Uso de algoritmos genéticos para detección de objetos en tiempo real

Jesús Martínez-Gómez[1], José A. Gámez[1], Ismael García-Varea[1], Vicente Matellán[2]
Departamento de Sistemas Informáticos[1], Dept. de Ingeniería Mecánica, Informática y Aeroespacial[2]
Universidad de Castilla-La Mancha[1], Universidad de Leon[2]
Albacete[1], Leon[2], España
{jesus_martinez, jgamez, ivarea}@dsi.uclm.es[1], vicente.matellan@unileon.es[2]

***Abstract*—El artículo presenta un sistema de Visión para la detección de objetos mediante el uso de algoritmos genéticos. La principal aportación del mismo consiste en la adaptación al tiempo real de algoritmos generalmente utilizados *offline*. Para conseguir esto se necesita una ejecución eficiente que reduzca al máximo la complejidad del algoritmo. El ámbito de aplicación del sistema es el campeonato de fútbol robótico RoboCup[1], en concreto la categoría *Standard Platform*, y debe permitir detectar y estimar distancias a los objetos relevantes dentro de un estadio de fútbol. Se realizaron diferentes experimentos dentro de un campo oficial RoboCup.**

***Index Terms*—Visión por computador, algoritmos genéticos, robótica, procesamiento de imágenes, Nao.**

I. Introduction

El procesamiento de imágenes es uno de los puntos clave dentro de la robótica móvil actual, ya que las cámaras de visión son el principal dispositivo, a partir del cual obtener información del entorno.

Este procesamiento debe realizarse en tiempo real, con una capacidad de procesamiento limitada y manejando imágenes ruidosas, o de baja calidad. Los sistemas de visión desarrollados para robótica móvil, poseen una serie de características, entre las que destaca un grado máximo de eficiencia, que permita procesar el mayor número de imágenes por segundo que la cámara pueda obtener.

Hasta el momento se han propuesto un gran número de soluciones, que en su mayoría han optado por el uso de procesamientos sencillos, utilizando como información base la obtenida mediante procesos de filtrado por color[1]. De entre estas soluciones, destaca el uso de líneas de escaneo[2] o soluciones basadas en la detección de bordes[3], que se utilizan ampliamente en la categoría *Standard Platform* de la RoboCup[4].

La propuesta que aquí se presenta ofrece un enfoque diferente, estudiando la viabilidad de utilizar algoritmos genéticos[5] en tiempo real, para realizar el reconocimiento de objetos. Para mejorar la eficiencia del sistema desarrollado, el número de iteraciones e individuos utilizados en cada ejecución del algoritmo deben reducirse lo máximo posible. Para que esta reducción no afecte a las prestaciones del algoritmo, la inicialización de cada población utilizará una gran cantidad de información base, de forma que la validez de los primeros individuos generados permita alcanzar soluciones óptimas tras un reducido número de iteraciones.

En lugar de reducir únicamente el número de iteraciones e individuos, ciertos trabajos proponen el uso de algoritmos evolutivos celulares[6], que muestran un buen comportamiento antes tareas de optimización, que necesiten una alta eficacia con un número reducido de iteraciones.

La información utilizada para inicializar una nueva población la obtendremos de un procesamiento previo de la imagen capturada, así como de la última población evolucionada por el anterior algoritmo. Clonando individuos de poblaciones anteriores para formar una nueva población, aprovecharemos la alta similitud entre sucesivas imágenes capturadas por una cámara de visión acoplada a un robot móvil.

Si una captura recoge un objeto situado a una determinada distancia, la siguiente captura tomada por la cámara posee una alta probabilidad de mostrar el mismo objeto situado a una distancia similar.

Nuestra hipótesis consiste en que utilizando esta similitud entre imágenes, así como la información obtenida en un procesamiento previo, podemos desarrollar un sistema de visión basado en algoritmos genéticos capaz de trabajar en tiempo real.

Para evaluar la propuesta se han realizado pruebas sobre un escenario real con el robot bípedo Nao, oficial dentro de la competición RoboCup Standard Platform, realizando la detección de objetos clave dentro de un campo de fútbol robótico, como las porterías y la pelota.

El resto del artículo se organiza de la siguiente forma. La sección II está dedicada a las restricciones del problema. El sistema completo de Visión se presenta en la sección III, mientras que la sección IV recoge los experimentos y resultados. Por último, la sección V muestra las conclusiones y trabajo futuro.

II. Restricciones del problema

El sistema de Visión debe ser válido para su uso dentro de la categoría *Standard Platform*, por lo que será aplicado con imágenes capturadas por la cámara del robot Nao[2], que obtiene 30 imágenes por segundo de 320 x 240 píxeles, en el

[1] http://www.robocup.org/

[2] http://www.aldebaran-robotics.com/eng/Nao.php

espacio de color YUV[7].
Para reducir la información con la que trabajamos, las imágenes captadas por el robot se filtran previamente, eliminando los píxeles que no hayan pasado un cierto filtro de color. Dentro del entorno Robocup, los colores clave que poseen los objetos son el amarillo y azul para las porterías, verde para el césped, naranja para la pelota, y blanco para las líneas del campo, los contrarios y compañeros. Además de esto, la equipación de cada jugador está formada por una serie de elementos de color rojo o azul oscuro.
El filtrado se realiza definiendo los valores máximo y mínimo de cada componente de color Y, U y V, de forma que un píxel únicamente pasará un filtro si todas sus componentes se encuentran dentro de estos límites.
Un ejemplo de filtrado lo tenemos en la imagen 1, donde se puede observar la imagen original y la obtenida tras aplicar un filtro con el color azul de la portería.

