

# ADAPTACIÓN EVOLUTIVA DE UN SISTEMA VISUAL DE RECONOCIMIENTO DE OBJETOS PARA EL CAMPEONATO DE FÚTBOL ROBÓTICO ROBOCUP

Juan Cristóbal Zagal, Javier Ruiz-del-Solar

Dept. de Ing. Eléctrica, Casilla 412 -3, Universidad de Chile

Email: [jzagal@ing.uchile.cl](mailto:jzagal@ing.uchile.cl), [jruizd@ing.uchile.cl](mailto:jruizd@ing.uchile.cl)

## RESUMEN

*En este artículo se presenta un sistema de aprendizaje para robots inspirado en técnicas de computación evolucionaria. El sistema permite seleccionar y ajustar automáticamente reglas útiles para la detección visual de objetos presentes en una cancha de fútbol robótico. La representación de las reglas y el aprendizaje se efectúa empleando algoritmos genéticos. El sistema pretende potenciar las habilidades del equipo de robots cuadrúpedos UChile1, que participará próximamente en el campeonato mundial de fútbol robótico RoboCup, representando a la Universidad de Chile. Otros sistemas propuestos para la detección visual de objetos, en el contexto de RoboCup, se basan en criterios de diseño arbitrario y en heurísticas. Nuestro sistema apunta a emplear la experiencia que es posible adquirir, a través de la observación de un conjunto de imágenes reales capturadas dentro del campo de juego. En el artículo se describe brevemente nuestro equipo UChile1 y se presentan resultados del sistema propuesto.*

## 1. Introducción

### 1.1. Competencia RoboCup

El campeonato mundial de fútbol entre robots, RoboCup, es un evento internacional que reúne anualmente a una extensa comunidad de investigadores de los más variados ámbitos de la robótica, como, por ejemplo, control automático, visión computacional, sistemas inteligentes e incluso ciencias del conocimiento. El principal objetivo de RoboCup es promover la investigación y enseñanza de la robótica, entregando un desafío común consistente en lograr que un equipo de robots juegue fútbol, en forma autónoma, bajo reglas de juego y condiciones del entorno de juego, claramente establecidas. La primera versión de este campeonato mundial se realizó en julio del año 1997, en Nagoya Japón. En dicha oportunidad, el desafío del evento se centraba en tres categorías de competición, dos categorías de robots con ruedas de tamaños pequeño y mediano, y una categoría de simulación computacional de

juegos de fútbol. Actualmente, RoboCup se divide en nueve categorías correspondientes a simulación, fútbol entre robots con ruedas de tamaños pequeño y mediano, fútbol entre robots cuadrúpedos AIBO, simulación de rescate de robots, rescate real de robots, un evento para estudiantes consistente en rescatar robots o hacerles bailar y, finalmente, la recientemente introducida liga de robots humanoides, ver algunas imágenes en la figura 1. RoboCup también incluye una conferencia satélite donde se discuten, en forma colaborativa, los nuevos aportes tecnológicos y científicos. El grado de compromiso y espíritu del evento han llevado a los organizadores de RoboCup a apostar que, para el año 2050, un equipo de robots humanoides será capaz de vencer al correspondiente equipo humano campeón mundial de fútbol. Existe un campeonato paralelo de fútbol robótico organizado por la Federación de Fútbol Robótico Internacional FIRA.



(a) Categoría *tamaño pequeño*.



(b) Categoría *tamaño mediano*.



(c) Categoría *robots cuadrúpedos*.



(d) Categoría *robots humanoides*.



(e) Categoría *simulación de fútbol*.



(f) Categoría *rescate*.



(g) Categoría *simulación de rescate*.



(h) Categoría *rescate junior*.



(i) Categoría *danza*.

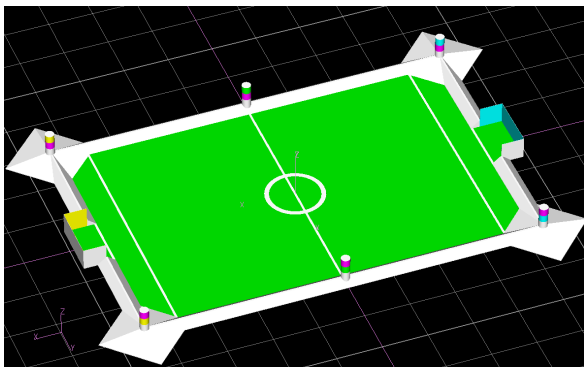
Figura 1. Fotografías de las principales categorías presentes en RoboCup. Fuente: Página oficial de RoboCup en Internet ([www.robocup.org](http://www.robocup.org)).

## 1.2. Categoría de Robots Cuadrúpedos

La categoría de robots cuadrúpedos, *Four Legged League*, fue introducida durante la tercera versión de RoboCup realizada el año 1999 en Estocolmo, Suecia. Actualmente, en esta categoría participan equipos compuestos por cuatro robots Sony

AIBO. Estos robots son muy sofisticados, están provistos de un procesador de arquitectura RISC de 64 bits, 384 MHz, y una memoria RAM de 32MB. Pueden sentir su entorno con una cámara a color de tecnología CMOS, y una gran variedad de sensores, tales como giroscopios, acelerómetros, micrófonos, sensores de infrarrojo,

tacto, etc. Las extremidades de los robots son controladas por 15 servomotores de precisión. Los robots son además capaces de comunicarse entre ellos por medio de una red inalámbrica de alta velocidad que utiliza el protocolo 802.11b. El juego se realiza sobre una cancha plana de dimensiones 3.1mts x 4.6mts, donde existen seis barras cilíndricas, o faros, marcados con combinaciones de los colores rosado, celeste y amarillo, indicando la posición de las esquinas de la cancha y los extremos de la mitad de ésta. Los arcos también están marcados, uno de color amarillo y el otro de color celeste, ver figura 2. Todo lo anterior permite a los robots estimar su propia localización dentro del campo de juego, para esto ellos deben ser capaces de detectar los elementos relevantes de la cancha, así como también a los otros robots marcados con los colores de equipos rojo y azul. Los partidos se dividen en dos tiempos de juego y un intermedio, de diez minutos cada uno. El objetivo del juego, al igual que en el fútbol, es hacer la mayor cantidad de goles sobre el arco contrario, cumpliendo con una serie de reglas a ser satisfechas por los jugadores, tales como no sobrepasar un tiempo de permanencia en el área del arquero, o un tiempo de posesión de la pelota, etc.



**Figura 2.** Ilustración de la cancha de juego, es posible apreciar los seis faros dispuestos en el contorno de la cancha.

