

Ensenada, B.C., a 4 de abril de 2016

**Dr. Silvio Guido Lorenzo Marinone Moschetto**  
Director General del Centro de Investigación Científica y de  
Educación Superior de Ensenada, Baja California (CICESE)  
PRESENTE

El suscrito, **Luis Arturo Guerra García**, bajo protesta de decir verdad declaro que me encuentro adscrito al Programa de Posgrado del **Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California**, en lo sucesivo **CICESE**, denominado **Ciencias de la Computación** y que soy autor de la tesis de **Maestría en Ciencias**, titulada:

**Atención visual integrando el paradigma de potencialidades en función de la distancia**

De igual forma declaro que dicha Tesis fue realizada como parte de los resultados de los estudios de posgrado que me encuentro realizando en el CICESE, Tesis que ha sido dirigida por **Gustavo Olague Caballero y Jose Luis Briseño Cervantes**, por lo que reconozco que la tesis fue realizada como consecuencia de las actividades desarrolladas por el suscrito en colaboración y bajo la dirección del personal del CICESE, inclusive con el uso de su infraestructura e información propiedad del CICESE, en tal sentido ratifico que el CICESE es titular de los derechos patrimoniales de la Tesis, conforme a lo previsto en el artículo 83 de la Ley Federal del Derecho de Autor, reservando para el suscrito los derechos morales de la misma obra.

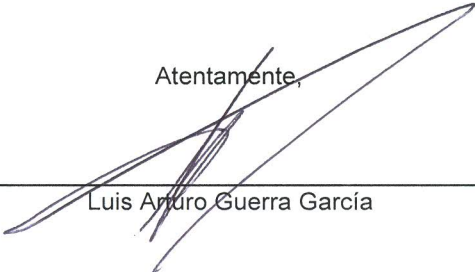
Conforme a los intereses de CICESE la obra citada puede estar disponible en el catálogo de su Biblioteca, en texto completo (formato PDF), en el sitio Web de la biblioteca (<http://biblioteca.cicese.mx>) y pueda ser copiada, reproducida, distribuida, transmitida, difundida y almacenada en cualquier tipo de formato impreso y medio electrónico, analógico o digital que se disponga en el presente y el futuro, así como de explotar cualquier parte de la misma.

De igual manera, es de mi conocimiento que la publicación de la tesis no tiene una finalidad lucrativa, sino académica; por lo que otorgo la autorización correspondiente para que la difusión pueda efectuarse ampliamente, a través de cualquier tipo de medios, tanto en red local como por vía Internet o de cualquier otro sistema o tecnología creada o por crearse, todo esto en apego al artículo 27 de la Ley Federal de Derechos de Autor.

Así mismo, manifiesto que el CICESE queda liberado y deslindado de cualquier responsabilidad moral, social y jurídica que pueda darse u ocasionar conflictos con terceros a raíz de la información contenida en la tesis de mi autoría, como lo es el contenido, el estilo de la redacción y uso de las gráficas, imágenes, fotografías, tablas, datos, textos y el uso del software que se hubiera utilizado.

Sin otro asunto en particular, quedo de usted como su atento y seguro servidor.

Atentamente,



---

Luis Arturo Guerra García

---

11262679

**Centro de Investigación Científica y de Educación  
Superior de Ensenada, Baja California**



**Programa de Posgrado en Ciencias  
en Ciencias de la Computación**

---

**Atención visual integrando el paradigma de potencialidades  
en función de la distancia**

Tesis

para cubrir parcialmente los requisitos necesarios para obtener el grado de  
Maestro en Ciencias

Presenta:

**Luis Arturo Guerra García**

Ensenada, Baja California, México

2016



Tesis defendida por

**Luis Arturo Guerra García**

y aprobada por el Comité

---

**Dr. Gustavo Olague Caballero**

*Codirector del Comité*

---

**M. en C. José Luis Briseño Cervantes**

*Codirector del Comité*

**Dr. Israel Marck Martínez Pérez**

**Dr. Roberto Conte Galván**



---

**Dr. Jesús Favela Vara**

*Coordinador del Programa de Posgrado en Ciencias de la Computación*

---

**Dra. Rufina Hernández Martínez**

*Director de Estudios de Posgrado*

*Luis Arturo Guerra García © 2016*

*Queda prohibida la reproducción parcial o total de esta obra sin el permiso formal y explícito del autor*



Resumen de la tesis que presenta Luis Arturo Guerra García como requisito parcial para la obtención del grado de Maestro en Ciencias en Ciencias de la Computación.

## **Atención visual integrando el paradigma de potencialidades en función de la distancia**

Resumen aprobado por:

---

**Dr. Gustavo Olague Caballero**

Codirector de Tesis

---

**M. en C. José Luis Briseño Cervantes**

Codirector de Tesis

El concepto de potencialidades, la gama de posibles acciones que un organismo percibe visualmente de los objetos, ha sido adoptada con interés por parte de la comunidad de visión artificial. Además de proponer otro camino para comprender el proceso de visión, la idea ha mostrado ser versátil, lo que permite su aplicación en distintas formas, ya sea como agente principal para la solución de un problema de visión, o como parte de un sistema que lo utiliza en conjunto con otros modelos.

El presente trabajo de investigación propone integrar dicho concepto al modelo de ruta dorsal artificial (ADS), el cual, combina las dimensiones de forma, color, orientación e intensidad de la imagen capturada para producir un foco de atención. La ruta dorsal biológica utiliza más dimensiones, y una de ellas es la distancia; la información de esta dimensión permite realizar las tareas visomotoras (tarea importante de dicha ruta), por lo tanto, se considera necesario integrarla a la ADS. Para ello, se propone el paradigma de potencialidades en función de la distancia, idea que establece el propósito y la forma de dicha integración.

El entrenamiento del sistema propuesto se realiza de dos formas: con una base de imágenes, y “en vivo”. Este último utiliza sólo las imágenes capturadas durante el entrenamiento.

Los experimentos muestran que la integración de la distancia a la ADS mejora la capacidad de detección, especialmente, en la capacidad de ubicar el punto de foco de atención dentro del objeto. Dicho punto es indispensable para la ubicación correcta del objeto en el espacio. También se implementa con éxito la automatización para crear las imágenes de entrenamiento; un paso básico para el entrenamiento en vivo, y benéfico para el entrenamiento con base de imágenes. Sin embargo, el método de entrenamiento en vivo muestra una serie de problemas que necesitan ser resueltos antes de poder producir los resultados deseados.

Palabras Clave: **Atención visual, Potencialidades, Distancia, Ruta dorsal artificial**



Abstract of the thesis presented by Luis Arturo Guerra García as a partial requirement to obtain the Master of Science degree in Master in Sciences in Computer Science.

## **Visual attention that integrates the paradigm of affordances as a function of distance**

Abstract approved by:

---

**Dr. Gustavo Olague Caballero**

Codirector Thesis

---

**M. en C. José Luis Briseño Cervantes**

Codirector Thesis

The concept of affordances, the range of possible actions that an organism can perceive visually from objects, has been taken with interest by the computer vision community. Besides proposing a different way to understand the vision process, the idea has shown to be versatile, which allows to use the concept in a variety of ways, whether as the main agent for the solution of a computer vision problem, or as a component of a system that applies it together with other models.

With this research, we propose to integrate the concept of affordances with the artificial dorsal stream model (ADS). The ADS combines the dimensions of shape, color, orientation and intensity from the captured image to produce a focus of attention. The biological dorsal stream includes more dimensions, one of them being the distance; the information of this dimension allows to perform visuomotor tasks (a main task of the dorsal stream), so we believe it is necessary to add it to the ADS. To accomplish this, we propose the paradigm of affordances as a function of distance, which states the purpose and form of this integration.

The training phase of the proposed system is executed in two ways: one with an image database, the other with a “live” training. The latter only uses the images that the system captures during the training phase.

The experiments show that the ADS improves its detection capabilities with the addition of distance information. Its capability to place the point of focus of attention on the object showed a good improvement. This point is critical for the correct location of an object in space. We successfully automated the process to create training images; this is a basic step for the “live” training, and it’s beneficial for the image database training as well. Nevertheless, the live training method itself showed various problems that we need to solve before said method can produce the desired results.

**Keywords: Visual attention, Affordances, Distance, Artificial dorsal stream**





## **Dedicatoria**

***A mis padres***

***Rafael Guerra Barajas y***

***Adreana Cecilia García Campos***



## Agradecimientos

“La fuerza personal de cada ser humano tiene un límite, pero esta pareciera hacerse infinita cuando recibe la ayuda de los que le rodean.”

- Luis Arturo Guerra García

A mi padre, Rafael Guerra Barajas, porque siempre me ha impulsado a seguir adelante, a prepararme, a no rendirme, a ser mejor de forma personal y profesional cada día. Me ha dado el ejemplo de trabajar con responsabilidad y amor a la profesión. Su apoyo y su amor incondicional no me han faltado incluso en los momentos más difíciles de su vida.

A mi madre, Adreana Cecilia García Campos, por ser el cimiento de mi vida. Su consejo, su apoyo y su amor incondicional me han permitido seguir cumpliendo mis metas, y formando nuevas. Me ha enseñado a alcanzar la felicidad a pesar de los constantes problemas de la vida.

A mi tío Héctor Velasco Rodríguez, mi tía Lucía Guillermina García Campos, y mis primos Ulises Velasco García y Homero Velasco García, porque siempre han procurado mi bienestar y me han brindado su apoyo incondicional en mis momentos de mayor dificultad.

A mi tío José Luis Guerra Barajas y mi tía Teresa Guillén Cruz, quienes siempre han estado al pendiente de mí, cuidando que no me faltara nada durante mi estancia en la ciudad de Ensenada.

A mi abuela Guillermina Campos Alfaro, y mi tío abuelo Guillermo Enrique Campos Alfaro, que en paz descansen. De pensamiento revolucionario, y un ejemplo para mí de fuerza y preparación constante. Me enseñaron que uno puede seguir desarrollándose en el arte y en la ciencia durante toda la vida.

A mis asesores, el Dr. Gustavo Olague Caballero, y el M. en C. José Luis Briseño Cervantes, y mis sinodales, el Dr. Israel Marck Martínez Pérez, y el Dr. Roberto Conte Galván, por su amistad, su consejo, su enseñanza, su paciencia, su apoyo, su confianza, y la ayuda que me ofrecieron, tanto personal como profesionalmente.

Al Dr. José Antonio García Macías, la Dra. Ana Isabel Martínez García, y el Dr. Jesús Favela Vara, pues tuvieron confianza en mi trabajo para el desarrollo y finalización de esta tesis, dándome el apoyo de la institución para continuar con mis estudios a pesar de las múltiples dificultades que surgieron. Así pues agradezco de igual forma a los que durante mi estancia formaron parte del Consejo de Programa de Posgrado y el Comité de Docencia, que comprendieron mi situación y me permitieron seguir adelante.

A Lina Ivonne Best Guzmán, por su amistad, porque que estuvo pendiente de mi salud, y me guió para conservar la beca de CONACyT a pesar de los distintos problemas que surgieron.

A mis amigos y compañeros de laboratorio Jessica Arballo García, Eddie Helbert Clemente Torres, León Felipe Dozal García, y Daniel Hernández, por su amistad, su guía, y sus valiosos aportes para el desarrollo de mi tesis.

A mis amigos Hugo Armando Guillén Ramírez, Nelson Emmanuel Ordoñez Guillén, y Andersen Herrera Hidalgo que siempre han estado al pendiente de mi y me han ayudado de formas múltiples y valiosas.

A María Alicia Tsuchiya Enriques y Karla Gabriela Moreno Duarte, quienes me dieron su amistad, su confianza, y apreciaron mi trabajo, lo que me permitió obtener una parte esencial de los recursos que necesité para mi sustento durante las etapas más austeras y difíciles durante el desarrollo de esta tesis.

Al Dr. Daniel Hernández del área de traumatología en el Hospital Regional Universita-

rio en Colima, y su equipo de estudiantes, que me trataron de forma humana, responsable, oportuna, y eficaz. Ellos evitaron que perdiera mi brazo izquierdo por una negligencia médica, la cual me obligó a interrumpir mis estudios de maestría. Gracias a ellos pude regresar para finalizar mi trabajo de investigación.

Al Centro de Investigación Científica y de Educación Superior de Ensenada, por otorgarme los recursos y la infraestructura para realizar mis estudios y mi trabajo de tesis.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por brindarme el apoyo económico para realizar mis estudios de maestría.

Gracias a Dios por poner en mi camino a todas estas personas, sin ellas, las dificultades que surgieron en esta etapa de mi vida seguramente me hubieran vencido. A todos ellos y a Dios, les doy las gracias de todo corazón.



# Tabla de contenido

	Página
<b>Resumen en español</b>	<b>v</b>
<b>Resumen en inglés</b>	<b>vii</b>
<b>Dedicatoria</b>	<b>ix</b>
<b>Agradecimientos</b>	<b>xi</b>
<b>Lista de figuras</b>	<b>xix</b>
<b>Lista de tablas</b>	<b>xxvii</b>
<b>Capítulo 1. Introducción</b>	<b>1</b>
1.1 Planteamiento del problema . . . . .	2
1.2 Objetivo general . . . . .	3
1.3 Objetivos específicos . . . . .	3
1.4 Alcance de la tesis . . . . .	4
1.5 Aportaciones . . . . .	4
1.6 Organización de la tesis . . . . .	4
<b>Capítulo 2. Estado del arte</b>	<b>7</b>
2.1 El sistema visual . . . . .	7
2.2 La atención visual . . . . .	9
2.3 Modelo de atención visual . . . . .	10
2.4 Computación evolutiva . . . . .	15
2.5 Programación de cerebros artificiales . . . . .	17
<b>Capítulo 3. Representación artificial de la corteza visual</b>	<b>21</b>
3.1 La corteza visual . . . . .	21
3.2 Modelos artificiales . . . . .	24
3.2.1 Ruta dorsal artificial . . . . .	24
3.2.2 Algoritmos de la programación cerebral de la ruta dorsal artificial	26
3.2.3 Corteza visual artificial . . . . .	41
3.2.4 Algoritmos de la programación cerebral de la corteza visual artificial . . . . .	42
<b>Capítulo 4. Potencialidades</b>	<b>51</b>
4.1 Origen . . . . .	51
4.2 Usos en la inteligencia artificial . . . . .	53
<b>Capítulo 5. La profundidad</b>	<b>57</b>
5.1 Sensores RGB-D . . . . .	58
5.2 Kinect . . . . .	61
5.3 Los sensores RGB-D en la visión artificial . . . . .	66



<b>Capítulo 6. Paradigma de potencialidades en función de la profundidad</b>	<b>69</b>
6.1 Modelo de FOA-DA y algoritmos . . . . .	72
6.1.1 Modelo de ADS para FOA . . . . .	72
6.1.2 FOA-LDA y FOA-HDA . . . . .	73
6.1.3 Algoritmo para el FOA-LDA . . . . .	74
6.1.4 Cálculo de prominencia en función de la máscara de profundidad	80
6.1.5 Algoritmo para el FOA-HDA . . . . .	82
6.1.6 Diagrama general de FOA-DA . . . . .	83
<b>Capítulo 7. Hacia un aprendizaje automatizado</b>	<b>87</b>
7.1 Idea principal . . . . .	88
7.2 Justificación . . . . .	89
7.3 Montaje . . . . .	89
7.4 Descripción del procedimiento . . . . .	90
7.4.1 Imágenes de entrenamiento . . . . .	90
7.5 Algoritmo del experimento . . . . .	93
7.5.1 Algoritmo de entrenamiento con base de imágenes . . . . .	93
7.5.2 Algoritmo de entrenamiento en vivo . . . . .	94
7.6 Ventajas y limitaciones . . . . .	94
<b>Capítulo 8. Experimentos y resultados</b>	<b>97</b>
8.1 Experimentos realizados . . . . .	97
8.2 Resultados de entrenamiento con base de imágenes . . . . .	98
8.2.1 FOA . . . . .	98
8.2.2 FOA-LDA . . . . .	109
8.2.3 FOA-HDA . . . . .	116
8.3 Resultados de entrenamiento en vivo . . . . .	123
8.3.1 FOA en vivo . . . . .	123
8.3.2 FOA-LDA en vivo . . . . .	130
8.3.3 FOA-HDA en vivo . . . . .	137
8.4 Comparación de resultados . . . . .	144
8.4.1 Con base de imágenes . . . . .	144
8.4.1.1 Progreso de la aptitud . . . . .	144
8.4.1.2 Uso de las dimensiones . . . . .	145
8.4.1.3 Desempeño de detección . . . . .	149
8.4.1.4 Composición del proto-objeto . . . . .	150
8.4.1.5 Punto de FOA . . . . .	151
8.4.2 Entrenamiento en vivo . . . . .	152
8.4.3 Tiempos de ejecución . . . . .	154
<b>Capítulo 9. Conclusiones</b>	<b>157</b>
<b>Capítulo 10. Trabajo a futuro</b>	<b>161</b>
<b>Lista de referencias bibliográficas</b>	<b>163</b>
<b>Apéndices</b>	<b>167</b>
A Biblioteca libfreenect . . . . .	167

B	Funcionamiento entre Kinect y MATLAB . . . . .	169
C	Proceso de validación . . . . .	171
	C.1 Validación cruzada . . . . .	171
	C.2 Tipo de muestreo . . . . .	172
D	Ontologías . . . . .	175
E	Glosario de funciones y terminales . . . . .	177



## Lista de figuras

Figura		Página
1	La corteza visual (Clemente Torres, 2015). . . . .	21
2	Diagrama de contexto: programación cerebral de la ruta dorsal artificial. . .	25
3	Representación de un individuo. . . . .	31
4	La pirámide, producto de una imagen cambiada de tamaño por una sucesión de veces. Para el ejemplo de la figura, se muestra una pirámide de 6 niveles. . . . .	40
5	(A) Proyección y recepción de infrarroja. (B) La proyección de luz infrarroja sobre el objeto crea una sombra. (C) Por el cambio de ángulo, la sombra queda en el campo del receptor. (D) Como resultado, habrá puntos en la imagen de distancia que no tendrán dicha información debido a esta sombra.	61
6	En la parte superior se observa la imagen de color, y la de profundidad (escala de grises) sin empatado. En la parte inferior se puede observar que la imagen de profundidad ha sido empatada con la de color, esto por medio del registro de fábrica del Kinect; de esta manera los objetos representados en color y en profundidad coinciden. Las líneas de guía muestran la alineación vertical y horizontal. Se debe notar que la imagen de profundidad empatada reduce considerablemente las sombras de la luz infrarroja (Figura 5). . . . .	63
7	Se puede observar como la distancia de los objetos va perdiendo precisión mientras más alejados estén de la cámara. Fuente: Crock (2011). . . . .	64
8	Adaptador de Kinect a USB. . . . .	65
9	Montura de Kinect de Xbox 360. . . . .	65
10	Diagrama de contexto de la ADS. Describe el proceso “5” del diagrama de la Figura 2 en la Sección 3.2.1. . . . .	73
11	Diagrama de contexto de FOA-LDA. . . . .	75
12	Diagrama de contexto de la función “Spread”/“Estimate_Shape”. . . . .	76
13	Cálculo de prominencia en profundidad general. . . . .	78
14	Diagrama de contexto de FOA-HDA. . . . .	83
15	Sistema general. . . . .	84
16	Diagrama de contexto del sistema general. . . . .	85
17	Base Giratoria para fotografía. . . . .	92
18	Arriba: Imagen RGB. Abajo: Imagen binaria de entrenamiento. . . . .	92
19	Evolución promedio en 30 ejecuciones para FOA entrenados con base de imágenes. . . . .	98

Figura	Página
20	Cantidad de uso de funciones de color en 30 individuos de FOA entrenados con base de imágenes. . . . . 99
21	Cantidad de uso de terminales de color en 30 individuos de FOA entrenados con base de imágenes. . . . . 100
22	Cantidad de uso de funciones de orientación en 30 individuos de FOA entrenados con base de imágenes. . . . . 100
23	Cantidad de uso de terminales de orientación en 30 individuos de FOA entrenados con base de imágenes. . . . . 101
24	Cantidad de uso de funciones de forma en 30 individuos de FOA entrenados con base de imágenes. . . . . 101
25	Cantidad de uso de terminales de forma en 30 individuos de FOA entrenados con base de imágenes. . . . . 102
26	Cantidad de uso de funciones de FI en 30 individuos de FOA entrenados con base de imágenes. . . . . 102
27	Cantidad de uso de terminales de FI en 30 individuos de FOA entrenados con base de imágenes. . . . . 103
28	Clasificación de pixeles en la imagen del proto-objeto. . . . . 104
29	Etapa de prueba para FOA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada. . . . . 105
30	Etapa de prueba para FOA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos del proto-objeto para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada. . . . . 106
31	Estadística general de la etapa de prueba para 30 individuos de FOA entrenados con base de imágenes. . . . . 107
32	Etapa de prueba para FOA entrenado con base de imágenes - Certeza de punto de foco de atención para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada. . . . . 108
33	Evolución promedio en 30 ejecuciones para FOA-LDA entrenados con base de imágenes. . . . . 109
34	Cantidad de uso de funciones de color en 30 individuos de FOA-LDA entrenados con base de imágenes. . . . . 110
35	Cantidad de uso de terminales de color en 30 individuos de FOA-LDA entrenados con base de imágenes. . . . . 110

Figura	Página
36	Cantidad de uso de funciones de orientación en 30 individuos de FOA-LDA entrenados con base de imágenes. . . . . 111
37	Cantidad de uso de terminales de orientación en 30 individuos de FOA-LDA entrenados con base de imágenes. . . . . 111
38	Cantidad de uso de funciones de forma en 30 individuos de FOA-LDA entrenados con base de imágenes. . . . . 112
39	Cantidad de uso de terminales de forma en 30 individuos de FOA-LDA entrenados con base de imágenes. . . . . 112
40	Cantidad de uso de funciones de FI en 30 individuos de FOA-LDA entrenados con base de imágenes. . . . . 113
41	Cantidad de uso de terminales de FI en 30 individuos de FOA-LDA entrenados con base de imágenes. . . . . 113
42	Etapa de prueba para FOA-LDA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada. . . . . 114
43	Etapa de prueba para FOA-LDA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos del proto-objeto para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada. . . . . 114
44	Estadística general de la etapa de prueba para 30 individuos de FOA-LDA entrenados con base de imágenes. . . . . 115
45	Etapa de prueba para FOA-LDA entrenado con base de imágenes - Certeza de punto de foco de atención para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada. . . . . 115
46	Evolución promedio en 30 ejecuciones para FOA-HDA entrenados con base de imágenes. . . . . 116
47	Cantidad de uso de funciones de color en 30 individuos de FOA-HDA entrenados con base de imágenes. . . . . 117
48	Cantidad de uso de terminales de color en 30 individuos de FOA-HDA entrenados con base de imágenes. . . . . 117
49	Cantidad de uso de funciones de orientación en 30 individuos de FOA-HDA entrenados con base de imágenes. . . . . 118
50	Cantidad de uso de terminales de orientación en 30 individuos de FOA-HDA entrenados con base de imágenes. . . . . 118

Figura	Página
51	Cantidad de uso de funciones de forma en 30 individuos de FOA-HDA entrenados con base de imágenes. . . . . 119
52	Cantidad de uso de terminales de forma en 30 individuos de FOA-HDA entrenados con base de imágenes. . . . . 119
53	Cantidad de uso de funciones de FI en 30 individuos de FOA-HDA entrenados con base de imágenes. . . . . 120
54	Cantidad de uso de terminales de FI en 30 individuos de FOA-HDA entrenados con base de imágenes. . . . . 120
55	Etapa de prueba para FOA-HDA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada. . . . . 121
56	Etapa de prueba para FOA-HDA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos del proto-objeto para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada. . . . 121
57	Estadística general de la etapa de prueba para 30 individuos de FOA-HDA entrenados con base de imágenes. . . . . 122
58	Etapa de prueba para FOA-HDA entrenado con base de imágenes - Certeza de punto de foco de atención para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada. . . . . 122
59	Evolución promedio en 5 ejecuciones para FOA entrenados en vivo. . . . . 123
60	Cantidad de uso de funciones de color en 5 individuos de FOA entrenados en vivo. . . . . 124
61	Cantidad de uso de terminales de color en 5 individuos de FOA entrenados en vivo. . . . . 124
62	Cantidad de uso de funciones de orientación en 5 individuos de FOA entrenados en vivo. . . . . 125
63	Cantidad de uso de terminales de orientación en 5 individuos de FOA entrenados en vivo. . . . . 125
64	Cantidad de uso de funciones de forma en 5 individuos de FOA entrenados en vivo. . . . . 126
65	Cantidad de uso de terminales de forma en 5 individuos de FOA entrenados en vivo. . . . . 126
66	Cantidad de uso de funciones de FI en 5 individuos de FOA entrenados en vivo. . . . . 127

Figura	Página	
67	Cantidad de uso de terminales de FI en 5 individuos de FOA entrenados en vivo. . . . .	127
68	Etapa de prueba para FOA entrenado en vivo - Promedio de porcentaje de positivos y negativos para 5 individuos que utilizan el mismo grupo imágenes de prueba. . . . .	128
69	Etapa de prueba para FOA entrenado en vivo - Promedio de porcentaje de positivos y negativos del proto-objeto para 5 individuos que utilizan el mismo grupo imágenes de prueba. . . . .	128
70	Estadística general de la etapa de prueba para 5 individuos de FOA entrenados en vivo. . . . .	129
71	Etapa de prueba para FOA entrenado en vivo - Certeza de punto de foco de atención para 5 individuos que utilizan el mismo grupo imágenes de prueba.	129
72	Evolución promedio en 5 ejecuciones para FOA-LDA entrenados en vivo. .	130
73	Cantidad de uso de funciones de color en 5 individuos de FOA-LDA entrenados en vivo. . . . .	131
74	Cantidad de uso de terminales de color en 5 individuos de FOA-LDA entrenados en vivo. . . . .	131
75	Cantidad de uso de funciones de orientación en 5 individuos de FOA-LDA entrenados en vivo. . . . .	132
76	Cantidad de uso de terminales de orientación en 5 individuos de FOA-LDA entrenados en vivo. . . . .	132
77	Cantidad de uso de funciones de forma en 5 individuos de FOA-LDA entrenados en vivo. . . . .	133
78	Cantidad de uso de terminales de forma en 5 individuos de FOA-LDA entrenados en vivo. . . . .	133
79	Cantidad de uso de funciones de FI en 5 individuos de FOA-LDA entrenados en vivo. . . . .	134
80	Cantidad de uso de terminales de FI en 5 individuos de FOA-LDA entrenados en vivo. . . . .	134
81	Etapa de prueba para FOA-LDA entrenado en vivo - Promedio de porcentaje de positivos y negativos para 5 individuos que utilizan el mismo grupo imágenes de prueba. . . . .	135
82	Etapa de prueba para FOA-LDA entrenado en vivo - Promedio de porcentaje de positivos y negativos del proto-objeto para 5 individuos que utilizan el mismo grupo imágenes de prueba. . . . .	135



Figura	Página
83 Estadística general de la etapa de prueba para 5 individuos de FOA-LDA entrenados en vivo. . . . .	136
84 Etapa de prueba para FOA-LDA entrenado en vivo - Certeza de punto de foco de atención para 5 individuos que utilizan el mismo grupo imágenes de prueba. . . . .	136
85 Evolución promedio en 5 ejecuciones para FOA-HDA entrenados en vivo. . . . .	137
86 Cantidad de uso de funciones de color en 5 individuos de FOA-HDA entrenados en vivo. . . . .	138
87 Cantidad de uso de terminales de color en 5 individuos de FOA-HDA entrenados en vivo. . . . .	138
88 Cantidad de uso de funciones de orientación en 5 individuos de FOA-HDA entrenados en vivo. . . . .	139
89 Cantidad de uso de terminales de orientación en 5 individuos de FOA-HDA entrenados en vivo. . . . .	139
90 Cantidad de uso de funciones de forma en 5 individuos de FOA-HDA entrenados en vivo. . . . .	140
91 Cantidad de uso de terminales de forma en 5 individuos de FOA-HDA entrenados en vivo. . . . .	140
92 Cantidad de uso de funciones de FI en 5 individuos de FOA-HDA entrenados en vivo. . . . .	141
93 Cantidad de uso de terminales de FI en 5 individuos de FOA-HDA entrenados en vivo. . . . .	141
94 Etapa de prueba para FOA-HDA entrenado en vivo - Promedio de porcentaje de positivos y negativos para 5 individuos que utilizan el mismo grupo imágenes de prueba. . . . .	142
95 Etapa de prueba para FOA-HDA entrenado en vivo - Promedio de porcentaje de positivos y negativos del proto-objeto para 5 individuos que utilizan el mismo grupo imágenes de prueba. . . . .	142
96 Estadística general de la etapa de prueba para 5 individuos de FOA-HDA entrenados en vivo. . . . .	143
97 Etapa de prueba para FOA-HDA entrenado en vivo - Certeza de punto de foco de atención para 5 individuos que utilizan el mismo grupo imágenes de prueba. . . . .	143
98 Se fuerza al programa a buscar una manera de diferenciar correctamente un objeto de otro a pesar de la similitud en color. . . . .	148

Figura	Página
99 Validación cruzada. Construcción de conjuntos de entrenamiento y prueba a partir de un conjunto de datos. . . . .	172
100 El objeto se encuentra encima de una plataforma que lo hace girar. Las flechas que provienen del centro muestran hacia donde mira la misma cara del objeto en cada estrato. A cada estrato se le asigna una letra que servirá para ordenar las imágenes posteriormente. . . . .	173



## Lista de tablas

Tabla		Página
1	Lista de Experimentos. . . . .	97
2	Resultados del K-Fold para FOA entrenado con base de imágenes. .	108
3	Resultados del K-Fold para FOA-LDA entrenado con base de imágenes. . . . .	109
4	Resultados del K-Fold para FOA-HDA entrenado con base de imágenes. . . . .	116
5	Resultados de cinco ejecuciones para FOA en vivo. . . . .	130
6	Resultados de cinco ejecuciones para FOA-LDA en vivo. . . . .	137
7	Resultados de cinco ejecuciones para FOA-HDA en vivo. . . . .	144
8	Comparación de aptitud en la etapa de entrenamiento (promedio inicial, promedio final, y mejor individuo) y la etapa de prueba para el entrenamiento con base de imágenes. . . . .	145
9	Comparación de aptitud en la etapa de entrenamiento (promedio inicial, promedio final, y mejor individuo) y la etapa de prueba para el entrenamiento en vivo. . . . .	152



# Capítulo 1. Introducción

---

Para el ser humano, así como para la gran mayoría de los seres vivos que pueden ver, el identificar un objeto en un determinado ambiente por medio de un proceso visual es una tarea que se lleva a cabo de manera rápida, eficiente y eficaz. Y con justa razón, pues el cumplir con esta tarea es un asunto de supervivencia. A pesar de la evidente facilidad con la que el ser humano puede detectar e identificar una herramienta, un vehículo, o a otros seres vivos, el poder comprender y recrear este proceso ha resultado en un problema extraordinario.

