo premeditadamente debido a las siguientes razones:

1. Las buenas prácticas de programación moderna enseñan que la interfaz gráfica debe separarse totalmente de la implementación del software particular.
2. El poder que ofrece el hecho de poder correr el software desde la línea de comandos cuyo ejemplo más práctico se da en los archivos de procesamiento por lotes que posibilitaron la ejecución de decenas de experimentos en una sola orden y
3. La visión de que en algún momento del desarrollo futuro el sistema pueda ser embebido a un microcontrolador o algún otro hardware especializado.

A.1. Módulo Optik

Optik es un software que genera bases de datos de imágenes de objetos destinadas para entrenamiento y pruebas del clasificador neuronal LIRA. Se parte de imágenes generales de múltiples objetos ordenados

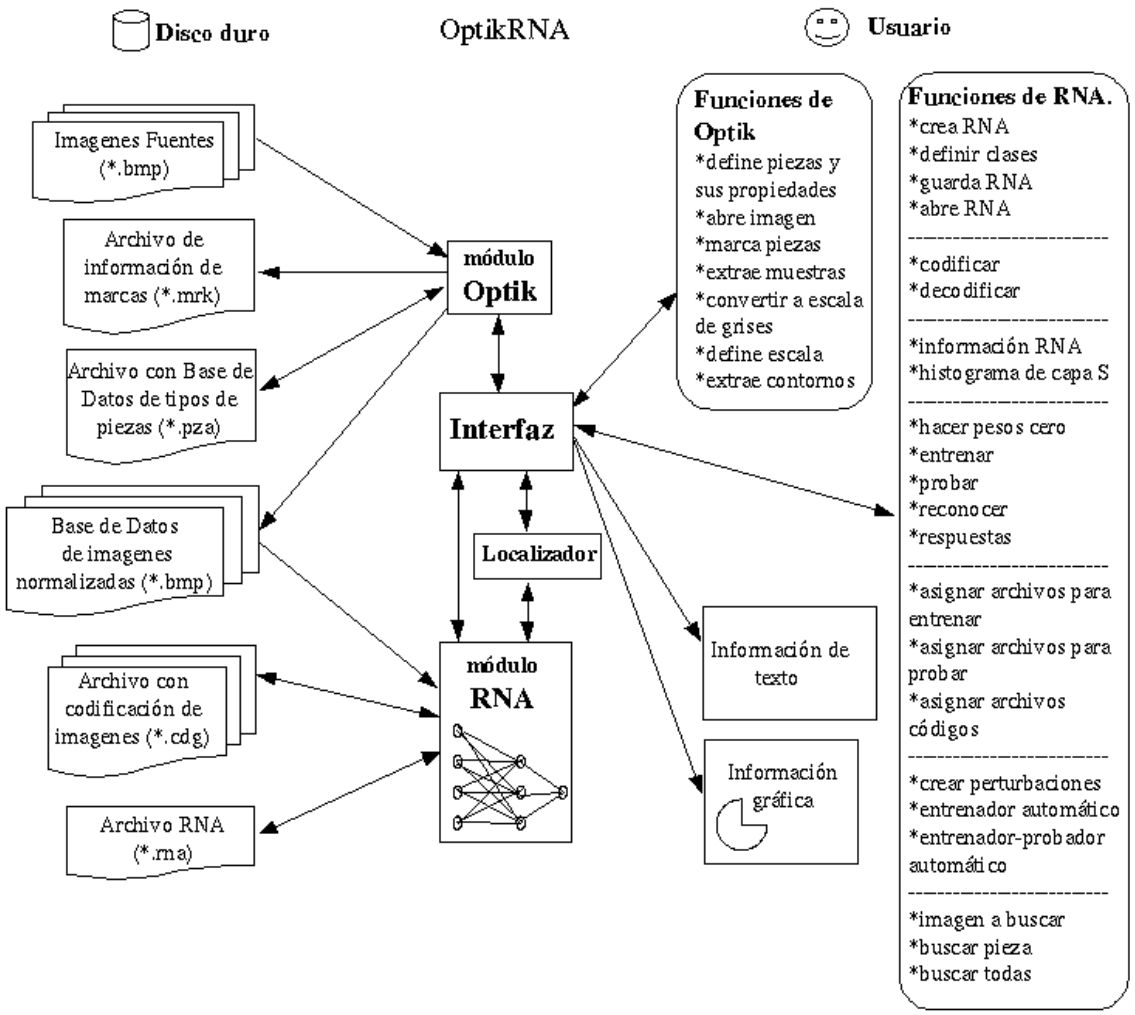


Figura A.1: Diagrama a bloques del software OptikRNA. Se muestran los cuatro módulos que lo componen, los archivos utilizados y sus principales funciones así como las intercomunicaciones entre todos estos elementos.

aleatoriamente, con ayuda de marcas colocadas por el usuario, genera una base de datos de imágenes normalizadas mediante un proceso de extracción. imágenes normalizadas se refiere a que éstas tienen iguales características: contienen una pieza centrada y con orientación fija de 0° respecto a su eje mayor, son de dimensión constante y tienen una ventana circular que facilita la rotación (Fig. A.2). Las imágenes extraídas son nombradas de acuerdo al tipo de pieza correspondiente y a un número consecutivo. También el sistema crea un archivo MRK que contiene los datos de todas las marcas realizadas.

Antes de el proceso de colocación de muestras, deben definirse los nombres y otras propiedades de los distintos tipos de piezas con que se va a trabajar (i. e. dimensiones, peso y material), éstos datos son de utilidad al sistema para calcular automáticamente el centro de la pieza además de futuras operaciones. Esta información se almacena en el sistema en un archivo PZA y puede ser cargada cuando se requiera. Optik además realiza algunas tareas de preprocesamiento de imágenes, como extracción de contornos y conversión de imágenes de color a escala de grises.

Debido al hecho de que las marcas en las escenas son puestas por el usuario mediante el ratón, las imágenes muestras para las bases de datos no están centradas ni orientadas exactamente. Este no es un problema ya que los clasificadores utilizados son capaces de llevar a cabo la tarea de reconocimiento con muestras imperfectamente centradas o giradas, sobre todo en conjuntos grandes de entrenamiento.

Este software en primera instancia se elaboró con Borland^{MR} C++, luego se decidió en colaboración con

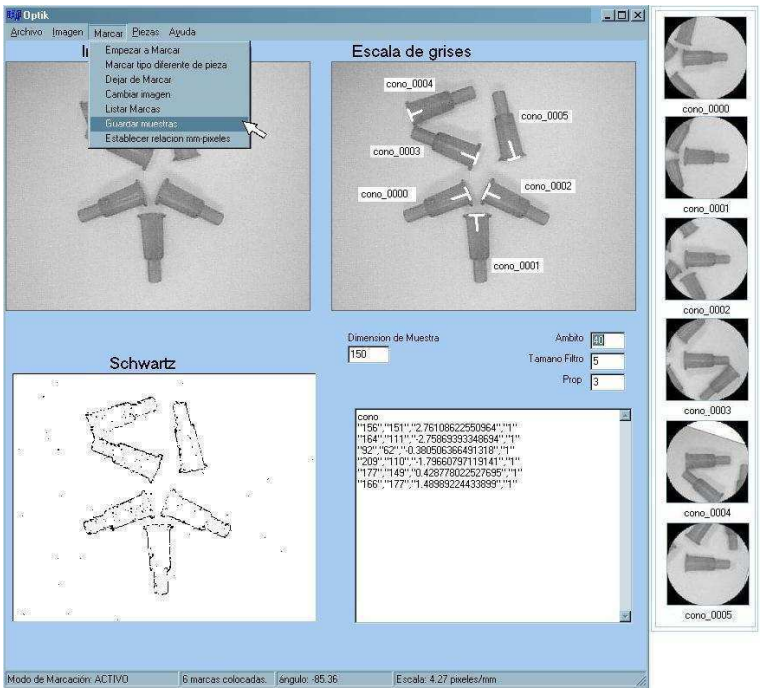


Figura A.2: IGU Optik, la cual permite marcar y extraer las muestras de imágenes a partir de las imágenes escena.

otros colegas pasarlo a sistema operativo GNU/Linux [90].