### 1.3. Nuestro Equipo “UChile1”

El equipo de fútbol robótico, categoría cuadrúpedo, de la Universidad de Chile, UChile1 ([www.robocup.cl](http://www.robocup.cl)), es uno de los dos equipos latinoamericanos que existe en esta categoría, el

otro pertenece al Instituto Tecnológico de Monterey (<http://www.cem.itesm.mx/robocup/>).

El sistema que gobierna la acción del equipo UChile1 está compuesto por cinco módulos de procesamiento, claramente distinguibles. Estos corresponden a: (i) percepción visual o visión, (ii) localización, (iii) comunicación, (iv) estrategia y, (v) actuación motriz. El módulo de visión recibe datos de la cámara montada en la cabeza del robot AIBO y entrega, para cada objeto presente en la imagen, un descriptor de su distancia y azimut relativas a la cámara. El módulo de localización debe realizar una estimación de la posición de todos los jugadores y de la pelota dentro de la cancha. Para esto recibe información odométrica del módulo de actuación y la información generada por el módulo de percepción visual. El módulo de estrategia envía comandos de alto nivel a los jugadores según el estado del juego. El módulo de actuación recibe comandos desde el módulo de estrategia y se encarga de transformarlos en secuencias de acciones motrices, tomando control sobre los 15 motores del robot. Finalmente, el módulo de comunicación se encarga de garantizar el flujo eficiente de datos entre los robots.

### 1.4. Computación Evolucionaria

El concepto de Evolución liga a cada organismo con una cadena histórica de eventos. En términos generales, la evolución puede describirse como un proceso iterativo de dos etapas: una de variación aleatoria y una de selección; en pocas palabras, es un proceso continuo de optimización. Basado en esta idea, surgió la Computación Evolucionaria, una técnica que busca plasmar el concepto de la Evolución en diversas técnicas computacionales que permiten resolver problemas de optimización. Para resolver un problema de optimización, se sigue un enfoque muy similar al encontrado en la Naturaleza. Tal como la Evolución Biológica comienza con una población inicial de individuos, el algoritmo evolutivo parte con un conjunto inicial de soluciones. Este conjunto puede construirse de manera aleatoria, o bien, aplicando el conocimiento a priori que se tenga del problema. Estas soluciones, que denominaremos “progenitoras”, generan una “descendencia” por medio de un mecanismo de reproducción. Las

soluciones resultantes son evaluadas por su *fitness* (el valor de la función objetivo evaluado en la solución candidata en cuestión), para poder seleccionar cuáles de ellas formarán el conjunto de progenitores de la siguiente generación. Este mecanismo de selección es similar al empleado en la Naturaleza, pues “sobreviven sólo los mejores”. Este proceso se repite con las generaciones sucesivas, hasta lograr que la mejora introducida por una generación sea menor que un error mínimo aceptable. Los algoritmos evolutivos ofrecen ventajas considerables sobre las técnicas tradicionales de optimización. En primer lugar, la función objetivo o de *fitness* no tiene restricciones en cuanto a su forma, sólo debe permitir discriminar entre dos soluciones en competencia (“cuál es la mejor”). Otra ventaja corresponde a la capacidad que tiene para manejar cambios en las condiciones del problema. Habitualmente, el cambio de una de las condiciones de un problema de optimización implica realizar todos los cálculos nuevamente, desde el comienzo. Sin embargo, en los algoritmos evolutivos es posible iniciar iteraciones a partir de la última población de soluciones que se había obtenido originalmente. Una de las características más importantes del enfoque evolucionario es la rapidez con que se obtienen soluciones, pues, aun cuando no son necesariamente la “mejor solución, son lo suficientemente buenas y fueron generadas lo suficientemente rápido como para ser útiles”.

Los pasos que se deben seguir para resolver un problema mediante computación evolucionaria, son los siguientes:

- Elegir una representación de la solución: se debe definir una estructura de datos que permita codificar todas las soluciones posibles que sean deseables de evaluar, llamada código genético.
- Determinar una regla de selección o función de *fitness*: corresponde al mecanismo que permitirá discriminar entre las soluciones “buenas” y “malas”.
- Determinar el Mecanismo de Reproducción: éste corresponde a un operador variacional aleatorio (o varios).
- Inicializar la Población: si no se conoce nada acerca de cómo resolver el problema, las soluciones pueden elegirse aleatoriamente dentro del espacio de solución.

Dentro de la computación evolucionaria podemos distinguir las siguientes disciplinas: algoritmos genéticos, programación genética, estrategias evolutivas y programación evolucionaria.

### 1.5. Robótica Evolucionaria

La robótica evolucionaria [15] corresponde a una metodología emergente destinada al diseño y generación de robots en forma automática. En ella, los robots son considerados como agentes autónomos artificiales capaces de desarrollar sus habilidades o comportamientos, al interactuar estrechamente con su entorno. Dicho de otro modo, se asume que los robots pueden aprender a partir de su propia experiencia. Si bien es posible simular la interacción entre robots y entorno, la tendencia actual apunta a hacer que los robots interactúen con su medio físico real, minimizando la simulación [7] y cualquier tipo de intervención humana. Las principales herramientas de modelación y aprendizaje empleadas en robótica evolucionaria, son computación evolucionaria, redes neuronales y, en general todos los métodos de aprendizaje de máquinas. La robótica evolucionaria ha demostrado ser exitosa en la generación de controladores robóticos reactivos simples [4,5,6], y también en la adaptación de sistemas preceptuales [8,10,11,12,13,14].

### 1.6. Estructura del Artículo

Este artículo se organiza de la siguiente manera. En la sección 2 se presenta el problema a resolver. La sección 3 presenta trabajos relacionados. La sección 4 describe el módulo de visión de nuestra arquitectura de software. La sección 5 presenta la metodología empleada para aprender a detectar objetos. La sección 6 presenta los resultados obtenidos en los experimentos realizados. Y finalmente la sección 7 presenta las conclusiones de este trabajo.