Los científicos dedicados a la Visión Artificial (CV<sup>1</sup>), subcampo de la Inteligencia Artificial (AI<sup>2</sup>), han realizado una variedad de investigaciones que le han otorgado a las computadoras, robots, y otros sistemas artificiales, la capacidad de emular ciertas funciones de la visión, acercándose lenta pero seguramente a la descripción del proceso visual.

La visión es un proceso muy complejo; para poder entender cómo funciona, los científicos en CV estudian las investigaciones realizadas en otros campos como la neurociencia y la psicología. La neurociencia permite indagar en la configuración biológica del cerebro tratando de explicar cómo está conectado entre sí, cómo realiza sus funciones y cómo interactúa con el ambiente que lo rodea. La psicología trata de explicar la conducta de los seres y trata conceptos de gran utilidad para la CV como la percepción, la atención, el pensamiento, y la inteligencia, entre otros.

Como resultado de la combinación de estas áreas han surgido ideas o paradigmas, tales como el proceso de *foco de atención*, o la percepción de las *potencialidades* de los objetos y el ambiente a través de la vista.

El trabajo de investigación propuesto implementa una combinación de estos dos paradigmas para explorar sus efectos sobre el sistema visual.

---

<sup>1</sup>Del Inglés: Computer Vision.

<sup>2</sup>Del Inglés: Artificial Intelligence.

## 1.1. Planteamiento del problema

Las investigaciones realizadas sobre el funcionamiento de la corteza visual muestran que esta descompone la imagen que entra por los ojos en distintas dimensiones. No se sabe con exactitud cómo realiza estas funciones ni cuáles son las dimensiones que utiliza de manera específica. Han sido múltiples las investigaciones sobre este tema en psicología y neurociencias, y en las últimas décadas se ha introducido la IA, en específico la Visión Artificial, para ayudar en el desarrollo de tales investigaciones y probar las distintas teorías que surgen sobre el tema.

Dozal *et al.* (2014) y Clemente *et al.* (2012) han desarrollado representaciones artificiales de las rutas dorsal y ventral, componentes de la corteza visual. Hasta el momento integran las características de color, forma, orientación e intensidad. Además, con la introducción de las cámaras RGB-D, como el Kinect, ha surgido un interés en la comunidad de visión artificial por explorar sus capacidades, otorgando información de la dimensión de profundidad a distintos modelos de visión, como por ejemplo, al de la atención visual.

Junto con lo anterior, también existen investigaciones recientes que intentan abordar el tema de la visión desde el concepto de potencialidades, un paradigma de psicología existente desde la década de los 70, y que propone un enfoque diferente a la manera en que el ser humano, y en general los organismos que usan la visión, realizan la identificación de los objetos que los rodean, esto con gran versatilidad.

Por tanto, para el presente trabajo de tesis se pretende usar como base el algoritmo de la ruta dorsal artificial (Dozal *et al.*, 2014) para continuar con su desarrollo, y experimentar las capacidades de este sistema al ser modificado para que utilice la información de profundidad. Para construir un algoritmo que lleve a cabo este trabajo, se considera utilizar la idea de las potencialidades desde un particular punto de vista, que permita darle un propósito a la integración de la información de profundidad.

Adicionalmente, los resultados que han arrojado los experimentos hechos con la ruta dorsal artificial, y su capacidad para encontrar buenas soluciones, motivan a preguntarse si sería posible implementar dicho sistema en una situación de aprendizaje que simule al aprendizaje natural, es decir, sin la necesidad de grandes bases de imágenes, únicamen-

te las imágenes capturadas en el momento de ejecución.

## 1.2. Objetivo general

En este trabajo de tesis se modificará el algoritmo de la Ruta Dorsal Artificial de tal manera que integre a su funcionamiento una nueva dimensión, la profundidad, pero explorando esta integración desde el concepto de las potencialidades, el cual es de interés reciente (Myers *et al.*, 2014; Tünnermann y Mertsching, 2014; Kjellström *et al.*, 2011).

Además se desea construir un procedimiento que permita realizar un aprendizaje de manera automática y similar a la forma en que un ser humano aprende a detectar objetos por medio de la vista.

## 1.3. Objetivos específicos

Para lograr los objetivos generales, deberemos cumplir con los siguientes objetivos específicos:

- Comprender el concepto de potencialidades, y buscar una manera de que funcione en base a la profundidad.
- Analizar si es necesario integrar la dimensión de distancia al proceso de aprendizaje evolutivo de la ruta dorsal artificial.
- Desarrollar un método que permita disminuir la intervención humana en el aprendizaje de la ruta dorsal artificial.
- Definir el material necesario para llevar a cabo tal método de aprendizaje y conseguir ese material.
- Una vez integrados los conceptos deseados, realizar las pruebas comparando el funcionamiento de la ruta dorsal artificial, bajo los nuevos parámetros establecidos, con el modelo que surgirá de combinar la dicho sistema con la profundidad y las potencialidades.
- Registrar los resultados y hacer el análisis correspondiente.



#### **1.4. Alcance de la tesis**

Debido a que es necesario hacer pruebas de la ADS y su nueva versión bajo los mismos parámetros, se considera usar sólo un objeto para realizar las pruebas y comparaciones.

#### **1.5. Aportaciones**

Se incursiona más en el funcionamiento de la ADS y sus capacidades con una nueva dimensión, la profundidad, comparándola con su versión previa. Es interesante desde el punto de vista biológico, ya que la profundidad otorga información muy importante para las tareas visomotoras, tareas que corresponden a la ruta dorsal. Por lo tanto, al agregar esta nueva dimensión, se estaría completando aún más la emulación que se busca hacer de esta zona de la corteza visual.

También se explora el concepto de las potencialidades, y se analizará su utilidad y versatilidad en el área de visión artificial.

Con los recientes avances en robótica y sistemas de inteligencia artificial, una metodología que permitiera realizar un entrenamiento de manera automática sería un aporte interesante para la comunidad en IA. Los métodos de enseñanza siempre han requerido una gran parte del tiempo de los expertos. Con este trabajo se quiere explorar la posibilidad de entrenar los sistemas de IA de manera que la intervención humana sea la menor posible, aprovechando mejor los recursos computacionales y el tiempo de los investigadores.

#### **1.6. Organización de la tesis**

El Capítulo 2, el estado del arte, revisa los conceptos que permitieron la construcción de las rutas dorsal y ventral artificiales. Debido a que este trabajo se centra más en la ADS, se explica el tema de la atención visual. También se ven las herramientas y procedimientos con los cuales fueron creadas las rutas artificiales.

El Capítulo 3 trata sobre la corteza visual artificial, centrándose en su mayoría en la ruta dorsal artificial. También se detallan los algoritmos principales que forman estos

sistemas.

El Capítulo 4 explica el tema de las potencialidades. Se detalla su origen y cómo se ha ido introduciendo en la inteligencia artificial, junto con ejemplos.

El Capítulo 5 trata sobre el tema de la profundidad, y cómo funciona esta para los sistemas naturales. Después se explica el funcionamiento de las cámaras RGB-D; con este tipo de cámaras se obtendrá la información de profundidad para proporcionársela al programa de la ruta dorsal artificial. Finalmente se verán algunos ejemplos del uso de este tipo de cámaras en investigaciones de visión artificial.

En el Capítulo 6, una vez tratados los temas de profundidad y potencialidades, se analiza el tema central del presente trabajo de investigación: el paradigma de potencialidades en función de la distancia. Junto con ello, se presentan los algoritmos propuestos para integrar la ruta dorsal con la información de profundidad por medio de las potencialidades.

El Capítulo 7 trata sobre cómo se propone realizar el aprendizaje automatizado. En este se ven todas las herramientas necesarias, y el procedimiento general.

En el Capítulo 8 se presenta la estadística de los resultados de los seis tipos de experimentos realizados, y se hace un análisis de estos.

En el Capítulo 9 se exponen las conclusiones, y finalmente en el Capítulo 10 se expone el trabajo a futuro.

En la sección de apéndices se encuentran las fuentes para instalar el software necesario. Se introduce al tema de ontologías, un concepto que es de interés para trabajos futuros. Se explica el método de la verificación cruzada y la forma en que se realizó el muestreo para realizar dicho método. Finalmente, se muestra un glosario donde se definen las funciones y terminales usadas por el sistema.



## Capítulo 2. Estado del arte

---

### 2.1. El sistema visual

El sistema visual humano puede ser dividido en dos partes: Una es la *corteza visual*; la otra son *los ojos*.

La *corteza visual* es la parte del cerebro que recibe la información de la imagen y la procesa para cumplir funciones como detección, reconocimiento de objetos, seguimiento, entre otras. Está dividida en dos áreas o rutas llamadas *ruta ventral* (VS<sup>1</sup>) y *ruta dorsal* (DS<sup>2</sup>) (Olague *et al.*, 2014).

La ruta ventral, o *ruta del qué*, ha sido definida como el área donde se llevan a cabo los procesos que permiten el reconocimiento de los objetos o estímulos visuales que se observan. La ruta dorsal, o *ruta del dónde o cómo*, es el área que está relacionada con la localización en el espacio y la dirección de la atención hacia las zonas de interés en la escena (Itti y Koch, 2001). Estas dos áreas trabajan en conjunto para procesar la imagen entregada por los ojos y llevar a cabo las distintas funciones visuales que servirán posteriormente a otras zonas como el sistema motriz, la memoria, o el sistema de aprendizaje.

La otra parte del sistema visual son los ojos. Estos órganos funcionan como sensores de entrada para el sistema; por medio de ellos se adquiere la imagen de la escena que se observa. Funcionan de la siguiente manera: la luz del ambiente entra a los ojos a través de la *córnea*, la *pupila* y el *crystalino*. Posteriormente atraviesa el *humor vítreo* y finalmente llega a la pared interna del ojo llamada *retina*. La retina está cubierta de células foto-receptoras llamadas *bastones* y *conos*. Los bastones no detectan colores y son muy sensibles a la luz; su función es permitir la visión en ambientes con poca luminosidad. Los conos sí detectan los colores y trabajan con niveles mucho más altos en luminosidad que los bastones. Los conos se dividen en tres clases correspondientes a los colores a los que son sensibles: rojo, verde y azul.

---

<sup>1</sup>Del Inglés: Ventral Stream.

<sup>2</sup>Del Inglés: Dorsal Stream.

La luz que entra al ojo es recibida por los conos y bastones, y son estos los que convierten la imagen recibida, en forma de luz, a un conjunto de señales eléctricas que son enviadas a la corteza visual en el cerebro para su procesamiento. Es importante notar que la imagen reproducida por las células de la retina no tiene la misma *nitidez* en todas sus partes. La razón de esto es debido a la distribución de los conos y bastones.

Cuando se habla de la nitidez de una imagen, se hace referencia al detalle que esta puede representar, y está ampliamente ligado a la resolución de la imagen, es decir, a cuántos puntos existen por unidad de medida. En la retina, cada punto de la imagen es representado por un grupo de conos (rojo, verde, o azul) o un bastón. La forma en que estas células están distribuidas a lo largo y ancho de la retina no es uniforme; existe una zona específica del ojo que posee la mayor densidad de conos y se le llama *fóvea*.

La fóvea es la zona especial en la retina responsable de reproducir la parte más nítida de la imagen. En comparación con el campo visual del ojo, cuyo ángulo es de 200 grados aproximadamente, la fóvea sólo recibe la luz de un ángulo visual de alrededor de 15 grados (Hecht, 2002). A pesar de su tamaño, del total de información que el ojo manda al cerebro, cerca de la mitad pertenece a la fóvea; la otra parte proviene del resto de la retina. Como resultado de esta configuración en su estructura, el ojo puede capturar imágenes en las que sólo una parte pequeña de esta posee gran detalle.

Actividades, como la lectura, requieren de la capacidad de ver con detalle. Debido a que el ojo sólo puede ver con detalle una zona, es necesario que este se mueva para posicionar el área que desea leer dentro del campo de la fóvea. Además de la lectura, una infinidad de actividades visuales requieren la mayor información posible del objeto que observan, por ejemplo, el reconocimiento de rostros, la identificación y seguimiento de objetos, el aprendizaje visual, etc.

Al analizar una imagen o al recibir algún estímulo (e.g. movimientos, cambios súbitos en la imagen), la corteza visual (y en especial la ruta Dorsal) tiene la capacidad de influir en el movimiento del ojo, posicionando a este para que ubique el área de la fóvea en la zona de la imagen que necesita procesar. A este conjunto de acciones se les conoce como *Mecanismo de Atención Visual*.

## 2.2. La atención visual

Se puede definir a la atención visual como el mecanismo por el cual una persona puede seleccionar la información más relevante de la escena en observación; esta relevancia es acorde con el comportamiento en un momento dado. Un mecanismo así es importante para un sistema visual debido a que casi cualquier escena o ambiente visual es complejo y posee mucha más información de la que el cerebro puede procesar (Chun y Wolfe, 2008).

Para tal sobrecarga, el cerebro posee varios mecanismos de atención que cumplen con dos funciones críticas. La primera función es el poder seleccionar la información que sea relevante al comportamiento presente, y a su vez ignorar la información que no sea relevante o que genere interferencia (Chun y Wolfe, 2008). Por ejemplo: Cuando un niño busca su juguete favorito de entre un montón de juguetes, él no inspecciona cada uno de ellos para encontrarlo; lo que hará será guiarse rápidamente por las características visuales de tal juguete como el color o el tamaño, incluso la textura. Si lo que busca es un oso de peluche de entre un montón de cubos de colores, de manera casi inmediata fijará su vista en el oso e ignorará a los demás juguetes que no cumplen con características como el color o el material del que está hecho el objeto buscado: los cubos se verán simétricos, lisos, de colores sólidos, mientras que el oso tendrá un color muy específico, su textura será de tela con diferentes pliegues asimétricos, y tendrá un tamaño mucho mayor que los cubos.

La segunda función tiene como propósito modular, o realzar, la información que se seleccionó; esto de acuerdo al estado y los objetivos del perceptor (Chun y Wolfe, 2008). En el mismo ejemplo del niño y el oso, suponga ahora que tal juguete se mezcla con un conjunto de otros peluches, entre los cuales puede haber incluso otros osos con características similares. Así como en el primer ejemplo, la primera función del sistema de atención visual del niño permite filtrar los peluches que no corresponden a las características del oso. La tarea de la segunda función es más abstracta y relativa, ya que el cerebro realiza varios procesos que “mejoran” las características percibidas por el infante. Por ejemplo, puede realizar un esfuerzo para percibir mejor ciertos colores o texturas. De esta manera, la capacidad de profundizar en el detalle será lo que le permita fijar su

atención en los osos que más se parezcan al que busca.

### 2.3. Modelo de atención visual

Fue en en la década de los 60 que surgieron las primeras descripciones del mecanismo de atención visual, pero no fue sino hasta la década de los 90 que los científicos comenzaron a implementar estas ideas en los sistemas artificiales de visión. Los primeros trabajos para hacer reconocimiento, u otras tareas visuales, consideraban el análisis previo de toda la escena por distintos métodos como segmentación, clasificación y métodos estadísticos. Sin embargo se dieron cuenta que, para hacer estas tareas, un análisis de la totalidad del espacio visual llegaba a ser compleja y de gran carga para los sistemas computacionales. Fue entonces cuando el mecanismo de atención visual comenzó a cobrar fuerza como el camino para realizar tareas visuales en los sistemas artificiales. A partir de entonces distintas investigaciones se han llevado a cabo para modelar el mecanismo de atención visual.

Una de las teorías más influyentes que describen la atención visual es la *Teoría de integración de características*<sup>3</sup> (Treisman y Gelade, 1980). Su idea propone que, antes de identificar un objeto, las diferentes características de una escena son percibidas de manera paralela y automática (características como: color, intensidad, orientación, disparidad, forma, movimiento, etc.). La atención visual tiene entonces como propósito permitir la unión correcta de estas características para construir la representación de un objeto. El concepto pues, está dividido en 3 fases: percepción, atención y finalmente identificación o reconocimiento.

Wolfe (1994) presentó su segunda versión del *modelo de búsqueda guiada*<sup>4</sup>, siendo la primera versión publicada en 1989. Basándose en la propuesta de Treisman y Gelade, Wolfe propone un modelo que busca explicar la forma en que el ser humano puede encontrar los estímulos visuales deseados de una escena común. Para lograr tal propósito el modelo guiará la atención por medio de un valor al cual llama "activación". Así, mientras mayor sea el valor de activación de una zona en la escena procesada, mayor disposición tendrá el modelo para dirigir la atención a tal zona. Se consideran dos tipos de activación:

---

<sup>3</sup>Conocido en Inglés como Feature-Integration Theory.

<sup>4</sup>Conocido en Inglés como Guided Search Model.

de *abajo hacia arriba* y de *arriba hacia abajo*.

La activación *de abajo hacia arriba*<sup>5</sup>, también nombrada como *impulsada por estímulo*<sup>6</sup>, es la medida que define que tan diferente es un elemento del resto de elementos en su vecindario o localidad, *i.e.*, se da un valor al contraste existente de un elemento con sus vecinos para cada una de las dimensiones que lo componen. Wolfe propone una metodología para obtener estos valores. Hay que mencionar que el cálculo de esta medida no es trivial, por lo que pueden existir muchas otras formas de obtenerlo. Sin embargo, la idea principal de esta activación es guiar al foco de atención para que, sin la necesidad de conocimiento previo, se centre sobre el elemento más contrastante en la escena, de ahí el nombre de *impulsada por estímulo*. Hay que ser puntuales en el significado que conlleva la palabra contraste, pues no se refiere únicamente a las diferencias en brillo o intensidad, sino a las diferencias en cada uno de las dimensiones que componen a la escena, e.g., orientación, forma, color, intensidad, disparidad, movimiento.

La activación *de arriba hacia abajo*<sup>7</sup>, también nombrada como *impulsada por usuario*<sup>8</sup>, es la medida que guía al foco de atención hacia objetos que, aunque no sean inherentemente atractivos o conspicuos, son objetos que el usuario busca, es decir, objetos que el usuario conoce previamente y tiene por objetivo encontrar en la escena observada. Este tipo de activación es muy diferente a la impulsada por estímulo, cuya información generalmente le provoca ruido. Con el propósito de guiar la atención a los elementos que posean las características que componen al objeto buscado, la activación por usuario filtra la información obtenida por estímulo. Esto lo logra definiendo las dimensiones o canales que le ayuden para su objetivo.

Los dos tipos de activación son representados en un *mapa de activación*. Abstractamente, se puede pensar en este mapa como un conjunto de valles y montañas en un área del tamaño de la imagen procesada. La atención se guía siempre hacia el punto más alto de este mapa; la topografía de este dependerá del tipo de activación empleada.

Otro trabajo que ha logrado gran influencia en el tema es el *modelo de atención visual*

---

<sup>5</sup>Conocida en Inglés como Bottom-Up Activation.

<sup>6</sup>Conocida en Inglés como Stimulus-Driven.

<sup>7</sup>Conocida en Inglés como Top-Down Activation.

<sup>8</sup>Conocida en Inglés como User-Driven.



*basado en prominencia*<sup>9</sup> (Koch y Ullman, 1987), y posteriormente los algoritmos de dicho modelo fueron desarrollados por Itti *et al.* (1998).

En 1985, Cristof Koch y Shimon Ullman concluyeron que los primates han desarrollado un sistema especializado de procesamiento por foco, el cual se mueve a lo largo de la escena que observan. Propusieron que tal sistema está compuesto de los siguientes elementos:

- *La etapa de representación temprana*, que consiste en un conjunto de mapas topográficos donde, de forma paralela, cada uno representa una característica elemental, como por ejemplo: color, orientación, dirección de movimiento, disparidad, forma, etc.
- *Un mapeo selectivo*, que va de la representación topográfica temprana a una más centralizada y no-topográfica que contenga sólo la información de un área específica, es decir, la zona seleccionada.
- *Las reglas de selección del área*, que determinan cuáles zonas serán incluidas en la representación central. La regla principal determina la zona de foco usando las características destacadas o conspicuas, obtenidas de la representación temprana, y que son seleccionadas por medio de un proceso llamado *el ganador se lleva todo* (WTA)<sup>10</sup>. Al inhibir la zona seleccionada, se resalta la siguiente localidad más sobresaliente. También existen dos reglas adicionales: *de proximidad*, que determina que los cambios de foco se hacen con referencia al vecindario del foco de atención vigente; y *de preferencias por similitud*, que determina que los cambios de foco se hacen con referencia a la semejanza en características elementales del siguiente foco de atención en comparación al vigente.

En 1998 Laurent Itti escribió los algoritmos para representar la propuesta de Koch y Ullman en un modelo computacional. Este modelo está relacionado con la *teoría de integración de características*. Su funcionamiento representa la atención *de abajo hacia arriba* y el procedimiento es altamente paralelizable. También cuenta con una versión extendida

---

<sup>9</sup>Conocido en Inglés como Model of Saliency-Based Visual Attention

<sup>10</sup>Del Inglés: Winner-Take-All. Es un principio computacional aplicado en redes neuronales donde cada neurona compite con las demás para su activación. Sólo la neurona con el valor de activación más alto queda encendida, mientras que el resto se apaga.

para representar la atención *de arriba hacia abajo*; esta usa la información de otras áreas (como el conocimiento) y un sistema de pesos que determina cuales características influyen más en el cálculo de atención.

El modelo está compuesto de las siguientes etapas: la descomposición de la entrada en sus dimensiones, la obtención de los *mapas de características*<sup>11</sup>, la producción de los *mapas de conspicuidad*<sup>12</sup>, el cálculo del *mapa de prominencia*<sup>13</sup>, y finalmente el proceso de WTA y la *inhibición de retorno*.

La entrada consiste de una imagen de color RGB<sup>14</sup> que es procesada para obtener los mapas de tres dimensiones diferentes: intensidad, color y orientación. La dimensión de *intensidad* se obtiene como el promedio de los tres valores (rojo, verde y azul) de la imagen de entrada. La dimensión de *color* se representa por medio de los tres colores que componen a la imagen, es decir, se obtiene un mapa para cada color. Finalmente, la dimensión de *orientación* se consigue procesando a la imagen de entrada con *pirámides de gabor orientadas* en nueve escalas para cuatro orientaciones: 0°, 45°, 90° y 135°.

Los mapas obtenidos son convertidos en *pirámides de Gaussianas* de nueve escalas, *i.e.*, cada uno de los mapas es pasado por un filtro pasa-bajas y un submuestreo que da como resultado la reducción progresiva del tamaño original de la imagen yendo de la escala 1:1 o *escala cero* hasta la escala 1:256 o *escala ocho*. Este conjunto de pirámides son introducidas a un procedimiento llamado *centro-contorno*<sup>15</sup>.

El procedimiento de *centro-contorno* representa la manera en que trabajan las neuronas que construyen el espacio visual; los estímulos fuertes de una pequeña región del espacio visual son inhibidos con los estímulos más débiles y amplios de los alrededores de dicha región. Esto significa que tales conjuntos de neuronas son muy sensibles a los cambios de continuidad del espacio local, permitiendo así, detectar localidades que sobresalen en comparación a sus alrededores. El proceso de *centro-contorno* se imple-

<sup>11</sup> Conocidos en Inglés como Feature Maps

<sup>12</sup> Conocido en Inglés como Conspicuity Maps

<sup>13</sup> Conocido en Inglés como Saliency Map

<sup>14</sup> Del Inglés: Red, Green, Blue. Modelo de color que usa la combinación de diferentes intensidades de los colores primarios Rojo, Verde y Azul para representar cualquier color.

<sup>15</sup> Conocido en Inglés como center-surround.

menta como la diferencia entre la escala fina y la gruesa (más pequeña) de la imagen; esto se hace para varias escalas lo que permite la extracción a multiescala de las características de la imagen. Como resultado de este proceso se obtienen 42 mapas llamados *mapas de características*: seis de intensidad, 12 de color, y 24 de orientación.

Una vez obtenidos los *mapas de características* se procede a combinarlos en *mapas de conspicuidad*. Estos mapas son el resultado de la suma de todas los *mapas de características*, *i.e.*, se reduce cada mapa a la escala cuatro y se suma punto por punto. Como resultado se obtiene un *mapa de conspicuidad* por cada dimensión.

Posteriormente se produce el *mapa de prominencia*. Esto se logra promediando los tres *mapas de conspicuidad* normalizados. El proceso de normalización intensifica la influencia de los mapas que tengan un número pequeño de picos fuertes de actividad, mientras que suprime los mapas que contengan picos más generalizados.

Finalmente el *mapa de prominencia* es introducido a una red neuronal WTA, la cual obtiene el punto más prominente del mapa. Este punto es precisamente el punto del *foco de atención* o FOA<sup>16</sup>. El FOA como tal es representado por un disco cuyo centro es el resultado del WTA y su radio es un sexto de la medida más pequeña entre la altura y la anchura de la imagen de entrada. La razón de que se use un disco para representar al FOA es porque el modelo no posee ninguna referencia o guía, pues es un procedimiento de *abajo hacia arriba*.

El proceso de *inhibición de retorno* permite buscar el siguiente punto más prominente que no pertenezca al FOA anterior, por lo que vuelve a correr la red neuronal WTA pero esta vez con las neuronas que pertenecen al área del FOA apagadas.