A.2. Módulo RNA

El módulo RNA implementa al clasificador neuronal LIRA, es decir, es la realización de la red neuronal completa, neurona a neurona, sus interconexiones y toda su funcionalidad. Este es el módulo más elaborado, su eficiencia es crítica, por lo que fue necesario cuidar dos aspectos fundamentales, memoria y velocidad. Un clasificador neuronal puede necesitar cientos de miles de neuronas y una magnitud mayor de interconexiones además de ser indispensable tiempos de ejecución total aceptables, tanto en las fases de entrenamiento como en las de prueba, siendo crítico en una aplicación real de reconocimiento (módulo Localizador). Para cuidar la velocidad se decidió programar este módulo y por extensión todo OptikRNA con lenguaje C++, es decir, se utilizó la velocidad y el poder de C combinado con las ventajas de la POO. Se utilizaron punteros, estos permiten operar a bajo nivel mejorando la velocidad y realizar las interconexiones entre las distintas clases utilizadas más eficientemente. Para cuidar la eficiencia evitando operaciones de punto flotante se utilizaron únicamente números enteros. Este módulo se constituye de diversas clases, las principales, junto con su relación de herencia se muestran en la Fig. A.3.
Existe una súper clase llamada RNA la cual construye con las clases descritas antes y otras menores no mencionadas el clasificador neuronal LIRA. Esta clase ofrece las mismas funciones que tiene el clasificador Lira descritas en la Sección 2 además de otras necesarias para su implementación y funcionamiento, entre las principales están: creación, codificación, entrenamiento, reconocimiento y borrado de memoria (pesos a cero). La clase RNA controla completamente a todo el módulo y es con la cuál se comunica realmente el módulo Interfaz.

A.3. Módulo Localizador

El módulo Localizador se encarga de buscar una pieza requerida en una imagen fuente y definir la posición de ésta. La pieza que será capaz de localizar este módulo debe estar en el conjunto de piezas con que el clasificador se haya entrenado previamente. La imagen fuente no tiene por que ser una imagen ocupada

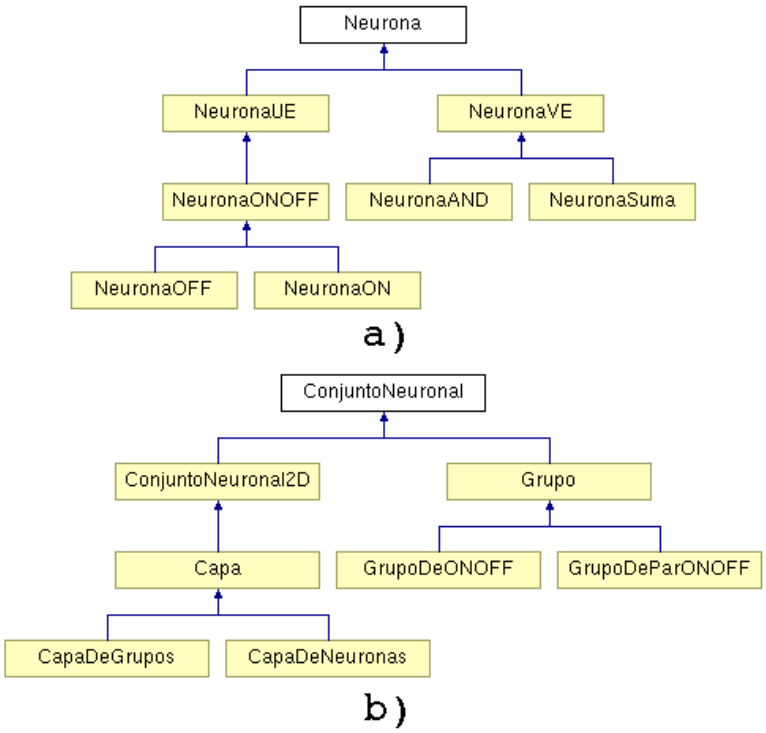


Figura A.3: Clases principales del módulo RNA y su relación de herencia. a) Clases a nivel neurona. b) Clases a nivel capa.

para la extracción de imágenes normalizadas, puede ser cualquier imagen siempre que contenga el objeto a buscar en la misma escala en que existe en las imágenes ocupadas.
Sobre una imagen dada, este módulo aplica un algoritmo de búsqueda que consiste en ir tomando sub-imágenes empezando por el centro y desplazándose en forma espiral (Fig. 5b). Para cada posición se pide al módulo RNA identificar la pieza requerida y si no se encuentra se rota un cierto ángulo y se repite el proceso hasta encontrar la pieza o cuando una rotación es completada, si no se encuentra, se continua el desplazamiento espiral hasta encontrar la pieza buscada o alcanzar los límites de la imagen. Los desplazamientos lineales y angulares pueden ser definidos por el usuario.
El módulo Localizador constituye una aplicación práctica concreta en microensamble y puede funcionar independientemente de los módulos Optik e Interfaz con el fin de ser acoplado a sistemas automáticos de manipulación de piezas. En la Fig. 5c se muestra un resultado de la búsqueda de una pieza ("tornillo de cabeza redonda") sobre una imagen que contiene diversas piezas. El sistema despliega las coordenadas de la pieza así como la orientación. Cuando la pieza se localiza lejos de su centro, la orientación es errónea, por esta razón este módulo debe ser mejorado.

A.4. Interfaz

El módulo Interfaz es el único módulo que se comunica con el usuario, además intercomunica los demás módulos para administrarlos. Este módulo en si es una Interfaz Gráfica de Usuario (IGU), cuenta con todas las funciones disponibles de OptikRNA y con áreas para desplegar imágenes y resultados. En la Fig. A.4 se muestra ésta interfaz. El área llamada "parámetros del clasificador.[es] a donde el usuario ingresa los parámetros para la creación de un clasificador neuronal LIRA en particular. Abajo está el panel de funciones básicas de la RNA, desde aquí se envían los comandos para el módulo RNA, como crear, guardar, cargar y reconocer. En el área de mensajes se despliega información general del clasificador cargado. En el panel de funciones avanzadas existen funciones para el módulo RNA como entrenar, probar, asignar bases de datos y entrenar-probar automáticamente. En el área de resultados se muestran los resultados de las pruebas o el reconocimiento de piezas. Por último, desde el panel del módulo Localizador accesamos a las

funciones y parámetros de este módulo. En la figura referida se muestra un clasificador cargado junto con su información general y los resultados de una prueba aplicada. En el área vacía es donde se despliega la imagen utilizada por el Localizador. Las funciones del módulo Optik están en otra IGU que no se muestra.