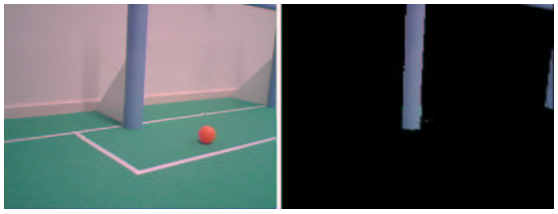

Fig. 1. Filtrado de color para la portería azul

El procesamiento debe permitir detectar los objetos durante un partido, lo que implica oclusiones parciales de los objetos, donde la cámara solamente detecta un porcentaje total del objeto. Ante este tipo de situaciones, los métodos tradicionales basados en líneas de escaneo presentan muchos problemas, mientras que el método aquí propuesto actúa de forma robusta, como se comprobará en la sección de resultados.

III. Sistema de Visión

Un algoritmo genético evoluciona una población de individuos, los cuales representan soluciones potenciales al problema general. En nuestro caso, un individuo debe representar la detección de un objeto a una cierta distancia, con una orientación relativa determinada.
Para evaluar la bondad o *fitness* de cada individuo, contrastamos la información que transporta (objeto o situado a distancia d) con la información extraída de la última imagen capturada por la cámara del robot. Si la información que porta un individuo es verosímil, dada la última captura, se le otorgará un alto valor de bondad. Si por contra, en la imagen capturada no aparece el objeto o que representa el individuo, o aparece a una distancia muy distinta a d, el individuo obtendrá un valor bajo valor de bondad.

A. Esquema general de procesamiento

Nuestro esquema estará dirigido por la llegada de nuevas imágenes a la cámara, de forma que cada vez que llegue una captura, evolucionaremos un conjunto de poblaciones para detectar cada uno de los objetos que queremos detectar.
En nuestro caso, detectamos hasta 3 objetos distintos (Portería Azul, Portería Amarilla y Pelota), por lo que mantenemos 3 poblaciones separadas de individuos. La elección de si es necesario o no evolucionar una nueva población tras la llegada de una nueva imagen, dependerá de si al aplicar el filtro de color específico, hemos obtenido una cantidad suficiente de píxeles. El esquema general es el mostrado en la figura 2.

```
Capturar una nueva imagen
Filtrar la imagen inicial con los filtros de color
para cada objeto
    si hemos obtenido un número de suficiente de píxeles
        Evolucionar una nueva población
        Aplicar búsqueda local sobre el mejor individuo
        Devolver distancia estimada al objeto
    fin si
fin para
```

Fig. 2. Esquema general de procesamiento del sistema

Para evitar caer en óptimos locales, la población de individuos se inicializará de nuevo tras realizar un determinado número de iteraciones sin mejora. Durante el proceso de evolución, se almacena el mejor valor de *fitness* obtenido hasta el momento, de forma que al terminar cada iteración aumentaremos el número de iteraciones sin mejora (si el mejor *fitness* de la iteración no supera el máximo global), o estableceremos este contador a cero (si la iteración proporciona un nuevo máximo global).

B. Representación de Individuos

Además de la distancia entre un objeto y la cámara, también es importante conseguir detectar la orientación entre ambos elementos.
En primer lugar, esta información adicional será necesaria para la futura toma de decisiones, así como para tareas de localización[8]. De forma adicional, necesitamos esta información para poder aplicar la función de *fitness* utilizando imágenes capturadas por la cámara, ya que las capturas obtenidas dependen de la distancia d y con la orientación relativa entre el objeto y la cámara.

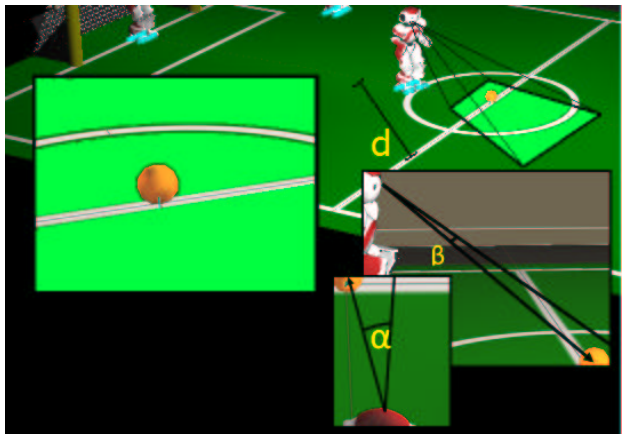

Fig. 3. Captura de imagen y orientación relativa entre cámara y pelota

La figura 3 muestra como la imagen capturada por la cámara representa la pelota, cuando se encuentra a una determinada distancia d, y la diferencia de orientación entre el objeto y la cámara en el eje x es α, y en el eje y β.
Si los valores de α o β variasen en la figura 3, manteniendo la distancia d, la captura de la cámara mostrará una pelota de similares proporciones, pero situada en una zona distinta. Incluso si la variación es grande, la cámara no capturará la pelota.
No es necesario utilizar una componente adicional con la diferencia de orientación en el eje z, debido a que mediante técnicas de detección del horizonte[9], podemos aplicar transformaciones a las imágenes capturadas para obtener todos los objetos de la imagen paralelos al suelo.
Cada individuo de la población almacena la siguiente información (genes):

- Distancia del objeto al robot: d.
- Diferencia de orientación en el eje x: α.
- Diferencia de orientación en el eje y: β.