Aunque el modelo establece que la combinación de los *mapas de conspicuidad* es un promedio de sus versiones normalizadas, el equivalente biológico (*i.e.*, el método por el cual el cerebro produce un *mapa de prominencia*) es aún desconocido y no trivial. Por esta razón Itti y Koch (2001b) proponen cuatro métodos para realizar esta combinación: la estrategia *ingenua*, en la que todos los *mapas de características* son normalizados entre 0 y 1, y posteriormente sumados para formar el *mapa de prominencia*. La estrategia

---

<sup>16</sup>Del Inglés: Focus Of Attention.

*entrenada*, en la que por medio de entrenamiento se asignan pesos a los *mapas de características*; tales pesos modifican la participación de estos mapas en el *mapa de prominencia*. La estrategia  $N(.)$ , cuyo método es el implementado en el modelo original. Y por último la estrategia de *interacciones localizadas iterativas*, en la que buscan que cada ubicación en el *mapa de características* esté provista de auto-excitación e inhibición inducida por su vecindario, i.e, simula la competencia local entre vecindarios de neuronas que tengan ubicaciones prominentes.

Aún con las cuatro propuestas, Itti y Koch dejan claro que estos métodos pueden no ser los más eficientes para realizar la detección, tanto de los objetivos del estudio como muchos otros nuevos; por ello dejan esta parte como un problema abierto, lo que abre las posibilidades a nuevos métodos e incluso el uso de otras técnicas para diseñar estos métodos.

La descripción del sistema de atención visual es importante debido a que ayuda a entender mejor los mecanismos por los cuales el cerebro humano es tan eficiente en los procesos visuales. Aunque aún queda más investigación por realizar para comprender toda su complejidad, su estudio y los resultados que se han obtenido ya no sólo transitan en las áreas de biología, neurología y psicología, sino que también se explotan en las áreas de tecnología y ciencias computacionales. Esto ha sido muy beneficioso, pues aparte de obtener algoritmos, programas o sistemas que implementen estos conocimientos, también ayudan en el desarrollo del entendimiento de la atención visual, y retroalimenta con sus resultados a otras áreas para que la investigación continúe.

## **2.4. Computación evolutiva**

La *computación evolutiva* es un conjunto de métodos clasificados como de *búsqueda*, *optimización* o *diseño*. Su funcionamiento básico está fundado en la *teoría de evolución* de Charles Robert Darwin en la cual se establece un principio conocido coloquialmente como *la supervivencia del más apto*. Este principio indica que todo los organismos de una población atraviesan por un proceso de selección que determina cuáles individuos son los más indicados para sobrevivir en el medio en el que habitan, y por tanto, estos individuos son los que heredan sus características a las siguientes generaciones. Así se entra en un

ciclo de constante adaptación y competencia que da como resultado individuos cada vez más adaptados a su medio, el cual, también puede sufrir cambios.

En la computación evolutiva se emula este principio con el objetivo de encontrar la solución de un problema. Este problema se considera como *el medio* en el cual competirán un conjunto de individuos. Los *individuos* son propuestas de solución a dicho problema y estos pasarán por un proceso de evolución determinado por las siguientes etapas: inicialización, evaluación, selección, cruce y Mutación (que permiten crear la siguiente generación), y el criterio de paro.

La *inicialización* es la etapa donde se crea la primer población de individuos. Los individuos tienen dos propiedades básicas: la *solución* y su *aptitud*. La solución es creada de forma aleatoria a partir de un conjunto de bloques determinados; a estos bloques se les llama *genes* y al conjunto completo se le llama *acervo genético*. La solución es representada por un *cromosoma*, que a su vez está conformado de un número de genes. Dependiendo del problema a solucionar, el cromosoma puede representar algo sencillo (e.g., números, valores lógicos) o algo complejo (e.g., funciones, programas). La segunda propiedad es un valor que determina qué tan propicio es el individuo para solucionar el problema, o en términos de evolución, que tan apto es para sobrevivir al medio.

Una vez creada la población, se procede a la etapa de *evaluación*. Cada uno de los individuos es evaluado por medio de la *función de aptitud*. Esta función es crítica para el ciclo evolutivo ya que determina la dirección que tomará el proceso de evolución, i.e., la función decide cuáles individuos resuelven “mejor” el problema, lo que define cuáles características tienen mayor posibilidad de pasar a la siguiente generación.

La etapa de *selección* determina, por medio del valor de aptitud, cuáles individuos se usarán para formar descendencia. Aquellos individuos con un valor de aptitud más alto tienen mayor probabilidad de participar en la siguiente etapa: cruce y mutación.

El *cruce* es un procedimiento donde los cromosomas de dos individuos son intercambiados en un punto intermedio. Éste punto es determinado de manera aleatoria, pero siempre cuidando que la estructura de la nueva solución combinada sea congruente. El objetivo de dicho procedimiento es explorar nuevas soluciones basadas en las soluciones

existentes.

La *mutación* es un procedimiento en el que uno de los genes de la solución es cambiado de manera aleatoria por alguno de los genes del acervo genético. El fin de esto es introducir nuevas opciones a las soluciones existentes, explorando así nuevas posibilidades que tal vez no estaban contempladas en la población actual.

Hay muchos caminos para formar la nueva población. Dependiendo del algoritmo evolutivo usado, la nueva generación se puede componer totalmente de los individuos nuevos, o puede ser una combinación de nuevos individuos con viejos individuos basándose en la aptitud de los mismos. Esta nueva generación pasará de nuevo por los procesos de selección, mutación y cruce, formando nuevas generaciones de población. El ciclo finalizará hasta que el criterio de paro se cumpla, por ejemplo, cuando se haya encontrado una solución lo suficientemente satisfactoria, o después de un determinado número de generaciones.

Uno de los algoritmos evolutivos más populares es la *programación genética* (GP<sup>17</sup>). Esta técnica se especializa en desarrollar, por medio del proceso evolutivo, un conjunto de programas con el fin de resolver un problema; por esto, se le considera como un método de *aprendizaje de máquina*. Esta estrategia se deriva de otra muy conocida: los *algoritmos genéticos*. La diferencia entre los dos radica en que, mientras que los algoritmos genéticos tienen cromosomas que representan variables, en la programación genética los cromosomas representan estructuras de árboles, los cuales a su vez representan programas. El proceso evolutivo es similar: se crea una población inicial de programas, se calcula su aptitud, se introducen a un proceso de selección, los individuos seleccionados se cruzan o se mutan, y se forma una nueva población. La operación de cruce se realiza intercambiando las ramas de dos árboles, y la operación de mutación se hace cambiando una de las ramas del árbol por alguna rama construida al azar.

## 2.5. Programación de cerebros artificiales

Cuando Itti y Koch propusieron el modelo del sistema de atención, lo construyeron basándose completamente en el modelo biológico de la corteza visual, i.e, cada función,

---

<sup>17</sup>Del Inglés: Genetic Programming

procedimiento y estructura es una representación de las capas y funciones que forman dicha parte del cerebro. Sin embargo, el cerebro sigue siendo un órgano muy complejo, y aunque las investigaciones han esclarecido cómo está formado y qué zonas cumplen con ciertas funciones, aún no se sabe cómo realiza muchas de estas tareas. Itti y Koch proponen diversos métodos para cumplir estas funciones, pero dejan claro que este es un problema abierto.

En los trabajos de Olague *et al.* (2014) y Dozal *et al.* (2014) se propone la *programación de cerebros* (BP<sup>18</sup>) como una metodología que busca recrear las funciones del cerebro basándose en un modelo biológico, y poniendo especial atención al propósito. Este nuevo paradigma está compuesto de dos partes: un proceso de evolución por medio de la programación genética, y un modelo biológico del cerebro o parte de corteza cerebral.

El modelo biológico introduce la estructura y el procedimiento que sigue alguna zona de la corteza cerebral, sin embargo, existen un conjunto de funciones que, o no están definidas de manera precisa, o simplemente no son estáticas, *i.e.*, dependiendo del tipo de organismo, o incluso de la experiencia o el ambiente de éste, tales funciones pueden cambiar para adaptarse a sus condiciones. Se sabe que los sistemas visuales de varios seres cuentan con las mismas estructuras en el sistema visual, y aún así, sus funciones cambian para adaptarse a su ambiente. Es justo en esta parte en la que se introduce la programación genética.

La programación genética permite crear programas de manera automatizada, y por medio de un proceso de evolución, estos programas se van adaptando al problema en cuestión. En un cerebro artificial, la programación genética permite encontrar varias propuestas de solución a las distintas funciones que la componen. Aunque posiblemente no entreguen respuestas definitivas, como en muchas soluciones encontradas por computación evolutiva, estas suelen tener mejor desempeño que las creadas manualmente.

Olgaue *et al.* son puntuales en aclarar que la metodología no es programación genética en sí, pues el algoritmo evolutivo no define la estructura del cerebro artificial. Los

---

<sup>18</sup>Del Inglés: Brain Programming.

programas que evoluciona no influyen en la estructura del modelo, y evolucionar sin este sería insuficiente para encontrar una solución dado a la gran complejidad que implicaría reconstruir un modelo cerebral completo. La programación genética sirve meramente para crear soluciones a las funciones internas del modelo, y el valor de su aptitud estará medido conforme a los resultados que arrojen al trabajar en conjunto dentro de la estructura cerebral para cumplir con su propósito como un todo.

Dozal *et al.* (2014) y Clemente *et al.* (2012) demuestran la implementación de esta metodología (para el foco de atención y el reconocimiento de objetos, respectivamente) y su gran capacidad, utilizando el modelo propuesto por Itti y Koch para la atención visual.

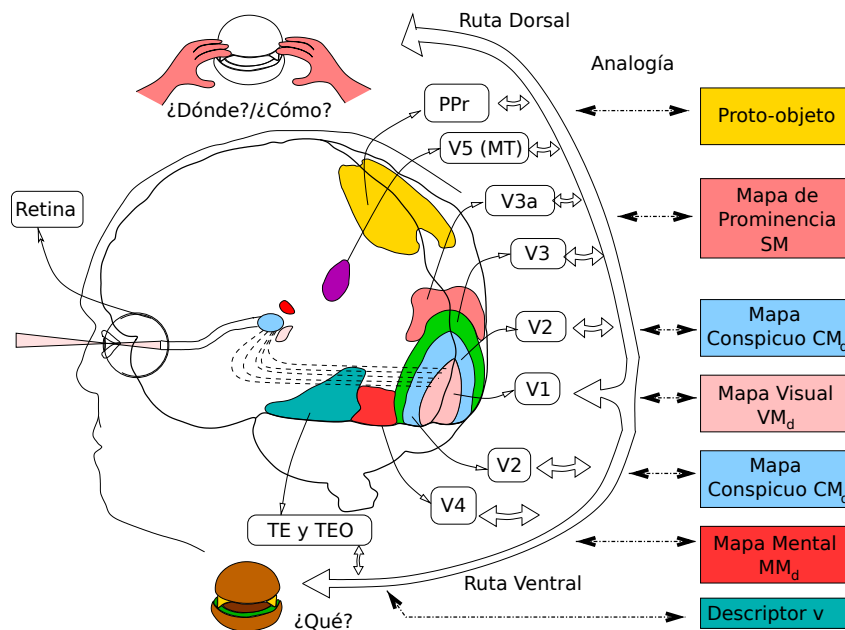




## Capítulo 3. Representación artificial de la corteza visual

### 3.1. La corteza visual

La *corteza visual* (Figura 1) es la parte del cerebro encargada de procesar la información visual. Esta información es obtenida por los ojos y es transformada a través de diferentes funciones para cumplir con distintas tareas visuales (e.g., detección, reconocimiento de objetos, determinación de profundidad) y visomotoras (e.g., centrado de la fovea en el foco de atención, seguimiento de un objeto). La corteza visual está localizada en el lóbulo occipital y abarca los dos hemisferios del cerebro. Su composición fisiológica está definida por un conjunto de zonas o capas a las cuales se les atribuyen diferentes funciones para la transformación de la información visual. Están nombradas como V1<sup>1</sup> o corteza visual primaria, y las áreas extraestriadas V2, V3, V4, y V5/MT<sup>2</sup>.



**Figura 1:** La corteza visual (Clemente Torres, 2015).

La información recibida por el ojo viaja por el nervio óptico hacia el *núcleo geniculado lateral* (LGN<sup>3</sup>) localizado en el *tálamo* en el cerebro. El LGN recibe directamente la información visual y la comunica hacia la corteza visual primaria a través de una vía llamada

<sup>1</sup>Del Inglés: Visual area *N*, donde *N* es un número. Esta definición también aplica para V2, V3, V4 y V5.

<sup>2</sup>Del Inglés: Middle Temporal.

<sup>3</sup>Del Inglés: Lateral Geniculate Nucleus.

*radiación óptica* o *ruta geniculoestriada*. Adicionalmente, el LGN también tiene conexiones por las que recibe información de retroalimentación proveniente de la corteza visual primaria.

La *corteza visual primaria* (V1) es la zona que recibe la información visual directamente desde el LGN. Esta información es mapeada por completo y con una mayor densidad de información dedicada a las señales provenientes de la fovea. Esta capa se especializa en procesar la información de los objetos estáticos y en movimiento, además de que cumple con las funciones de reconocimiento de patrones. La respuesta neuronal de V1 puede identificar cambios en la orientación visual, los colores, y las frecuencias espaciales. Esta área también es sensible a la organización general de la escena, lo cual se logra después de que se retroalimenta con los resultados de las capas siguientes.

La capa V2 o *corteza preestriada* recibe una gran cantidad de información de V1 y a su vez manda información a las capas V3, V4, y V5; también retroalimenta a V1. Así como en V1, las neuronas de V2 también responden a las propiedades de orientación visual, color y frecuencia espacial; sin embargo, también son moduladas por propiedades más complejas, por ejemplo, en monos macacos la capa V2 responde a los contornos ilusorios que resultan de la combinación de orientaciones locales.

La capa V3 es un área más o menos definida que recibe información de V2 y posee neuronas que responden a diferentes combinaciones de estímulos visuales. Algunos estudios sugieren que podría tener un rol en el procesamiento del movimiento global. También se cree que, al igual que las capas anteriores, esta podría tener una representación visual completa. Esto podría significar que la capa abarca un área más grande de lo que actualmente se considera.

Lo que se sabe sobre el área V4 se debe a los estudios realizados en monos macacos. Sin embargo, el área V4 equivalente en los humanos aún es un tema de debate, es decir, existen varias opiniones sobre su estructura física en un cerebro humano, y si esta estructura es también similar a la de los primates no humanos (Goddard *et al.*, 2011). Esta área recibe información desde V2, y en menor medida desde V1. Esta capa se relaciona con la atención selectiva. V4 responde a las mismas características que V1, pero

se diferencia de esta porque se especializa en características medianamente complejas como las formas geométricas, pero no tan complejas como los rostros.

El área V5/MT, localizada en la corteza visual extraestriada, es considerada como la pieza principal en la percepción del movimiento, la integración del movimiento local en referencia al espacio global, y el movimiento de los ojos. V5 recibe información de V1, V2, V3, la *región konioceular* del LGN, y el *pulvinar inferior* (también localizado en el tálamo). La información que sale de V5 se distribuye a lo largo de las áreas ubicadas alrededor. Estas áreas se relacionan con las tareas visomotoras.

Existen diferentes propuestas que describen cómo se organizan estas capas o áreas, incluso agregan o modifican las características de estas. El trabajo presente se basa en el modelo que define a la corteza visual como una composición de dos áreas: la *ruta ventral* y la *ruta dorsal*.

El modelo ventral/dorsal fue descrito por Mishkin *et al.* (1983) que propusieron a estas dos rutas como la ruta del *qué* y del *dónde*, respectivamente; según la propuesta, la ruta del *qué* realiza las tareas de identificación visual, mientras que la ruta del *dónde* cumple con las tareas de localización en el espacio visual. Posteriormente fue expandido por Goodale y Milner (1992), quienes redefinieron las áreas como la ruta del *qué* y del *cómo*; en este modelo se introduce la idea de que la corteza visual, además de cumplir con las tareas de visión de objetos y visión espacial, también realiza las funciones motoras relacionadas a la visión, *i.e.*, la ruta del *qué* define lo que se percibe en la imagen y la ruta del *cómo* procesa la información visual para tener una respuesta de movimiento con respecto a lo observado (e.g., mover la mano para agarrar un objeto, mover los ojos para posicionar la fovea en el foco de atención).

De esta forma, la ruta ventral queda compuesta de la capa V1, V2, V4, y termina en la corteza temporal inferior (cuyas funciones se relacionan con la memoria visual); la ruta dorsal queda conformada por la capa V1, V2, V3, V5, y finaliza en la corteza parietal posterior (la cual cumple con papel de producir el plan de movimientos).

### 3.2. Modelos artificiales

Con el propósito de crear una representación artificial de las dos rutas que componen la corteza visual, Dozal *et al.* (2014) y Clemente *et al.* (2012) se basaron en los algoritmos desarrollados Itti *et al.* (1998), la programación genética, y el modelo ventral/dorsal (combinación que más tarde nombrarían como *programación cerebral*) para construir un modelo que fuera capaz de emular las funciones de las rutas cumpliendo con las tareas de detección y reconocimiento.

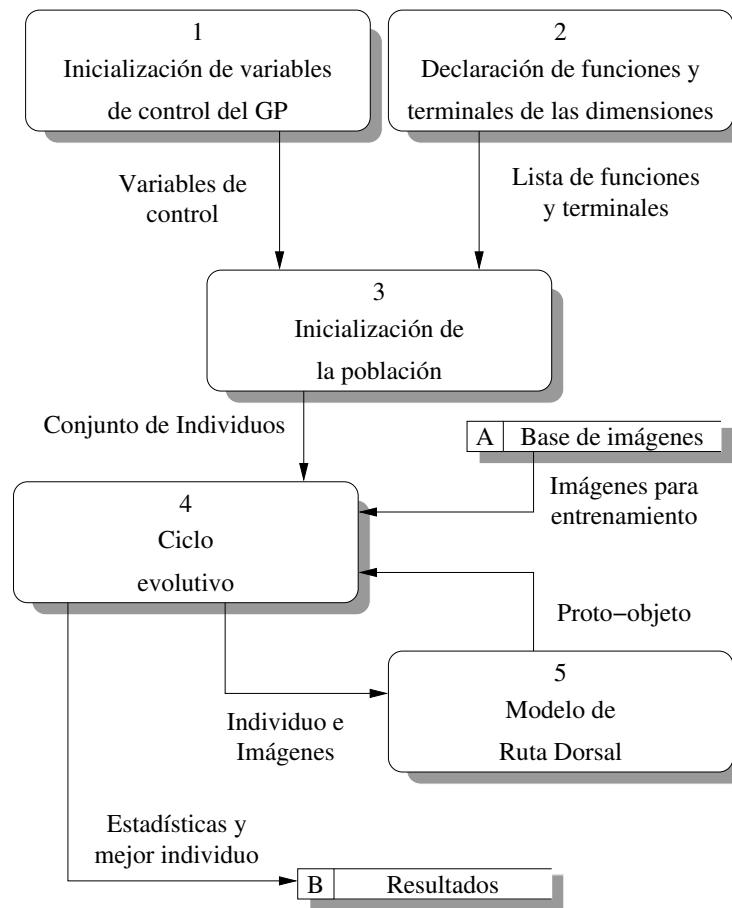
Los dos modelos se componen de dos partes principales:

- El *módulo de aprendizaje*, que usa la programación genética para buscar soluciones que, embebidas en el modelo de una de las rutas de la corteza visual, cumplan con la tarea de detección y reconocimiento de algún objeto o grupo de ellos.
- EL *modelo de la ruta*, que proporciona la base para que el conjunto de funciones, producidas por programación genética, realicen sus tareas en una estructura definida.

#### 3.2.1. Ruta dorsal artificial

La ruta dorsal es la parte de la corteza visual que se encarga de detectar objetos y realizar movimientos en el cuerpo en base a lo detectado. *Detectar* significa que, aunque aun no se identifique al objeto, las características que posee dan suficiente razón como para prestar atención a dicho objeto. Esta decisión es influida por una variedad de aspectos, por ejemplo, cuando un determinado objeto entra en escena, o cuando sus características resaltan del resto de su ambiente por contraste; a este tipo de atención se le llama *de abajo hacia arriba*. Por otro lado, cuando se busca un objeto determinado, la ruta dorsal se predispone para seleccionar ciertas características; como resultado la atención se fijará en objetos parecidos al objetivo que busca, por ejemplo, si se busca una libreta roja es posible que comience por fijar la vista en los objetos rojos, o en los objetos rectangulares y planos, o una combinación de estas y otras características. Este tipo de atención tiene por nombre *de arriba hacia abajo*.

El modelo de *ruta dorsal artificial* (ADS<sup>4</sup>), desarrollado por Dozal *et al.*, busca emular este tipo de comportamiento por medio de un sistema computacional que, basado en el modelo de la ruta dorsal, produce conjuntos de funciones que señalan un objeto en la imagen. Este señalamiento se imprime en una *imagen binaria*<sup>5</sup> en la que los pixeles con valor igual a “1” cubren el objeto. Este conjunto de pixeles representan el *foco de atención*. La Figura 2 muestra el diagrama de flujo de la ruta dorsal con programación cerebral.



**Figura 2:** Diagrama de contexto: programación cerebral de la ruta dorsal artificial.

Se comienza creando el conjunto de variables de inicialización que definen el funcionamiento de la programación genética (GP), e.g., el tamaño de la población, tamaño de las soluciones o individuos, probabilidades de cruce y mutación. Los individuos de la población estarán compuestos de cuatro funciones diferentes, estas funciones son representadas por árboles cuyos *nodos* son representados por un conjunto de funciones y

<sup>4</sup>Del Inglés: Artificial Dorsal Stream.

<sup>5</sup>Una imagen binaria es una imagen cuyos pixeles sólo pueden tener el color *negro* o *blanco*. Estos colores son usualmente representados por los valores numéricos “0” y “1” respectivamente.

terminales. Estas funciones y terminales se definen en listas. Por ello, para este modelo se implementan cuatro pares (funciones-terminales) de listas diferentes.

Se crea un conjunto inicial de individuos y se procede con el ciclo evolutivo de la GP. Así, dicho proceso lee un conjunto de imágenes definidas para el entrenamiento de los individuos. Por medio de la ADS, cada individuo es probado con cada imagen.

El modelo de la ADS, para su funcionamiento, convierte la información de la imagen en cuatro dimensiones: intensidad, color, forma, y orientación. De las cuatro dimensiones, sólo la intensidad tiene un método definido para calcularla. Ver Ecuación (1).

$$I_{x,y} = \frac{R_{x,y} + G_{x,y} + B_{x,y}}{3} \quad (1)$$

Las otras tres dimensiones son calculadas por medio de los métodos construidos por la GP, es decir, los individuos. Adicionalmente, se necesita una cuarta función para el método de integración de características, nombrado como *función F<sup>6</sup>*. Como resultado, se obtiene una imagen binaria llamada proto-objeto que representa al foco de atención. La aptitud del individuo se calcula comparando esta imagen con la respectiva imagen de entrenamiento; mientras más exacto sea el proto-objeto producido, mejor será la aptitud del individuo que lo produjo.

Ya que la GP termina su ciclo, se producen los resultados y se guarda el individuo con mejor aptitud del proceso evolutivo.

### 3.2.2. Algoritmos de la programación cerebral de la ruta dorsal artificial

El trabajo de tesis realizado aquí implica hacer cambios en el funcionamiento del modelo de la ADS, por lo que es necesario detallar su funcionamiento original para poder entender los cambios. En esta sección se presentan y describen los algoritmos, en formato de pseudocódigo, de la programación cerebral de la ruta dorsal artificial de León Dozal *et al.* Dozal *et al.* (2014).

El proceso evolutivo es descrito por el Algoritmo 1.

---

<sup>6</sup>Del inglés: Feature-integration.

---

**Algoritmo 1** Algoritmo de GP - Ciclo Evolutivo
 

---

**Propósito:** Este proceso de programación genética (GP) crea un conjunto de individuos. Cada individuo está compuesto de cuatro funciones distintas. Estas funciones son usadas por la Ruta Dorsal Artificial.

**Entrada:**

- *num\_gen*: Número de ciclos/generaciones en el proceso evolutivo.
- *pop\_size*: Número límite de los individuos de la población.

**Variables:**

- *max\_level*: Número máximo de niveles que cualquier árbol puede tener.
- *max\_trees*: Número de árboles (genes) que un individuo (cromosoma) debe tener.
- *prob\_chrom\_cross*: Número que define la probabilidad de aplicar un Cruce de Cromosoma.
- *prob\_chrom\_mut*: Número que define la probabilidad de aplicar una Mutación de Cromosoma.
- *prob\_gen\_cross*: Número que define la probabilidad de aplicar un Cruce de Genes.
- *prob\_gen\_mut*: Número que define la probabilidad de aplicar una Mutación de Genes.
- *pop*: Un arreglo con todos los individuos de la población actual.
- *offspring<sub>chrom</sub>*: Un arreglo que guarda uno o dos individuos nuevos, producto de algún operador genético aplicado a miembros seleccionados de la población actual.
- *new\_pop*: Un arreglo que contiene los individuos de la nueva generación de población. Tiene el mismo tamaño que *pop*.
- *parents<sub>chrom</sub>*: Un arreglo que guarda los individuos seleccionados a los que se les aplicará alguna operación genética.
- *images<sub>tr</sub>*: Un arreglo de pares de imágenes usadas para el entrenamiento. La primera es una imagen RGB y la segunda es una imagen binaria que representa el proto-objeto de la imagen RGB.
- *fitness*: Un arreglo que guarda los valores de aptitud promedio de cada individuo.
- *fitness<sub>chrom</sub>*: Un arreglo que guarda los valores de aptitud de un solo individuo.
- *f<sub>O</sub>*, *f<sub>C</sub>*, *f<sub>S</sub>*, *f<sub>FI</sub>*: Conjuntos de funciones de Orientación (O), Color (C), Forma (S) e Integración de Características (FI).
- *t<sub>O</sub>*, *t<sub>C</sub>*, *t<sub>S</sub>*, *t<sub>FI</sub>*: Conjuntos de terminales de Orientación (O), Color (C), Forma (S) e Integración de Características (FI).
- *data\_base\_train*: Cadena de caracteres que representa la dirección en disco de la base de imágenes de entrenamiento.

**Salida:**

- *best\_ind*: guarda al individuo con el mejor valor de aptitud.

**Inicio**

- 1: *max\_level*  $\leftarrow$  9
- 2: *max\_trees*  $\leftarrow$  4
- 3: *prob\_chrom\_cross*  $\leftarrow$  0.4
- 4: *prob\_chrom\_mut*  $\leftarrow$  0.1
- 5: *prob\_gen\_cross*  $\leftarrow$  0.4
- 6: *prob\_gen\_mut*  $\leftarrow$  0.1
- 7: *f<sub>O</sub>*  $\leftarrow$  {Funciones de Orientación}



```

8:  $f_C \leftarrow \{\text{Funciones de Color}\}$ 
9:  $f_S \leftarrow \{\text{Funciones de Forma}\}$ 
10:  $f_{FI} \leftarrow \{\text{Funciones de Integración de Características}\}$ 
11:  $t_O \leftarrow \{\text{Terminales de Orientación}\}$ 
12:  $t_C \leftarrow \{\text{Terminales de Color}\}$ 
13:  $t_S \leftarrow \{\text{Terminales de Forma}\}$ 
14:  $t_{FI} \leftarrow \{\text{Terminales de Integración de Características}\}$ 
15:  $pop \leftarrow \text{Init\_Pop}(f_O, f_C, f_S, f_{FI}, t_O, t_C, t_S, t_{FI}, pop\_size, max\_level, max\_trees)$ 
16:  $images_{tr} \leftarrow \text{Load}(data\_base\_train)$ 
17: for  $gen \leftarrow 1$  to  $num\_gen$  step 1 do
18:   if  $gen \neq 1$  then
19:      $parents_{chrom}.\text{Clear}()$ 
20:      $parents_{chrom} \leftarrow \text{Roulette}(pop, fitness, pop\_size)$ 
21:      $offspring_{chrom}.\text{Clear}()$ 
22:     while  $offspring_{chrom}.length < parents_{chrom}.length$  do
23:        $operator \leftarrow \text{Roulette\_Op}(prob\_chrom\_cross, prob\_chrom\_mut, prob\_gen\_cross,$ 
 $prob\_gen\_mut)$ 
24:        $i \leftarrow offspring_{chrom}.length$ 
25:        $offspring_{chrom}.\text{Add}(\text{Apply\_Gen\_Op}(operator, parents_{chrom}, i))$ 
26:     end while
27:      $pop \leftarrow offspring_{chrom}$ 
28:      $pop.\text{Add}(best\_ind)$ 
29:   end if
30:    $fitness \leftarrow 0$ 
31:   for  $i \leftarrow 1$  to  $pop.length$  step 1 do
32:      $fitness_{chrom} \leftarrow 0$ 
33:     for  $j \leftarrow 1$  to  $images_{tr}.length$  step 1 do
34:        $proto \leftarrow \text{ADS}(pop[i], images_{tr}[j, 1])$ 
35:        $fitness_{chrom}[j] \leftarrow \text{Calc\_Proto\_Fitness}(proto, images_{tr}[j, 2])$ 
36:     end for
37:      $fitness[i] \leftarrow \text{Mean}(fitness_{chrom})$ 
38:   end for
39:    $best\_ind \leftarrow \text{Get\_Best}(pop, fitness, best\_ind)$ 
40: end for
41: return  $best\_ind$ 

```

**Fin**

---

Se inicializan las variables de control de la GP (1-6). Se declaran los conjuntos de funciones y terminales de las 3 dimensiones (color, forma y orientación) y de la función de integración de características (7-14). Se crea la primer generación de la población (15). Véase el Algoritmo 2. Se crea la lista de referencia de las imágenes de entrenamiento (16). Se inicia el ciclo del GP (17). Sólo en el primer ciclo, la instrucción condicional **if** es

negativa; la función de ese bloque es crear a una nueva generación a partir de la actual (18-29).