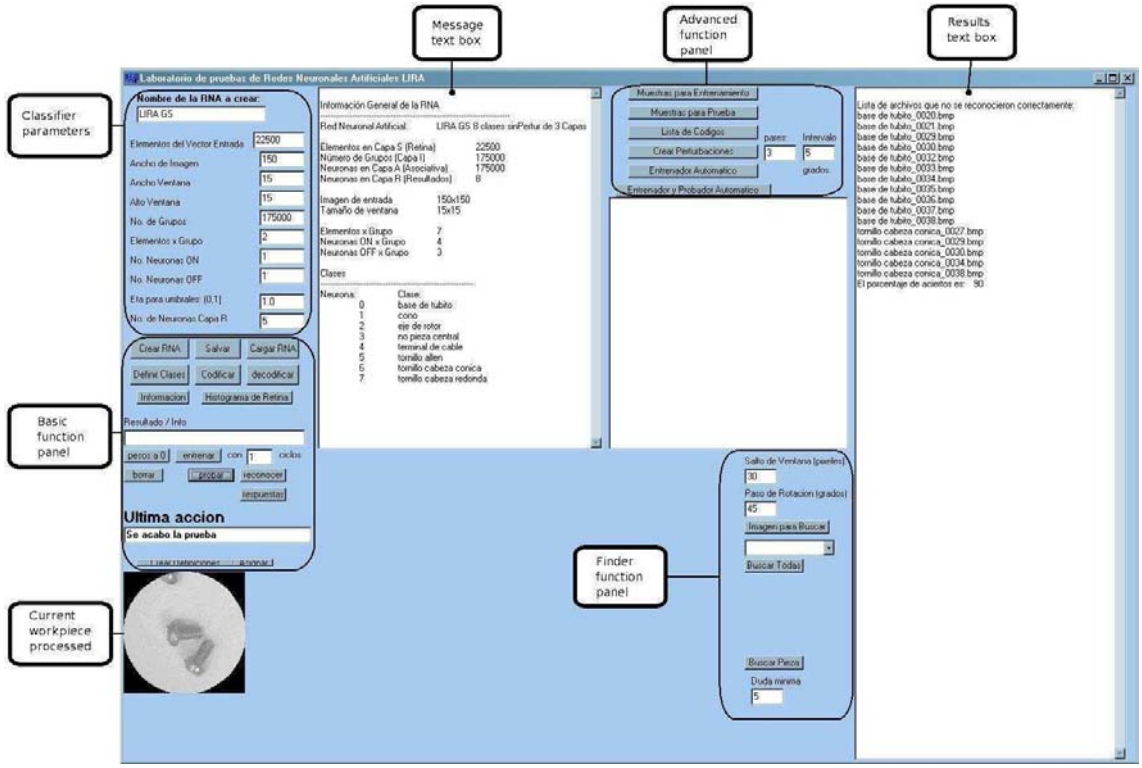


Figura A.4: Interfaz gráfica de usuario del software RNA.

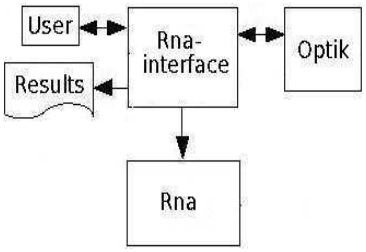


Figura A.5: Diagrama a bloques del software OptikRNA.

El usuario trabaja con el módulo de software de redes neuronales que implementa el clasificador LIRA, esto lo hace a través de la IGU Rna (Fig. A.4). Con la ayuda de esta interfaz el usuario puede crear el clasificador neuronal LIRA, entrenar, probar y usar el clasificador así como utilizar el localizador de piezas. La IGU Rna con un clasificador entrenado ya cargado es mostrada en la Fig. A.4. En el cuadro de texto de la izquierda se muestra información sobre el clasificador. Los resultados se dan en el cuadro de texto de la derecha: el porcentaje de reconocimiento obtenido y los nombres de los archivos de las muestras que el sistema no pudo reconocer.
En el centro de la Fig. A.6 se muestra una escena en la cuál se ha llevado a cabo el procesamiento con el localizador de piezas. Un clasificador previamente entrenado y probado ha sido ya cargado. En esta misma imagen se muestra también en el cuadro de texto de la derecha, una marca sobre la pieza localizada así como las coordenadas y orientación asociada a la misma.

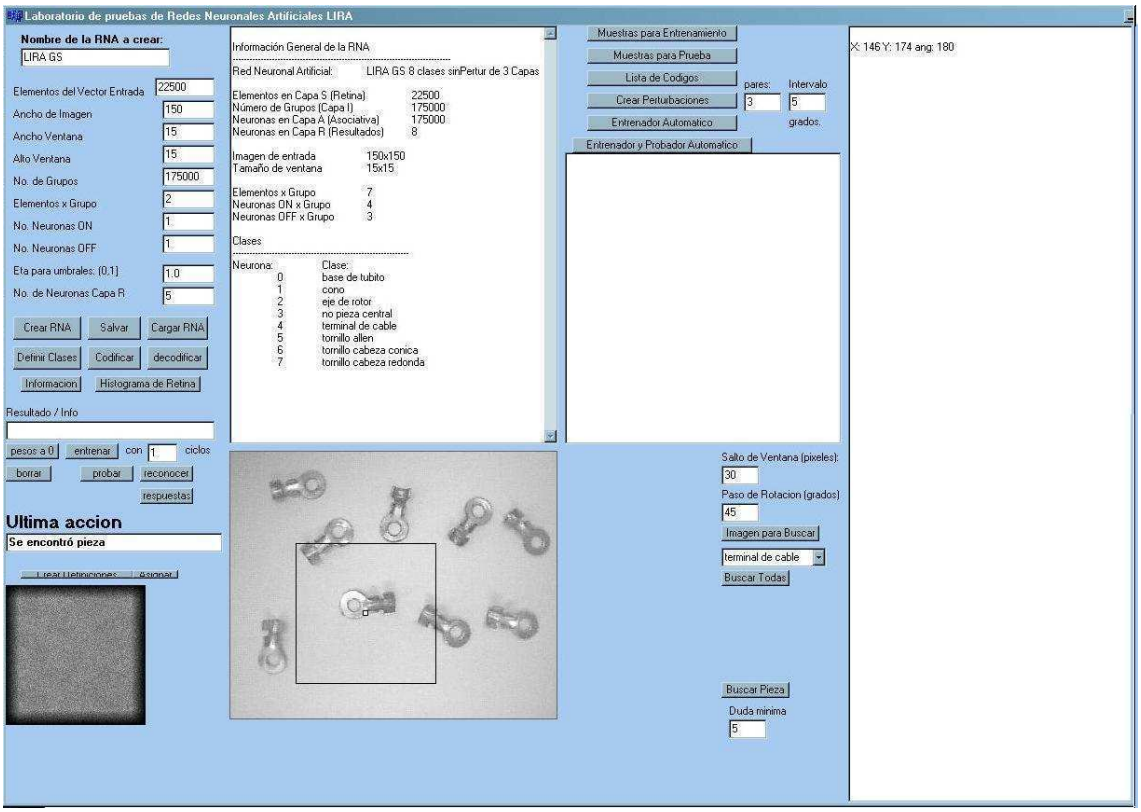


Figura A.6: IGU Rna. Localizador de piezas.