Cada uno de estos tres genes está representado por un valor numérico, limitado el caso de α y β al ángulo de visión de la cámara del Nao, y en el caso de la distancia, a la distancia mínima y máxima a la que deseamos realizar la detección.
Para poder realizar correctamente la función de *fitness*, los individuos utilizados para detectar las porterías deben contener un gen adicional denotado por θ, que represente la orientación que la portería muestra en la captura. Este parámetro es necesario, ya que dos capturas de una misma portería, con los mismos parámetros $< d, \alpha, \beta >$, variarán enormemente si la portería posee diferentes orientaciones. Esta situación se puede comprobar en la figura 4.

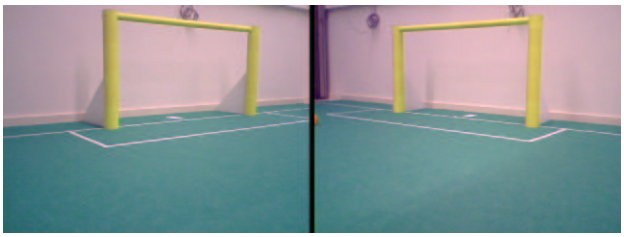

Fig. 4. Imágenes obtenidas variando el parámetro β

C. *Obtención del parámetro* β

Si conocemos en el momento de capturar una imagen el ángulo que posee la cámara del robot en el eje y respecto al suelo, el parámetro β puede obviarse. En lugar de modelar este parámetro β, se calculará utilizando la distancia d, la orientación en el eje x α, y la orientación relativa γ en el eje y entre la cámara del robot y el suelo. Gracias a ello, evitaremos explorar zonas del espacio que representen soluciones que no puedan darse.
Si conocemos γ y el campo de visión de la cámara, podemos obtener la distancia mínima y máxima a la que somos capaces de detectar objetos. Si γ es cercano a 90 grados, el robot detectará objetos situados a distancia lejanas, pero no detectará la pelota cuando esta se encuentre a pequeñas distancias.
El principal problema de no modelar β y obtener este parámetro a partir de otros, es que el algoritmo será totalmente dependiente de una correcta obtención de γ, y dejará de funcionar si la precisión de esta medida se reduce.
En robots equipados con articulaciones, como es nuestro caso, los movimientos del robot provocan grandes movimientos y oscilaciones en la cámara, haciendo realmente complicado obtener de forma precisa γ.
En robots con ruedas, el movimiento del robot afectará en menor medida a la cámara, y la obtención de γ podrá realizarse con mayor precisión, de forma que no será necesario modelar el parámetro β, ya que puede calcularse de forma fiable.

D. *Función de* fitness

La función de *fitness* seleccionada debe ser eficaz, y devolver valores que evalúen la validez de la proyección generada por el individuo $< d, \alpha, \beta >$. Para evaluar la información que representa el individuo, traducimos esta información en una proyección del objeto que representa. La proyección necesita una posición $< x, y >$ de comienzo en la imagen, obtenida a partir de los parámetros α y β. El tamaño del objeto proyectado dependerá del parámetro d.
Para evaluar esta proyección, la comparamos píxel a píxel con la información obtenida en el proceso de filtrado, utilizando el color específico del objeto a detectar. Cada píxel $< x, y >$ de la proyección se considera válido si el píxel $< x, y >$ de la imagen capturada ha superado el filtro de color, y no válido en caso contrario.
Un ejemplo puede verse en la figura 5, donde a la izquierda se muestra la imagen filtrada con el color naranja, y a la derecha el resultado de evaluar 12 individuos, donde los píxeles rojos sumarían valores negativos, y los verdes positivos, por corresponderse con píxeles que han superado el filtro de color.

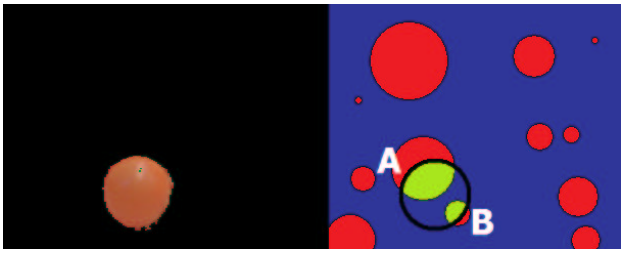

Fig. 5. Imagen filtrada(izq) y estudio de 12 individuos para detectar la pelota(dcha)

Dado que obtenemos la cantidad de píxeles positivos y negativos de una proyección, la primera idea para la función de *fitness* consistió en utilizar el porcentaje de píxeles de la proyección que han superado el filtro. Esto conlleva un importante problema, ya que favorece individuos que representan objetos lejanos, que obtienen proyecciones más pequeñas con mayor probabilidad de obtener un alto porcentaje de píxeles que pasen el filtro de color.
Debido a este problema, y aprovechando que durante el proceso de filtrado obtenemos la cantidad de píxeles que ha superado cada filtro de color, devolvemos como valor de *fitness* el valor mínimo de los dos siguiente valores:

- % de píxeles de la proyección que han superado el filtro

– % de píxeles de la muestra de color que se utilizan en la proyección superando el filtro

Para ilustrar como funciona, estudiamos en la figura 5 los individuos A y B. Mientras que B posee un mayor porcentaje de píxeles que han superado el filtro que A (70 frente a 45), únicamente un 5% de los píxeles que superaron el filtro naranja pertenecen al objeto B, frente al 35% de A aproximadamente. El *fitness* final será de 0.35 para A y 0.05 para B.