## 2. Definición del Problema

Los equipos que han participado en la categoría de robots cuadrúpedos han reportado, en general, buenas estrategias para la detección de objetos dentro del campo de juego [2,3,17], objetos tales

como la pelota, faros, y los arcos. Estas estrategias usualmente se basan en la aplicación secuencial de un conjunto de reglas de reconocimiento, como por ejemplo, hacer comparaciones de tamaños y distancias entre regiones de píxeles conexos de determinados colores. Estas reglas son diseñadas en forma arbitraria y sus parámetros se ajustan manualmente hasta que son útiles para la detección invariante de objetos, i.e. detección en diferentes localizaciones, condiciones de iluminación, y posturas de los objetos en la imagen. De esta manera, luego de emplear la teoría y modelar el problema, la tarea ingenieril cae frecuentemente en un proceso de optimización del tipo prueba y error. Algunos equipos [17] han reportado incluso su incertidumbre respecto a si la aplicación de ciertas reglas es efectiva o no. Nosotros creemos que este proceso ingenieril puede ser apoyado, e incluso automatizado, por medio del empleo de un sistema de aprendizaje supervisado, en el cual, un extenso conjunto de imágenes pre-clasificadas se emplea como datos de entrenamiento para un sistema de aprendizaje. Estas imágenes reales deben contener la mayor cantidad de ejemplos interesantes a imaginar. La principal ventaja de este enfoque es que tal sistema obtendría su conocimiento sobre la base de su propia experiencia mas que ser el fruto de un diseño arbitrario. En este trabajo vamos a explorar un método para evolucionar la selección y ajuste de un conjunto de reglas de reconocimiento visual empleadas en la detección de objetos de una cancha de juego en el contexto de RoboCup. Exploraremos resultados de la detección de pelota, arcos, y faros. Estamos actualmente trabajando en la aplicación más avanzada del método propuesto, que consiste en su utilización para la detección de otros robots dentro del campo de juego. Dada la complejidad de este último problema, aún no ha sido abordado con suficiente atención por la comunidad de investigadores de RoboCup.

### 3. Trabajos Relacionados

En el trabajo desarrollado por Cliff *et al.* [4,5] se emplean algoritmos genéticos para evolucionar controladores de robots visualmente guiados, los controladores se implementan en forma de redes neuronales. En el trabajo se emplean modelos de computación gráfica para simular la visión de los robots. El modelo empleado considera la adición

de cierta cantidad de ruido que impide al sistema volverse completamente determinístico. Sin embargo, este sistema tiene un elevado costo computacional y una pobre resolución de imagen. El empleo de computación evolucionaria en problemas de visión computacional ha sido ampliamente explorado, por ejemplo, Köppen *et al.* [11] propone un sistema para la generación automática de filtros para texturas, empleando tanto algoritmos genéticos como programación genética. Una de las primeras aplicaciones de computación evolucionaria aplicada al reconocimiento de objetos en imágenes corresponde al caso de reconocimiento de caracteres. En el trabajo de Koza [9] se presenta un experimento en el empleo de programación genética para la clasificación de solo cuatro caracteres contenidos en pequeños mapas de bits, este sistema se basa en el empleo de un método, de alto costo computacional, para marcar puntos de atención dentro de la imagen. Koza también propone un sistema que emplea funciones automáticamente definidas (ADF) para encontrar soluciones de clasificación [10], sin embargo, requiere poblaciones de gran tamaño (8000 individuos). Andre [1] emplea ambos, programación genética y algoritmos genéticos, en forma simultánea. Primero un algoritmo genético determina marcos de características y luego un sistema de programación genética se emplea para la clasificación de mapas de caracteres. En el trabajo de Teller y Veloso [16] se emplea programación genética para el sistema propuesto *Parallel Algorithm Discovery and Orchestration* (PADO). Este sistema efectúa reconocimiento de objetos en imágenes reales de escala de grises. La programación genética se emplea para inducir programas que operan sobre los píxeles de la imagen y retornan valores de confianza respecto a si cierta imagen corresponde o no, a la clase que se pretende reconocer.

### 4. Nuestro Módulo de Visión

Tal como se indicó en el punto 1.3, la arquitectura de software empleada por el equipo UChile1 se divide en módulos claramente definidos por sus funcionalidades. Uno de ellos, el módulo de visión, esta a cargo de detectar los objetos relevantes presentes en las imágenes capturadas

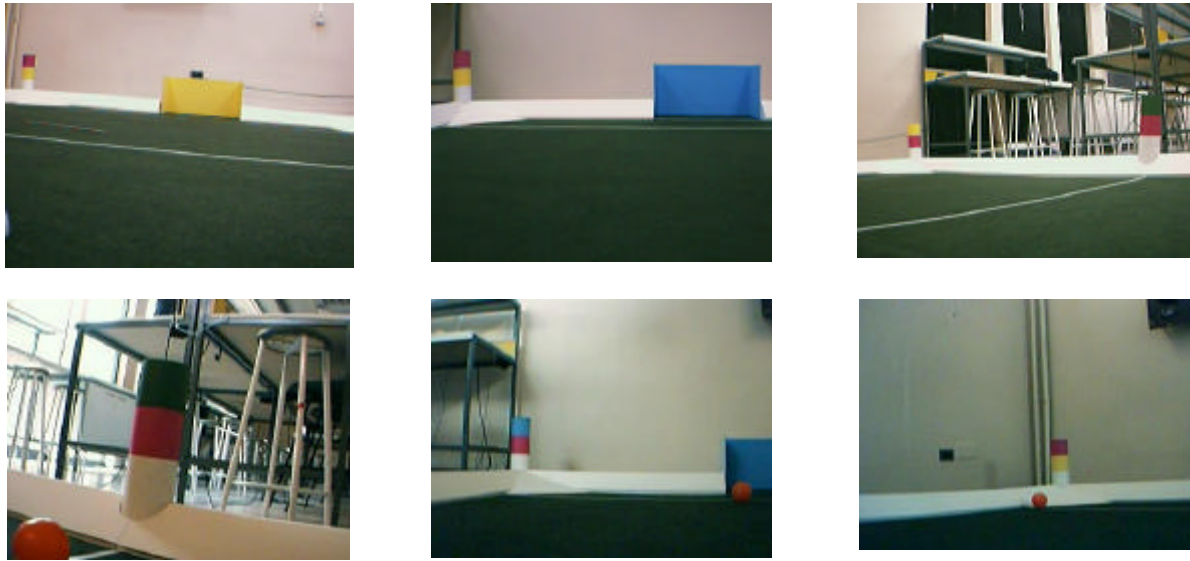
por las cámaras de los robots. Este módulo, en particular, está inspirado en los enfoques propuestos por equipos de larga experiencia en RoboCup, tales como los equipos UNSW [3] y CMPack [17]. Nuestro módulo de visión está compuesto por cuatro sub-módulos de procesamiento, que son: (i) segmentación de colores, (ii) codificación *run-length*, (iii) etiquetado de regiones conexas y, finalmente, (iv) reconocimiento de objetos. El proceso de segmentación de colores se lleva a cabo mediante el empleo de una tabla de búsqueda rápida (*lookup table*) de 64 niveles en cada dimensión del espacio de color YUV. La tabla se genera al tomar una gran cantidad de muestras de colores (alrededor de 5000) desde imágenes capturadas en el interior del campo de juego, por cada muestra se llena un elemento de la tabla con el identificador del color asociado. Una vez que todas las muestras se han recolectado, un filtro de mediana se aplica sobre los identificadores de la tabla con el objetivo de despejar la interfaz entre agrupaciones de colores diferentes, así como también para asignar un identificador a elementos de la tabla que no fueron asignados a momento de recolección de los datos. Este proceso es particularmente útil, por ejemplo, para despejar ambigüedades entre agrupaciones del color rojo y naranja. Un detalle importante de este proceso es que no solo entrenamos los siete colores relevantes en la cancha, sino también, una clase representativa de todos los colores que no son relevantes al juego. El proceso de etiquetado tiene por resultado describir regiones conexas de cierto color. Cada región puede caracterizarse por un conjunto de descriptores, tales como el tamaño o cantidad de píxeles asociados, un índice de color que en este caso toma los valores {0,1,2,3,4,5,6}, las coordenadas que describen los vértices del menor rectángulo que encierra a la región (*bounding box*), y las coordenadas de su centro de masa. La tarea del sub-módulo de reconocimiento es identificar regiones dentro de la imagen que estén relacionadas a objetos relevantes dentro del campo de juego. El reconocimiento de estas regiones se realiza al evaluar la respuesta frente a un conjunto de reglas. Dichas reglas operan sobre todas las regiones de la imagen o sobre pares o, en general, combinaciones de estas. Por ejemplo, al detectar la pelota, se verifica que la región asociada sea del