A.5. Implementación de LIRA

Our neural classifier software is based on integer numbers operations in order to avoid large time effort that is necessary for floating point operations. The most general classes were made, that means that the same software modules can be used to create different neural network topologies. The user interface was made specially for LIRA neural classifier. It is not hard to use this software modules to construct other types of neural networks or make changes to the current topology, e. g. add more layers.

The most important classes created for artificial neural network realization were: Dentrita, Neuron, NeuralSet and Rna.

The Dentrita class is a basic one. It has only two attributes, stimulus and weight. Dentrita instances are used widely by neuron objects in order to create fully functional neurons. In Fig. A.7 the Neuron class and its derived classes are shown. The neuron types used in LIRA classifier are ON, OFF, AND and adding neurons. The ON and OFF neurons are one input (OI) neurons and the rest are several input (SI) neurons. The Neuron class is the fundamental part of the neural network software.

In order to construct neuron sets a general NeuralSet class was created. Rna is the class that combines all mentioned classes. This class we use to construct a LIRA neural network. Examples of the parameters are the number of neurons or groups in each layer, the number of ON and OFF neurons in each group, the input vector and the output classes. The Rna object contains information about itself. It is possible to store the complete neural network including all its internal parameters, to load a previously stored neural network and to perform training and recognition processes.

The neural classifier is controlled by an object called Rna-Interface. That interrelates the user by means the GUI with the databases used for training and testing of the classifier.

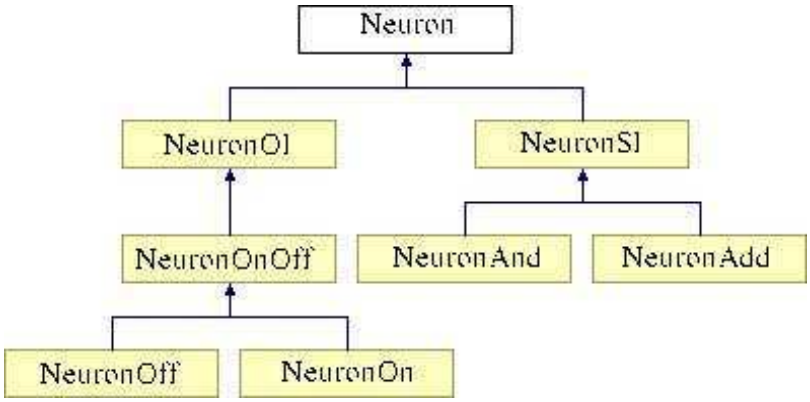


Figura A.7: Clase neurona y sus clases derivadas.

A.6. Software de línea de comandos

A.6.1. lira2007

A continuación se presenta la ayuda propia del software *lira2007* que ha sido usado para implementar y experimentar con el clasificador LIRA así como su interacción con las bases de datos.

```
Example to use follow()-functions:

USAGE:
--help, -h
       display this help.

*** Setup ***
--create adn='<RNA name (string)> <tamVectEnt> <ImageWidth> <windowWidth> <windo
wHeight> <numGroup> <elemXGroup> <numNeuON> <numNeuOFF> <eta (double)> <numClass
es>', you can use -c instead of --create. Non especified are (int).
For example use:
 lira -c adn='liraSUN 22500 150 15 15 170000 7 4 3 1.0 8'
       Create a Lira Neural Classifier from scrath.
--load <filename.rna>, -l <filename.rna>
       load a neural classifier from specified file.
--info, -i
       displays info about the loaded classifier.
--classes-file <filename.ent>, -cf <filename.ent>
       assign the classes file to LIRA classifier.

*** Preparing ***
--train-dir <dir>, -td <dir>
       assign the training directory, default is "./train".
--code-dir <dir>, -cd <dir>
       assign the code directory, default is "./code".
--code, -k
       code all the images in the train directory, they will put in the code di
rectory
--reset, -kill
       reset the internal connections of the loaded ANN, Erase its memory
--train num, -t num
       train using the coded images from the given code dir using "num" cycles.
 Default is 40
--prove-dir <dir>, -pd <dir>
```

```
       assign the prove directory, default is "./prove".
--images-dir <dir>, -id <dir>
       assign the images directory, default is "./images".
--train-percentage num, -tp num
       Percentaje of images in images-dir to be selected for the training set.
Default is 50%

*** Using ***
--prove, -p
       proves the classifier with the images files stored in the "proving direc
tory".
--recognize <filename.png>, -r <filename.png>
       recognize a class in the given normalized image file.
pos='<x (int)> <y (int)> <O (double)>', Parameters especification for recogniz
ed a big (not normalized) image in certain point and orientation. All values s
hould be (int).
You might specified a "filename.png" by -r to look for.
Example use:
 lira -l lira.rna -r imagen.png pos='10 20 -45.0'
An image file called "cut.png" will be created with the used subimage.

*** Ending***
--save, -s
       Save the current classifier to a .rna file.
--close, -q
       (NOW FAIL!) Save the current classifier to a .rna file.
       close the current classifier loaded in memory.
--output, -o
       Print values from the output layer.

*** Search ***
--searchImage <filename.png>, -si <filename.png>
       Search for some recognized clase.
--searchClass <className>, -sc <className>
       Class to be search by the Search command. Use "prueba" to make a prove
 and "cualquiera" to search anyone. Default is anyone
--searchQuantity num, -sq num
       Number of objects to search for. Default is 1.
--searchStep num, -ss num
       Step in pixels for the searcher to jump (/\x & /\y). Default is 20.
--searchStepAngle num, -sa num
       Angle in degrees for the searcher to jump. Default is 45.

*** Tools ***
--distortion dis='<dx> <nx> <dy> <ny> <dO> <nO>', you can use -d instead of -
-distortion. All parameters should be (int).
You might specified the "training dir".
For example use:
 lira -td ./trainimages -d dis='5 2 5 2 5 2'
```

A.6.2. pcnc2007

A continuación se presenta la ayuda propia del software *pcnc2007* que ha sido usado para implementar y experimentar con el PCNC así como su interacción con las bases de datos.

```
Example to use follow()-functions:

USAGE:
--help, -h
```

```
       display this help.

*** Setup ***
--create adn='<PCNC name (string)> <method> <Tmin> <Tmax> <w> <h> <p> <n> <S>
 <N> <K> <Dc> <q> <numClasses>', you can use -c instead of --create. Non espe
cified are (int).
For example use:
 peco -c adn='PCNCprueba 0 1 32000 5 5 3 2 500 1000 12 8 5 8'
       Create a Permutative Code Neural Classifier (PCNC) from scrath.
--load <filename.pcnc>, -l <filename.pcnc>
       load a PCNC from specified file.
--info, -i
       displays info about the loaded PCNC.
--classes-file <filename.ent>, -cf <filename.ent>
       assign the classes file to PCNC.

*** Preparing ***
--train-dir <dir>, -td <dir>
       assign the training directory, default is "./train".
--code-dir <dir>, -cd <dir>
       assign the code directory, default is "./code".
--code, -k
       code all the images in the train directory, they will put in the code
directory
--reset
       reset the internal connections of the loaded PCNC, Erase its memory
--train num, -t num
       train using the coded images from the given code dir using "num" cycles
. Default is 40
--prove-dir <dir>, -pd <dir>
       assign the prove directory, default is "./prove".
--images-dir <dir>, -id <dir>
       assign the images directory, default is "./images".
--train-percentage num, -tp num
       Percentaje of images in images-dir to be selected for the training set.
 Default is 50%

*** Using ***
--prove, -p
       proves the PCNC with the images files stored in the "proving directory".
--recognize <filename.png>, -r <filename.png>
       recognize a class in the given normalized image file.

*** Ending***
--save, -s
       Save the current PCNC to a .pcnc file.
--close, -q
       (NOW FAIL!) Save the current PCNC to a .pcnc file.
       close the current PCNC loaded in memory.
```