E. Inicialización de la población

La inicialización de la población de un algoritmo genético suele ser aleatoria, generando uniformemente en los límites de los genes, pero en nuestro caso usaremos poblaciones informadas. En primer lugar podemos utilizar individuos que provengan de la población de capturas anteriores. Además, durante el proceso de filtrado obtuvimos el número de píxeles de la muestra de cada color, así como la componente x e y del centroide de la distribución, lo cual puede utilizarse para inicializar los nuevos individuos.
Aprovechando esta información, generamos individuos de 3 maneras distintas:

– Aleatoriamente
– Utilizando la información obtenida en el proceso de filtrado
– Clonando un individuo de poblaciones anteriores

Las dos primeras formas para generar un individuo pueden utilizarse siempre. La tercera opción se podrá utilizar únicamente cuando tengamos poblaciones anteriores válidas. Para definir esta validez necesitamos conocer el tiempo transcurrido entre la captura actual, y la última que evolucionó una población p, dando como resultado la detección del objeto o que intentamos detectar. Si este tiempo supera cierto umbral, no generaremos ningún individuo utilizando esta población anterior.
Para inicializar una población, por cada nuevo individuo realizamos un sorteo, donde cada una de las opciones contará con una probabilidad. Estas 3 probabilidades serán variables y estarán condicionadas al número de capturas entre la actual y la última que detectó el objeto que estamos estudiando. Para ello definimos 2 parámetros:

– MW: Máxima probabilidad de clonar un individuo de una población anterior
– MFN: Número máximo de capturas que pueden existir entre la actual y la última que detectó el objeto estudiado.

La suma de las probabilidades de las tres maneras de generar un individuo debe sumar 1.0. La probabilidad de escoger un individuo de una anterior población disminuirá a medida que aumente el número de capturas sin actualizar esta población, Las otras dos probabilidades se obtienen utilizando $CloneProb$.

$$CloneProb : MW - MW * (NFWU/MNF)$$
$$InitialInfoProb : (1 - (CloneProb)) * 0.66$$
$$RandomlyProb : (1 - (CloneProb)) * 0.34$$

Aumentar el número de individuos generados aleatoriamente fomentará la diversidad de la población inicial. Utilizando la información de filtrado, aumentaremos el elitismo del algoritmo, así como el riesgo de caer en óptimos locales. El uso de individuos clonados de iteraciones anteriores, permitirá una rápida convergencia ante pequeñas variaciones entre capturas. Combinando correctamente estas tres opciones, aseguraremos el equilibrio entre elitismo y generalidad. Seleccionamos los valores de 0.66 y 0.34 con el objetivo de obtener una población inicial heterogénea, en base a los test empíricos preliminares.

F. Oclusión parcial de objetos

En entornos complicados y cambiantes como los escenarios RoboCup, es indispensable que el sistema de visión funcione de forma robusta ante oclusiones de objetos, ya que los objetos que deseamos detectar se encontrarán la mayor parte del tiempo ocultos, parcial o totalmente tras otros objetos. De forma adicional, la cámara del robot obtendrá un gran número de capturas que recojan parcialmente el objeto, debido a la orientación entre la cámara y el objeto.
La propuesta actual funciona correctamente ante oclusiones, ya que durante la evaluación de un individuo se estudia su proyección completa, y no características parciales del objeto. Cuando se presenten obstáculos entre la cámara y el objeto a detectar, la función de *fitness* obtendrá valores inferiores, pero permitirá detectar el objeto. Para aquellas situaciones en las que una captura muestre parcialmente un objeto, como la de la figura 6, existirán individuos dentro del espacio de búsqueda que permitan representar esta situación, a los cuales la función de *fitness* otorgue un alto valor. Esto se consigue aumentando el número de orientaciones posibles entre cámara y el objeto, a todas aquellas que proyecten objetos dentro de los límites de la captura.

Fig. 6. Captura parcial de un objeto

IV. EXPERIMENTOS Y RESULTADOS

El entorno donde se han realizado los experimentos se corresponde con un escenario RoboCup, con un campo de fútbol de 6 x 4 metros y utilizando la pelota y porterías oficiales. Hemos utilizado un robot Nao, capturando 2 imágenes por segundo en formato YUV, de tamaño 320 x 240 píxeles.
Durante los experimentos, por cada captura de la cámara, se almacenó la diferencia absoluta entre la distancia real y la distancia estimada por el individuo de la población con mayor *fitness*, para cada uno de los elementos detectados.
Las condiciones de iluminación permanecieron invariables durante los experimentos, de forma que los filtros de color fueron los óptimos durante la realización de las pruebas.
Realizamos dos estimaciones por segundo, ya que a pesar de

que el tiempo de procesamiento es variable, en ningún caso una captura necesitó más de 500 milisegundos para realizar el procesamiento completo.
Después del proceso de filtrado ($\approx$ 80 msec), el tiempo de ejecución nunca fue superior a 370 milisegundos (183 para la portería y 187 para la pelota).