Una vez creados los individuos de *pop*, se procede a calcular su aptitud. Todos los valores de *fitness* son igualados a “0” (30). Se inicia el ciclo para el cálculo de la aptitud (31). Los valores de *fitness<sub>chrom</sub>* son igualados a “0” (32). Se inicia un segundo ciclo (33) que calcula la aptitud del individuo señalado por el índice *i* para cada imagen de la lista *images<sub>tr</sub>*. Para esto, se ejecuta la **ADS** que devuelve un proto-objeto como resultado (34). Este proto-objeto es comparado con la imagen binaria correspondiente (35); el resultado es el valor de aptitud del individuo *i* en la imagen *j*. Los resultados son guardados en *fitness<sub>chrom</sub>* y una vez terminado el ciclo, se calcula el promedio de los valores de aptitud (37). Al final de cada ciclo de la GP, se guarda el individuo con mayor aptitud promedio generado hasta el momento (39).

A partir del segundo ciclo, se crean nuevas generaciones de la población (18-29). Se vacía el arreglo *parents<sub>chrom</sub>* (19) y se le asignan *pop\_size* individuos por medio de una selección de ruleta basada en los valores de aptitud de cada individuo de la población; mientras mayor sea su aptitud, mayor probabilidad tienen de ser seleccionados para ser un padre (20). Ahora se vacía el arreglo *offspring<sub>chrom</sub>* (21) y se inicia el ciclo que genera los nuevos individuos a partir de los individuos padres (22). Primero se elige, por medio de una selección por ruleta, uno de los cuatro operadores genéticos: Cruce de Cromosoma (Algoritmo 3), Mutación de Cromosoma (Algoritmo 4), Cruce de Genes (Algoritmo 5), o Mutación de Genes (Algoritmo 6) cuyas probabilidades de suceso fueron definidas en la inicialización (23). Los operadores genéticos de cruce producen dos individuos hijo usando a dos individuos padre. Los operadores genéticos de mutación producen un individuo hijo a partir de un individuo padre. Estos individuos nuevos son guardados en el arreglo *offspring<sub>chrom</sub>* (25). Una vez producidos todos los individuos de la nueva generación, estos pasan a sustituir a la población anterior (27). Adicionalmente, se agrega a la nueva población el mejor individuo de la generación pasada (28).

El Algoritmo 2 se encarga de crear la población inicial.

---

### Algoritmo 2 Inicialización de Población (Init\_Pop)

---

**Propósito:** Construir un número definido de individuos. Cada individuo se compone de un número de árboles y cada árbol representa una función. Los árboles se construyen por medio de un arreglo aleatorio de funciones y terminales que son seleccionadas de un conjunto definido.

**Entrada:**

- $f_O, f_C, f_S, f_{FI}, t_O, t_C, t_S, t_{FI}$ : Conjuntos de funciones y terminales usadas para construir los árboles que componen a los individuos. Véase Algoritmo 1.
- $pop\_size$ : Número que define el tamaño de la población.
- $max\_level$ : Número que define el nivel máximo de profundidad de los árboles.
- $max\_trees$ : Número que define cuantos árboles son construidos para un individuo.

**Variables:**

- $new\_ind$ : Un arreglo que guarda los árboles construídos para el individuo actual.
- $set_{func}$ : Un arreglo que guarda los conjuntos de funciones.
- $set_{term}$ : Un arreglo que guarda los conjuntos de terminales en el mismo orden que  $set_{func}$ .

**Salida:**

- $pop$ : Un arreglo con  $pop\_size$  individuos.

**Inicio**

```

1:  $set_{func}[1] \leftarrow f_O$ 
2:  $set_{func}[2] \leftarrow f_C$ 
3:  $set_{func}[3] \leftarrow f_S$ 
4:  $set_{func}[4] \leftarrow f_{FI}$ 
5:  $set_{term}[1] \leftarrow t_O$ 
6:  $set_{term}[2] \leftarrow t_C$ 
7:  $set_{term}[3] \leftarrow t_S$ 
8:  $set_{term}[4] \leftarrow t_{FI}$ 
9: for  $i \leftarrow 1$  to  $pop\_size$  step 1 do
10:    $new\_ind.Clear()$ 
11:   for  $t \leftarrow 1$  to  $max\_trees$  step 1 do
12:      $new\_ind[t] \leftarrow new\_tree(set_{func}[t], set_{term}[t], max\_level)$ 
13:   end for
14:    $pop[i] \leftarrow new\_ind$ 
15: end for

```

**Fin**

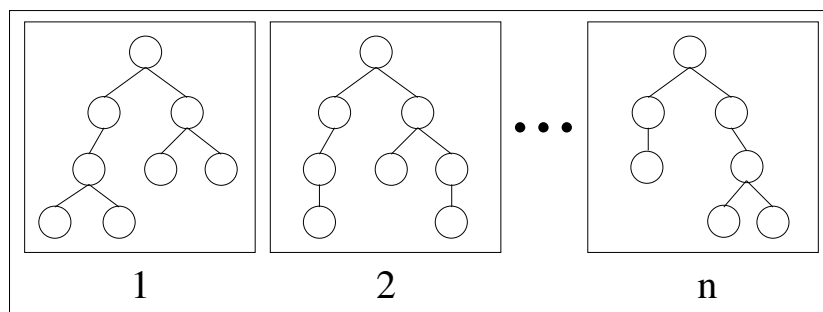
---

Primero, los conjuntos de funciones y terminales son organizados en dos arreglos (1-8). El algoritmo se compone de un ciclo principal en el que se genera cada miembro de la población (9). Para esto, se vacía el arreglo  $new\_ind$  (10) y se entra al segundo ciclo,

en el cual, se construyen los árboles que componen al nuevo individuo (11); esto lo hace por medio del método *Half-and-half*<sup>7</sup> (12). Los individuos creados son guardados en el arreglo *pop* (14).

Como se define en el Algoritmo 1, existen cuatro operadores para esta modalidad de la GP: Cruce de Cromosoma (Algoritmo 3), Mutación de Cromosoma (Algoritmo 4), Cruce de Genes (Algoritmo 5), y Mutación de Genes (Algoritmo 6).

Para entender la diferencia entre las operaciones de cromosomas y de genes, se debe comprender cómo está construido un individuo. Un individuo (Figura 3) está representado como un **arreglo** cuyos objetos componentes son las **estructuras de datos** llamadas *árboles*. A su vez, un árbol está compuesto de estructuras básicas llamadas *nodos*. En el algoritmo, los nodos representan distintos tipos de funciones, valores estáticos y entradas.



**Figura 3:** Representación de un individuo.

Ahora, considere la analogía de que un cromosoma es un arreglo cuyos objetos componentes son unos conjuntos llamados *loci*, y que a su vez, estos conjuntos están compuestos de genes. De esta manera, una operación de cromosomas realiza su procedimiento en un arreglo, que representa al cromosoma, tomando como unidad base a los loci. En cambio, una operación de genes realiza su acción en un árbol, que representa a un locus<sup>8</sup>, tomando como unidad base a los genes, es decir, los nodos.

<sup>7</sup>El método *Rampa Mitad y Mitad* (En Inglés: Ramped Half-and-Half) de J. R. Koza lleva ese nombre debido a que combina dos métodos básicos para la inicialización de árboles en Programación Genética: el *método de crecimiento* (En Inglés: grow) y el *método completo* (En Inglés: full). Dividiendo el número de individuos entre el máximo de niveles del árbol, el método obtiene el siguiente resultado: la mitad de los individuos creados con el método de crecimiento, irregulares y de diferentes profundidades; y la otra mitad con el método completo, cuyos árboles serán más regulares.

<sup>8</sup>Del latín: *Loci* es la forma plural de *locus*.

---

**Algoritmo 3** Cruce de Cromosoma
 

---

**Propósito:** Intercambiar, entre dos cromosomas (padres), una parte de sus loci. Con esto se producen dos nuevos cromosomas (hijos).

**Entrada:**

- $P1, P2$ : Dos arreglos, que representan los cromosomas.

**Variables:**

- $len1, len2$ : Números que indican el tamaño de los cromosomas.
- $min\_len$ : Número que indica el tamaño del cromosoma más pequeño.
- $cp$ : Número que indica el punto de cruce.

**Salida:**

- $O1, O2$ : Dos arreglos, que representan a los nuevos cromosomas, producto de la operación genética.

**Inicio**

```

1:  $len1 \leftarrow P1.length$ 
2:  $len2 \leftarrow P2.length$ 
3:  $min\_len \leftarrow \text{Minimum}(len1, len2)$ 
4:  $O1 \leftarrow P2$ 
5:  $O2 \leftarrow P1$ 
6: if  $len1 = len2$  then
7:    $min\_len \leftarrow min\_len - 1$ 
8: end if
9:  $cp \leftarrow \text{Random}(1, min\_len)$ 
10: for  $i \leftarrow 1$  to  $cp$  step 1 do
11:    $O1[i] \leftarrow P1[i]$ 
12:    $O2[i] \leftarrow P2[i]$ 
13: end for
14: return  $O1, O2$ 

```

**Fin**


---

Las tres primeras instrucciones sirven para encontrar cuál de los dos cromosomas es el menor (1-3), y de esta manera asegurar que el punto de cruce no sea mayor al menor de los cromosomas. En el caso específico de la ruta dorsal artificial, como la proponen Dozal *et al.* (2014), todos los cromosomas siempre van a tener el mismo tamaño; para este y otros casos en los que se dé esta situación,  $min\_len$  es reducido por uno (7). Debido a que  $min\_len$  se usa como límite superior inclusivo para obtener el punto de cruce aleatorio (9), no se puede permitir que sea igual al tamaño máximo de los dos cromosomas, ya que el cruce consistiría en un intercambio total de loci, lo que perdería sentido para el propósito de la operación genética. Para hacer el cruce, primero se hace una copia de los cromosomas (4, 5) y una vez obtenido el punto de cruce, se procede

a intercambiar todos los loci desde la primer posición hasta el número indicado por  $cp$ . Nótese el cruce que se hace al hacer la copia de los cromosomas (4, 5) y al hacer la copia de los loci (11,12). Finalmente, se retornan los nuevos cromosomas (14).

---

#### Algoritmo 4 Mutación de Cromosoma

---

**Propósito:** Cambiar cada árbol del cromosoma con uno nuevo creado de manera aleatoria.

**Entrada:**

- $P$ : Un arreglo que representa a un cromosoma.
- $max\_level$ : Número que indica el nivel máximo que puede tener un árbol.
- $set_{func}$ ,  $set_{term}$ : Las listas de conjuntos de funciones y terminales, necesarios para crear nuevos individuos.

**Variables:**

- $len$ : Número que indica el tamaño del cromosoma.

**Salida:**

- $O$ : Un arreglo que representa el cromosoma resultante.

**Inicio**

```

1:  $len \leftarrow P.length$ 
2: for  $i \leftarrow 1$  to  $len$  step 1 do
3:    $new\_tree \leftarrow Create\_Tree(set_{func}i, set_{term}[i], max\_level)$ 
4:    $O[i] \leftarrow new\_tree$ 
5: end for
6: return  $O$ 

```

**Fin**

---

La mutación de cromosomas es una operación sencilla. Sólo es necesario sustituir todos los loci del cromosoma por nuevos (3), por ello, es necesario tomar en cuenta el nivel máximo que puede tener el árbol, además de tener acceso a la lista de funciones y terminales para cada locus.

---

#### Algoritmo 5 Cruce de Genes

---

**Propósito:** Realizar un cruce entre los genes de dos cromosomas. Si los dos cromosomas son de diferente tamaño, se intercambiarán los genes de tantos loci como tenga el menor de los cromosomas.

**Entrada:**

- $P1, P2$ : Arrays que representan los cromosomas.

**Variables:**

- $len1, len2$ : Números que indican el tamaño de los cromosomas.
- $cp1, cp2$ : Referencias a los nodos que indican los puntos de cruce de cada locus.
- $min\_len$ : Número que indica el tamaño del cromosoma menor.

**Salida:**

- $O1, O2$ : Arrays que representan a los cromosomas resultantes.

**Inicio**

```

1:  $len1 \leftarrow P1.length$ 
2:  $len2 \leftarrow P2.length$ 
3:  $min\_len \leftarrow \text{Minimum}(len1, len2)$ 
4:  $O1 \leftarrow P1$ 
5:  $O2 \leftarrow P2$ 
6: for  $i \leftarrow 1$  to  $min\_len$  step 1 do
7:    $cp1 \leftarrow \text{Random}(P1[i])$ 
8:    $cp2 \leftarrow \text{Random}(P2[i])$ 
9:    $O1[i] \leftarrow \text{Swap}(P1[i], P2[i], cp1, cp2)$ 
10:   $O2[i] \leftarrow \text{Swap}(P2[i], P1[i], cp2, cp1)$ 
11: end for
12: return  $O1, O2$ 

```

**Fin**

Se inicia obteniendo el tamaño de los dos cromosomas padre (1, 2) y el tamaño del menor de ellos (3). Como se explica en el Algoritmo 3, para el caso de la ADS siempre se tienen cromosomas del mismo tamaño. Continuando con el algoritmo, se hace una copia de los cromosomas (4, 5) y se procede al ciclo que intercambia los genes de cada locus (6). Para lograr ésto, primero se obtienen los puntos de cruce,  $cp1$  y  $cp2$ , del locus  $i$  para los dos cromosomas padres (7, 8). Un punto de cruce es una referencia a un nodo aleatorio de cada cromosoma, de esta forma, el nodo referenciado por  $cp1$  pasa a sustituir al nodo referenciado por  $cp2$ , trayéndose consigo a los nodos que le descienden (9); de la misma forma, el nodo referenciado por  $cp2$  pasa a sustituir al nodo referenciado por  $cp1$ , trayéndose consigo a todos sus nodos descendientes (10). En términos simples, cada par de árboles  $i$  están intercambiando una de sus ramas. Una vez que cada locus ha hecho un intercambio de sus genes, se retornan los nuevos cromosomas  $O1$  y  $O2$  (12).

**Algoritmo 6** Mutación de Genes

**Propósito:** Cambiar una de las ramas de cada árbol del arreglo de entrada.

**Entrada:**

- $P$ : Un arreglo que representa un cromosoma.
- $max\_level$ : Número que indica el nivel máximo que puede tener un árbol.

**Variables:**

- $mp$ : Nodo de referencia usado como punto de mutación.
- $mp\_level$ : Número que indica el nivel del punto de mutación.

- *new*: Una estructura de datos tipo árbol.
- *set<sub>func</sub>*, *set<sub>term</sub>*: Las listas de conjuntos de funciones y terminales, necesarios para crear nuevos individuos.

**Salida:**

- *O*: Un arreglo con el cromosoma resultante.

**Inicio**

```

1: len ← P.length
2: for i ← 1 to len step 1 do
3:   mp ← Random(P[i])
4:   mp_level ← Get_Node_Level(mp)
5:   new ← Create_Tree(setfunc[i], setterm[i], max_level – mp_level + 1)
6:   O[i] ← Join_Trees(P[i], new, mp)
7: end for
8: return O

```

**Fin**

Se obtiene el tamaño del cromosoma (1), pues se va a hacer un ciclo por cada locus que lo compone (2). Para realizar la mutación a un locus, representado por un árbol, primero se obtiene la referencia de uno de sus nodos de forma aleatoria (3). También se calcula su nivel, necesario para obtener el límite máximo de la nueva rama (4). Al igual que en la mutación de cromosomas (Algoritmo 4), la función que crea al nuevo árbol (que en este caso sería una rama) recibe de entrada el conjunto de funciones y terminales con los que conformará a la nueva estructura, junto con el límite que debe respetar (5). Ya creada la nueva rama, esta sustituye al nodo señalado por *mp*, así como el resto de sus nodos descendientes (6). Al final del ciclo, una rama de cada árbol ha sido cambiada por otra completamente nueva.

Como se ve en el diagrama de flujo (Figura 2) y el algoritmo del ciclo evolutivo (Algoritmo 1), el modelo de la ruta dorsal artificial se ejecuta dentro de la GP para probar las capacidades de la variedad de individuos creados y, por medio de sus resultados, calcular su aptitud. El Algoritmo 7 describe el funcionamiento de la ADS.

**Algoritmo 7** Ruta Dorsal Artificial (ADS)

**Propósito:** Contruir un proto-objeto a partir de una imagen y un conjunto de funciones (recibidas en grupo como un individuo). El algoritmo busca emular el funcionamiento de la ruta dorsal; tiene una estructura definida, pero son las funciones las que le permiten cambiar la forma en que obtiene y mezcla las características de la imagen, permitiéndole



adaptarse a diferentes situaciones. Recibe cuatro funciones distintas: tres para las dimensiones de color, forma y orientación; una para la función de integración de características. Así pues, cada individuo es una propuesta de solución a una instancia del problema del Foco de Atención.

#### Entrada:

- *image*: Una fotografía, en formato RGB, tomada con una cámara digital.
- *ind*: Arreglo que representa a un individuo. Está compuesto de cuatro funciones, representadas por árboles, que servirán para el funcionamiento de la ADS.

#### Variables:

- $I_{color}$ : Un conjunto de diez matrices que representan a la imagen en los componentes de color  $R, G, B, C, M, Y, K, H, S,$  y  $V$ .
- $D$ : Conjunto que sirve como referencia a las dimensiones de orientación, color, forma, e intensidad.
- $VM_d$ : Mapa visual que resulta de la extracción de características de la dimensión  $d$  del conjunto  $D$ .
- $CM_d$ : Mapa de conspicuidad de la dimensión  $d$ .
- $SM$ : Mapa de prominencia. Es el resultado de la integración de una selección de los mapas de conspicuidad.
- $S$ : Resultado del procesamiento del mapa de prominencia.

#### Salida:

- *Proto\_Object*: Imagen binaria resultante.

#### Inicio

```

1:  $I_{color} \leftarrow \text{Convert\_Image}(image)$ 
2:  $D \leftarrow \{orientation, color, shape, intensity\}$ 
3:  $EVO_{orientation} \leftarrow ind[1]$ 
4:  $EVO_{color} \leftarrow ind[2]$ 
5:  $EVO_{shape} \leftarrow ind[3]$ 
6:  $EFI \leftarrow ind[4]$ 
7: for each  $d$  in  $D$  do
8:   if  $d = intensity$  then
9:      $VM_d \leftarrow \frac{I_R + I_G + I_B}{3}$ 
10:  else
11:     $VM_d \leftarrow EVO_d(I_{color})$ 
12:  end if
13:   $CM_d \leftarrow \text{Center\_Surround}(VM_d)$ 
14: end for each
15:  $SM \leftarrow EFI(CM_{color}, CM_{orientation}, CM_{shape}, CM_{intensity})$ 
16:  $S \leftarrow \text{WTA}(SM)$ 
17:  $Proto\_Object \leftarrow \text{Estimate\_Shape}(S, SM)$ 
18: return  $Proto\_Object, S$ 

```

#### Fin

En la Sección 2.3 se describen las etapas por las que pasa la información visual al

momento de entrar a la ruta dorsal. Cada mapa producido por la ADS representa una de estas etapas.

Primero, se obtienen los componentes de tres modelos de color de la imagen de entrada (1); se producen diez matrices, del mismo tamaño de la imagen, representadas por la variable  $I_{color}$ :  $I_R$  (Rojo),  $I_G$  (Verde),  $I_B$  (Azul),  $I_C$  (Cyan),  $I_M$  (Magenta),  $I_Y$  (Amarillo),  $I_K$  (Negro),  $I_H$  (Matiz),  $I_S$  (Saturación), e  $I_V$  (Valor). También se inicializa el conjunto  $D$ , el cual sirve para hacer referencia a una determinada dimensión al momento de organizar los mapas y ejecutar las funciones del individuo (2).

Se asignan las tres primeras funciones del individuo como *operadores visuales evolucionados* ( $EVO_d^9$ ), y la cuarta como función de *integración de características evolucionada* ( $EFI^{10}$ ) (3-6). Se ejecuta un ciclo por cada dimensión en  $D$  (7), creando así un mapa de conspicuidad por cada dimensión.

Como se especifica en la Sección 3.2.1, la dimensión de intensidad es la única que tiene una fórmula definida para la producción de su mapa visual (ver Ecuación (1)), la cual equivale a un promedio de los componentes de color rojo, verde y azul (9). Para el resto de las dimensiones, se implementa el EVO correspondiente, cuya única entrada son los componentes de color de la imagen (11). Como resultado se obtiene un mapa visual para cada dimensión.

El mapa visual ( $VM^{11}$ ) es el resultado de la *etapa de representación temprana*. Cada VM es un mapa topográfico que representa, de alguna manera, una característica elemental. Por esa razón se realiza un mapa por cada dimensión. Hasta el momento no existe una idea clara de cuáles son los procedimientos o cálculos que realiza el cerebro para obtener estos mapas; con ello se abren las puertas para que se propongan una variedad de métodos para su cálculo, y es precisamente lo que se quiere lograr con el uso de la GP.

Cada VM produce un mapa de conspicuidad por medio de una función de centro-

---

<sup>9</sup>Del Inglés: Evolved Visual Operator.

<sup>10</sup>Del Inglés: Evolved Feature-Integration

<sup>11</sup>Del Inglés: Visual Map.

contorno (Algoritmo 8) (13). El mapa de conspicuidad ( $CM^{12}$ ), biológicamente, representa la etapa en la que el cerebro comienza a ser sensible a los cambios sobresalientes en un espacio local. También es la etapa en la que se obtienen las características invariantes a la escala.

Una vez producidos los mapas de conspicuidad de cada dimensión, estos son procesados por la función EFI (15). Hay que notar que, dependiendo de la EFI, es posible que uno o más mapas de conspicuidad sean omitidos en la integración. Como resultado se produce un mapa de prominencia.

El mapa de prominencia ( $SM^{13}$ ) es introducido a una red neuronal “Winner Take All” (16). Su trabajo es obtener el punto que genere mayor prominencia en todo el  $SM$ ; evalúa y realiza encendidos e inhibiciones de los puntos que conforman al mapa de prominencia, deteniéndose hasta que sólo uno de los puntos queda encendido; este punto es  $S$  y se guarda como una coordenada del mapa. Visualmente significa que este punto genera un cambio tan llamativo que sobresale del resto de la imagen, en un inicio. Puede tratarse de un cambio en el movimiento, o pudiera ser el contraste de dos colores, también pudiera ser una parte vacía o sencilla dentro de una imagen caótica.

Con estos dos resultados ya es posible formar el proto-objeto. La función de “Estimate\_Shape<sup>14</sup>” (17) (también nombrada como función “Spread<sup>15</sup>” en los diagramas de contexto) se encarga de esto utilizando como base a la coordenada  $S$ ; a partir de ella irá formando una mancha, por medio de un umbral aplicado a  $SM$ , hasta que cumpla con una serie de condiciones. El algoritmo de esta función se analiza en la Sección 6.1. El programa finaliza con el retorno del proto-objeto y la coordenada de prominencia  $S$  (18). El proto-objeto es la representación gráfica del *foco de atención*, es decir, representa la zona de la imagen en la que se va a centrar la atención, de los ojos y del cerebro, para una tarea más exhaustiva; un ejemplo de ello es el reconocimiento de objetos.

El procedimiento de centro-contorno es detallado en el Algoritmo 8.

---

<sup>12</sup>Del Inglés: Conspicuity Map.

<sup>13</sup>Del Inglés: Saliency Map.

<sup>14</sup>En español: Estimar\_Forma.

<sup>15</sup>En Español: Propagación

---

**Algoritmo 8** Centro-Contorno (Center\_Surround)
 

---

**Propósito:** El proceso genera un mapa de conspicuidad a partir de un conjunto de imágenes. Este conjunto es producido al transformar el mapa visual en una *pirámide Gaussiana*. El método centro-contorno representa la funcionalidad de las células ganglionares, las cuales, miden la diferencia de actividad neuronal de un punto de la imagen con respecto a los puntos de su alrededor.

**Entrada:**

- *VM*: Una imagen resultante de alguna función de extracción de características.

**Variables:**

- *Q*: Matriz que guarda el resultado de la resta entre dos niveles de la pirámide Gaussiana.
- $\sigma$ : Número que define la cantidad de niveles para la pirámide Gaussiana.
- *P*: Un conjunto de matrices. Cada matriz es una imagen correspondiente a un nivel de la pirámide Gaussiana. *P*[1] se conoce como el nivel *inferior*, mientras que a *P*[ $\sigma$ ] se le llama nivel *superior*.
- *minLevel*: Número que define el nivel más bajo de la pirámida que se usará para los cálculos del centro-contorno. "1" es el valor más bajo que puede tener esta variable.
- *maxLevel*: Número que indica el nivel más alto a usar en el cálculo del centro-contorno. Su valor máximo es  $\sigma - 1$ .
- *minDelta*: Número que define la distancia mínima, en niveles, entre el centro y el contorno.
- *maxDelta*: Número que indica la distancia máxima entre el centro y el contorno.
- *mapLevel*: Número que indica el nivel usado como referencia para definir el tamaño de todos los mapas envueltos en el proceso.
- *l*: Número que indica el índice del mapa usado como *centro*.
- *l2*: Número que indica el índice del mapa usado como *contorno*.
- *d*: Número que indica el desplazamiento *delta*. Es decir, la distancia en niveles entre el centro y el contorno.

**Salida:**

- *CM*: Matriz que representa al mapa de conspicuidad.

**Inicio**

```

1:  $\sigma \leftarrow 9$ 
2: minLevel  $\leftarrow 3$ 
3: maxLevel  $\leftarrow 5$ 
4: minDelta  $\leftarrow 3$ 
5: maxDelta  $\leftarrow 4$ 
6: mapLevel  $\leftarrow 5$ 
7: P  $\leftarrow$  Gaussian_Pyramid(VM,  $\sigma$ )
8: CM  $\leftarrow 0$ 
9: for l  $\leftarrow$  minLevel to maxLevel step 1 do
10:   for d  $\leftarrow$  minDelta to maxDelta step 1 do
11:     l2  $\leftarrow$  l + d
12:     if l2 >  $\sigma$  then
13:       continue
14:     end if

```

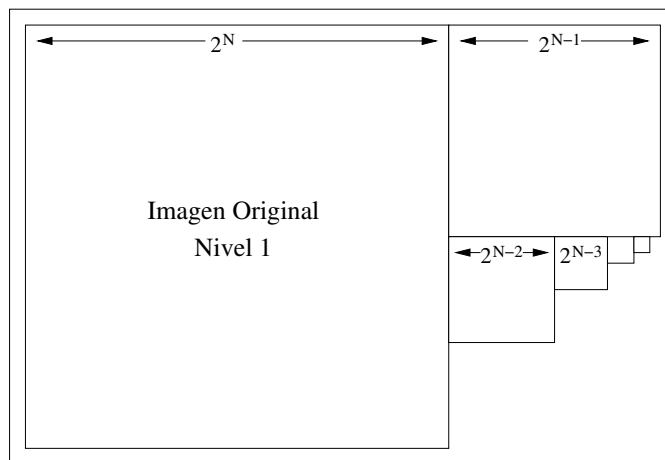
```

15:     PA ← Scale_To(P[l], P[mapLevel])
16:     PB ← Scale_To(P[l2], P[mapLevel])
17:     Q ← PA - PB
18:     Normalize(Q, 0, 1)
19:     CM ← CM + Q
20:   end for
21: end for
22: Normalize(CM,0,1)
23: CM ← Scale_To(CM, P[0])
24: return CM
Fin

```

---

Antes de comenzar el proceso que corresponde al comportamiento de centro-contorno, el mapa visual de entrada es convertido a una pirámide Gaussiana (7). Una pirámide (Figura 4) está formada por una serie de imágenes referenciadas como *niveles*. Cada nivel es el resultado de algún procesamiento y cambio de tamaño del nivel anterior. En la pirámide Gaussiana, cada nivel se suaviza con un kernel<sup>16</sup> Gaussiano simétrico. Después es submuestreado para formar la imagen del siguiente nivel; la escala queda en 1:2 con respecto al nivel anterior. El algoritmo produce nueve niveles para la pirámide (1).