color rosado de la pelota, y en caso de no ser así la región es rechazada como candidato.

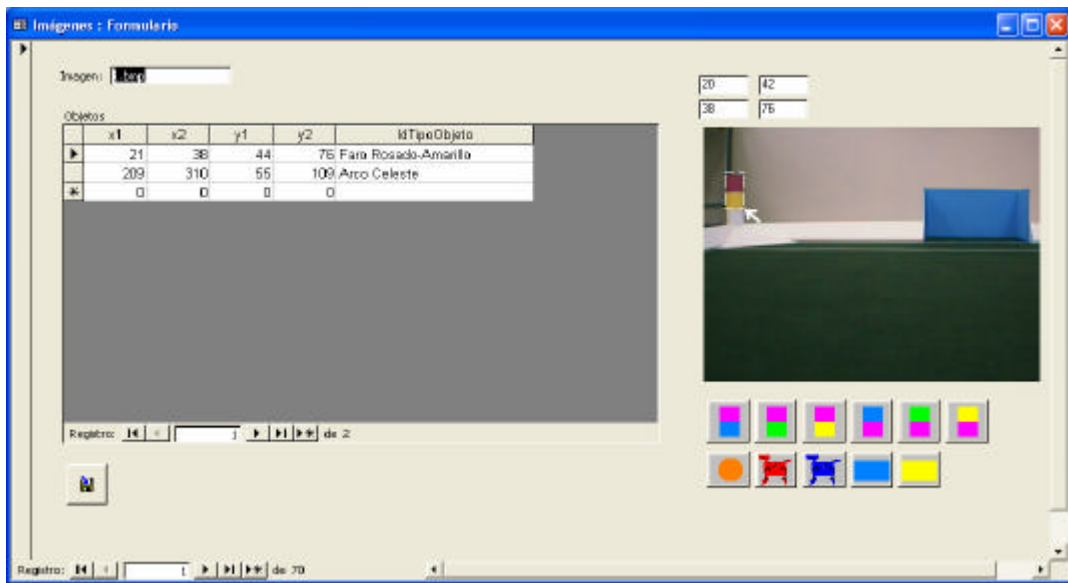
## 5. Aprendiendo a Detectar Objetos

### 5.1. Metodología

El primer paso necesario para realizar la evolución del sistema de detección visual de objetos, consiste en extraer, desde un conjunto de imágenes reales, descriptores de regiones de referencia, es decir, de aquellas regiones de la imagen efectivamente asociadas a objetos de interés. Esta etapa se realiza con la ayuda de un usuario experto. Luego se definen las regiones candidato como las regiones individualizadas por el proceso de visión, o como regiones derivadas de combinaciones de estas, ver figura 5. Entonces, bajo un proceso de aprendizaje supervisado de reglas, las regiones candidato, se comparan con las correspondientes regiones de referencia en cada imagen, y el grado de correspondencia general sirve como función de *fitness* del sistema, que luego es empleada por el algoritmo genético que aprende las reglas de detección del sistema. Claramente, la bondad del método está relacionada con el grado de correspondencia entre las regiones candidato y las regiones de referencia seleccionadas por el usuario experto. Para la extracción supervisada de referencias se emplea, en este trabajo, un conjunto de 180 imágenes reales. Estas imágenes, capturadas desde el interior de la cancha, contienen objetos tales como la pelota, faros, y arcos, así como también objetos irrelevantes, presentes en los alrededores de la cancha de juego. Las imágenes consideran un amplio rango de perspectivas, rotaciones, y posiciones no canónicas de los objetos, así como también variaciones en las condiciones de iluminación. La figura 3 muestra ejemplos de estas imágenes. Con el fin de generar una base de datos de los identificadores de regiones de referencia, se ha desarrollado un software que permite a un usuario experto, definir las regiones de la imagen que están relacionadas con los objetos relevantes, esta definición se realiza en términos de los rectángulos que encierran a las regiones (*bounding box*). Una vez que una región ha sido seleccionada en la imagen, esta es relacionada con su correspondiente identificador al presionar el botón correspondiente



**Figura 3.** Ejemplos de imágenes recolectadas desde el interior del campo de juego para el entrenamiento del sistema. Cada imagen es sujeta a inspección y evaluación por parte de un usuario experto, en la etapa de asignación de regiones de referencia, y por parte del sistema visual bajo adaptación.

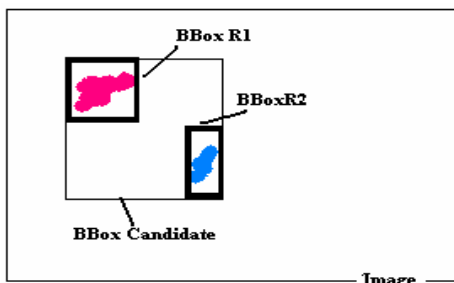


**Figura 4.** Una ilustración del software desarrollado con el objeto de permitir al usuario experto, definir las regiones de interés en cada imagen (derecha) y luego almacenar las respectivas coordenadas e identificadores en una base de datos (izquierda). Esta operación se realiza al presionar el botón de clase correspondiente (derecha, abajo).

al objeto de interés. La figura 4 presenta un ejemplo de este software en operación. Durante el proceso de aprendizaje, el algoritmo genético evoluciona una población de las reglas de detección de objetos. Estas reglas operan sobre los descriptores de las regiones automáticamente detectadas por el proceso de visión. En el caso que

una región, o una combinación de regiones, sea considerada como un objeto, su grado de correspondencia con la referencia es calculado por medio de una función que mide la calidad del ajuste. Luego, el nivel de correspondencia global, definido como la suma de los niveles de correspondencia de las regiones detectadas en

todas las imágenes, es empleado como función objetivo de cada individuo generado por el algoritmo genético.