A.6.3. Potencial para la experimentación

Como un ejemplo del uso ventajoso del software de línea de comandos se presenta uno de los múltiples archivos de procesamiento por lotes[1] empleado para realizar decenas de experimentos con cambió de parámetros de forma automática, es decir, en una sola corrida.

```
#!/bin/bash
```

[1]Más conocidos por su denominación en inglés: *scripts*

```
#Decenas de pruebas para estudiar comportamiento de parámetros
#La base es el Mejor:
#../../bin/pcnc2007 -c adn='PCNCmejorBD-A 1 0 65535 10 10 5 4 1000 300000 20 5
 2 8' -cf BD-A.ent -td ../../muestras/entrenamiento -cd ../../muestras/codigo
-pd ../../muestras/prueba -k -t 40 -p -s


#*************PARAMETRO W
#Inicializacion
PARAM=W
PARw=10
PARp=5
PARn=4
PARS=1000
PARN=300000
PARK=20
PARDc=5
PARq=2
#Usando "for" para probar distintos parámetros
for PARw in 05 08 09 11 12 15 20;
  do
  echo "**************************"
  echo "***** CORIDA: "$PARAM:$PARw" *******"
  echo "**************************"
  #Hace todo de una vez, también salva
  ../../bin/pcnc2007 -c adn='PCNCparam'$PARAM''$PARw' 1 0 65535 '$PARw' '$PARw
' '$PARp' '$PARn' '$PARS' '$PARN' '$PARK' '$PARDc' '$PARq' 7' -cf BD-D.ent -td
 ../../muestrasBD-D/entrenamiento -cd ../../muestrasBD-D/codigo -pd ../../mues
trasBD-D/prueba -k -t 30 -p -s

# ../../bin/pcnc2007 -c adn='PCNCparam'$PARAM' 1 0 65535 '$PARw' '$PARw' '$PAR
p' '$PARn' '$PARS' '$PARN' '$PARK' '$PARDc' '$PARq' 7' -cf BD-D.ent -i
done

#*************PARAMETRO K
#Inicializacion
PARAM=K
PARw=10
PARp=5
PARn=4
PARS=1000
PARN=300000
PARK=20
PARDc=5
PARq=2
#Usando "for" para probar distintos parámetros
for PARK in 05 10 15 25 30;
  do
  echo "**************************"
  echo "***** CORIDA: "$PARAM:$PARK" *******"
  echo "**************************"
  #Hace todo de una vez, también salva
  ../../bin/pcnc2007 -c adn='PCNCparam'$PARAM''$PARK' 1 0 65535 '$PARw' '$PARw
' '$PARp' '$PARn' '$PARS' '$PARN' '$PARK' '$PARDc' '$PARq' 7' -cf BD-D.ent -td
 ../../muestrasBD-D/entrenamiento -cd ../../muestrasBD-D/codigo -pd ../../mues
trasBD-D/prueba -k -t 30 -p -s
done
```

```
#*************PARAMETRO Dc
#Inicializacion
PARAM=Dc
PARw=10
PARp=5
PARn=4
PARS=1000
PARN=300000
PARK=20
PARDc=5
PARq=2
#Usando "for" para probar distintos parámetros
for PARDc in 02 04 06 08 10 15 20 25;
  do
  echo "*************************"
  echo "***** CORIDA: "$PARAM:$PARDc" *******"
  echo "*************************"
  #Hace todo de una vez, también salva
  ../../bin/pcnc2007 -c adn='PCNCparam'$PARAM''$PARDc' 1 0 65535 '$PARw' '$PARw
' '$PARp' '$PARn' '$PARS' '$PARN' '$PARK' '$PARDc' '$PARq' 7' -cf BD-D.ent -td
 ../../muestrasBD-D/entrenamiento -cd ../../muestrasBD-D/codigo -pd ../../mues
trasBD-D/prueba -k -t 30 -p -s
done

#*************PARAMETRO q
#Inicializacion
PARAM=q
PARw=10
PARp=5
PARn=4
PARS=1000
PARN=300000
PARK=20
PARDc=5
PARq=2
#Usando "for" para probar distintos parámetros
for PARq in 00 01 03 04 05 10;
  do
  echo "*************************"
  echo "***** CORIDA: "$PARAM:$PARq" *******"
  echo "*************************"
  #Hace todo de una vez, también salva
  ../../bin/pcnc2007 -c adn='PCNCparam'$PARAM''$PARq' 1 0 65535 '$PARw' '$PARw
' '$PARp' '$PARn' '$PARS' '$PARN' '$PARK' '$PARDc' '$PARq' 7' -cf BD-D.ent -td
 ../../muestrasBD-D/entrenamiento -cd ../../muestrasBD-D/codigo -pd ../../mues
trasBD-D/prueba -k -t 30 -p -s
done

#*************PARAMETROS p y q
#Inicializacion
PARAM=PyN
PARw=10
PARp=5
PARn=4
PARS=1000
PARN=300000
PARK=20
```

```
PARDc=5
PARq=2
#Usando "for" para probar distintos parámetros
for PARpn in "3 6" "4 5" "6 3" "7 2" "2 5" "3 4" "4 3" "5 2" "6 1" "4 7" "5 6"
 "6 5" "7 4" "8 3" "9 2";
  do
  echo "**************************"
  echo "***** CORIDA: "$PARAM:$PARpn" *******"
  echo "**************************"
  #Variable auxiliar para nombrar sin espacios
  PARpnNom=${PARpn/ /y}
  #Hace todo de una vez, también salva
  ../../bin/pcnc2007 -c adn='PCNCparam'$PARAM''$PARpnNom' 1 0 65535 '$PARw' '$P
ARw' '"$PARpn"' '$PARS' '$PARN' '$PARK' '$PARDc' '$PARq' 7' -cf BD-D.ent -i -td
 ../../muestrasBD-D/entrenamiento -cd ../../muestrasBD-D/codigo -pd ../../mues
trasBD-D/prueba -k -t 30 -p -s
done



#*************PARAMETRO N
#Inicializacion
PARAM=N
PARw=10
PARp=5
PARn=4
PARS=1000
PARN=300000
PARK=20
PARDc=5
PARq=2
#Usando "for" para probar distintos parámetros
for PARN in 100000 200000 250000 400000 500000;
  do
  echo "**************************"
  echo "***** CORIDA: "$PARAM:$PARN" *******"
  echo "**************************"
  #Hace todo de una vez, también salva
  ../../bin/pcnc2007 -c adn='PCNCparam'$PARAM''$PARN' 1 0 65535 '$PARw' '$PARw
' '$PARp' '$PARn' '$PARS' '$PARN' '$PARK' '$PARDc' '$PARq' 7' -cf BD-D.ent -td
 ../../muestrasBD-D/entrenamiento -cd ../../muestrasBD-D/codigo -pd ../../mues
trasBD-D/prueba -k -t 30 -p -s
done


#*************PARAMETRO S
#Inicializacion
PARAM=S
PARw=10
PARp=5
PARn=4
PARS=1000
PARN=300000
PARK=20
PARDc=5
PARq=2
#Usando "for" para probar distintos parámetros
for PARS in 0400 0600 0800 1200 1500 2000 2500 3000 4000 5000 6000 10000;
```

```
  do
  echo "*************************"
  echo "***** CORIDA: "$PARAM:$PARS" *******"
  echo "*************************"
  #Hace todo de una vez, también salva
  ../../bin/pcnc2007 -c adn='PCNCparam'$PARAM''$PARS' 1 0 65535 '$PARw' '$PARw
' '$PARp' '$PARn' '$PARS' '$PARN' '$PARK' '$PARDc' '$PARq' 7' -cf BD-D.ent -td
 ../../muestrasBD-D/entrenamiento -cd ../../muestrasBD-D/codigo -pd ../../mues
trasBD-D/prueba -k -t 30 -p -s
done

#Crea el resumen de resultados
grep "El porcentaje" PCNCparam*.proveresults >> resumen.proveresults

#FIN

#  LocalWords:  PARw
```