**Figura 4:** La pirámide, producto de una imagen cambiada de tamaño por una sucesión de veces. Para el ejemplo de la figura, se muestra una pirámide de 6 niveles.

Se definen las variables de control para la resta que se hace entre los niveles (2-6). También se inicializa la matriz de CM en ceros (8); sus dimensiones son las mismas que tiene el mapa del nivel *mapLevel*. Se inician dos ciclos (9, 10): el primero es para

<sup>16</sup>Un kernel es la matriz usada en la operación de convolución de una imagen. Es la base del filtro, y la configuración de los valores de dicha matriz son definidos por el tipo de filtro.

definir el índice del nivel que es usado como *centro*. El segundo es para definir el valor de desplazamiento, es decir, la distancia que existe entre el mapa de *centro* y el mapa de *contorno*. Por tanto,  $I_2$  representa el índice de este último (11).

Calculados los índices, los niveles correspondientes son redimensionados al tamaño del mapa designado por *mapLevel* (15, 16). Se calcula  $Q$  como la diferencia entre las matrices del centro y el contorno (18); el resultado es normalizado en el rango  $[0, 1] \in \mathbb{R}$ .  $CM$  va guardando la suma de cada resultado en  $Q$  (19).

Al terminar los ciclos, se normaliza  $CM$  de nuevo con el mismo rango usado para  $Q$  (22). Después,  $CM$  es redimensionado al tamaño del primer nivel de la pirámide (23). Finalmente se retorna la matriz del mapa  $CM$ .

### 3.2.3. Corteza visual artificial

Como se explica en la Sección 3.1, la corteza visual se compone de dos rutas principales: la ruta dorsal y la ruta ventral. Mientras que la ruta dorsal se caracteriza por ser la parte del cerebro que realiza las tareas de detección y acción visomotora, la ruta ventral es caracterizada por ser la zona que realiza la identificación de los objetos observados. Basado en el modelo de esta ruta, Clemente *et al.* (2012) propusieron un modelo artificial bajo el mismo paradigma de la *programación cerebral*. Al modelo se le nombró como *ruta ventral artificial (AVS)*<sup>17</sup>.

Para las fechas en las que la AVS fue propuesta, las neurociencias ya mostraban que la dicotomía de las rutas no es total, es decir, existe la evidencia de actividad que define a las rutas, pero también se ha encontrado que estas no son independientes una de la otra. No sólo comparten áreas en la corteza, sino que los resultados obtenidos por una ruta sirven a la otra. Por ejemplo, Farivar (2009) explica sobre cómo existen áreas de la ruta dorsal que también juegan un papel importante en la tarea de reconocimiento de objetos, tarea que antes era sólo atribuible a la ruta ventral.

Así pues, Clemente *et al.* proponen el modelo de la *corteza visual artificial (AVC)*<sup>18</sup> como un modelo que implementa a las dos rutas para realizar el reconocimiento de obje-

---

<sup>17</sup>Del Inglés: Artificial Ventral Stream

<sup>18</sup>Del Inglés: Artificial Visual Cortex

tos. El modelo funciona bajo la idea de que para cumplir con dicha tarea, la ruta ventral utiliza a los resultados de la ruta dorsal, ordenando estas acciones como *detección*, y posteriormente *reconocimiento*.

### 3.2.4. Algoritmos de la programación cerebral de la corteza visual artificial

La AVC implementa los algoritmos de la ADS, mostrados en la Sección 3.2.2. En esta sección se describen los algoritmos que muestran la integración de las etapas de detección y reconocimiento, y el que muestra el funcionamiento general del ciclo evolutivo.

El proceso evolutivo es descrito en el Algoritmo 9.

---

#### Algoritmo 9 Algoritmo de GP - Ciclo Evolutivo

---

**Propósito:** Crear un conjunto de individuos. Cada individuo se compone de un mínimo de 4 funciones. Estas funciones son usadas por la Corteza Visual Artificial y sus resultados son introducidos a una SVM. La SVM es entrenada para identificar un objeto en base a los resultados de tales funciones.

**Entrada:**

- *num\_gen*: Número de ciclos/generaciones en el proceso evolutivo.
- *pop\_size*: Número límite de los individuos de la población.

**Variables:**

- *max\_level*: Número máximo de niveles que cualquier árbol puede tener.
- *max\_trees*: Número máximo de árboles (genes) que un individuo (cromosoma) debe tener.
- *prob\_chrom\_cross*: Número que define la probabilidad de aplicar un Cruce de Cromosoma.
- *prob\_chrom\_mut*: Número que define la probabilidad de aplicar una Mutación de Cromosoma.
- *prob\_gen\_cross*: Número que define la probabilidad de aplicar un Cruce de Genes.
- *prob\_gen\_mut*: Número que define la probabilidad de aplicar una Mutación de Genes.
- *pop\_chrom*: Un arreglo que guarda los miembros de la población.
- *parents\_chrom*: Un arreglo que guarda los individuos seleccionados para ser padres.
- *offspring\_chrom*: Un arreglo que guarda los individuos nuevos, producidos por algún operador genético.
- *images<sub>tr</sub>*: Un arreglo de pares de imágenes usadas para el entrenamiento. La primera es una imagen RGB y la segunda es una imagen binaria que representa al proto-objeto de la imagen RGB.
- *images<sub>val</sub>*: Un arreglo con la misma configuración que *images<sub>tr</sub>*. Sin embargo, éstas imágenes son usadas para la etapa de validación.
- *desc*: Un arreglo que guarda los *Vectores Descriptores* que son usados en la SVM para construir el clasificador de la función.
- *pop*: Un arreglo que guarda a los individuos y su modelo de predicción SVM correspondiente.

- *fitness*: Un arreglo que guarda los valores de aptitud de los cromosomas.
- $f_O, f_C, f_S, f_{MM}$ : Conjuntos de funciones de Orientación (O), Color (C), Forma (S) y Mapas Mentales (MM).
- $t_O, t_C, t_S, t_{MM}$ : Conjuntos de terminales de Orientación (O), Color (C), Forma (S) y Mapas Mentales (MM).
- *data\_base\_train*: Cadena de caracteres que representa la dirección en disco de la base de imágenes de entrenamiento.
- *data\_base\_val*: Cadena de caracteres que representa la dirección en disco de la base de imágenes de validación.

#### Salida:

- *best\_sol*: Es el mejor individuo que posee las funciones que mejor clasifican entre *clase* y *no-clase*.

#### Inicio

```

1:  $max\_level \leftarrow 9$ 
2:  $max\_trees \leftarrow 15$ 
3:  $prob\_chrom\_cross \leftarrow 0.4$ 
4:  $prob\_chrom\_mut \leftarrow 0.1$ 
5:  $prob\_gen\_cross \leftarrow 0.4$ 
6:  $prob\_gen\_mut \leftarrow 0.1$ 
7:  $f_O \leftarrow \{Orientation\ functions\ set\}$ 
8:  $f_C \leftarrow \{Color\ functions\ set\}$ 
9:  $f_S \leftarrow \{Shape\ functions\ set\}$ 
10:  $f_{MM} \leftarrow \{Mental\ Map\ functions\ set\}$ 
11:  $t_O \leftarrow \{Orientation\ terminals\ set\}$ 
12:  $t_C \leftarrow \{Color\ terminals\ set\}$ 
13:  $t_S \leftarrow \{Shape\ terminals\ set\}$ 
14:  $t_{MM} \leftarrow \{Mental\ Map\ terminals\ set\}$ 
15:  $images_{tr} \leftarrow Load(data\_base\_train)$ 
16:  $images_{val} \leftarrow Load(data\_base\_val)$ 
17:  $pop_{chrom} \leftarrow Init\_Pop(f_O, f_C, f_S, f_{MM}, t_O, t_C, t_S, t_{MM}, pop\_size, max\_level, max\_trees)$ 
18: for  $gen \leftarrow 1$  to  $num\_gen$  step 1 do
19:   if  $gen \neq 1$  then
20:      $parents_{chrom}.Clear()$ 
21:      $parents_{chrom} \leftarrow Roulette(pop, fitness, pop\_size)$ 
22:      $offspring_{chrom}.Clear()$ 
23:     while  $offspring_{chrom}.length < parents_{chrom}.length$  do
24:        $operator \leftarrow Roulette\_Op(prob\_chrom\_cross, prob\_chrom\_mut, prob\_gen\_cross,$ 
 $prob\_gen\_mut)$ 
25:        $i \leftarrow offspring_{chrom}.length$ 
26:        $offspring_{chrom}.Add(Apply\_Gen\_Op(operator, parents_{chrom}, i))$ 
27:     end while
28:      $pop_{chrom} \leftarrow offspring_{chrom}$ 
29:      $pop_{chrom}.Add(best\_sol)$ 
30:   end if
31:   for  $i \leftarrow 1$  to  $pop_{chrom}.length$  step 1 do
32:      $desc \leftarrow 0$ 

```



```

33:   for  $j \leftarrow 1$  to  $images_{tr}.length$  step 1 do
34:      $desc[j] \leftarrow AVC(pop_{chrom}[i], images_{tr}[j], 1)$ 
35:   end for
36:    $model[i] \leftarrow SVM(desc)$ 
37:    $pop[i] \leftarrow \{pop_{chrom}[i], model[i]\}$ 
38: end for
39:  $fitness \leftarrow 0$ 
40: for  $i \leftarrow 1$  to  $pop.length$  step 1 do
41:    $fitness[i] \leftarrow evaluation(images_{val}, pop[i])$ 
42: end for
43:  $best\_sol \leftarrow Get\_Best(pop, fitness, best\_ind)$ 
44: end for
45: return  $best\_ind$ 
Fin

```

---

Aunque el ciclo evolutivo de la ADS y de la AVC pudieran parecer similares a simple vista, estos dos procedimientos contienen varias diferencias.

El algoritmo comienza por la inicialización de los siguientes conjuntos de variables: las de control de los árboles (1, 2), las de probabilidades de los operadores genéticos (3-6), los conjuntos de funciones y terminales (7-14), y las bases de imágenes (15, 16).

La primer diferencia está en  $max\_trees$ , definida con el valor de 15 (2). Esto es debido a que la función EFI (de la ADS) es sustituida por un número aleatorio de funciones (llamadas  $EVO_{MM}$ ) para obtener unos mapas llamados *mapas mentales*. Entonces, el valor máximo que puede tener este número aleatorio es  $max\_trees - 3$ , pues se restan los árboles de orientación, color y forma.

Un mapa mental tiene el objetivo de describir el contenido de la imagen. El operador visual evolutivo (EVO) de un mapa mental es homogéneo; significa que el mismo operador se puede aplicar a todos los mapas de conspicuidad. La operación para obtener un mapa mental de la dimensión  $d$  está definida como:

$$MM_d = \sum_{i=1}^k (EVO_{MM_i}(CM_d)) \quad (2)$$

donde  $k$  es el número de operadores  $EVO_{MM}$  del individuo. Como se menciona con anterioridad, este número es aleatorio para cada individuo al momento de crearse.

También se utilizan dos bases de imágenes diferentes. La primera se usa para la etapa de entrenamiento; la segunda es utilizada en una etapa adicional llamada *validación*. Esta última se implementa para solucionar el sobreentrenamiento de la Máquina de Vectores de Soporte (SVM<sup>19</sup>).

Terminada la inicialización, se procede a construir la primer generación por medio de la función *Init\_Pop* (Algoritmo 10) (17). Después se inicia con el ciclo evolutivo (18) que consta de tres bloques.

El primer bloque, la condición **if** (19-30), se ignora en el primer ciclo. Es la parte del algoritmo encargada de crear la siguiente generación. Su funcionamiento es el mismo que en la ADS. Los individuos designados como padres son elegidos por medio de una selección de ruleta; la probabilidad de ser seleccionados depende de su aptitud. Después se van aplicando operadores genéticos, seleccionados también por ruleta, a los padres para producir los individuos hijos. Se usan los mismos operadores genéticos: cruce de cromosoma o genes (Algoritmo 3 y 5), y mutación de cromosoma o genes (Algoritmo 4 y 6). Una vez creada la población de hijos, esta sustituye a la población actual guardada en *pop<sub>chrom</sub>*; también se agrega el mejor individuo obtenido hasta el momento.

El segundo bloque, un ciclo **for** (31-38), es el encargado de calcular los *vectores descriptores*<sup>20</sup> por medio de la AVC. Por cada individuo de la población se crea un arreglo *desc*. Este arreglo guarda cada vector producido por la acción de la AVC (Algoritmo 11) sobre una imagen de entrenamiento en base al individuo actual, por lo que se producen tantos vectores descriptores como imágenes de entrenamiento, esto para un solo individuo. Una vez calculados los vectores, estos se introducen a la Máquina de Vectores de Soporte para producir un *modelo*. El individuo, el modelo y la SVM son las herramientas que se necesitan para realizar el reconocimiento de objetos. Finalmente, el arreglo *pop* guarda los pares *individuo-modelo*.

El siguiente y último bloque (39-42) se encarga de calcular la aptitud de cada par

<sup>19</sup>Del Inglés: Support Vector Machine. Se trata de un conjunto de modelos de aprendizaje supervisado. El algoritmo de aprendizaje analiza un conjunto de ejemplos de entrenamiento. Como resultado produce un modelo que clasifica los datos nuevos en dos o más categorías.

<sup>20</sup>Un vector descriptor es una variable, usualmente numérica, que caracteriza a un objeto. La idea es que estos descriptores capturen las características importantes de los patrones que producen estos objetos, para que luego una SVM pueda crear una relación entre ambos.

*individuo-modelo*. Esto lo hace obteniendo nuevos vectores descriptores para las imágenes de la etapa de validación. La SVM, usando el modelo del individuo, realiza una predicción sobre la imagen para identificar el objeto basada en el patrón mostrado por los vectores descriptores. Dependiendo de los aciertos que obtenga, se le dará un valor de aptitud al individuo.

Una vez obtenida la aptitud de todos los individuos, se guarda al mejor de todos ellos (43). Al finalizar todos los ciclos del proceso evolutivo, se retorna al mejor individuo junto con su modelo (45).

Para crear la primer generación de individuos, se siguen los pasos descritos por el Algoritmo 10.

---

#### **Algoritmo 10** Inicialización de Población en AVC (Init.Pop)

---

**Propósito:** Construir un número definido de individuos. Cada individuo se compone de un número de árboles y cada árbol representa una función. Los árboles se construyen por medio de un arreglo aleatorio de funciones y terminales que son seleccionadas de un conjunto definido.

**Entrada:**

- $f_O, f_C, f_S, f_{MM}, t_O, t_C, t_S, t_{MM}$ : Conjuntos de funciones y terminales usadas para construir los árboles que componen a los individuos. Véase Algoritmo 9.
- $pop\_size$ : Número que define el tamaño de la población.
- $max\_level$ : Número que define el nivel máximo de profundidad de los árboles.
- $max\_trees$ : Número que define el máximo de árboles que pueden ser construidos para un individuo.

**Variables:**

- $new\_ind$ : Un arreglo que guarda los árboles construídos para el individuo actual.
- $set_{func}$ : Un arreglo que guarda los conjuntos de funciones.
- $set_{term}$ : Un arreglo que guarda los conjuntos de terminales en el mismo orden que  $set_{func}$ .
- $mmTrees$ : Un número aleatorio que indica el número de árboles para MM.

**Salida:**

- $pop$ : Un arreglo con  $pop\_size$  individuos.

**Inicio**

- 1:  $set_{func}[1] \leftarrow f_O$
- 2:  $set_{func}[2] \leftarrow f_C$
- 3:  $set_{func}[3] \leftarrow f_S$
- 4:  $set_{func}[4] \leftarrow f_{MM}$
- 5:  $set_{term}[1] \leftarrow t_O$
- 6:  $set_{term}[2] \leftarrow t_C$
- 7:  $set_{term}[3] \leftarrow t_S$

```

8:  $set_{term}[4] \leftarrow t_{MM}$ 
9: for  $i \leftarrow 1$  to  $pop\_size$  step 1 do
10:    $new\_ind.Clear()$ 
11:    $mmTrees \leftarrow Random\_Int(1, max\_trees - 3)$ 
12:   for  $t \leftarrow 1$  to  $3 + mmTrees$  step 1 do
13:     if  $t > 3$  then
14:        $new\_ind[t] \leftarrow new\_tree(set_{func}[4], set_{term}[4], max\_level)$ 
15:     else
16:        $new\_ind[t] \leftarrow new\_tree(set_{func}[t], set_{term}[t], max\_level)$ 
17:     end if
18:   end for
19:    $pop[i] \leftarrow new\_ind$ 
20: end for
Fin

```

---

Este es similar al Algoritmo 2 de la ADS. Se diferencia en dos puntos: el primero es la sustitución de las funciones y terminales de *FI* por las de *MM* (4, 8); el segundo es el modo en que crea las funciones MM.

Por cada individuo a crear, se obtiene un número aleatorio entre 1 y  $max\_trees - 3$ . Este número,  $mmTrees$  (11), indica el número de árboles que se van a crear usando como fuente las terminales y funciones para mapas mentales. Véase el Algoritmo 11. De esta manera, a partir del cuarto árbol, todos los árboles serán creados en base al cuarto conjunto de funciones y terminales (14).

Como resultado final, cada individuo de la población estará compuesto de tres árboles (orientación, color, y forma) y de un número aleatorio de árboles para mapas mentales. El individuo más pequeño que puede existir consta de 4 árboles, mientras que el más grande puede tener hasta  $max\_trees$  árboles.

El funcionamiento de la AVC es detallado en el Algoritmo 11.

---

#### **Algoritmo 11** Corteza Visual Artificial (AVC)

---

**Propósito:** Contruir un descriptor a partir de una imagen y un conjunto de funciones (recibidas en grupo como un individuo). El algoritmo busca emular el funcionamiento de la corteza visual; tiene una estructura definida, pero son las funciones las que establecen la forma en que se obtienen las características y los métodos que emplea para construir los vectores que servirán para describir y clasificar a un objeto. El individuo está compuesto de al menos 4 funciones, pero pueden ser más: las primeras tres son para las dimensiones de color, forma y orientación; a partir de la cuarta función, y hasta la última, son

funciones para la construcción del *mapa mental*.

#### Entrada:

- *image*: Una fotografía, en formato RGB, tomada con una cámara digital.
- *ind*: Un arreglo que representa a un individuo. Está compuesto de al menos cuatro funciones, representadas por árboles, que servirán para el funcionamiento de la AVC.

#### VARIABLES:

- $I_{color}$ : Un conjunto de diez matrices que representan a la imagen en los componentes de color  $R$ ,  $G$ ,  $B$ ,  $C$ ,  $M$ ,  $Y$ ,  $K$ ,  $H$ ,  $S$ , y  $V$ .
- $D$ : Conjunto que sirve como referencia a las dimensiones de orientación, color, forma, e intensidad.
- $VM_d$ : Mapa visual que resulta de la extracción de características de la dimensión  $d$  del conjunto  $D$ .
- $CM_d$ : Mapa de conspicuidad de la dimensión  $D$ .
- $MM_d$ : Mapa mental de la dimensión  $d$ . Combina los resultados de cada operador visual aplicado al mapa de conspicuidad  $d$ . El número de operadores aplicados depende del individuo.
- $k$ : Número que indica la cardinalidad del conjunto de operadores visuales para el mapa mental. Su tamaño es un número aleatorio entre “1” y “12”.
- $gamma$ : Un arreglo que guarda todos los valores del mapa mental.
- $n$ : Número que define cuantos valores de  $gamma$  son usados para el vector descriptor. Este número es definido por el usuario.
- $v$ : Vector descriptor, usado por la Máquina de Vectores de Soporte (SVM). Representa una instancia de la clasificación.

#### Salida:

- *label*: La etiqueta designada por la SVM.

#### Inicio

```

1:  $I_{color} \leftarrow \text{Convert\_Image}(image)$ 
2:  $D \leftarrow \{color, orientation, shape, intensity\}$ 
3:  $VO_{orientation} \leftarrow ind[1]$ 
4:  $VO_{color} \leftarrow ind[2]$ 
5:  $VO_{shape} \leftarrow ind[3]$ 
6: for  $i \leftarrow 1$  to  $ind.length() - 3$  step 1 do
7:    $VO_{MM_i} \leftarrow ind[i + 3]$ 
8: end for
9: for each  $d$  in  $D$  do
10:   if  $d = intensity$  then
11:      $VM_d \leftarrow \frac{I_R + I_G + I_B}{3}$ 
12:   else
13:      $VM_d \leftarrow VO_d(I_{color})$ 
14:   end if
15:    $CM_d \leftarrow \text{Center\_Surround}(VM_d)$ 
16:   for  $i \leftarrow 1$  to  $k$  step 1 do
17:      $MM_d \leftarrow MM_d + VO_{MM_i}(CM_d)$ 
18:   end for

```

```

19: end for each
20:  $\gamma \leftarrow \text{MM\_Cat}(\text{MM}_{\text{color}}, \text{MM}_{\text{orientation}}, \text{MM}_{\text{shape}}, \text{MM}_{\text{intensity}})$ 
21: for  $i \leftarrow 1$  to  $n$  step 1 do
22:    $v[i] \leftarrow \text{Get\_Max}(\gamma)$ 
23:    $\gamma \leftarrow \text{Subtract}(\gamma, v[i])$ 
24: end for
25:  $\text{label} \leftarrow \text{SVM}(v)$ 
26: return label
Fin

```

---

El algoritmo de la AVC combina los procesos de la ADS y la AVS. La primera parte de este algoritmo corresponde a los procesos de la ADS, que es cuando el cerebro genera un foco de atención. A partir de ese punto, en vez de generar un proto-objeto, el algoritmo construye un vector descriptor: este proceso representa una parte de las funciones de la ruta ventral, pues estos descriptores son usados para la clasificación del objeto señalado por el foco de atención.

El primer paso es convertir la imagen de entrada en sus componentes de color (1). También se definen las dimensiones que se extraerán de la imagen (2). Se asignan los operadores visuales VO (3-8). Se debe notar que las tres primeras funciones corresponden a los operadores de orientación, color y forma; el resto son operadores para mapas mentales.

El bloque **for each** (9 - 19) realiza un ciclo por cada dimensión definida. Para cada dimensión  $d$  se construye un mapa mental ( $\text{MM}_d$ ) por medio de tres pasos: calcular el mapa visual, calcular el mapa de conspicuidad y finalmente realizar la suma de un número determinado de mapas mentales.

El cálculo del mapa visual es igual que en el proceso de la ADS. Para la dimensión de intensidad se usa una formula previamente definida (11); las otras tres dimensiones son calculadas con los operadores  $\text{VO}_{\text{orientation}}$ ,  $\text{VO}_{\text{color}}$  y  $\text{VO}_{\text{shape}}$  (13).

EL mapa de conspicuidad se calcula también por el mismo proceso de centro-contorno de la ADS (15). Este procedimiento está definido en el Algoritmo 8.

El mapa mental de la dimensión  $d$  es el resultado de la suma de cada mapa obtenido al aplicar cada  $\text{VO}_{\text{MM}}$  al mapa de conspicuidad (17). Como resultado del ciclo **for each**,

se obtienen cuatro mapas mentales. Cada mapa mental está estructurado en una matriz del tamaño del CM, y es una representación numérica de las características presentes en cada dimensión de la imagen.

Los cuatro mapas mentales son transformados de matrices a arreglos, y se concatenan en un solo arreglo *gamma* (20). El ciclo **for** (21 - 24) se encarga de formar el vector descriptor. Para construirlo, simplemente sustrae en orden descendente los *n* valores más altos de *gamma*.

Con el vector descriptor *v*, se produce una etiqueta por medio de la SVM (25). Esta etiqueta es la que se regresa como resultado de la AVC y se utilizará junto con otro conjunto de etiquetas para que la SVM pueda construir un modelo de clasificación.

En conclusión, este capítulo muestra en detalle de pseudocódigo los procedimientos que siguen la Ruta Dorsal Artificial (ADS) y la Corteza Visual Artificial (AVC). Para el desarrollo de esta tesis, se usa como base el funcionamiento de la ADS. Debido a que la AVC incluye el funcionamiento de la ADS, los cambios que se hagan en este último pueden ser transportados con relativa facilidad al funcionamiento de la corteza visual artificial.

En los Capítulos 4 y 5 se detallan los temas de las potencialidades y la profundidad, y sus aplicaciones en la inteligencia artificial. Estos dos temas centrales del trabajo de tesis son lo que se implementarán en conjunto con la ruta dorsal artificial.

## Capítulo 4. Potencialidades

---

### 4.1. Origen

El concepto de *potencialidades* proviene de la palabra en inglés “affordances” que a su vez proviene de la palabra “afford” que significa *permitirse* o *proporcionar*. El término de “affordances” (al que de ahora en adelante será referido como *potencialidades*) fue acuñado por Gibson (2014) en su documento llamado “The Ecological Approach to Visual Perception”, en el capítulo 8, “Theory of Affordances” (publicado originalmente en 1979).

Gibson describe que los organismos tienen la capacidad de percibir potencialidades de las sustancias, las superficies, los objetos, otros seres vivos, e incluso de la forma en que está acomodado un ambiente. Tales propiedades se perciben visualmente, no como valores físicos (e.g. color, forma), sino como significados de esos valores. Estos significados proveen a la entidad observada un sentido de uso e identidad dentro de un contexto ambiental.

Considere el siguiente ejemplo: Un cuchillo puede identificarse como aquel objeto que esté compuesto de una hoja afilada y un mango. La hoja afilada proporciona la habilidad de cortar, mientras que el mango permite que el usuario agarre el objeto sin lastimarse. Estas dos potencialidades son las que describen al objeto (no que sean las únicas), y por medio de ellas el cuchillo puede ser identificado como tal.

Según Gibson, la percepción de una potencialidad no es el proceso de percibir objetos físicos libres de valor, a los cuales se les asigna un significado posteriormente. Percibir las potencialidades es el proceso de percibir un *objeto ecológico rico en valores*. Para entender esto, se debe saber su origen: la *psicología ecológica*.

La psicología ecológica o ambiental estudia la relación que existe entre un organismo y su ambiente, y cómo este organismo percibe las diferentes acciones, oportunidades o posibilidades que le ofrece tal ambiente. Se propone al observador y al ambiente como un par inseparable, lo que quiere decir que todas las percepciones que tiene el organismo del ambiente son sólo para sí; un organismo diferente puede tener una percepción diferente. Sin embargo, las potencialidades de los ambientes son invariantes.



La invarianza de las potencialidades significa que tales propiedades no cambian con respecto al organismo que las observa. El organismo puede detectar una serie de potencialidades de un ambiente, sin embargo, el que no detecte alguna no significa que este ambiente no la posea. Por ejemplo, una silla muy alta puede que no sea adecuada para un niño pequeño que busca un asiento, y por tanto para el niño esta silla no es un asiento, sin embargo, un adulto no tendrá ningún problema en usarla de esta manera. Aunque el niño no pueda percibir el uso de la silla como asiento, no significa que la silla esté desprovista de esta potencialidad.