**Figura 5.** La figura indica cómo una región candidato de forma rectangular puede derivarse a partir de un par de regiones conexas extraídas por el proceso de visión, esto es particularmente útil en el caso de la detección de faros.

## 5.2. Función Objetivo o de *Fitness*

No resulta trivial, en este caso, definir una buena función objetivo. Esta medida debe asumir su valor máximo cuando existe un traslape perfecto entre la referencia y la región candidato, sin embargo no es necesariamente claro cómo manejar aquellas situaciones en que el traslape es parcial. En el trabajo de Köppen *et al.* [11] se propone una función de calidad de ajuste que se ajusta bien a nuestro problema. Esta consiste en medir el área A de una región de referencia que no posee traslape con la región candidato, el área B de la región traslapada, y el área C de la región candidato que no traslapa la referencia, ver figura 6. De lo anterior se derivan las tres medidas siguientes:

$r_1 = B / (A+B)$ , la cantidad relativa de píxeles correctamente traslapados dentro de la referencia,  
 $r_2 = 1 - (C / (Q - A - B))$ , la cantidad relativa de píxeles vacíos dentro de la imagen, donde  $Q$  corresponde al número total de píxeles en la imagen, y  
 $r_3 = B / (B+C)$ , la cantidad relativa de píxeles correctamente traslapados dentro del candidato.

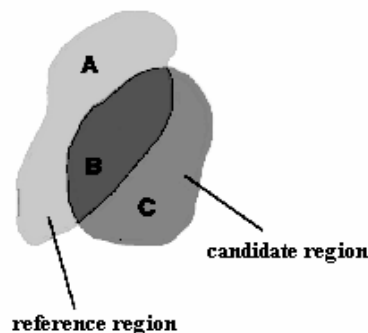
Empleando estas tres medidas, un grado de correspondencia del ajuste se define como:

$$CD = 0.1r_1 + 0.5r_2 + 0.4r_3 \quad (1)$$

Y la función objetivo de cada individuo se calcula como la suma de los grados de correspondencia obtenidos en todas las regiones detectadas por el proceso visual:

$$fitness = \sum_i CD_i \quad (2)$$

Hay que notar que en vez de incrementar simultáneamente las medidas  $r_i$ , se ha escogido el empleo de una suma ponderada de ellas. Consecuencia de ello, es que la búsqueda genética se concentra primero en incrementar la medida de mayor peso, en este caso  $r_2$ .



**Figura 6.** Un ejemplo de traslape parcial entre una región de referencia y una región candidato. Las regiones A, B, y C se definen tal como se muestra en la figura.

## 5.3. Representación Genética de Reglas

Para cada región candidato detectada por el proceso de visión, un conjunto de  $N$  reglas binarias  $R_i = f(\{p_{ij}\}, \{Region\_Descriptors_k\})$ , es evaluado. Cada regla es descrita por un conjunto de  $M$  parámetros  $p_{ij}$ . Las reglas tienen como argumento al conjunto de descriptores de las regiones candidato. En forma genérica las reglas binarias tienen la siguiente estructura

$$R_i = \begin{cases} 1 & \text{si } p_{i1} \geq COND \geq p_{i2} \\ 0 & \text{si } no \end{cases} \quad (3)$$

donde  $COND$  puede corresponder a un valor, como por ejemplo el tamaño de una región, el resultado de un cociente entre valores, y en general al resultado de operaciones lógicas o aritméticas entre los descriptores de las regiones. Cada región candidato recibe un puntaje

**Tabla 1.** Las seis reglas para el experimento de detección de pelota. Se presenta la estructura de cada regla y el rango de variación de sus parámetros. Se emplean los descriptores de regiones **width\_reg**; ancho de la región, **height\_reg** alto de la región, **area\_reg**; tamaño de la región, **w<sub>bb</sub>**; ancho del *bounding box*, **h<sub>bb</sub>**; alto del *bounding box* y **color\_reg** índice del color de la región.

Reglas para Detección de Pelota		
Regla	Condición de Activación	Rango de los Parámetros
R <sub>1</sub>	width_reg > P <sub>11</sub>	integer [0,image_width]
R <sub>2</sub>	height_reg > P <sub>21</sub>	integer [0,image_height]
R <sub>3</sub>	area_reg > P <sub>31</sub>	integer [0,image_size]
R <sub>4</sub>	min(w <sub>bb</sub> /h <sub>bb</sub> ,h <sub>bb</sub> /w <sub>bb</sub> ) > P <sub>41</sub>	double [0,1]
R <sub>5</sub>	P <sub>51</sub> < area_reg/area_bbox < P <sub>52</sub>	double [0,1]
R <sub>6</sub>	color_reg = P <sub>61</sub>	integer[1,num_colors=7]

**Tabla 2.** Las seis reglas empleadas para el experimento de detección de los faros. Se presenta la estructura de cada regla y el rango de variación sus parámetros. Se emplean los descriptores **distX**; distancia entre regiones en el eje x, **distY**; distancia entre regiones en el eje y, **Sreg**; tamaño de la región y **color\_reg**; índice del color de la región.

Reglas para Detección de Faros		
Regla	Condición de Activación	Rango de los Parámetros
R <sub>1</sub>	distX(reg1,reg2) < P <sub>11</sub>	integer [0,image_width]
R <sub>2</sub>	distY(reg1,reg2) < P <sub>21</sub>	integer [0,image_height]
R <sub>3</sub>	min(Sreg1/Sreg2,Sreg2/Sreg1) > P <sub>31</sub>	double [0,1]
R <sub>4</sub>	1/2 (Sreg1+Sreg2)/sqrt(dist(reg1,reg2)) > P <sub>41</sub>	double [0,1]
R <sub>5</sub>	color reg1 = P <sub>51</sub> OR color reg2 = P <sub>51</sub>	integer[1,num_colors=7]
R <sub>6</sub>	color_reg2 = P <sub>61</sub> OR color reg2 = P <sub>61</sub>	integer[1,num_colors=7]

correspondiente a la suma ponderada del resultado de cada regla. La región que recibe el mayor puntaje será considerada como el objeto bajo búsqueda solo en el caso que este exceda cierto valor umbral. Este puntaje se calcula de la siguiente manera:

$$Score = \sum_i w_i R_i \geq T \quad (4)$$

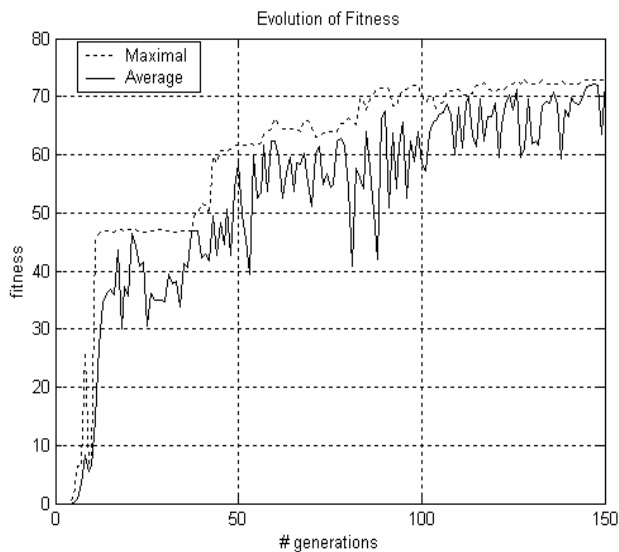
En este caso, los pesos  $w_i \in [0,1]$ , el umbral  $T \in [0, N]$  y todos los  $M$  parámetros  $p_{ij} \in [0,1]$  se representan por palabras de 16 bit. Por tanto el cromosoma que codifica la regla posee  $16x(N+M+1)$  bits. En ciertos casos, tal como será indicado, los parámetros se ajustan o se discretizan a un conjunto reducido de valores. Estos cromosomas serán evolucionados con un algoritmo genético. Este algoritmo emplea selección proporcional al *fitness*, con escalamiento lineal de *fitness*, esquema no elitista, combinación de dos puntos con probabilidad de combinación de

$P_c=0.75$  y mutación con probabilidad de mutación de  $P_m=0.015$  por bit. El tamaño de la población es de 8 individuos que evolucionan a lo largo de 100 a 150 generaciones.

## 6 Resultados

### 6.1. Detección de Pelota

Hemos escogido un conjunto de seis reglas para el experimento de detección de pelota. La estructura de estas reglas así como también el rango de sus parámetros, se presentan en la tabla 1. De los resultados expuestos en la figura 7, es posible apreciar primero que la prueba elemental de color, implícita en la regla R6, se satisface, por lo tanto, el sistema aprende a escoger el color correcto de la pelota. El parámetro P61 se discretiza como el número entero 7 que corresponde precisamente al índice del color naranja. Esta regla es también reconocida como la más importante puesto que



Individuo mejor evolucionado		
Pr.	Valor	Interpretación
w1	0.952347	Ranking 2
w2	0.45491	Ranking 5
w3	0.07095	Ranking 6
w4	0.805679	Ranking 3
w5	0.459213	Ranking 4
w6	0.961151	Ranking 1
P11	0.00372314	Ancho mínimo 2 píxeles.
P21	0.253232	1/4 del ancho de la imagen
P31	0.308493	1/3 del tamaño de la imagen
P41	0.575424	Relativamente cuadrado
P51	0.0270081	Acepta siluetas
P52	0.552963	Rechaza falsos candidatos
P61	0.98053	Índice del color naranja
T	0.166499	Equivalente a 0.99

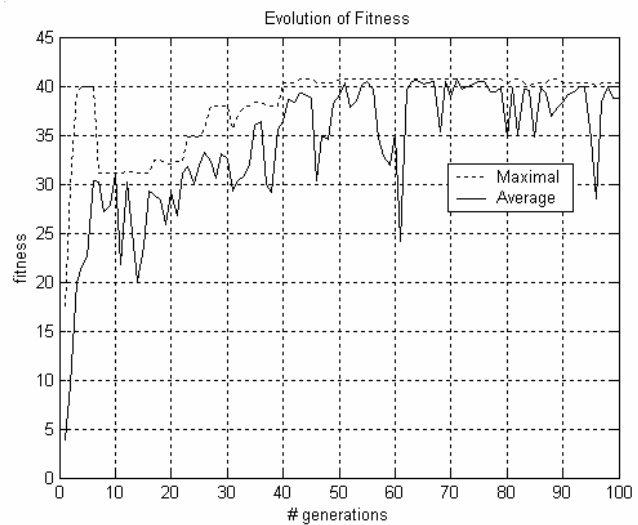
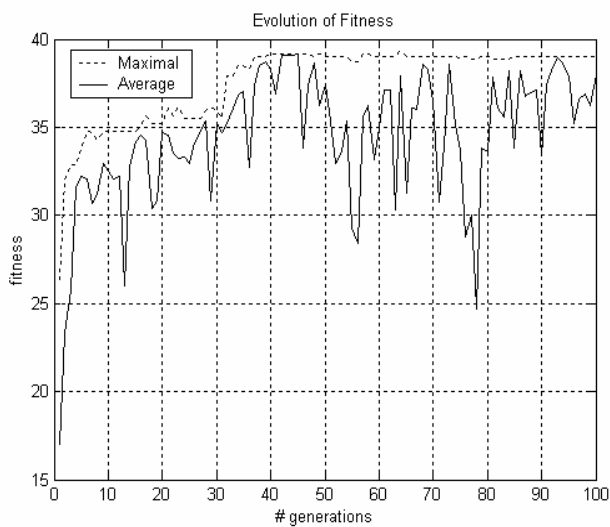
**Figura 7.** Evolución resultante del *fitness* (izquierda), y los pesos, parámetros y umbral correspondientes al mejor individuo resultante de la evolución (derecha), para el experimento de detección de la pelota.

posee el mayor peso. La segunda regla en importancia es R1, el bajo valor de su parámetro relacionado P11 indica que la región candidata debe tener un ancho mínimo de dos píxeles. La tercera regla en importancia, R4, indica que el rectángulo que encierra a la región debe ser relativamente cuadrado, con un cociente mínimo entre ancho y alto, o vice versa, de  $P41 = 0.5754$ . La cuarta regla, R5, indica que el cociente entre el tamaño de la región y el tamaño del rectángulo que la encierra debe estar entre  $P51=0.027$  y  $P52=0.553$ , lo cual es bastante lógico para nuestra implementación. El límite inferior sirve para aquellos casos en que la pelota se presenta como una silueta de píxeles correctamente segmentados del color naranja, i.e. regiones pequeñas en tamaño pero encerradas por grandes rectángulos. El límite superior sirve para rechazar las regiones de color naranja que están muy próximas a tener una forma cuadrada. La quinta regla, R2, indica que el alto de la región debe ser de al menos un cuarto del alto de la imagen. La última regla, R3, establece que el tamaño de la región debe ser por lo menos un tercio del tamaño de la imagen. Sin embargo, hay que notar que el peso de esta regla es bastante pequeño y por tanto no es una regla importante. El umbral, escalado por el número total de 6 reglas, corresponde a  $T = 0.99$ , el cual puede compararse al puntaje teórico máximo de

3.69, correspondiente a la suma de los pesos. Esto significa que el umbral debe fijarse al 26% del puntaje máximo.