A. Parámetros del Algoritmo Genético

Los experimentos se realizaron con los siguientes parámetros:

- Número de Individuos: 12
- Número de Iteraciones: 24
- Probabilidad de mutación: 5%
- Tipo de operador de cruce: por punto
- Tipo de reemplazo: Algoritmo Generacional
- Reinicio tras superar un 25% de iteraciones sin mejorar el óptimo global
- MW: 0.5
- MFN: 10

Con estos parámetros, el algoritmo posee número de individuos e iteraciones reducido, utilizando una probabilidad de mutación y operador de cruce estándar. Al ser un algoritmo generacional, cada nueva población de hijos obtenida tras el cruce y mutación reemplazará a sus progenitores, lo que puede provocar que la calidad de la población se devalúe con el paso de las iteraciones. Los individuos son evolucionados sin tener en cuenta la odometría del robot.
Al mejor individuo de cada iteración se le aplica un proceso de búsqueda local simple (*Hill Climbing*), que permite mejorar el *fitness* final obtenido tras cada captura. El proceso de búsqueda local consiste en evaluar variaciones positivas y negativas sobre los genes de los individuos. A los algoritmos que combinan conceptos y estrategias de diferentes metaheurísticas se les denomina algoritmos meméticos[10].

B. Experimento 1 - Validación de la Hipótesis

El primer experimento tiene como objetivo comprobar si el sistema desarrollado puede trabajar en tiempo real, detectando correctamente los elementos del entorno, estimando distancias correctamente. Para ello utilizamos los parámetros por defecto del algoritmo genético descritos anteriormente.
Realizamos un mismo recorrido por el terreno de juego en 6 ocasiones. En cada recorrido se realizan 30 capturas (tomadas durante 15 segundos) en las cuales se puede observar la portería amarilla, situada entre 360 y 300 cm., y la pelota naranja, entre 260 y 200 cm.
En total el experimento consta de 180 capturas distintas (6 x 30). Por cada captura se almacena la diferencia absoluta entre la distancia real y la estimada por el algoritmo (denotada por $DBRED$) y el *fitness* del mejor individuo evolucionado para realizar la detección. Este valor se utiliza para generar diversos conjuntos de datos, los cuales contienen únicamente las diferencias obtenidas con individuos cuyo valor de bondad supere un cierto umbral.
La tabla I muestra de forma separada para la pelota y la portería amarilla, y únicamente para las detecciones realizadas con individuos con *fitness* mayor que 0, 0.25, 0.5 y 0.75, el valor medio de las diferencias entre la distancia real y estimada ($DBRED$) para estas capturas, y el porcentaje del total de las 180 capturas que obtienen, durante la detección del elemento, su mejor individuo con un *fitness* superior a estos umbrales.

TABLE I
DIFERENCIA MEDIA ENTRE POSICIÓN REAL Y ESTIMADA Y PORCENTAJE DE CAPTURAS POR ENCIMA DE DIFERENTES UMBRALES DE FITNESS

	Fitness	> 0.0	> 0.25	> 0.5	> 0.75
Pelota	Dif.media (cm)	42.62	40.57	31.77	22.75
	Capturas (%)	68.89	68.33	57.78	8.89
Portería	Dif.media (cm)	40.03	37.88	33.1	32.69
Amarilla	Capturas(%)	99.44	93.33	44.44	8.89

Se puede comprobar cómo el valor obtenido con la función de *fitness* representa fielmente la validez de los individuos, de forma que a medida que utilizamos únicamente detecciones realizadas con un valor de *fitness* por encima de un cierto umbral, la media de las diferencias entre distancias reales y estimadas disminuye.
Para evaluar la robustez del algoritmo, también mostramos en la tabla II el porcentaje de capturas cuya diferencia entre la distancia real y estimada se encuentra por debajo de ciertos umbrales.

TABLE II
PORCENTAJE DE CAPTURAS POR DEBAJO DE DISTINTOS UMBRALES DE DIFERENCIA ENTRE DISTANCIA REAL Y ESTIMADA

	Porcentaje de capturas por debajo de			
	100cm	75cm	50cm	30cm
Pelota	63.63	56.11	44.44	35.55
Portería Amarilla	92.77	87.78	72.22	51.67

Los valores obtenidos muestran una gran robustez, especialmente en el caso de la portería, ya que en un escenario con una distancia máxima entre elementos de 721 centímetros, un alto porcentaje de capturas (37.37% y 51.67%) obtiene una diferencia entre la distancia real y estimada por debajo de 30 centímetros.
La detección de la pelota a través de un genético resulta más complicada que la portería, ya que sólo individuos cercanos al máximo global obtienen valores de *fitness* superiores a cero, evitando que el algoritmo converja a través de las iteraciones. Debido al pequeño tamaño de la pelota en las imágenes capturadas, sólo aquellos individuos que representen este objeto con una diferencia de orientación y distancia muy cercana a la real, obtienen valores de bondad distintos a 0. La convergencia de un algoritmo genético de estas características no será constante. Un 83.83% de las detecciones que se han realizado correctamente para la pelota (fit$>$ 0), obtuvieron valores de *fitness* superiores a 0.5. Para las porterías, este porcentaje baja a un 44.69%.