Así pues, un objeto ecológico es aquella sustancia, superficie, objeto, organismo o lugar que forma parte del ambiente del organismo observante, y con el cual, tal organismo tiene una relación. Entonces, percibir las potencialidades es el proceso de percibir las propiedades de uso, significados, y relaciones que tienen estos objetos de nuestro ambiente.

¿Cómo es esto diferente a “percibir objetos físicos libres de valor y asignar un significado después”? La percepción clásica describe que un organismo observa y sustrae primero las características y propiedades físicas del objeto (e.g., color, forma, orientación) y después de procesar esta información, asigna al objeto a alguna clasificación; esta clasificación es la que luego es relacionada con alguna propiedad. Por ejemplo, para identificar un cuchillo, tal organismo sustraería las características físicas como el color y la forma del cuchillo, después por algún método de clasificación lo asignaría como un objeto de tipo cuchillo y entonces podría deducir que los cuchillos permiten cortar, y por tanto, el objeto que observa sirve para cortar. En el sentido de las potencialidades, primero se detectaría que existe un objeto que corta y que puede agarrarse por un mango, por ende, es un cuchillo.

Esto explicaría la facilidad que tiene el ser humano de identificar una cantidad enorme de objetos con relativa y aparente facilidad, incluso aquellos que no ha visto con anterioridad. Los objetos de la vida diaria no son iguales: los asientos son todos diferentes en todas partes, dos herramientas que sirven para lo mismo pueden poseer distintos colores, formas y materiales, sólo por mencionar unos ejemplos. He aquí que métodos como el empaquetado de imágenes para la identificación es poco eficaz en este tipo de pro-

blemas. Los sistemas de visión artificial cada vez se enfocan más en obtener aquellas características que definen a un objeto por lo que es. En vez de preguntar cuáles colores puede tener una silla, o de qué material están formadas las sillas, se están comenzando a preguntar cosas como: ¿Por qué una silla es una silla? ¿Qué aspectos de la silla son las que nos permiten identificarla como tal? (Pregunta que se hacen Grabner *et al.* (2011)). Este tipo de preguntas son precisamente para las que Gibson propuso una respuesta: Una silla se puede identificar como tal porque el organismo la puede usar como asiento, las características de tal objeto le sirven al organismo para usarla como silla, por tanto, se identifica como tal.

Pero Gibson también es puntual en decir que no sólo los objetos poseen potencialidades, también los tienen los ambientes en general, las superficies, las sustancias, etc. Esto abre la posibilidad a muchas maneras de identificar un objeto por distintas vías, no sólo de lo que lo compone visualmente, sino de las potencialidades que posee debido a sus componentes.

La idea de potencialidades nació en la psicología, pero con el paso del tiempo ha encontrado su lugar y su aplicación en la inteligencia artificial. Dada la versatilidad del concepto, son varias las investigaciones que se han hecho con este tema. En la siguiente sección se hace mención de algunas de ellas.

#### **4.2. Usos en la inteligencia artificial**

Gibson otorga una lista de conceptos que contienen potencialidades: los animales, el terreno, los fluidos, las herramientas, en fin, todo lo que compone al ambiente del organismo. La idea de potencialidades dio nueva luz sobre cómo entender la percepción. Con esto en mente, los investigadores en inteligencia artificial han desarrollado este concepto proponiendo varias fuentes de potencialidades y realizando experimentos sobre su uso.

Myers *et al.* (2014) desarrollaron un sistema que infiere potencialidades en base a las características geométricas de los objetos. Proponen que un robot, además de identificar un objeto, debe saber qué parte del objeto cumple determinada función. Incluso debe entender aquellos objetos que no han sido vistos con anterioridad. En sus experimentos consideran dos tipos de agarres (la forma en que el robot agarra o sostiene un objeto)

y otras cinco potencialidades: cortar, contener, soportar, recoger (e.g., cuchara, cucharones) y aplastar.

El sistema de Myers *et al.* usa una cámara RGB-D<sup>1</sup> para extraer las características geométricas que componen a los objetos observados. Sus experimentos muestran que las potencialidades por información geométrica (e.g., profundidad, curvatura) mejoran satisfactoriamente los resultados al compararse con la clasificación por técnicas de apariencia (que usan el RGB y la escala de grises). También mostraron que las dos técnicas combinadas no obtienen mejoras visibles, concluyendo así que las potencialidades deducidas de información geométrica presentan un punto clave en la tarea de sus experimentos.

Giesel y Zaidi (2014) discuten la idea de que reconocer las propiedades de un material es tan importante como reconocer al objeto que componen. Por ejemplo, para saber cuando se va a caminar por un terreno que puede estar mojado (y por tanto resbaladizo) o con lodo, o cuando se busca una lija de papel para hacer un trabajo de carpintería. Su sistema se construye en base a los mecanismos de percepción a bajo nivel para la identificación de las propiedades de flexibilidad, grosor y aspereza. Tales mecanismos estiman las propiedades de superficies tridimensionales a partir de la representación de frecuencia bidimensional de las imágenes.

En el experimento que realizaron Giesel y Zaidi, se clasifican un conjunto de fotografías de distintos tipos de telas. Las clasificaciones se basan en cuatro potencialidades opuestas: suave-áspero, flexible-rígido, caliente-frío, y absorbente-impermeable. Para realizar la clasificación, se basan en la decisión de nueve personas. Posteriormente, implementan una serie de técnicas de análisis de imágenes para poder ver la correlación que existe entre la contribución de cada propiedad de material y la clasificación de potencialidad.

Tünnermann y Mertsching (2014) realizaron un trabajo en el que integran el concepto de prominencia visual con las potencialidades. En sus experimentos, hicieron una comparación entre el modelo de atención por prominencia y el de atención por potencialidad de agarre (*grasp-affordance*). Para esto, a un grupo de personas se les presentó una serie

---

<sup>1</sup>Una cámara RGB-D produce una imagen de color y una de profundidad. Véase la Sección 5.1.

de ciclos de imágenes en las que su tarea fue señalar la zona del cambio. Cada ciclo de imagen estaba compuesto de 4 fases: la imagen original, una imagen en blanco, la imagen que presenta un cambio respecto a la original, y nuevamente una imagen blanca. Cada imagen estaba representada en prominencia y en potencialidades de agarre. La tarea consistía en medir el tiempo de respuesta, por parte de los participantes, para encontrar el cambio en la imagen. Estos fueron los resultados: las personas podían identificar mucho más rápido los cambios hechos en las imágenes de potencialidades, que los señalados por el modelo de prominencia.

Se hizo un tercer experimento: se creó un tercer grupo de imágenes que combinara el modelo de potencialidades con el de prominencia, y se compararon con las imágenes de los modelos separados. Esta vez, los resultados mostraron que no hubo beneficio en combinar el modelo de prominencia con el de potencialidad. Sus conclusiones fueron que un modelo de atención por potencialidades es recomendado en lugar del modelo de prominencia, siempre y cuando sea posible extraer la información de potencialidades de la escena. Cuando tal información no esté disponible, se puede depender del modelo de prominencia.

Kjellström *et al.* (2011) investigaron la categorización de objetos de acuerdo a sus potencialidades. Estas potencialidades eran demostradas por las acciones de un humano. El sistema puede segmentar al humano y al objeto, y aprender de las acciones del humano sobre el objeto. Sus experimentos también muestran que la información contextual mejora la certeza de la clasificación tanto de los objetos como de las acciones. Con el sistema propuesto, un robot puede aprender a identificar un objeto desconocido observando las acciones que realiza un humano con dicho objeto, aprendiendo de esta forma cómo se manipula y cuál es su función.

Como se puede observar, existen varias formas de aplicar el concepto de potencialidades. Varias investigaciones concuerdan en que las potencialidades son una característica que tiene un impacto positivo en la certeza de la atención visual, así como en la identificación de objetos. Bajo esta misma idea, con el presente trabajo de tesis se propone integrar el concepto de potencialidades en función de la información de profundidad. Por ello, el Capítulo 5 introduce el concepto de profundidad, los métodos, y los instrumentos

necesarios para obtener esa información. Una vez abordados los tres conceptos (ruta dorsal, potencialidades, y profundidad), se detalla cómo funcionan juntos en el Capítulo 6.

## Capítulo 5. La profundidad

---

En la visión, la percepción de la profundidad es la capacidad que tiene un organismo de entender la distancia que existe entre el objeto observado y él mismo. Aunque el organismo no tenga la capacidad de asignarle una medida cuantitativa exacta, es lo suficientemente certera como para poder realizar las tareas visomotoras. Los métodos con los que el organismo puede lograr esta percepción son tres: por el tamaño de un objeto en la imagen, por el movimiento de la paralaje, y por la estereopsis.

La primer forma de percepción se trata de la capacidad del organismo de medir su distancia a un objeto, del cual conoce previamente su tamaño; esta medición la realiza por el tamaño que tenga dicho objeto en su retina, *i.e.*, cuando el cerebro recibe la imagen de un objeto conocido, puede estimar de una manera cualitativa su distancia a este. Por ejemplo, cuando una persona observa una pelota de básquetbol a una cierta distancia, si la persona está familiarizada con este tipo de pelotas, entonces tendrá la capacidad de saber qué tan lejos está de tal objeto. Incluso puede inferir si la posición en la que se encuentra le permite alcanzar dicho objeto con alguna extremidad. Este tipo de percepción es monocular, lo que significa que sólo necesita un ojo para estimar la distancia.

El segundo método es el movimiento de la paralaje. La palabra *paralaje* proviene de la palabra griega que significa *diferencia*, y se define como la diferencia de posición de un objeto al ser visto desde dos fuentes distintas. Tómese de ejemplo un organismo con visión binocular, como el ser humano. Desde una posición fija observando a un objeto fijo, tal organismo puede observar un cambio de posición del objeto al observar primero con un ojo y luego con el segundo. Este efecto es mucho más evidente mientras más cercano sea el objeto al organismo que lo observa. Suponga un ambiente donde existe una criatura que ve y está posicionada ante una serie de objetos acomodados a diferentes distancias; si la criatura se mueve, podrá observar que los objetos más cercanos se moverán mucho más rápido que los objetos que estén más lejos; esto es el movimiento de la paralaje. Cabe mencionar que contrario a la paralaje, el movimiento de la paralaje es un efecto que se puede observar de manera monocular.

La estereopsis es un fenómeno por el cual el cerebro tiene la capacidad de componer

una imagen tridimensional a partir de dos imágenes bidimensionales. Este efecto necesita obligatoriamente la visión binocular, y está relacionado con el concepto de paralaje. De alguna manera, el cerebro tiene la capacidad de identificar la posición del mismo objeto dentro de las dos imágenes. Con esas dos posiciones y por medio de la paralaje, calcula la distancia que existe entre el observador y el objeto. Un objeto que esté muy lejano tendrá una paralaje casi nula, mientras que los objetos cercanos tendrán una paralaje más prominente.

El cálculo de la profundidad es muy importante en visión artificial, pues es la dimensión que permite entender el mundo observado de manera tridimensional. Por ello se usan dos tipos de dispositivos que permiten obtener esta información: las cámaras estereoscópicas, y las cámaras RGB-D.

Las cámaras estereoscópicas o 3D, trabajan bajo el mismo concepto de la capacidad del cerebro para la estereopsis. Están formadas de dos objetivos<sup>1</sup>; cada uno captura una imagen y lo hacen al mismo tiempo. Calcular la distancia de los objetos por medio de estas dos imágenes es un problema al que se le conoce como *disparidad binocular*, y existen varias propuestas para realizar una reconstrucción tridimensional que le sea útil al sistema artificial que la implementa.

Los sensores RGB-D son cámaras relativamente nuevas en el mercado, y es el tipo de cámara que se usa en este trabajo de investigación.

### **5.1. Sensores RGB-D**

Un sensor RGB-D es un dispositivo que integra dos tipos de cámaras: una cámara RGB (Red, Green, Blue) y un sensor de profundidad (Depth). La cámara RGB es como cualquier otra cámara de video o fotográfica.

El sensor de distancia está compuesto de dos dispositivos: un emisor de luz infrarroja en forma de patrón, y el receptor de luz infrarroja. Con estos dos dispositivos, la cámara de distancia puede medir la profundidad existente entre un punto de la escena y la cámara.

---

<sup>1</sup>Un objetivo es un dispositivo que contiene un conjunto de lentes convergentes y divergentes, y posiblemente un sistema de enfoque y/o obturación. Forma parte de la óptica de una cámara.

La cámara de profundidad se basa nuevamente en el concepto de estereopsis y la paralaje para calcular la distancia de una distribución de puntos de la escena. Para hacer el cálculo necesita tres datos: la distancia existente entre el emisor y el receptor, el ángulo de un punto en la escena para el emisor, y el ángulo del mismo punto para la escena del receptor. Con esta información ya es posible realizar una triangulación y de esta manera conocer la distancia existente entre dicho punto y la cámara.

Es común que las imágenes de profundidad contengan zonas en las que no se puede calcular la distancia. Existen tres razones: los límites de distancia del sensor, ciertos materiales de las superficies, y la proyección de la sombra del infrarrojo.

Dependiendo del modelo de la cámara, hay límites para las distancias que puede calcular y la exactitud con las que puede medirlas. Si el punto a medir queda fuera del mínimo o el máximo de distancia, es imposible para la cámara calcular la profundidad. Si el objeto está muy cerca, interrumpe la luz infrarroja de manera que no puede ser captada por el receptor, y por tanto no hay medida. Si está muy lejos, el reflejo de la luz infrarroja no alcanza a ser detectado o se pierde en el ruido ambiental. Dentro del rango que sí puede verse, existe un rango óptimo. Este rango ofrece la mejor calidad de medición. Una vez fuera de él, ya sea que se aleje o se acerque a la cámara, la distancia comenzará a tener una cierta cantidad de error.

Al depender de la reflexión de la luz, la cámara de distancia tiene problemas para medir los objetos que están hechos de ciertos tipos de materiales. Entre estos materiales están: las superficies de reflexión no difusa, los objetos transparentes, y los materiales absorbentes de luz.

Una superficie de reflexión difusa es aquella que refleja la luz incidente hacia todos lados. Esta característica permite que pueda ser vista desde cualquier punto de observación, mientras no haya oclusión. Cualquier otro tipo de reflexión puede desviar, completa o parcialmente, la luz y evitar que pueda ser registrada por el sensor de infrarrojo. En este tipo de superficies están: los espejos, pantallas, y en general cualquier superficie muy pulida.

Por la misma razón, los objetos transparentes presentan un problema para el sensor,



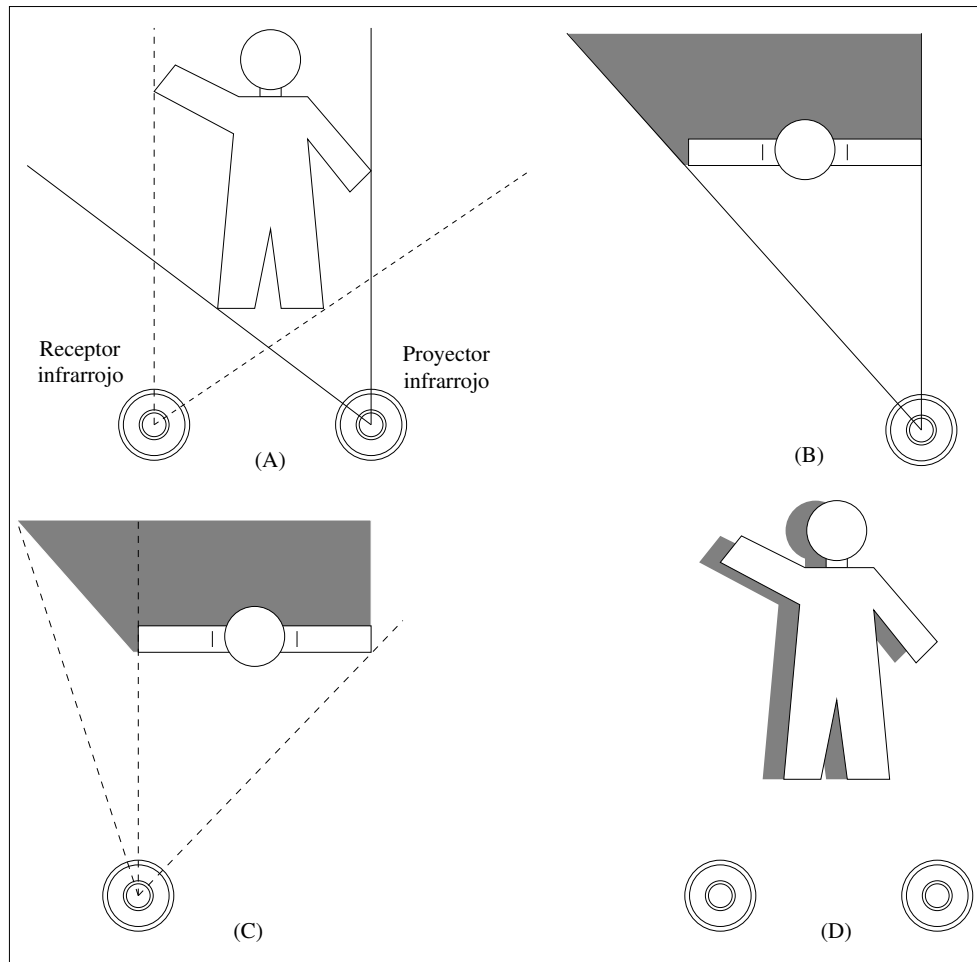
ya que pueden cambiar la dirección de la luz o dejarla pasar por completo. En el primer caso, generan reflexiones que pueden hacer inconsistentes el ángulo de emisión y el de reflexión, que finalmente se traduce en datos basura. El segundo, hace que el objeto sea invisible a la cámara de distancia. En este tipo de objetos están: las ventanas, las botellas, los líquidos, etc.

Finalmente, los materiales que absorben la luz también se hacen invisibles ante el sensor. Al no haber reflexión, el sensor infrarrojo no puede calcular el ángulo de la luz en esos puntos. Entre estos materiales, los más comunes son los que están pintados o son de tonalidades intensas de negro.

La tercer razón que impide el cálculo de la distancia es la sombra de la proyección del infrarrojo. Esta sombra se genera por la diferencia de ángulo entre el punto de proyección y el punto de recepción, y por la cercanía del objeto a la fuente de proyección. Un objeto que recibe la luz infrarroja proyecta una sombra detrás de si mismo. El receptor, al no tener ninguna información infrarroja de los puntos bajo la sombra, no puede medir la distancia de dichos puntos, ver Figura 5.

Aunque este tipo de cámaras permiten hacer cálculos menos complicados para obtener la distancia en las imágenes, también posee la desventaja de que sólo pueden ser usadas en interiores. Esto se debe a que la luz intensa, como la solar, distorsiona la luz del infrarrojo y dificulta (o imposibilita) la tarea del receptor.

Como se menciona, existen varias tipo de cámaras RGB-D. Para este trabajo de investigación se ha elegido usar un sensor Kinect para Xbox 360.



**Figura 5:** (A) Proyección y recepción de infrarroja. (B) La proyección de luz infrarroja sobre el objeto crea una sombra. (C) Por el cambio de ángulo, la sombra queda en el campo del receptor. (D) Como resultado, habrá puntos en la imagen de distancia que no tendrán dicha información debido a esta sombra.

## 5.2. Kinect

El Kinect es una cámara RGB-D de un costo relativamente accesible, con una comunidad de desarrolladores e investigadores extensa. Es sencilla de conectar a una computadora y puede usarse en los sistemas operativos Windows y Linux.

Como cualquier cámara RGB-D, se basa en el uso de la proyección y recepción de luz infrarroja para medir la distancia de los puntos de una escena. La forma en que identifica la correspondencia de los puntos del emisor con el receptor es la siguiente.

Primero, el emisor proyecta un patrón de puntos infrarrojos sobre la escena a medir. Este patrón debe cumplir la característica de que todos los puntos que componen a la imagen deben ser representados por un patrón de puntos diferentes, *i.e.*, no existen dos

puntos de la imagen que contengan el mismo patrón de puntos infrarrojos. Entiéndase como *puntos de la imagen* aquellas áreas de la imagen que componen un pixel en su representación digital. El dispositivo tiene la información de los ángulos en los que el emisor proyecta a los puntos.

Mientras tanto, el receptor infrarrojo está obteniendo una imagen de los puntos proyectados en la escena. Con esto, el dispositivo relaciona cada patrón de puntos con el ángulo visto desde el receptor. De esta manera se tiene la información de dos ángulos para el mismo punto de la imagen (el cual es identificado por el patrón de puntos infrarrojos). También se sabe la distancia existente entre el emisor y el receptor. Esta información es suficiente para conocer la distancia que existe entre el punto de la imagen y la cámara, por medio de un procedimiento llamado triangulación.

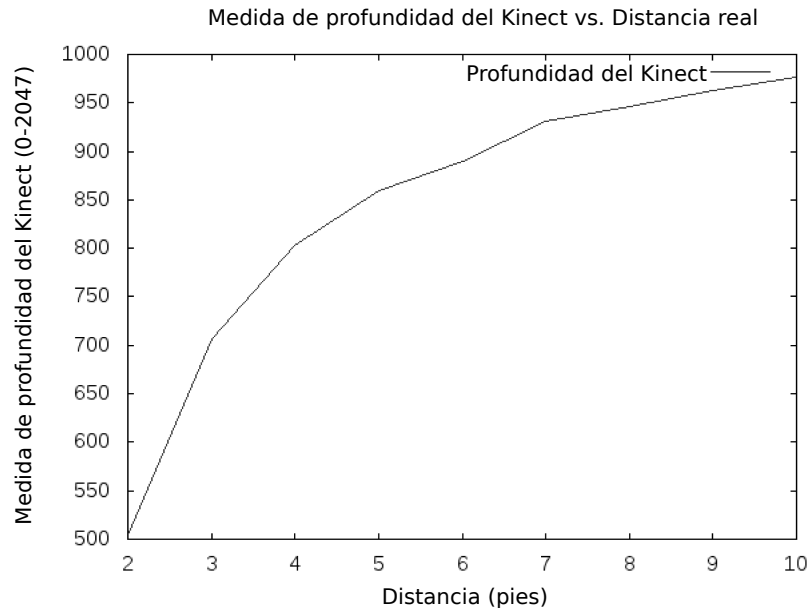
La cámara obtiene de manera sincronizada una imagen RGB (la imagen de color) y una imagen D (la imagen de profundidad) a la velocidad de 30 cuadros por segundo. Las dos imágenes tienen un tamaño de 640x480 (WxH) pixeles. Sin embargo la amplitud de visión de la imagen de profundidad es mayor a la de color. Agregado a esto, los objetos capturados en la imagen de profundidad tienen cierto desplazamiento con respecto a su posición original. Esto significa que la imagen RGB y la imagen de profundidad tienen un problema de empatado. Existen varias propuestas de solución para empatar estas dos imágenes, y también el propio Kinect viene con una rutina para realizar esta tarea: tiene integrado un registro de fábrica que le indica cuales pixeles de la imagen de profundidad corresponden a la imagen de color. En la Figura 6 se puede ver un ejemplo del par de imágenes sin y con empatado.

La imagen RGB está representada por un color de 8 bits, es decir, el valor de cada pixel de la imagen está en un rango de 0 a 255. Cada pixel se representa con 3 colores, por lo que se necesitan 24 bits para cada pixel. La imagen tiene un tamaño de 640x480, que se traduce en 3,072 pixeles; como cada pixel se representa con tres valores, la imagen tiene un total de 921,600 valores. El mapa de profundidad consta de 3,072 pixeles. El valor de cada pixel se encuentra en el rango de 0 a 2047; esto quiere decir que la imagen de profundidad necesita 11 bits para representar un pixel.



**Figura 6:** En la parte superior se observa la imagen de color, y la de profundidad (escala de grises) sin empatado. En la parte inferior se puede observar que la imagen de profundidad ha sido empatada con la de color, esto por medio del registro de fábrica del Kinect; de esta manera los objetos representados en color y en profundidad coinciden. Las líneas de guía muestran la alineación vertical y horizontal. Se debe notar que la imagen de profundidad empatada reduce considerablemente las sombras de la luz infrarroja (Figura 5).

Como se explica con anterioridad, la medida de la distancia tiene un rango óptimo. Mientras el objeto se encuentre más cercano al rango óptimo de distancia, será más exacta la profundidad calculada. Conforme los objetos medidos estén más lejanos, la nitidez de distancia será menor. La Figura 7 muestra el comportamiento de este sensor.



**Figura 7:** Se puede observar como la distancia de los objetos va perdiendo precisión mientras más alejados estén de la cámara. Fuente: Crock (2011).

Para poder usar el Kinect en nuestro trabajo, se necesitan dos cosas: un medio de conexión a la computadora y el software necesario para poder usarla en los programas.

El medio de conexión es un adaptador que permite conectar la salida del cable Kinect a una entrada USB 2.0. Ya que la energía de la conexión USB no puede sostener al Kinect, el adaptador también viene con una conexión individual de energía eléctrica, véase la Figura 8.

También fue necesario conseguir una base que pudiera montar un Kinect a un tripié estándar. Se adquirió una base “Monoprice 108682 - Montura de pared para Kinect de Xbox 360” (Figura 9). La montura de la pared se puede desprender, por lo que sólo utilizamos la base. También cabe mencionar que el orificio para el tornillo de seguridad para el tripié no era estándar sino de una medida M6, por lo que también se consiguieron tornillos de esta medida y se intercambiaron por los que usaba el tripié.



**Figura 8:** Adaptador de Kinect a USB.



**Figura 9:** Montura de Kinect de Xbox 360.

Para poder leer las imágenes capturadas por la Kinect se necesita instalar los controladores necesarios y la API<sup>2</sup>. Para el sistema operativo Windows existe el propio SDK<sup>3</sup> para Kinect de Microsoft; sin embargo, dado a que nuestro trabajo utiliza un ambiente Linux, no es posible usar tal SDK.

Para trabajar con Kinect en un ambiente Linux, se va a usar la biblioteca llamada *lib-freenect*. Esta biblioteca se encuentra mantenida por la comunidad llamada *OpenKinect*, y está bajo la licencia GNU. El proceso de instalación del OpenKinect es descrito en el Apéndice A. La forma en que se va a implementar esta biblioteca y el Kinect en el proyecto se especifica en el Apéndice B.

### 5.3. Los sensores RGB-D en la visión artificial

Aunque la visión artificial busca la manera de emular las funciones y procedimientos que realiza el cerebro para la visión, no está fuera de la discusión utilizar herramientas como las cámaras RGB-D. Finalmente, estas herramientas permiten obtener la información de profundidad sin tener que ocuparse del problema de la disparidad. Así los esfuerzos se centran en deducir qué es lo que se hace con la información de profundidad, no el cómo se obtiene.

La introducción de las cámaras RGB-D es relativamente nueva, sin embargo ya son muchos los trabajos en visión artificial que han utilizado esta herramienta. Hay trabajos de investigación que buscan integrar la información de profundidad a temas como el foco de atención o las potencialidades.

Varadarajan y Vincze (2012) crearon el sistema AfRob, que significa “Affordance Network Ontology for Robots”. Se trata de un sistema de ontologías<sup>4</sup> que usa las imágenes obtenidas con una cámara RGB-D para definir las regiones que pertenecen a los objetos, así como identificar las partes que los componen. Por medio de una ontología que

---

<sup>2</sup>Del Inglés: Application Programming Interface. Es un conjunto de subrutinas, funciones y métodos que serán utilizados por algún software como una capa de abstracción.

<sup>3</sup>Del Inglés: Software Development Kit. Es un conjunto de herramientas, APIs, y documentación sobre ellas. Sirven para que los programadores tengan todo lo necesario para programar en una plataforma específica.

<sup>4</sup>En inteligencia artificial, una ontología es un modelo de representación de conocimiento. Para mayor información, véase el Apéndice D.

clasifica estas propiedades, el sistema puede hacer inferencias para identificar el objeto observado.

Ogawa *et al.* (2014) crearon un algoritmo que simula la forma en que el ojo humano centra su vista en una serie de objetos puestos a distintos niveles de profundidad. El procedimiento obtiene un mapa de prominencia que después es filtrado por el mapa de profundidad. De esta manera, el algoritmo define movimientos de atención mucho más congruentes con respecto a la distancia que existe entre los mismos. Para sus experimentos, usaron una cámara Kinect.