## 6.2. Detección de Arcos

En general, en la literatura de RoboCup, se han reportado menos reglas para la detección de los arcos que en el caso de la pelota. Nosotros hemos escogido emplear el mismo grupo de reglas utilizadas para el experimento de la pelota, ver tabla 1. Este grupo parece ser un buen *súper conjunto* de reglas relevantes para nuestro análisis. En la figura 8 se muestran los resultados para la detección de los arcos celeste y amarillo. Nuevamente tenemos que la regla trivial de color obtiene los parámetros correctos, correspondientes a los índices del celeste y del amarillo para cada experimento. Esta regla obtiene el cuarto lugar en importancia en ambos experimentos. La regla más importante es, en ambos casos, R1 estableciendo límites inferiores para el ancho de las regiones candidato en un 9% y un 5% del ancho de la imagen, respectivamente. El segundo lugar es para R2 en el experimento del arco celeste y R3 en el experimento del arco amarillo. En ambos experimentos se establece que el alto mínimo de la región debe ser de un 6% del alto de la imagen. En forma similar, en ambos experimentos, R3



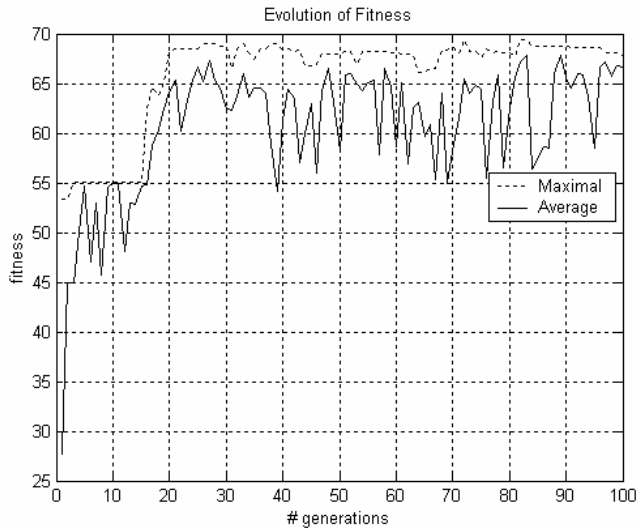
Individuo Mejor Evolucionado Arco Celeste		
Pr.	Valor	Interpretación
W1	0.802551	Ranking 1
W2	0.784454	Ranking 2
W3	0.348038	Ranking 5
W4	0.366501	Ranking 6
W5	0.606674	Ranking 3
W6	0.661896	Ranking 4
P11	0.0901642	Ancho mínimo 9%
P21	0.0627899	Alto mínimo 6%
P31	0.647183	Tamaño mínimo 6%
P41	0.65889	Grado de Similitud a un cuadrado
P51	0.400696	Límite inferior de área
P52	0.864853	Límite superior de área
P61	0.779999	Índice color celeste
T	0.0565796	Equivalente a 0.34

Individuo Mejor Evolucionado Arco Amarillo		
Pr.	Valor	Interpretación
W1	0.999039	Ranking 1
W2	0.524292	Ranking 5
W3	0.859741	Ranking 2
W4	0.0393372	Ranking 6
W5	0.83844	Ranking 3
W6	0.743652	Ranking 4
P11	0.0493469	Ancho mínimo 5%
P21	0.589996	Alto mínimo 6%
P31	0.503348	Tamaño mínimo 5%
P41	0.0301056	Grado de similitud a un cuadrado
P51	0.173755	Límite inferior de área
P52	0.9658875	Límite superior de área
P61	0.134277	Índice del color amarillo
T	0.0552063	Equivalente a 0.33

**Figura 8.** Resultados del experimento de detección del arco celeste (izquierda), y de la detección del arco amarillo (derecha). La evolución del fitness (arriba), y los correspondientes pesos, parámetros, y umbrales de los mejores individuos resultantes de la evolución (abajo).

establece que el tamaño mínimo de la región debe corresponder respectivamente a un 6% y a un 5% del tamaño total de la imagen. El tercer lugar es para R5 en ambos experimentos, esta regla establece límites para el cociente entre el tamaño de la región y el tamaño del rectángulo que la encierra. Los parámetros resultan ser similares en ambos experimentos. El quinto lugar es para R3 en el experimento del arco celeste y para R2 en el experimento del arco amarillo. La regla de menor relevancia, tal como es indicado en ambos experimentos, corresponde a R4, la cual asume

valores muy diferentes en sus parámetros. Una interpretación de esto es que la búsqueda genética se concentra en optimizar los parámetros de las reglas relevantes, dejando parámetros relativamente arbitrarios en el caso de las reglas irrelevantes. Los valores umbrales resultan ser bastante similares en ambos experimentos; sus correspondientes valores ajustados son 0.33 y 0.34. La suma de los pesos corresponde a 3.569 para el experimento del arco celeste, y a 4 para el experimento del arco amarillo. Por lo tanto el



Individuo Mejor Evolucionado		
Pr.	Valor	Interpretación
W1	0.549803	Ranking 3
W2	0.40321	Ranking 5
W3	0.101231	Ranking 6
W4	0.538501	Ranking 4
W5	0.633870	Ranking 2
W6	0.875532	Ranking 1
P11	0.032041	Distancia mínima en X 3%
P21	0.083218	Distancia mínima en Y 8%
P31	0.0031345	Cuociente de tamaños
P41	0.832451	Distancias relativas
P51	0.896420	Color es rosado
P61	0.7559361	Color es celeste
T	0.138913	Equivalente a 0.83

**Fig. 9.** Evolución del fitness (izquierda), y los resultantes pesos, parámetros, y umbrales para el individuo mejor evolucionado (derecha), para el experimento de detección de faros.

umbral queda establecido a un 10% del puntaje total.