C. Experimento 2 - Estudio de β

El objetivo del segundo experimento es estudiar si la obtención del parámetro β a partir del resto de parámetros puede ser realizada, además de evaluar cómo afecta al rendimiento general del algoritmo. Realizamos el mismo

recorrido del experimento anterior, utilizando los mismos parámetros.
Durante el experimento, los individuos no contendrán el gen que representa el parámetro β, y para realizar la función de evaluación, calcularemos este parámetro utilizando para ello la distancia al objeto d, la diferencia de orientación en el eje x entre cámara y objeto α, y la diferencia de orientación entre la cámara y el suelo en el eje y γ, obtenida gracias al estudio de los sensores del robot.
De nuevo obtuvimos 180 capturas en total, con los resultados mostrados en la tabla III.

TABLE III
DIFERENCIA MEDIA ENTRE POSICIÓN REAL Y ESTIMADA Y PORCENTAJE DE CAPTURAS POR ENCIMA DE DIFERENTES UMBRALES DE FITNESS

	Fitness	> 0.0	> 0.25	> 0.5	> 0.75
Pelota	Dif.media (cm)	18.70	18.05	16.89	27.7
	Capturas (%)	69.44	68.33	57.78	5.55
Portería Amarilla	Dif.media (cm)	33.66	32.81	34.31	27.5
	Capturas(%)	100.0	95.00	40.56	1.11

La primera conclusión que podemos extraer de estos resultados, es que realizamos un mayor número de detecciones correctas (*fitness* > 0), aunque disminuye el porcentaje de capturas que obtienen un valor de *fitness* superior a 0.5 y 0.75. Esto se debe a que modelar β permite generar individuos que no podrían ser obtenidos calculando este parámetro, ya que recogen situaciones incorrectas según la obtención de β a partir del resto de parámetros, pero que debido al ruido, o a diferencias entre el valor de γ real y estimado, son correctas. La diferencia media entre las distancias reales y estimadas se ha reducido considerablemente, ya que al tener un parámetro menos, con un número similar de iteraciones el algoritmo converge más rápido a individuos prometedores.
Para realizar una comparación completa entre el modelado de β y su obtención directa a través de otros parámetros, la tabla IV muestra el porcentaje de capturas con diferencia entre la distancia real y estimada por debajo de ciertos umbrales.

TABLE IV
PORCENTAJE DE CAPTURAS POR DEBAJO DE DISTINTOS UMBRALES DE DIFERENCIA ENTRE DISTANCIA REAL Y ESTIMADA

	Porcentaje de capturas por debajo de			
	100cm	75cm	50cm	30cm
Pelota	68.89	68.89	65.56	54.44
Portería Amarilla	96.67	93.33	76.67	48.89

Comparando la tabla IV con la II, podemos observar como mejora la robustez del algoritmo, ya que la rápida convergencia de las poblaciones, permite obtener un alto porcentaje de capturas que obtienen una reducida diferencia entre el valor real y estimado, para la distancia al objeto a procesar.
Como principal conclusión, destacamos la conveniencia de reducir el número de genes que posee cada individuo de una población, a los estrictamente necesarios. Para ello utilizamos la mayor cantidad posible de información del entorno, de la plataforma a utilizar y de los elementos a detectar. Esto nos permitirá reducir el espacio de búsqueda del algoritmo genético, explorando únicamente soluciones que se correspondan con situaciones reales. Al contar con un menor número de parámetros, el algoritmo convergerá más rápido, permitiendo alcanzar mejores óptimos con un número reducido de iteraciones.

D. Experimento 3 - Estudio de MW

El objetivo del tercer experimento consiste en mostrar la forma en la que el parámetro MW afecta al sistema de visión. Este parámetro define la probabilidad máxima de clonar un individuo de poblaciones anteriores, a la hora de generar una nueva población, y por tanto define el peso de poblaciones anteriores a la hora de inicializar una nueva población.
A medida que aumentemos el valor de este parámetro, un mayor número de individuos de los que conformarán la nueva población representarán soluciones para la detección de objetos en capturas anteriores.
El experimento consistió en una secuencia de 20 imágenes capturadas con el robot desde una posición estática, situada a 250 cm de la portería azul y 150 cm de la pelota. Mientras se realizó el experimento, la orientación de la cámara cambió de valores, capturando diferentes imágenes de objetos situados a la misma distancia. Debido a las diferentes orientaciones de la cámara, la mayoría de las capturas mostraron parcialmente y no en su totalidad la portería y la pelota.
Durante la realización del experimento utilizamos los parámetros por defecto, incluyendo el parámetro β entre la información que poseen los individuos. Los cambios en el valor de MW definieron las diferentes configuraciones que fueron probadas. El experimento se repitió en 9 ocasiones con cada distinta configuración, almacenando un conjunto total de 180 capturas. Evaluamos 4 diferentes configuraciones, con valores para la probabilidad máxima de seleccionar un individuo de la población anterior (MW) de 0, 25, 50 y 75%.