Penaloza *et al.* (2012) proponen un algoritmo para simular el comportamiento de aprendizaje de un infante. Para esto, utilizan un robot programado con un modelo de atención de abajo hacia arriba. Cuando una persona se pone frente a él, el robot modifica su proceder: combina el modelo de atención con la información de profundidad, con la finalidad de poner más atención al objeto que el humano le esté presentando. La información de profundidad se obtiene por medio de una cámara Kinect.

En la Sección 4.2 se hace referencia a los trabajos de Myers *et al.* (2014) y Tünnermann y Mertsching (2014) los cuales también utilizan cámaras RGB-D para obtener la información de profundidad, siendo esto necesario para sus experimentos.

Con las cámaras RGB-D se puede obtener la profundidad de manera directa y sencilla. Su uso está limitado a interiores, pero no es una limitante muy importante para iniciar la exploración de la integración de información de profundidad a los distintos sistemas visuales que se están desarrollando día con día. La Kinect se ha introducido como la herramienta preferida por la facilidad de adquisición, instalación, y gran soporte.

El presente capítulo, junto con los Capítulos 3 y 4, introducen los tres temas principales que componen el trabajo de tesis: la ruta dorsal artificial, las potencialidades, y la profundidad en una imagen. Estos temas se combinan bajo el paradigma explicado en el Capítulo 6, donde además, se detallan los algoritmos propuestos para llevar a cabo esta tarea.





## Capítulo 6. Paradigma de potencialidades en función de la profundidad

---

En el Capítulo 4 se dio a conocer el concepto de las potencialidades de los objetos, y cómo estas características pueden ser usadas para guiar la identificación de los mismos. También se describió que estas potencialidades no eran únicas para los objetos; elementos como el ambiente, las texturas, la configuración geométrica e incluso la presencia de ciertos elementos en un entorno, por mencionar algunos, podrían propiciar una potencialidad para la entidad observante.

En esta sección, la referencia a la palabra *elemento* se dará por entendida como cualquier cosa (e.g., ambiente, objeto, sustancia) que puede ofrecer una potencialidad para un organismo que lo observa. Se puede entender que un elemento puede estar compuesto de otros elementos, y que sus potencialidades pueden ser heredadas desde los elementos que los conforman. También existe la posibilidad de que una potencialidad al ser heredada, puede ser aumentada, disminuida, anulada o cambiada totalmente. Por ejemplo, dos elementos contenidos en un tercero pueden combinar sus potencialidades y crear una nueva.

Un ejemplo muy sencillo de todo esto son las herramientas. Un martillo está compuesto del mango, y un metal pesado y con una superficie plana unido a uno de los extremos del mango. Por sí solo, el mango, que por el momento es únicamente un palo, puede ser agarrado, lanzado, abanicado, y puede golpear cosas con la posibilidad de ser destruido en algún grado (dependiendo de la dureza de la madera de la que está hecho). El metal con forma de cilindro puede ser agarrado, se pueden golpear cosas con él sin que el mismo sufra un daño significativo, la fuerza con la que golpea también afecta directamente al usuario, por mencionar algunas. Unidos en un martillo, ofrece la potencialidad de martillar, donde se usan las potencialidades de ambos objetos trabajando en conjunto. Las potencialidades fueron combinadas en una y cambiadas en cuanto a los resultados que ofrece.

Considere la pregunta: ¿La distancia, o profundidad, puede ser un elemento? Si la distancia puede ofrecer alguna potencialidad, lo puede ser. Sin embargo, la distancia

es sólo un modificador de potencialidades; por ejemplo, el fuego a una distancia media puede brindar calor benéfico, a una distancia inmediata puede quemar. Para que una distancia pueda brindar una potencialidad necesita conformar algo con lo que los organismos puedan interactuar directamente. Por tanto, la distancia es sólo una característica que describe a un elemento. Ese elemento es el espacio. Y el espacio sí puede ofrecer potencialidades.

Las potencialidades de un espacio dependen tanto de los elementos que se encuentran dentro del mismo, y de la percepción de quien los observa. Un espacio sin nada en él, otorga la potencialidad de poder ser atravesado, servir como un lugar para estar o desplazarse. Con más elementos que compongan o hagan a ese espacio, le otorgan más potencialidades a este. Existen muchos ejemplos de espacios: una cueva es un espacio creado por las formaciones rocosas, el viento y las corrientes de agua. Y una cueva ofrece un sinnúmero de potencialidades para distintos seres vivos. Estos seres pueden usarla de escondite, casa, almacén, etc., y trayendo más elementos a este espacio, la cueva puede ofrecer más potencialidades.

Considere otro ejemplo. Un espacio rodeado por 4 paredes, un piso, un techo, y con una puerta en una de las paredes, es una habitación de una casa. Si se agrega una estufa, convierte el espacio en una cocina; si se agrega una cama, convierte el espacio en un cuarto; si se agregan tuberías e inodoro, es un baño.

Se entiende entonces que los espacios son elementos, y la distancia o profundidad es una dimensión que los describe. En términos de visión, la distancia es una *dimensión*, y los valores que la describen son *características* de esta dimensión.

Una variedad de investigaciones, como las mencionadas en las Secciones 4.2 y 5.3, muestran que, tanto la distancia de los objetos como las potencialidades de los mismos sirven de guía a la atención y a la identificación. Entonces ¿Se puede crear un sistema que considere las potencialidades de los espacios? ¿Qué características y mejoras podría ofrecer este tipo de sistemas al foco de atención, o a la identificación de objetos?

Antes de dar una respuesta tentativa a estas preguntas, se puede pensar en el siguiente ejemplo: Un robot tiene la tarea de preparar un platillo en una cocina. Va a necesitar

identificar varias herramientas y trastos; y algunos de estos necesitarán colocarse en lugares especiales o definidos. El robot comienza su función y decide buscar un cuchillo. Puede hacer un barrido general a la cocina por medio de un algoritmo de atención, e ir identificando todas las cosas que llamen su atención para saber si son un cuchillo. ¿Que pasaría si el robot supiera que los cuchillos generalmente se ponen en los cajones, o se cuelgan en un portacuchillos? ¿Podría encontrarlos de manera más eficiente? ¿Podría también aumentar su eficacia? Suponga que el robot no tiene información de dónde se ponen los cuchillos, pero sí los encuentra. ¿Qué pasaría si necesita buscar otro cuchillo?, o ¿qué pasaría si necesita buscar otra herramienta parecida, tal vez un cuchillo más pequeño?

Suponga que se introduce un humano al mismo escenario, y que este humano no conoce la cocina en la que se encuentra, pero sí puede identificar las herramientas de cocina. Si el humano aprende que existe un cajón con un cuchillo después de haberlo encontrado, cuando busque otro cuchillo ¿No iría a buscarlo primero al cajón? Si se ve la situación entendiendo el tema de las potencialidades, sabrá que el humano aprendió que hay un cajón, un espacio, que guarda utensilios, y que aunque una forma de encontrar otro cuchillo sería buscarlo de nuevo por todos lados hasta encontrarlo en un cajón, la forma más lógica para él sería comenzar por el cajón. El ser humano aprendió e identificó una potencialidad por el espacio que le ofrece el cajón. Esta identificación la hizo porque al interior del cajón ya había utensilios guardados.

Sin duda el ser humano sigue usando sus ojos para encontrar la herramienta que busca, pero ya no examina el escenario completo, entiende que tal escenario posee distintas potencialidades ubicadas en diferentes espacios a lo largo del ambiente en el que se encuentra.

Este tipo de cuestionamientos son los que nos motivan a pensar que los espacios, cuyas composiciones forman al ambiente, ofrecen potencialidades al organismo que vive en ellos, y que estas potencialidades pueden guiar tanto al foco de atención, como a la identificación de objetos.

Pero para poder describir un espacio, se necesita saber la composición tridimensional

del mismo, y para entender esta composición, se depende de la dimensión de profundidad. Por tanto, la profundidad forma al espacio, el espacio otorga potencialidades, y estas potencialidades pueden servir de guía para los distintos sistemas del cerebro, y por consecuencia a los organismos; este es el paradigma de las potencialidades en función de la profundidad.

## **6.1. Modelo de FOA-DA y algoritmos**

Este proyecto de investigación se limita a integrar 3 elementos: El modelo de foco de atención, el paradigma de potencialidades y la dimensión de distancia, esta última proporcionada por la información de profundidad de la cámara RGB-D; lleva por nombre Foco de Atención integrando Potencialidades por Profundidad<sup>1</sup> o su versión más corta FOA-DA<sup>2</sup>.

### **6.1.1. Modelo de ADS para FOA**

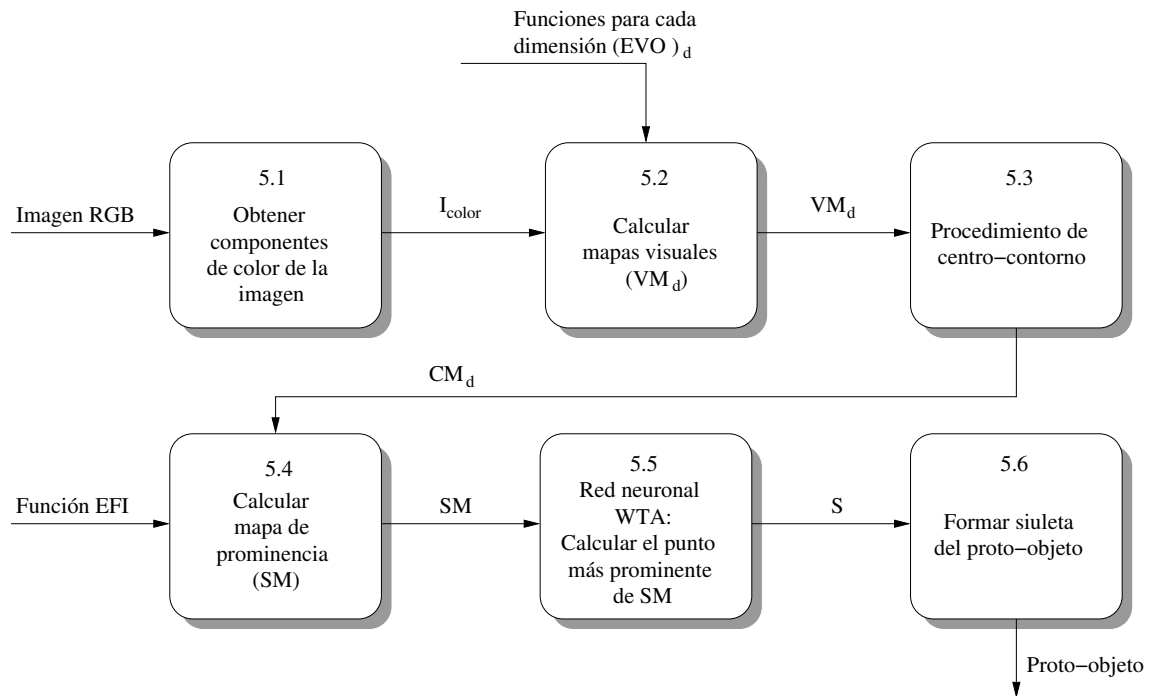
En la Sección 3.2.2 está especificado el algoritmo de la ADS. En esta sección se muestra un diagrama de este modelo para comprender cómo es modificado para integrar las potencialidades por profundidad. En la Figura 10 se puede observar el diagrama de la ADS.

Se debe recordar que existen dos formas de foco de atención. De abajo hacia arriba (bottom up) y de arriba hacia abajo (top-down). El primero es un foco de atención reactivo a la situación; el segundo es un foco de atención guiado por el conocimiento previo de lo que se busca. Bajo el mismo paradigma, se considera que la integración de las potencialidades de profundidad se pueden manejar de dos formas: de Bajo Nivel o Reactiva, y de Alto Nivel o Conocimiento previo.

---

<sup>1</sup>Nótese que se establece en el Capítulo 6 que la distancia no es en sí un elemento sino la descripción del elemento de interés, es decir, el espacio. Sin embargo, se conserva la palabra profundidad porque es la característica que se va a tener como fuente para describir al espacio.

<sup>2</sup>Del Inglés: Focus Of Attention with Depth Affordance.



**Figura 10:** Diagrama de contexto de la ADS. Describe el proceso “5” del diagrama de la Figura 2 en la Sección 3.2.1.

### 6.1.2. FOA-LDA y FOA-HDA

FOA-LDA<sup>3</sup> se refiere al *foco de atención con potencialidades por profundidad de bajo nivel*. Las potencialidades por profundidad de bajo nivel (LDA) son aquellas características de profundidad que guían al foco de atención sin algún conocimiento previo sobre lo que busca.

Se debe recordar que un foco de atención es representado por un *proto-objeto*. Este proto-objeto segmenta un área de la imagen; esta segmentación puede ser modificada por las características de la profundidad en el área. Por ejemplo, si los puntos de un área se encuentran a una distancia similar, quiere decir que se está observando una superficie, *i.e.*, el sistema reacciona ante un conjunto de puntos que pudieran estar formando un objeto. Así pues, podría prevenirse una formación de foco de atención sobre dos objetos que, aunque pudieran ser similares en apariencia, presentan una diferencia en profundidades que supera un determinado umbral.

En las LDA no se busca una potencialidad en específico, sino que se busca un con-

<sup>3</sup>Del Inglés: Focus Of Attention with Low-level Depth Affordances.

junto de características de profundidad que pudieran aportar una potencialidad, *i.e.*, un conjunto de puntos sean lo suficientemente cercanos como para formar un *elemento*<sup>4</sup>.

FOA-HDA<sup>5</sup> se refiere al *foco de atención con potencialidades por profundidad de alto nivel*. Las potencialidades en profundidad de alto nivel (HDA) son un grupo de puntos que pertenecen a una zona tridimensional. El sistema está predispuesto a buscar dentro de esta zona, ya que tiene conocimiento de que la misma le aporta una potencialidad. Dado a que es un espacio tridimensional, hay varias formas de delimitarlo; en realidad, tantas como figuras o formas tridimensionales se puedan definir. Sin embargo, en este trabajo de investigación sólo se consideran los límites de frente y fondo, es decir, una distancia límite cercana a la cámara y una distancia límite lejana a la cámara.

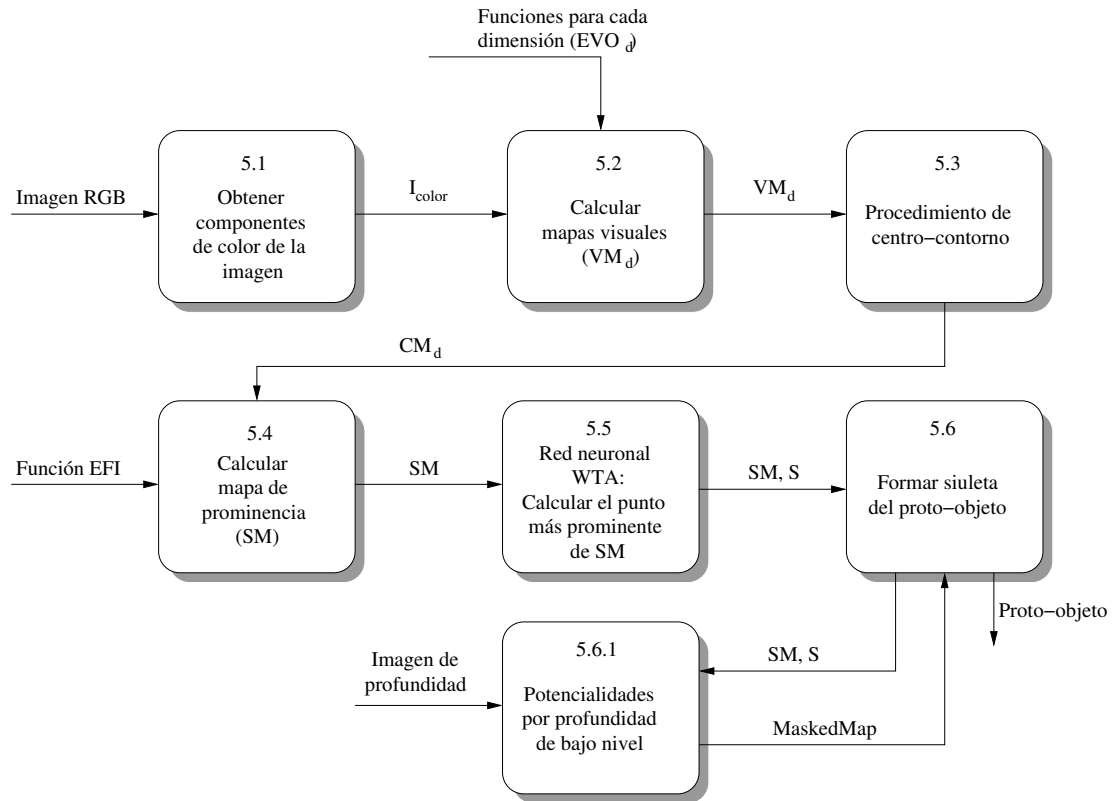
En pocas palabras, una LDA se calcula por un vecindario en profundidad sobre los puntos de la imagen; este vecindario se forma tomando como referencia el punto de foco de atención. Una HDA delimita la búsqueda en una imagen, cortando los estímulos que estén fuera del área de la potencialidad buscada. Para conocer esta delimitación, las HDA se basarán en un *mapa de potencialidades*. Como este proyecto sólo usa dos límites, el mapa de potencialidades consta únicamente de dos números. Para etapas más avanzadas del proyecto, se desea construir este mapa de manera que describa un conjunto de espacios tridimensionales, donde cada espacio representa una o más potencialidades conocidas.

### 6.1.3. Algoritmo para el FOA-LDA

El diagrama de flujo para el FOA-LDA se muestra en la Figura 11. Se puede observar que la modificación necesaria en el procedimiento de FOA se hace en la función *spread*; esta es la función que crea el proto-objeto. Se debe recordar que la distancia no representa una potencialidad en principio, por lo que la LDA necesita un sitio de referencia; para ello se usa el punto de foco de atención. El punto de foco de atención es el lugar más prominente obtenido por la red neuronal WTA. Este lugar conlleva un significado: representa la posición donde se centra la fovea, sin embargo, aún no se ha formado la mancha que define al foco de atención. Aquí es donde entra la LDA.

<sup>4</sup>Recuérdese la definición de elemento, expuesta en el Capítulo 4

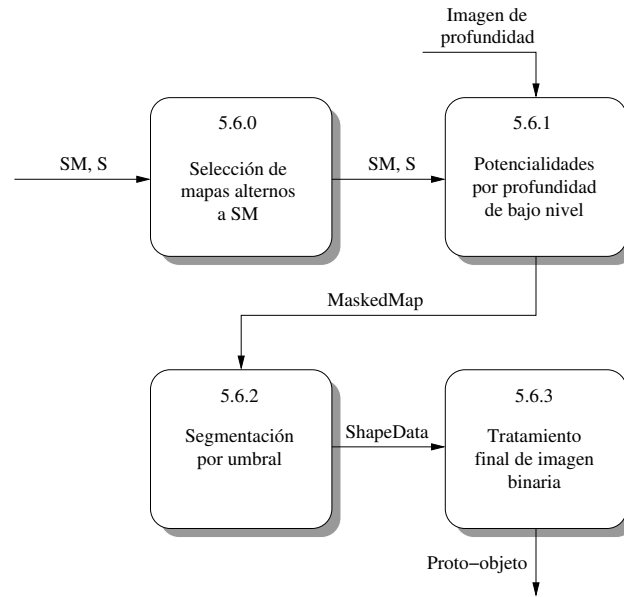
<sup>5</sup>Del Inglés: Focus Of Attention with High-level Depth Affordances.



**Figura 11:** Diagrama de contexto de FOA-LDA.

El proto-objeto es un producto del resultado de la WTA y el mapa de prominencia. La información del mapa de prominencia es delimitada por una serie de umbrales para formar la mancha, tomando como referencia el punto de foco de atención. Antes de que el mapa de prominencia sea modificado por estos umbrales, se introduce al procedimiento de integración con la distancia. La Figura 12 muestra el procedimiento que sigue la función de “spread” y el momento en el que se realiza la integración del mapa de prominencia con la información de profundidad.





**Figura 12:** Diagrama de contexto de la función “Spread”/“Estimate\_Shape”.

El Algoritmo 12 muestra el proceso que se le hace al mapa de prominencia para que sea modificado de acuerdo a las profundidades que lo conforman.

---

### Algoritmo 12 Integración de Profundidad (Combine\_With\_Depth)

---

**Propósito:** Modificar el mapa de prominencia de tal forma que la prominencia de cada punto sea aumentada o disminuida dependiendo de su relación con la profundidad del punto de foco de atención. Hay dos tipos de relaciones: de vecindario, y de profundidad general. Las dos relaciones de profundidad son promediadas y se usan como una máscara. Por medio de un cálculo, esta máscara aumenta o disminuye los valores del mapa de prominencia. Finalmente el nuevo mapa es devuelto para que se forme la mancha del proto-objeto.

**Entrada:**

- *SM*: Mapa de Prominencia.
- *win*: Un arreglo con dos valores que indican la coordenada en  $[x,y]$  del punto más prominente de *SM*.
- *depth*: Una imagen de profundidad.

**Variables:**

- *m*: Número que indica la pendiente para la función de profundidad general.
- *thl*: Número que indica el umbral (en milímetros) para la función de vecindario en profundidad.
- *posScaled*: Un arreglo que contiene la coordenada de *win* escalada al tamaño de *depth*.
- *dSal*: Una matriz con los valores de la máscara de profundidad general.
- *dNhd*: Una matriz con los valores de la máscara de vecindario en profundidad.
- *xLimit*, *yLimit*: Números que definen las dimensiones de las matrices.
- *x*, *y*: Números contadores.
- *max*: Número que guarda el valor del punto de foco de atención.

- *mask*: Matriz que guarda los valores de la máscara de profundidad.

**Salida:**

- *maskedMap*: Una matriz que representa el mapa de prominencia modificado por la máscara.

**Inicio**

```

1:  $m \leftarrow 5$ 
2:  $thl \leftarrow 5$ 
3:  $posScaled \leftarrow \text{GetScaledPosition}(depth, SM, win)$ 
4:  $dSal \leftarrow \text{GeneralDepthSaliency}(depth, posScaled, m)$ 
5:  $dNhd \leftarrow \text{DepthNeighborhood}(depth, posScaled, thl)$ 
6:  $[xLimit, yLimit] \leftarrow depth.size$ 
7: for  $y \leftarrow 1$  to  $yLimit$  step 1 do
8:   for  $x \leftarrow 1$  to  $xLimit$  step 1 do
9:      $mask(x, y) \leftarrow \frac{dSal[x, y] + dNhd[y, x]}{2}$ 
10:   end for
11: end for
12:  $mask \leftarrow \text{ScaleTo}(mask, SM)$ 
13:  $max \leftarrow SM[win[1], win[2]]$ 
14: for  $y \leftarrow 1$  to  $yLimit$  step 1 do
15:   for  $x \leftarrow 1$  to  $xLimit$  step 1 do
16:      $maskedMap[x, y] \leftarrow SM[x, y] * mask[x, y] +$ 
 $(max - SM[x, y] * mask[x, y]) * \left( \frac{SM[x, y]}{max} + \frac{mask[x, y] - \frac{SM[x, y]}{max}}{2} \right)$ 
17:     if  $maskedMap[x, y] < max * (0.4)$  then
18:        $maskedMap[x, y] \leftarrow maskedMap[x, y] * (0.1)$ 
19:     end if
20:   end for
21: end for
22: return  $maskedMap$ 
Fin

```

Se comienza por definir los valores de  $m$  y  $thl$  (1 - 2). Para este experimento se decidió que  $m$  tuviera un valor de 5 (que indica el valor de la pendiente para el método de profundidad general), y para  $thl$  el valor de 5 (el valor del umbral para el método de vecindario, en milímetros).

La siguiente fase consiste en formar una máscara de profundidad usando a la imagen  $depth$  y a la coordenada  $win$  como punto de referencia. Como la imagen de profundidad  $depth$  y el mapa de prominencia  $SM$  no tienen el mismo tamaño, no se puede usar  $win$  y  $depth$  directamente. Si la imagen  $depth$  se escala al tamaño de  $SM$ , se pierde información.

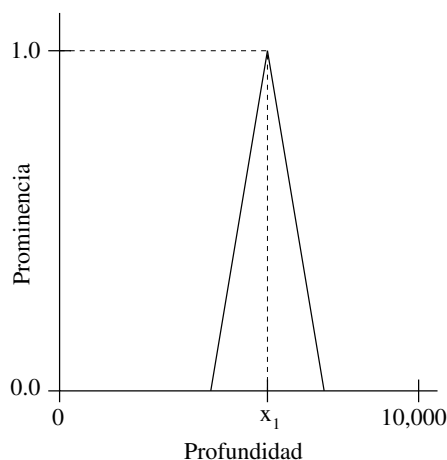
Entonces, el mejor camino es escalar la posición que tiene la coordenada de *win* para que sea congruente con el tamaño de *depth*. Con esto se obtiene la coordenada *posScaled* (3).

La máscara principal se construye de dos máscaras previas: *dSal* y *dNhd*, ambas representadas por matrices con un rango de valores de 0 a 1. Cada una representa la prominencia de cada punto de profundidad con respecto a un punto de referencia señalado por *posScaled*.

La máscara *dSal*, o de profundidad general, representa la prominencia que tiene cada distancia al compararse con la distancia de *posScaled*, *i.e.*, mientras más alejado se esté del punto de referencia, habrá menos prominencia. Para calcular este valor se usa la ecuación de la recta:

$$y - y_1 = m(x - x_1). \quad (3)$$

Con esta ecuación, podemos formar una gráfica como se ve en la Figura 13.



**Figura 13:** Cálculo de prominencia en profundidad general.

El *eje x* representa el valor de profundidad. El *eje y* representa la prominencia, cuyo rango es [0,1]. Se puede saber cuál es la profundidad del punto *win* por medio de *posScaled* en *depth*. Esta profundidad se asigna a  $x_1$ . Como  $x_1$  representa la profundidad del punto de mayor prominencia en *SM*, su valor en  $y_1$  es 1. A partir de aquí, la prominencia con respecto a la distancia es modificada por  $m$ , ya que define la inclinación que tendrá la recta que pasa por la coordenada  $(x_1, y_1)$ . Entonces, si la pendiente es muy pronunciada,

la prominencia de los puntos en la distancia también cambiará de manera más marcada al alejarse de la distancia que tiene *posScaled*.

Como se puede ver en la Figura 13, la pendiente de la recta se hace negativa cuando se hacen cálculos con distancias mayores a la del punto de mayor prominencia:

$$x \begin{cases} \leq x_1 : y = m * (x - x_1) + 1, \\ > x_1 : y = (-m) * (x - x_1) + 1. \end{cases} \quad (4)$$

Como resultado, sólo los puntos que tienen la misma distancia que *posScaled* tendrán el valor de 1. A partir de ahí, todos los demás puntos tendrán menor prominencia, calculada respecto a la diferencia que existe entre su distancia y la de *posScaled*.

La máscara de vecindario en profundidad, a diferencia de la de profundidad general, es una imagen binaria, es decir, sólo tiene los valores de “0” y “1”. Todos los píxeles que se encuentren dentro del vecindario tienen el valor de “1”. Este vecindario se calcula de la siguiente manera: Se establece la máscara inicial con todos sus valores igual a “0”, y se asigna al píxel señalado por *posScaled* como el píxel inicial con valor de “1”. A partir de ahí, se usa un procedimiento recursivo: los píxeles horizontales y verticales colindantes son marcados como vecinos si la diferencia de sus valores de distancia están dentro del umbral inclusivo. Los píxeles marcados como vecinos corren el mismo procedimiento que se usó con el píxel inicial. Sólo aquellos píxeles que no han sido marcados con anterioridad pueden ser usados en la recursión; esto evita que el programa recursivo se cicle marcando una y otra vez los mismos píxeles vecinos.

Después se calcula la máscara final; estase obtiene como el promedio de las dos máscaras (6 - 11). La máscara final *mask* se reduce en escala al tamaño de *SM*(12). Se asigna *max* como el valor de *SM* en la coordenada *win* (13). Se procede a calcular la prominencia de *SM* en función de la distancia por medio de *mask* (14 - 21); la ecuación se describe en la Sección 6.1.4.