Apéndice B

Publicaciones

El presente trabajo dió lugar a las siguientes publicaciones y congresos nacionales e internacionales:

1. G.K. Toledo, E. Kussul, T. Baidyk. Neural classifier LIRA for recognition of micro work pieces and their positions in the processes of microassembly and micromanufacturing. The Seventh All-Ukrainian International Conference on Signal/Image Processing and Pattern Recognition, UkrObraz2004, Kiev, Ukraine, October 2004, pp. 1720.
2. G. Toledo, E. Kussul, T. Baidyk. Clasificador neuronal LIRA para reconocimiento de piezas de trabajo y sus posiciones en los procesos de microensable y micromanufactura. Memorias del XIX Congreso de Instrumentación SOMI, Pachuca, Hidalgo, México, 25-29, octubre, 2004.
3. Ernst Kussul, Tatiana Baidyk, Gengis Kanhg Toledo Ramírez. Investigación y desarrollo del sistema de visión técnica para aplicaciones en micromaquinado y microensamble. Reporte técnico, CCADET-UNAM, 15 de junio de 2005, 29 p.
4. Gengis K. Toledo-Ramírez. Software orientado a objetos para investigación de redes neuronales. Memorias del XX Congreso de Instrumentación SOMI, León, Guanajuato, México, 24-28, Octubre, 2005.
5. Gengis K. Toledo, Ernst Kussul, Tatiana Baidyk. Neural classifier for micro work piece recognition. Image and Vision Computing. Elsevier. Vol 24/8, 2006, pp 827-836.
6. Gengis K. Toledo-Ramírez, Ernst Kussul, Tatiana Baidyk. Object oriented software for micro work piece recognition in microassembly. Journal of Applied Research and Technology. Vol 4/1, 2006, pp 59-74.
7. Josué Enríquez Zárate, Zaizar Betanzos Cortés, Gengis Kanhg Toledo Ramírez. Sistema Generador de Bases de Datos para Imágenes: una aplicación con Gambas. I Simposium de Linux de la Mixteca, 2-4, Marzo, 2006. http://www.utm.mx/gulmix/simposium_2006/ponencia.html
8. Gengis K. Toledo, Ernst Kussul, Tatiana Baidyk. Clasificación de piezas mediante un clasificador neuronal artificial. Memorias del XXI Congreso de Instrumentación SOMI, Ensenada, Baja California, México, 22-25, Octubre, 2006.
9. Gengis Kanhg, Kussul Ernst, Baidyk Tatiana. Reconocimiento de piezas mediante un clasificador neuronal con permutación de códigos. Artículo aceptado para el XXII Congreso de Instrumentación SOMI, Monterrey, Nuevo León, México, 1-4, Octubre, 2007.

Asi mismo al momento de terminar este trabajo una nueva publicación internacional está en proceso y un nuevo trabajo está postulándose para un congreso internacional.

Bibliografía

[1] Willian S. Trimmer, editor. *Micromechanics and MEMS, Classic and Seminal Papers to 1990*. IEEE Press, 1997.

[2] T. Ofeifer and G. Dussler. Process observation for the assembly of hybrid microsystems. In *IEEE International Sysposium on Micromechatronics and Human Science*.

[3] H. Bleuler, R. Clavel, J-M. Brequet, H. Langen, and E. Pernette. Issues in precision motion control and microhandling. In *Procedings of the IEEE International Conference on Robotics Automation*, pages 959–964, USA, 2000.

[4] E. Kussul, T. Baidyk, L. Ruiz-Huerta, A. Caballero, G. Velasco, and L. Kasatkina. Development of micromachine tool prototypes for microfactories. *Journal of Micromechanical and Microengineer*, 12(6):795–812, October 2002.

[5] E. Kussul, T. Baidyk, L. Ruiz, A. Caballero, and G. Velasco. Development of low cost microequipment. In *International Symposium on Micromechatronics and Human Science*, pages 125–134, Nagoya, Japan, October 2002.

[6] C.R. Friedrich and M.J. Vasile. Development of the micromilling process for high- aspect- ratio microstructures. *Journal of Microelectromechanical Systems*, 5:33–38, 1996.

[7] N. Ooyama, S. Koka ji, M. Tanaka, and et al. Desktop machining microfactory. In *Proceedings of the 2nd International Worshop on Microfactories*, pages 14–17, Switzerland, October 2000.

[8] R. Jain, B.G. Schunck, and R. Kasturi. *Machine Vision*. McGraw Hill, New York, U.S.A., 1995.

[9] Faugeras O. *Three-dimensional Computer Vision*. Cambridge, Massachusetts, MIT Press, 1993.

[10] P. Steinhaus. *Prinzipielle Komponenten-Analyse versus multimediale Filterhistogramme*. PhD thesis, Universitat Karlsruhe (TH), Germany, July 1997.

[11] Sun-Ho Lee, Hyun-Ki Hong, and Jong-Soo Choi. A study on assembly part recognition using part-based superquadratic model. *IEEE*, pages 78–82, 1999.

[12] M. Bennamoun and B. Boashash. A structural description based vision system for automatic object recognition. *IEEE Transactions on Systems, Man and Cybernetics*, Part B-27(6):893–906, December 1997.

[13] G.K. Toledo, E. Kussul, and T. Baidyk. Neural classifier LIRA for recognition of micro work pieces and their positions in the processes of microassembly and micromanufacturing. In *The Seventh All-Ukrainian International Conference on Signal/Image Processing and Pattern Recognition, UkrObraz2004*, pages 17–20, Kiev, Ukraine, October 2004.

[14] F. Rosenblatt. *Principles of Neurodynamics*. Spartan books, New York, 1962.

[15] E. Kussul and T. Baidyk. Improved method of handwritten digit recognition tested on MNIST database. *Image and Vision Computing, ELSEVIER*, 22(12):971–981, October 2004. doi:10.1016/j.imavis.2004.03.008.

[16] T. Baidyk, E. Kussul, O. Makeley, A. Caballero, L. Ruiz, G. Carrera, and G. Velasco. Flat image recognition in the process of microdevice assembly. *Pattern Recognition Letters, Elsevier*, 25(1):107–118, January 2004.