TABLE V
DIFERENCIA MEDIA ENTRE POSICIÓN REAL Y ESTIMADA DE CAPTURAS CON VALORES DE FITNESS POR ENCIMA DE DIFERENTES UMBRALES

	MW	$Fit > 0$	$Fit > 0.25$	$Fit > 0.5$	$Fit > 0.75$
Pelota	0.00	47.37	47.37	36.93	31.75
	0.25	43.10	41.43	34.26	34.27
	0.50	41.37	41.26	33.63	33.67
	0.75	43.48	42.08	32.72	33.49
Portería Azul	0.00	58.02	49.48	27.15	12.78
	0.25	53.64	42.63	26.71	19.72
	0.50	51.22	43.54	21.76	14.16
	0.75	44.16	37.60	24.45	15.39

La tabla V muestra de nuevo como la función de *fitness* representa fielmente la validez de los individuos durante el proceso, aunque se puede comprobar que las variaciones en el valor de MW no provocan grandes cambios en la diferencia media entre distancia real y estimada de las capturas, utilizando únicamente capturas por encima de varios umbrales de *fitness*.

La tabla VI muestra como a medida que utilizamos valores mayores de MW, el porcentaje de capturas que obtiene un mayor valor de *fitness*, y que por tanto realizan mejores estimaciones, aumenta. En el caso de la portería, esta tendencia se mantiene a medida que aumenta MW, mientras que para

TABLE VI
PORCENTAJE DE CAPTURAS POR ENCIMA DE DIFERENTES UMBRALES DE FITNESS

	MW	$Fit > 0$	$Fit > 0.25$	$Fit > 0.5$	$Fit > 0.75$
Pelota	0.00	93.59	93.59	78.84	35.26
	0.25	91.66	91.02	80.12	44.87
	0.50	89.74	89.10	75.64	29.49
	0.75	89.10	87.18	75.00	32.69
Portería Azul	0.00	100.0	82.68	47.49	12.85
	0.25	98.32	83.80	56.42	13.97
	0.50	98.88	87.71	55.87	13.97
	0.75	98.88	89.94	64.25	12.85

la pelota, parece que el punto óptimo se encuentra en 0.25, tras lo cual el rendimiento empeora.

TABLE VII
PORCENTAJE DE CAPTURAS POR DEBAJO DE DISTINTOS UMBRALES DE DIFERENCIA ENTRE DISTANCIA REAL Y ESTIMADA

		Porcentaje de capturas por debajo de			
	MW	100cm	75cm	50cm	30cm
Pelota	0.00	82.69	72.44	62.18	33.33
	0.25	85.90	79.49	71.79	31.41
	0.50	85.90	75.00	67.31	35.30
	0.75	80.77	73.72	64.10	32.69
Portería Azul	0.00	77.09	68.71	55.31	32.96
	0.25	81.00	73.74	62.57	34.08
	0.50	81.56	75.41	61.45	43.01
	0.75	87.15	78.77	72.07	48.60

Finalmente, la tabla VII presenta el porcentaje total de capturas, que obtienen una diferencia entre la distancia real y estimada por debajo de ciertos umbrales. En esta tabla se puede comprobar perfectamente que aumentar el valor de MW mejora enormemente el rendimiento del algoritmo en el caso de la portería. Para el caso de la pelota, al aumentar el valor de MW por encima de 0.25 obtenemos peores resultados, como se pudo intuir tras la tabla VI.

La explicación del porqué de estos comportamientos del sistema en función de MW proviene de las características propias de los objetos que intentamos detectar.

La pelota siempre aparece en las imágenes como un objeto redondo de pequeño tamaño y color naranja. En muy pocas capturas la pelota se encuentra parcialmente oculta (al contrario que ocurría en la liga que utilizaba robots AIBO), por lo que la información de filtrado resulta muy útil para inicializar nuevos individuos. La posición $< x, y >$ de la pelota en una captura, será en muchas ocasiones cercana al centroide $< x, y >$ de la muestra de píxeles naranja, obtenida tras el proceso de filtrado. Como se indicó en el primer experimento, únicamente individuos muy cercanos a la solución obtendrán valores de *fitness* mayores que cero.

Por estas razones, al aumentar el porcentaje de individuos que se clonan de iteraciones anteriores por encima de cierto umbral, las prestaciones del algoritmo se devalúan, ya que se crean demasiados individuos que representan soluciones a capturas anteriores y menos con la información obtenida con el proceso de filtrado. Los individuos de poblaciones anteriores pueden no corresponderse con soluciones válidas para la captura actual, y por tanto obtendrán valores de *fitness* 0, retrasando la convergencia del algoritmo.

A pesar de estos inconvenientes, introducir una proporción media de individuos provenientes de poblaciones anteriores (entre 0.25 y 0.5), mejorará las prestaciones del algoritmo, ya que la población contará con individuos que poseen información valiosa, y que tras cruzarse con el resto de individuos podrán dar lugar individuos con altos valores de *fitness*. Incluir individuos de iteraciones anteriores también permitirá al algoritmo converger más rápido cuando la información obtenida con el proceso de filtrado no sea valiosa, debido al ruido del entorno.