### 6.3. Detección de Faros

En este experimento, las regiones candidato se derivan de pares de regiones detectadas por el proceso de visión, ver figura 5. Las reglas, por tanto, deben evaluarse sobre todos los pares de regiones que es posible extraer de la imagen. Claramente las reglas poseen como entrada a dos regiones. Aquella región candidato que obtiene puntaje máximo se selecciona si satisface la ecuación 3. El grado de ajuste se obtiene según la ecuación 1. En este experimento hemos seleccionado un conjunto de seis reglas expuestas en la tabla 2. Vamos a explorar el caso particular de la detección de la categoría faro conformada por los colores rosado y celeste, y celeste rosado, sin hacer distinción entre el orden vertical de los colores. Es posible apreciar de los resultados expuestos en la figura 8 que las reglas para verificar el color R5 y R6 son consideradas como las más importantes, sus correspondientes parámetros se ajustan exactamente a los colores esperados de rosado y celeste. La tercera regla en

importancia corresponde a R1, la cual corresponde a una verificación de la distancia horizontal entre las regiones, el correspondiente parámetro asociado establece un umbral de un 3% del ancho de la imagen. La cuarta regla en importancia es R4, la cual verifica que las distancias entre las regiones con independencia de la escala de los objetos, esta regla fue propuesta en [14]. La quinta regla es R2 estableciendo una distancia vertical mínima entre las regiones de un 8% del alto de la imagen. Finalmente R3 es considerada como la regla de menor importancia. El puntaje teórico máximo es en este caso 3.1, el cual significa que el umbral fue establecido a un 27% del puntaje máximo. Respecto a la detección de los otros faros, se considera que el problema es equivalente.

## 7 Conclusiones

En este artículo se ha presentado un método para automatizar y reforzar la selección y ajuste de reglas empleadas para detectar visualmente objetos en el contexto del campeonato mundial de fútbol robótico RoboCup. El sistema muestra ser consistente con los datos de entrenamiento, y permite identificar las reglas más importantes en

un conjunto de reglas dado, así como también, la extracción de interesantes parámetros asociados a estas reglas. En particular se analizaron los casos de la detección de la pelota, arcos y faros de la cancha. Pretendemos extender esta investigación hacia la detección de otros jugadores dentro del campo de juego. El sistema presentado se basa en un método de visión basado en la caracterización de regiones conexas tipo mancha (*blob-based vision*). Una metodología similar podría emplearse para evolucionar otro tipo de estrategias de visión, tales como la visión basada en grillas o bordes.

### Agradecimientos

Esta investigación ha sido apoyada por el Departamento de Ingeniería Eléctrica de la Universidad de Chile.

### Referencias

1. Andre, D.: Automatically Defined Features – the Simultaneous Evolution of 2-Dimensional Feature Detectors and an Algorithm for Using Them. In: Kinnear, K. (eds.): *Advances in Genetic Programming*. MIT Press, Cambridge, Massachusetts (1994) 477-494.
2. Chalup, S., Creek, N., Freeston, L., Lovell, N., Marshall, J., Middleton, R., Murch, C., Quinlan, M., Shanks, G.: When NUbots Attack! The 2002 NUbots Team Report. Newcastle Robotics Laboratory. The University of New Castle. NSW 2308, Australia (2002) <http://robots.newcastle.edu.au/NUbotFinalReport.pdf>
3. Chen, S., Siu, M., Vogelgesang, T., Fai Yik, T., Hengst, B., Bao Pham, S., Sammut, C.: The UNSW RoboCup 2001 Sony Legged Robot League Team. In: Birk, A., Coradeschi, S., Tadokoro, S. (eds.): *RoboCup 2001: Robot Soccer World Cup V*. Springer-Verlag Berlin, Germany (2002) 39-48.
4. Cliff, D., Husbands, P., Harvey, I.: Evolving Visually Guided Robots. In: Meyer, J. A., Roitblat, H., Wilson, S. (eds.): *Proceedings of the Second International Conference on Simulation of Adaptive Behaviour (SAB92)*. MIT Press Bradford Books, Cambridge, MA (1993) 374-383.
5. Cliff, D., Harvey, I., Husbands, P.: General Visual Robot Controller Networks via Artificial Evolution. In: Casement, D. (eds.): *Proceedings of the Society of Photo-Optical Instrumentation Engineers Conference (SPIE-93), Session on Intelligent Robots and Computer Vision XII: Algorithms and Techniques*, Boston, MA (1993)
6. Hoffmann, F., Zagal, J.C.: Evolution of a Tactile Wall-Following Behavior in Real Time. In: Roy, R., Köppen, M., Ovaska, S., Huruhashi, T., Hoffmann, F. (eds.): *Soft Computing and Industry, Recent Applications*. Springer (2002) 747-755.
7. Jacobi, N.: Half-baked, ad-hoc and noisy: Minimal Simulations for Evolutionary Robotics. In: *Proceedings of the Fourth European Conference on Artificial Life*. MIT Press (1997)
8. Johnson, M., Maes, P., Darrell, T.: Evolving Visual Routines. In: Brooks, R., Maes, P. (eds.): *Artificial Life IV*. MIT Press (1994) 198-209.
9. Koza, J.: *Genetic Programming*. MIT Press, Cambridge, Massachusetts (1992)
10. Koza, J.: *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press, Cambridge, Massachusetts (1994)
11. Köppen, M., Teunis, M., Nickolay, N.: A Framework for the Evolutionary Generation of 2D-Lookup Based Texture Filters. In: *Proceeding of Lizuka Conference*, Lizuka, Japan (1998) 965-970
12. Martin, M. C.: Genetic Programming for Robot Vision. In: *SAB 2002, the Seventh International Conference on the Simulation of Adaptive Behaviour* (2002).
13. Martin, M.C.: Genetic Programming for Real World Robot Vision. In: *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, EPFL, Lausanne, Switzerland, September-October (2002) 67-72
14. Nguyen, T., Huang, T.: Evolvable 3d Modeling for Model-Based Object Recognition Systems, In: Kenneth, E. (eds.): *Advances In Genetic Programming*. MIT Press (1994) 459-475
15. Nolfi, S., Floreano, D.: *Evolutionary Robotics – The Biology, Intelligence, and Technology of Self-Organizing Machines*, In: *Intelligent Robotics and Automation Agents*. MIT Press (2000)
16. Teller, A., Veloso, M.: PADO: A New Learning Architecture for Object Recognition. In: Ikeuchi, K., Veloso, M. (eds.): *Symbolic Visual Learning*, Oxford University Press (1996) 81-116
17. Veloso, M., Lenser, S., Vail, D., Roth, M., Stroupe, A., Chernova, S.: *CMPack-02: CMU's Legged Robot Soccer Team*. Carnegie Mellon University Report (2002) [https://www.openr.org/page1\\_2003/code2002SDK/CMU/cmu\\_teamdesc.pdf](https://www.openr.org/page1_2003/code2002SDK/CMU/cmu_teamdesc.pdf)