[17] E. M. Kussul, D. A. Rachkovskij, and T.Ñ. Baidyk. On image texture recognition by associative-projective neurocomputer. In *Proc. of the ANNIE'91 conference "Intelligent engineering systems through artificial neural networks"*, pages 453–458. ASME Press, C.H. Dagli and S. Kumara and Y.C. Shin, 1991.

[18] E. M. Kussul, D. A. Rachkovskij, and T.Ñ. Baidyk. Associative-projective neural networks: architecture, implementation, applications. pages 463–476, Nimes, France, Nov. 4-8 1991.

[19] E. Kussul and T. Baidyk. Permutative coding technique for handwritten digit recognition system. *IEEE*, pages 2163–2168, 2003.

[20] F. Lara-Rosano, E. Kussul, T. Baidyk, L. Ruiz, A. Caballero, and G. Velazco. Artificial intelligence systems in micromechanics. In *The first IFIP International Conference on Artificial Intelligence Applications and Innovations.*, pages 1–10, Toulouse, France, August 2004. Ed. By Max Bramer, Boston/Dordrecht/London, Kluwer Academic Publishers.

[21] G.K. Toledo, E. Kussul, and T. Baidyk. Neural classifier for micro work piece recognition. *Image and Vision Computing, Elsevier.*, 24(8):827–836, 2006.

[22] G.K. Toledo, E. Kussul, and T. Baidyk. Object oriented software for micro work pieces recognition in microassembly. *Journal of Applied Research and Technology*, 4(1):59–74, April 2006.

[23] G.K. Toledo. Clasificación de piezas mediante un clasificador neuronal artificial. Ensenada, Baja California. México, October 22-25 2006. XXI Congreso de instrumentación.

[24] E. Kussul and T. Baidyk. Improved method of handwritten digit recognition tested on MNIST database. In *15-th International Conference on Vision Interface*, pages 192–197, Calgary, Canada, May 27-29 2002.

[25] Ernst Kussul, Dmitri Rachkovskij, Tatyana Baidyk, and Semion Talayev. Micromechanical engineering: a basis for the low cost manufacturing of mechanical microdevices using microequipment. *Journal of Micromechanics and Microengineering*, 6(4):410–425, August 1996. doi:10.1088/0960-1317/6/4/008.

[26] R.P. Feynman. There's plenty of room at the bottom. A speech given at the annual meeting of the American Physical Society, December 1959. Available on the web at http://www.zyvex.com/nanotech/feynman.html.

[27] Mohamed Gad el Hak, editor. *The MEMS Handbook*. CRC, first edition, september 27 2001.

[28] P. Rai-Choudhury, editor. *MEMS and MOEMS Technology and Applications*. SPIE–the International Society for Optical Engineering, December 1 2000.

[29] N. P. Mahalik. *Micromanufacturing and nanotechnology: fundamentals, techniques and platforms*. Springer Verlag, november 15 2005.

[30] Jeff Wolfe. Microtechnology, changing the world little by little. Northwest Science & Technology, Winter 2004.

[31] H. Detter and G. Popovic. Industrial demands on micromechanical products. In *IEEE International Conference on Microelectronics*, pages 61–67, NIS, Serbia, 2000.

[32] System Planing Corporation. Mems and spc market study, 1994.

[33] T Fukuda and W Menz, editors. *Micro Mechanical Systems: Principles and Technology*. Elsevier, Amsterdam, 1998.

[34] Nadim Maluf. *An introduction to Microelectromechanical Systems Engineering*. First edition, december 1999.

[35] Franck Chollet and Haobing Liu. A (not so) short introduction to micro electro mechanical systems. Available in the web at `http://memscyclopedia.org/introMEMS.html`.

[36] P. Rai-Choundhury, editor. *Handbook of Microlithography*, volume 2 of *Micromachining and Microfabrication*. SPIE Press, Bellingham. WA., 1997.

[37] H. Ishihara, F. Arai, and T. Fukuda. Micro mechatronics and micro actuators. *IEEE/ASME Transaction of Mechatronics*, 1:68–79, 1996.

[38] Makoto Tanaka. Development of desktop machining microfactory. In *RIKEN Review*, number 34 in Focused on Advances on Micro-mechanical Fabrication Techniques, pages 46–49. April 2001.

[39] Yuichi Okazaki, Nozomu Mishima, and Kiwamu Ashida. Microfactory - concept, history, and developments. *Journal of Manufacturing Science and Engineering*, 126:837–844, 2004.

[40] Some micro machine activities in japan. Technical report.

[41] Okazaki Yuichi and Kitahara Tokio. Micro-lathe equipped with closed-loop numerical control. In *Proceedings of the 2-nd International Workshop on Microfactories*, pages 87–90, Switzerland, October 2000.

[42] E.M. Kussul, T.N. Baidyk, D.A. Rachkovskij, and S.A. Talayev. Método de manufactura de productos micromecánicos., 1997. Patente No. 96102125.02.

[43] E.M. Kussul, T.N. Baidyk, D.A. Rachkovskij, and S.A. Talayev. Método de producción de productos micromecánicos., 1998. Patente No. 24091.

[44] T Mazuzawa. An approch to micromachining through machine tool technology. In *Procedings of 2nd International Symposium of Micro Machine and Human Science*, pages 47–52, Nagoya, Japan, 1991.

[45] C.R. Friedrich and S.D. Kang. Micro heat exchangers fabricated by diamond machining. *Precision Engineering*, 16:56–59, 1994.

[46] Y. Yamagata and T. Higuchi. Four axis ultra precision machine tool and fabrication of micro parts by precision cutting technique. In *Proceeding 8th International Precision Engineering Seminar*, pages 467–470, Compiegne, France, 1995.

[47] Karoly Santa, Sergej Fatikow, and Gabor Felso. Control of microassembly-robots by using fuzzy-logic and neural networks. *Computer in industry. Elsevier*, 39:219–227, 1999.

[48] A. Mardanov, J. Seyfried, and S. Fatikow. An automated assembly system for a microassembly station. *Computer in Industry, Elsevier*, 38:93–102, 1999.

[49] Arun D. Kulkarni. *Artificial Neural Networks for Image Understanding*. Van Nostrand Reinhold, 1994.

[50] Jon Stenerson. *Industrial Automation and Process Control*. Prentice Hall, first edition, September 17 2002.

[51] Jonathan Wu Q.M., M.F. Ricky Lee, and Clarence W. de Silva. Intelligent 3-D sensing in automated manufacturing processes. In *Proceedings IEEEASME International Conference on Advanced Intelligent Mechatronics*.

[52] J.Y Kim and H.S. Cho. A vision based error-corrective algorithm for flexible parts assembly. In *Proceedings of the IEEE International Symposium on Assembly and Task Planning*.

[53] Lee S.J, K. Kim, D.-H. Kim, J.-O. Park, and G.T. Park. Recognizing and tracking of 3-D shaped micro parts using multiple visions for micromanipulation. In *IEEE International Symposium on Micromechatronics and Human Science*.

[54] H. Detter and M.J. Radjenovic. Recognition of thin, flat microelectromechanical structures for automation of the assembly process. *Journal of intelligent Manufacturing*, 8(3):191–201, June 1997.

[55] T. Baidyk and E. Kussul. Application of neural classifier for flat image recognition in the process of microdevice assembly. In *Procceding of IEEE International Join Conference on Neural Networks*, volume 1, pages 160–164, Honolulu, Hawaii, USA, May 12-17 2002. doi:10.1109/IJCNN.2002.1005462.