Para la detección de las porterías, la situación es completamente distinta. Las porterías se captan como objetos de distinta forma y tamaño, en función de la posición desde la que se observen. Su tamaño es mucho mayor que el de la pelota, por lo que ocuparán una mayor superficie en las capturas, como puede comprobarse en la figura 6. A diferencia de la pelota, individuos muy alejados de la mejor solución pueden poseer valores de *fitness* superiores a cero, por portar en sus genes información valiosa. Durante la detección de una portería, existe un alto riesgo de caer en óptimos locales (como individuos que representen porterías que compartan únicamente uno de los postes con la portería captada en una imagen), por lo que se debe evitar un excesivo elitismo, inicializando individuos mediante los tres sistemas disponibles.

La información obtenida con el proceso de filtrado de color, no resulta tan valiosa como en el caso de la pelota, ni permite crear individuos cercanos a la mejor solución. Sin embargo, los individuos de poblaciones anteriores, a pesar de no representar el óptimo global de la población, obtendrán valores de *fitness*, que permitirán tras un pequeño número de mutaciones y cruces, alcanzar valores cercanos a la solución global del problema.

V. CONCLUSIONES Y TRABAJO FUTURO

En base a los resultados obtenidos en el primer experimento, el sistema ofrece una robusta alternativa a los sistemas tradicionales de detección de objetos, utilizando para ello los principios de los algoritmos genéticos. El sistema posee unos tiempos de ejecución que le permiten ser ejecutado en entornos RoboCup. El sistema funciona correctamente ante la aparición de oclusiones, sin necesidad de enfoques basados en casos

El parámetro β debe calcularse a partir del resto de parámetros. Este parámetro se puede obtener correctamente los valores de los sensores del robot se obtienen sin error. El número de genes de cada individuo debe tan pequeño como sea posible.

En base a los resultados obtenidos tras el tercer experimento, se ha demostrado que la relación existente entre imágenes capturadas de forma consecutiva, puede ser utilizada para mejorar el rendimiento del sistema desarrollado, acelerando la convergencia hacia mejores soluciones.

El sistema se ha realizado para la detección de porterías y de la pelota, pero en vista a los resultados obtenidos, las alternativas actuales para la detección de la pelota, y la especial dificultad en la detección de porterías, el principal campo de aplicación del sistema sería únicamente la detección de porterías.

Como principal línea de trabajo futuro, encontramos la integración del sistema desarrollado con un método formal de localización como Montecarlo[11] o Filtros de

Kalman[12], dado que estos métodos combinan la información de movimiento (odometría) y la información visual. El método seleccionado deberá utilizar la distancia y orientación estimada a las porterías, objetos clave para localizar al robot, además de aprovechar el valor de *fitness* para integrar las dos fuentes de información de la mejor manera posible[13].

Otra línea de trabajo estaría relacionada con la inclusión de restricciones durante la creación de nuevos individuos, utilizando para ello la creencia sobre la posición y orientación del robot, limitando el rango de opciones a los objetos que sea posible observar desde la pose estimada actual.

REFERENCES

[1] Wasik, Z., Saffiotti, A.: Robust color segmentation for the robocup domain. Pattern Recognition, Proc. of the Int. Conf. on Pattern Recognition (ICPR) **2** (2002) 651–654

[2] Jüngel, M., Hoffmann, J., Lötzsch, M.: A real-time auto-adjusting vision system for robotic soccer. In: In 7th International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences), Lecture Notes in Artificial Intelligence, Springer (2004) 214–225

[3] Coath, G., Musumeci, P.: Adaptive arc fitting for ball detection in robocup. In: APRS Workshop on Digital Image Analysing. (2003) 63–68

[4] Rofer, T., Brunn, R., Dahm, I., Hebbel, M., Hoffmann, J., Jungel, M., Laue, T., Lotzsch, M., Nistico, W., Spranger, M.: GermanTeam 2004. Team Report RoboCup (2004)

[5] Mitchell, M.: An Introduction to Genetic Algorithms. (1996)

[6] Whitley, L.: Cellular Genetic Algorithms. In: Proceedings of the 5th International Conference on Genetic Algorithms table of contents, Morgan Kaufmann Publishers Inc. San Francisco, CA, USA (1993)

[7] Foley, J.D., van Dam, A., Feiner, S.K., Hughes, J.F.: Computer graphics: principles and practice. Addison-Wesley Longman Publishing Co., Inc. (1990)

[8] Borenstein, J., Everestt, H., Feng, L.: Where am I? Sensors and Methods for Mobile Robot Positioning. (1996)

[9] Bach, J., Jungel, M.: Using pattern matching on a flexible, horizon-aligned grid for robotic vision. Concurrency, Specification and Programming-CSP **1**(2002) (2002) 11–19

[10] Moscato, P.: Memetic algorithms: a short introduction. Mcgraw-Hill'S Advanced Topics In Computer Science Series (1999) 219–234

[11] Fox, D., Burgard, W., Thrun, S.: Active markov localization for mobile robots (1998)

[12] Negenborn, R.: Robot localization and kalman filters. (2003)

[13] Jesús Martínez-Gómez, José A. Gámez, I.G.V.: An improved markov-based localization approach by using image quality evaluation. In: Proceedings of the 10th International Conference on Control, Automation, Robotics and Vision (ICARCV). (2008) 1236 – 1241