[56] T. Baidyk. Application of flat image recognition technique for automation of micro device production. In *Proceedings of the International Conference on Advanced Intelligent Mechatronics AIM01*, pages 488–494, Italy, July 2001.

[57] D.A. Mitzias and B.G. Mertzios. A neural multiclassifier system for object recognition in robotic vision applications. *Measurement, Elsevier*, 36:315–330, 2004.

[58] Antonio Torralba, Kevin P. Murphy, Willian T. Freeman, and Mark A. Rubin. Context-based vision system for place and object recognition. In *Proceeding of the ninth IEEE International Conference on Computer Vision*, 2003.

[59] Sun-Ho Lee, Hyun-Ki Hong, and Jong-Soo Choi. Assembly part recognition using part-based super-quadric model. In *IEEE TENCON*, pages 479–482, 1999.

[60] Markus Ehrenmann, Despina Ambela, Peter Steinhaus, and Rudiger Dillmann. A comparison of four fast vision based object recognition methods for programing by demonstration applications. In *Proceedings of the IEEE International Conference on Robotics & Automation*, pages 1862–1867, San Francisco, CA. USA, April 2000.

[61] Yasushi Sumi, Mikko Sallinen, Matti Sirviö, and Jukka Väinölä. Recognition of large work objects in difficult industrial environments. In *Proceedings of the International Conference on Systems, Man and Cybernetics*, pages 5285–5289, Hague, Netherlands, 2004. IEEE SMC.

[62] M. Hagedoom. *Patter matching using similarity measures*. PhD thesis, Utrecht University, Netherlands, 2000.

[63] E. Kefalea, O. Rehse, and v.d. Malsburg. Object classification based on contours with elastic graph matching. In *Proccedings of the 3rd International Workshop on Visual Form*, Singapure, 1997.

[64] D.H. Ballard. Generalizing the hough transformation to detect arbitrary shapes. *Pattern Recognition*, 13:111–122, 1981.

[65] Kiyomi Nakamura, Hironobu Takano, and Tsukata Sakamoto. Real-time face shape and position recognition by a two-D spreading associative neural network. *IEEE*, pages 449–454, 2004.

[66] R.C. Gonzalez and P. Wintz. Addison-Wesley, second edition, 1987.

[67] Yasushi Sumi, Yutaka Ishiyama, and Fumiaka Tomita. Robot-vision architecture for real-time 6-DOF object localization. *Computer Vision and Image Understanding*, 2007. Article in press. doi:10.1016/j.cviu.2006.11.003.

[68] D. Rachkovskij and E. Kussul. Binding and normalization of binary sparse distributed representations by context-depending thinning. *Neural Computation*, 13:411–452, 2001.

[69] Ernst Kussul. Micromechanics and perspectives of neurocomputing. neuron-like networks and neuro-computers. Technical report, Institute of Cybernetics, Kiev, Ukraine, 1993.

[70] R. Herbrich. *Learning Kernel Classifiers: Theory and Algorithms*. MIT Press, 2001.

[71] K.S. Fu. *Syntactic pattern recognition and application*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[72] Maria Petrou and Panagiota Bosdogianni. *Image Processing: The Fundamentals*. John Wiley & Sons, Inc., New York, NY, USA, 1999.

[73] S. Haykin. *Neural networks: A comprehensive foundation*. Prentice Hall, second edition, 1999.

[74] R.J. Schalkoff. *Pattern recognition: Statistical, structural and neural approaches*. John Wiley and Sons, New York, 1992.

[75] Ernst M. Kussul, Tatiana Baidyk, Donald C. Wunsch II, Oleksandr Makeyev, and Anabel Martín. Permutative coding technique for image recognition systems. *IEEE Transactions on neural networks*, 17(6):1566–79, november 2006.

[76] Ernst M. Kussul, Tatiana Baidyk, Donald Wunsch, Oleksandr Makeyev, and Anabel Martín. Image recognition systems based on random local descriptors. pages 4722–27, Vancouver, BC, Canada, July 16-21 2006. International Joint Conference on Neural Networks.

[77] Ernst M. Kussul, Tatiana Baidyk, and Donald C. Wunsch II. Image recognition systems with permutative coding. In *International Joint Conference on Neural Networks*, pages 1788–93, Montreal, Canada, July 31 2005.

[78] E. Kussul, T. Baidyk, L. Kasatkina, and V. Lukovich. Rosenblatt perceptrons for handwritten digit recognition. pages 1516–1520, Washington. USA, July 15-19 2001.

[79] E. Kussul and T. Baidyk. Neural random threshold classifier in OCR application. pages 154–157, Kiev, Ukraine, 1994.

[80] T. Wiesel D. Hubel. Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat. *Journal on Neurophysiology*, 28(2):229–289, 1965.

[81] Yann LeCun and Corinna Cortes. The MNIST database, 1998. Available in the web at `http://yann.lecun.com/exdb/mnist`.

[82] L. Bottou, C. Cortes, J. Denker, H. Drucker, L. Guyon, L. Jackel, Y. LeCun, U. Muller, E. Sackinger, P. Simard, and V. Vapnik. Comparison of classifier methods: a case study in handwritten digit recognition. In *Proccedings of the 12th IAPR International Conference on Pattern Recognition*, volume 2, pages 77–82. Proc. 12th IAPR Int. Conf. Pattern Recognition, 1994.

[83] P.Y. Simard, D. Steinkraus, and J.C. Platt. Best practices for convolution neural networks applied to visual document analysis. In *International Conference on Document Analysis and Recognition (ICDAR). IEEE Computer Society.*

[84] AT&T Laboratories Cambridge. The database of faces from the olivetti research laboratory.

[85] Ernst M. Kussul and Tatiana Baidyk. Permutative coding technique for handwritten digit recognition. In *International Joint Conference on Neural Networks*, volume 3, pages 2163–2168, Portland, Oregon, USA, July 20-24 2003.

[86] T. Baidyk and E. Kussul. Neural network based vision system for micro workpieces manufacturing. *WSEAS Transactions on Systems*, 3(2):483–488, April 2004 2004.

[87] E.Kussul., T. Baidyk, V. Lukaviteh, and D. Rachkovskij. Adaptive high performance classifier based on random threshold neurons. *Cybernerics and Systems'94*, pages 1687–1695, 1994. in R. Trappl (Ed.) Singapore: World Scientific Publishing Co. Pte. Ltd.

[88] I. Sobel and G. Feldman. A 3x3 isotropic gradient operator for image processing. presented at a talk at the Stanford Artificial Project in 1968, unpublished but often cited, orig. in Pattern Classification and Scene Analysis, Duda, R. and Hart, P., John Wiley and Sons, 1973, pp 271-2, 1968.

[89] S Artikutsa, T. Baidyk, E Kussul, and D Rachkovskij. Texture recognition with the neurocomputer. Preprint 91-8 20, Institute of Cybernetics, Ukraine, 1991. (in Russian).

[90] Josué Enríquez-Zárate, Zaizar Betanzos-Cortés, and Gengis Kanhg Toledo-Ramíre. Sistema generador de bases de datos para imágenes: una aplicación con gambas. Ixtepec, Oaxaca. México., Marzo 2-4 2006. I Simposium de Linux de la Mixteca.