Como paso final, todos aquellos resultados cuyo valor sea menor al 40% de *max* son disminuidos (17 - 19). Este paso se realiza para ayudar en la formación de la mancha del

proto-objeto. Una vez calculados los valores de *maskedMap*, este pasa a ser el nuevo mapa de prominencia, que se usará para formar la mancha del proto-objeto en la función *spread*.

#### 6.1.4. Cálculo de prominencia en función de la máscara de profundidad

El objetivo de la ecuación para calcular la prominencia en función de la máscara de profundidad es modificar la prominencia original, de tal forma que los puntos que guarden mayor relación con el punto de foco de atención puedan aumentar su valor de prominencia, mientras que los que guardan menor relación disminuyan este valor.

La ecuación para calcular la modificación de la prominencia de *SM* está definida como:

$$\begin{aligned} maskedMap[x, y] \leftarrow & SM[x, y] * mask[x, y] + \\ & (max - SM[x, y] * mask[x, y]) * \\ & \left( \frac{SM[x, y]}{max} + \frac{mask[x, y] - \frac{SM[x, y]}{max}}{2} \right). \end{aligned} \quad (5)$$

Esta ecuación se puede entender mejor si la representamos de la siguiente manera:

$$maskedMap[x, y] \leftarrow A + B * C \quad (6)$$

donde

$$A \leftarrow SM[x, y] * mask[x, y], \quad (7a)$$

$$B \leftarrow (max - SM[x, y] * mask[x, y]), \quad (7b)$$

$$C \leftarrow \left( \frac{SM[x, y]}{max} + \frac{mask[x, y] - \frac{SM[x, y]}{max}}{2} \right). \quad (7c)$$

Se debe recordar que la máscara contiene valores en un rango de [0,1], y la variable *max* contiene el valor de la prominencia más alto de todo el mapa *SM*. Como se puede observar, la ecuación está compuesta de tres subecuaciones: A, B, y C; cada una con un significado.

La *subecuación A* cambia el valor de prominencia de acuerdo al valor que representa la máscara. Se debe pensar en la máscara como un índice en porcentaje de cero a 100, donde un 0% indica que tal punto guarda una relación nula en distancia con el punto de foco de atención; por el contrario un 100% indica que tal punto está completamente relacionado, en distancia, con el punto de foco de atención. Esta sola subecuación tiene un problema, todos los pixeles que no sean el punto de foco de atención sólo podrán disminuir su prominencia o quedarse en su valor original. Para sortear esto, se suma con el resultado de la operación de las otras dos subecuaciones.

La *subecuación B* obtiene la diferencia que existe entre *max* y el valor de *SM* modificado por la máscara. En otras palabras, se calcula cuánto le falta al valor de *SM* modificado para tener el mismo valor que *max*. Esto es importante ya que el objetivo de modificar la prominencia es lograr que los puntos que no son tan prominentes ganen valor, dependiendo de la relación que existe en distancia entre tales puntos y el punto de foco de atención.

La *subecuación C* calcula el porcentaje que se usará del valor de la *subecuación B*, *i.e.*, la *subecuación C* define que tanto podrá aumentar la prominencia de los pixeles dependiendo de su valor en la máscara y el valor de prominencia en *SM*. La subecuación está compuesta de una suma de dos términos:

$$C_1 \leftarrow \frac{SM[x, y]}{max}, \quad (8a)$$

$$C_2 \leftarrow \frac{mask[x, y] - \frac{SM[x, y]}{max}}{n}. \quad (8b)$$

La subecuación  $C_1$  indica el porcentaje que representa el valor de prominencia del pixel  $[x, y]$  con respecto del valor de *max*. La subecuación  $C_2$  define si el resultado de  $C_1$  aumenta o disminuye; para ello, al valor de la máscara se le resta el valor de  $C_1$ . Si la máscara es mayor a  $C_1$ , entonces  $C_1$  aumentará. Si la máscara es menor a  $C_1$ , el valor de  $C_1$  disminuirá. Finalmente, si los dos son iguales,  $C_1$  quedará igual. El valor de  $n$  es un número mayor o igual a “1” que define la fuerza con la que afectará  $C_2$  a  $C_1$ . Mientras más grande sea  $n$ , menor será el efecto de  $C_2$  en  $C_1$ . Para este experimento, se definió el valor de “2” para  $n$  por defecto.

Por la naturaleza de las subecuaciones,  $C$  sólo puede tener un valor en el rango de  $[0,1]$ .  $B$  es complemento de  $A$  para  $max$ . Por ende, la Ecuación (5) nunca obtendrá un valor de prominencia fuera del rango  $[0,max]$ . Así, la Ecuación (5) define de manera matemática el valor de prominencia de un pixel tomando en cuenta el valor de prominencia original y el valor de la máscara de profundidad.

#### 6.1.5. Algoritmo para el FOA-HDA

La Figura 14 muestra el diagrama del Foco de Atención con Potencialidades en Profundidad de Alto Nivel. El objetivo de FOA-HDA es darle prioridad a las características que se encuentran dentro de un área determinada, ya que tal área representa una potencialidad del espacio. Esta área se define por medio de un mapa de potencialidades (Ver la Sección 6.1.2).

El algoritmo de FOA-HDA es sencillo. Justo después de que se calculan los *mapas visuales*, estos se enmascaran por medio de los rangos definidos en el mapa de potencialidades. Los mapas visuales y la imagen de profundidad tienen las mismas dimensiones. Por tanto, se recorre cada pixel de cada mapa visual; si el valor de distancia en un pixel está fuera de los rangos definidos por el mapa de potencialidades, su valor correspondiente en el mapa visual es igualado a "0".

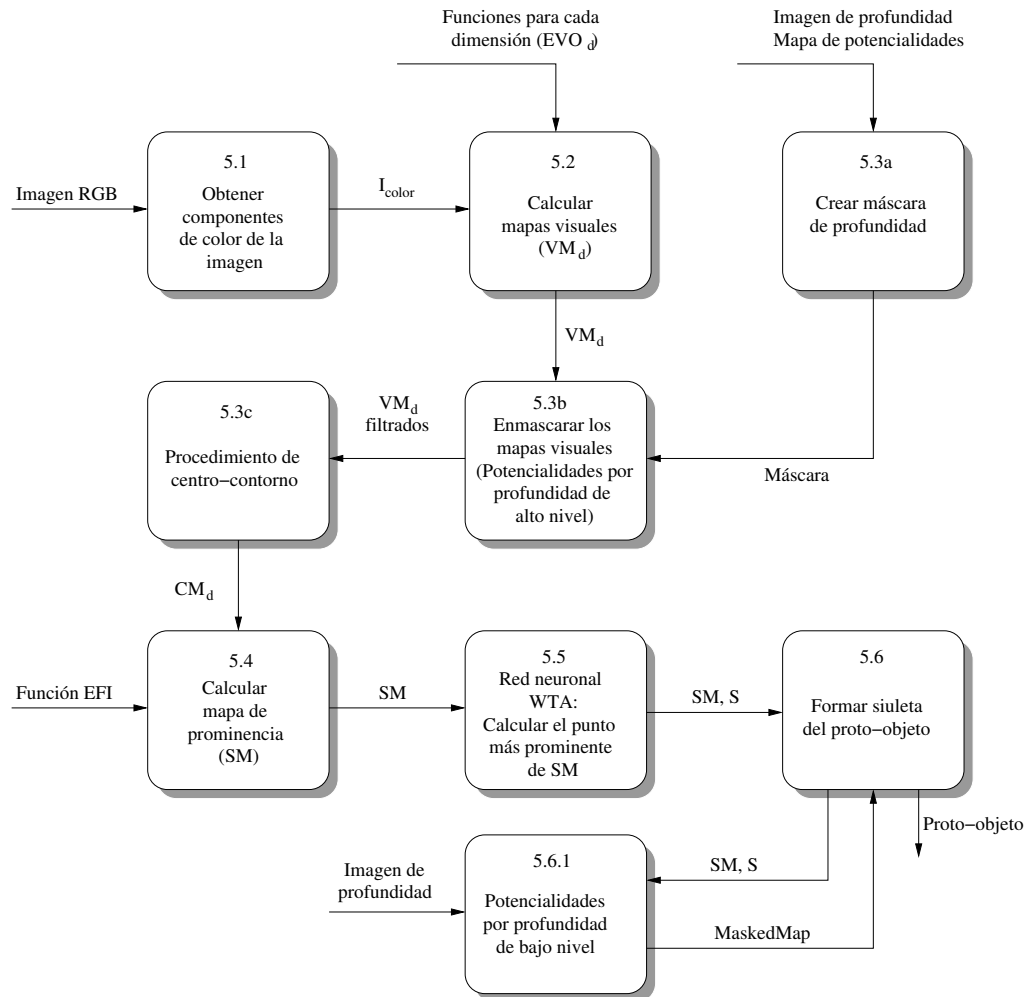
Otra forma de hacer el procedimiento es crear una imagen binaria basada en la imagen de profundidad. Sólo aquellos pixeles que tienen un valor de profundidad dentro del rango, se representan con "1" en la imagen binaria; al resto se les asigna el valor de "0". A esta imagen binaria se le nombra como *máscara*. Después, se obtiene el producto Hadamard de la máscara y el mapa visual.

Se le llama producto Hadamard a la operación binaria que toma dos matrices,  $A$  y  $B$ , para obtener una tercera matriz  $C$  de tal manera que cada elemento  $C_{ij}$  es igual  $A_{ij} \times B_{ij}$ . Tanto la matriz  $A$  como  $B$  deben tener el mismo tamaño; la matriz resultante  $C$  tendrá las mismas dimensiones. Este tipo de operación se usa en los algoritmos de compresión con pérdida, por ejemplo, para las imágenes de formato JPEG.

Así pues, a cada mapa visual se le aplica una operación de producto Hadamard con

la máscara. Esto hará que los píxeles de los mapas visuales sean igualados a “0” si su distancia correspondiente no se encuentra en los rangos establecidos por el mapa de potencialidades. Los píxeles que sí se encuentran dentro de los rangos no se ven afectados por esta máscara.

El propósito de este procedimiento es simular de forma sencilla la manera en que se le da prioridad a la información visual que se encuentra a cierta distancia y lugar.



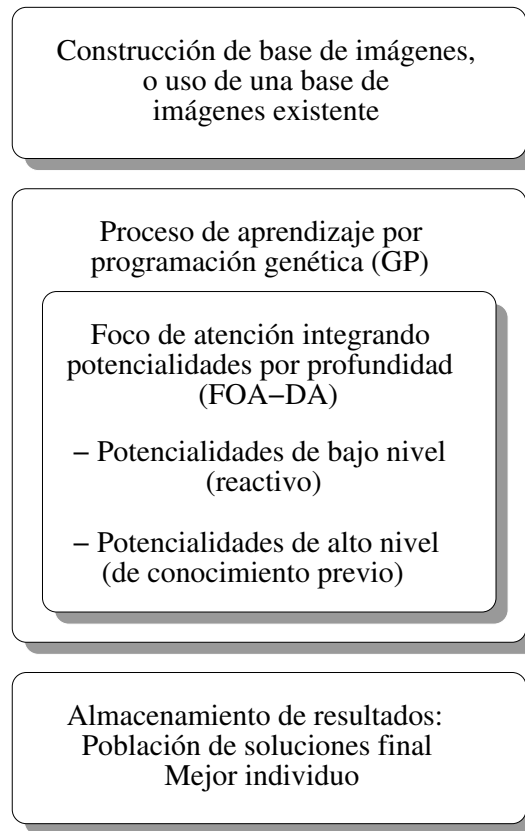
**Figura 14:** Diagrama de contexto de FOA-HDA.

### 6.1.6. Diagrama general de FOA-DA

La Figura 15 muestra cómo está conformado el sistema en general. De manera más detallada, se puede ver el diagrama de contexto del sistema completo en la Figura 16. Se puede observar que se agregó un procedimiento para construir bases de imágenes,

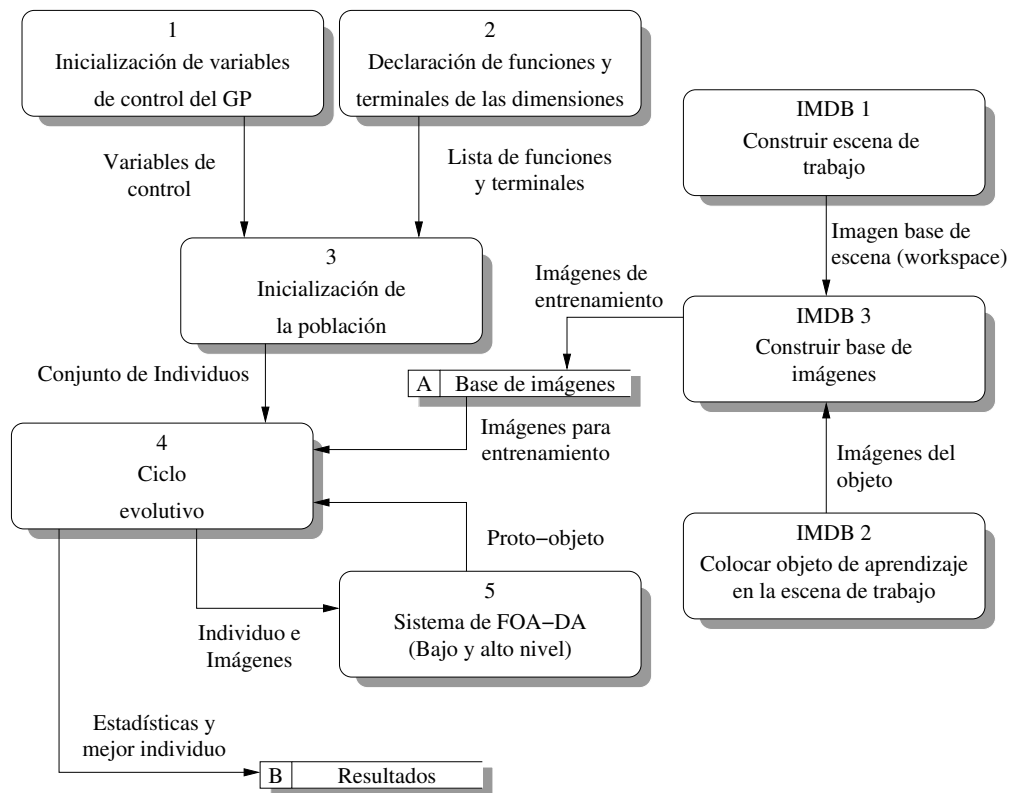


sin embargo, el sistema general es independiente de este procedimiento, y puede trabajar con bases creadas con anterioridad o construidas en el momento de ejecución. Este tema se detalla en el Capítulo 7.



**Figura 15:** Sistema general.

En el presente capítulo se explica el paradigma de potencialidades en función de la distancia, y los algoritmos que se implementaron para representar en principio este paradigma. En el Capítulo 7 se proponen un conjunto de procedimientos para integrar los elementos necesarios para realizar los experimentos. Además también se propone un método para llevar a cabo la etapa de aprendizaje de forma semi-automática, sin la necesidad de tener una base de imágenes previa.



**Figura 16:** Diagrama de contexto del sistema general.



## Capítulo 7. Hacia un aprendizaje automatizado

---

En inteligencia artificial existen una gran variedad de sistemas. Todos ellos trabajan con el objetivo de resolver un problema, o un conjunto de ellos, por medio de distintas técnicas llamadas *heurísticas*. Estas heurísticas se basan en un conjunto de reglas o normas que el programa debe seguir para lograr obtener una solución. Estas reglas son escritas por los programadores o diseñadores de estos sistemas, e igualmente depende de ellos cambiarlas. Tales sistemas tendrán la capacidad de resolver un número de situaciones. Sin embargo, a diferencia de un ser inteligente como lo es el ser humano, no podrán resolver problemas que estén más allá de lo que dicten este conjunto de reglas. En otras palabras, no tienen la capacidad de deducir nuevas formas de resolver el mismo problema, o de intentar resolver un nuevo problema.

Mientras más se avanza en la inteligencia artificial, más se acerca a simular el comportamiento de los seres inteligentes. Para ello, se crean nuevas formas y propuestas de sistemas para IA, los cuales tienen la capacidad de deducir nuevas soluciones a los problemas, esta vez, sin que el diseñador del sistema conozca la forma de la solución, aunque sí el resultado deseado.

Ya que las reglas de las heurísticas no son escritas por el diseñador, los sistemas son construidos con la capacidad de buscar estas reglas y formar soluciones. Y para lograr esto, todos estos sistemas pasan por la etapa conocida como “aprendizaje”, que como si fuera un ser inteligente natural, el sistema pasa por una etapa en la que se enfrenta a distintas *instancias de un problema*; también se le dan *ejemplos de los resultados* que debería obtener. Dependiendo de la técnica de aprendizaje, el sistema construye distintas soluciones y crea una base de conocimientos, con los cuales solucionará instancias similares para las cuales ha sido entrenado.

Para los sistemas de visión artificial, la fuente básica de información son las imágenes, o secuencias de video, en una variedad de formatos. A estos conjuntos se les llama bases de imágenes; estas bases son utilizadas como el conjunto de ejemplos de problemas y resultados necesarias para la etapa de aprendizaje.

Hacer una base de imágenes lleva tiempo y esfuerzo. Es por eso que muchas bases

de imágenes, creadas con distintos propósitos, son publicadas. También se hace para permitir la corroboración de los resultados obtenidos con los distintos sistemas que se prueban.

Aunque es una metodología que ha funcionado bien, existe otro tipo de modelo que funciona sin la necesidad de una base de imágenes del modo tradicional. Estos son precisamente los sistemas naturales, como por ejemplo, el cerebro humano. Un ser humano, y de hecho cualquier otro organismo que observa y aprende con base a la observación, con sólo unas cuantas imágenes de un objeto, es capaz de identificarlo en una cantidad enorme de escenarios, bajo condiciones de luz diferentes, e incluso en oclusión. Adicional a esto, un ser humano común no necesita mucho tiempo para aprender un nuevo objeto; por otro lado, los tiempos del aprendizaje de sistemas artificiales pueden ir desde unos minutos u horas, hasta meses.

Es evidente entonces que la etapa de aprendizaje utiliza una buena cantidad de recursos y tiempo, y por ello es interesante proponer nuevas formas que permitan agilizar el proceso, analizando sus ventajas y desventajas.

### **7.1. Idea principal**

El sistema de FOA de Dozal *et al.* (2014) ha mostrado una característica interesante, y es que, en las estadísticas de entrenamiento, la GP pareciera encontrar las mejores soluciones al problema durante las primeras generaciones. Esto da paso a la pregunta: ¿Se puede utilizar esta característica para mejorar los tiempos de entrenamiento?

Los resultados mostrados por el GP hacen considerar que con algunos cambios, se puede encaminar la forma de aprendizaje para ser más parecida a la manera en que se cree que funciona el cerebro humano. También, si la ADS y AVC son emulaciones del funcionamiento cerebral, sería interesante que se sometiesen a situaciones similares a las que se enfrenta el cerebro humano todos los días. Se debe notar que para lograr estos cambios, también es necesario modificar la forma en que se obtienen las imágenes para el aprendizaje.

En este trabajo de investigación, sólo se usará la ADS para probar una primera versión

de lo que se llamará entrenamiento o aprendizaje *en vivo*.

## 7.2. Justificación

La primer razón por la que se quiere realizar este experimento es porque se ha observado la rapidez con la que los sistemas basados en programación cerebral encuentran las soluciones para las instancias en las que se han entrenado. Ahora se quiere probar la capacidad de este sistema para buscar esas soluciones en un ambiente más cercano a la forma en que los seres humanos aprenden día con día.

Se debe considerar que es necesario desarrollar sistemas de este tipo. El entrenamiento con bases de imágenes es muy bueno para probar las capacidades de los sistemas. Pero hoy en día la robótica y los sistemas artificiales se enfrentan a situaciones cada vez más cercanas al mundo natural, y deben ser capaces de reaccionar en base al conocimiento que están adquiriendo en el momento, mientras realizan otras tareas.

Incluso si no se puede lograr el 100 % de automatización, se estará aportando ideas en esa dirección y se podrán descubrir y enfrentar nuevos problemas para lograr finalmente la automatización del aprendizaje.

## 7.3. Montaje

El sistema para esta investigación se compone de las siguientes partes:

- Una computadora con el software del sistema de FOA-DA, y la API de libfreenect instalada.
- Una cámara Kinect Xbox 360 (RGB-D) montada en un tripié, y con una vía de conexión a una computadora.
- Una escena de trabajo, preferentemente con luz controlada, en donde se colocarán los objetos para el aprendizaje.

Como se usa una cámara RGB-D, todos los experimentos deben ser hechos en interiores. Por la naturaleza del procedimiento, es necesario que el escenario esté libre de movimientos y cambios ajenos al objeto de aprendizaje.

## 7.4. Descripción del procedimiento

### 7.4.1. Imágenes de entrenamiento

Uno de los trabajos que más utilizan la intervención humana es la creación de *bases de imágenes*. Esto es con justa razón: los humanos son los expertos en visión que embeben el conocimiento en estas bases.

Para su entrenamiento, la ADS utiliza como unidad básica un par de imágenes. Una imagen de color, y otra imagen llamada *de entrenamiento*. La imagen de entrenamiento es una imagen binaria que representa en blanco el objetivo de la imagen, y en negro el resto. Respectivamente, esos valores están representados con los valores numéricos “1” y “0”. La mancha de blanco representa el objeto de interés para la observación y debe ser creada de forma manual. Evidentemente, debe ser lo más exacta posible, de manera que todos los píxeles que correspondan al objeto estén pintados de blanco.

Como parte de la automatización, es necesario que la computadora construya esta imagen sin intervención humana, utilizando las imágenes que la cámara está captando en el momento. Para lograrlo se siguieron estos pasos:

1. Se captura una imagen base (nombrada como “workspace” o *espacio de trabajo*). Esta imagen debe contener todos los elementos que se usarán en la escena, a excepción del objeto de aprendizaje. Es importante que la iluminación de la escena sea estable durante todo el experimento, es decir, sin cambios de intensidad.
2. Se captura una imagen de la misma escena, pero con el objeto de aprendizaje presente.
3. La imagen del objeto se contrasta con la imagen base. El resultado se pasa por una variedad de filtros para suavizar el ruido y obtener una figura que sea lo más exacta posible a la figura del objeto.

Existen una gran variedad de algoritmos para obtener la figura de la diferencia entre la imagen del objeto y la imagen base. El tema de este tipo de algoritmos se llama *sustracción de fondo*. Varwani *et al.* (2013) presentan un resumen y la referencia de varios de estos algoritmos. Por cuestiones de tiempo, para este trabajo de investigación se im-

plementó un algoritmo provisional y práctico, pero de capacidad limitada; simplemente se usa un umbral para filtrar los valores de la diferencia de color entre la imagen de base y del objeto. Si el valor de la diferencia está dentro del umbral, el pixel es clasificado como objeto; el resto se clasifican como fondo.

Debe notarse que este procedimiento implica que la cámara debe estar fija en todo momento. Cualquier movimiento en el ángulo de la escena genera interferencias en los algoritmos de sustracción de fondo. Por la misma razón, también se debe asegurar de que sólo el objeto de interés es la única cosa diferente en la escena.

Como se busca minimizar la intervención humana, se da paso a un problema para la representación de la *generalidad del objeto*. Es decir, si únicamente se toma la fotografía de un solo lado del objeto, se estará sesgando el aprendizaje; el sistema sólo aprenderá sobre el objeto desde el punto de vista que se le presenta en la fotografía. Una solución sería intercalar la toma de fotografías con cambios manuales de la posición del objeto. Sin embargo esto implicaría de nuevo la intervención humana. Tampoco se puede mover la cámara al rededor del objeto, pues se invalidaría el algoritmo de sustracción de fondo.

Para resolver el problema, se consiguió una base giratoria de motor (Figura 17). Esta base gira a una velocidad de un ciclo cada 40 segundos aproximadamente (dependiendo del peso que esté soportando), tiene un diámetro de 23 cm, y soporta hasta 10 kg. Este tipo de bases son usadas para mostradores, como soportes de maniquíes, y como bases para fotografía profesional.

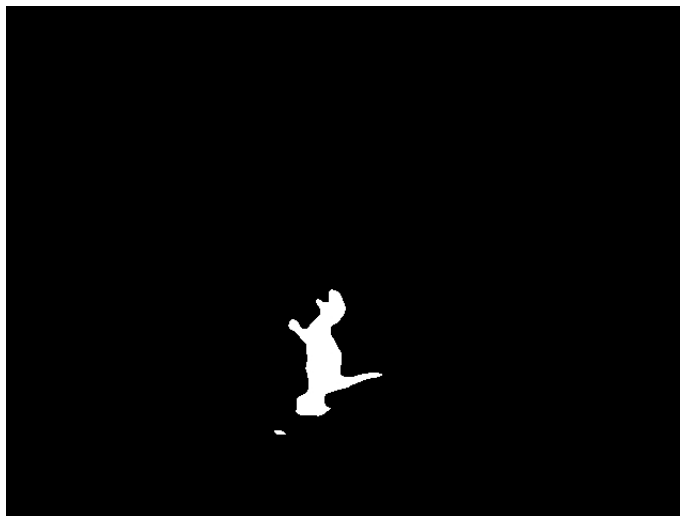
Colocando la base giratoria en la escena, ya no es necesaria la intervención humana para mover el objeto a distintas posiciones, únicamente para encenderla y apagarla. Además, gira lo suficientemente lento como para que la cámara no tenga problemas para captar una imagen nítida.

Con esto es posible obtener cualquier serie de imágenes del objeto con distintas posiciones, y de forma automática. En la Figura 18 se puede observar un ejemplo del resultado.





**Figura 17:** Base Giratoria para fotografía.



**Figura 18:** Arriba: Imagen RGB. Abajo: Imagen binaria de entrenamiento.

## 7.5. Algoritmo del experimento

Se usaron dos enfoques para realizar los experimentos: el entrenamiento con base de imágenes, y el entrenamiento en vivo. Para el primero, la base de imágenes se crea de manera automática. En el segundo, cada generación de la GP usará sólo una imagen; esta imagen se captura en el momento de la ejecución.

### 7.5.1. Algoritmo de entrenamiento con base de imágenes

1. Se prepara la escena de trabajo. Se debe asegurar de que no haya objetos que se muevan mientras se estén capturando las imágenes. También deben evitarse los espejos y las pantallas, es decir, cosas que puedan mostrar cambios aunque no se muevan.
2. Se coloca la cámara en una posición fija, y la base giratoria sin el objeto.
3. Se captura la imagen de la escena. Esta imagen es nombrada como “workspace” o *espacio de trabajo*.
4. Se coloca el objeto de aprendizaje sobre la base giratoria y se enciende la base.
5. Se toma una serie de fotografías al objeto. Considerando que el ciclo de esta base toma 40 segundos, aproximadamente, y que la Kinect puede capturar video a 30 frames por segundo, se pueden tomar al rededor de 1,200 imágenes diferentes sin necesidad de intervención humana. Para el caso de este trabajo de tesis, sólo se tomaron secuencias de 300 imágenes aproximadamente, de las cuales se realizó un muestreo para el entrenamiento.
6. Se sigue el algoritmo para crear las imágenes de entrenamiento (Véase la Sección 7.4.1).
7. La base de imágenes se organiza en una carpeta. En esta carpeta se crea un archivo que contiene la lista de las imágenes. Este archivo siempre tiene el mismo nombre: *imdb* (en este trabajo se usó MATLAB, por lo que este archivo tiene extensión “.mat”). El nombre del archivo viene del inglés *Image-Data-Base*.
8. Dentro de la misma carpeta se crea otra llamada *imgTrDb*. En estase guardan todas las imágenes de entrenamiento. Toda imagen de entrenamiento tiene el mismo nombre que su par RGB. Los formatos de las imágenes son: color - ppm, profundidad - pgm, entrenamiento - pgm. Se eligieron estos formatos por que son los más

básicos que existen para imágenes, lo que aumenta su versatilidad y compatibilidad. Su desventaja es el tamaño del archivo, debido a que la información no se comprime como en un jpg o png. La organización de los archivos se hace para facilitar su acceso durante la ejecución del programa de FOA-DA.

9. Con la base de imágenes creada, se apaga la base giratoria, y se puede proceder a correr el código de FOA-DA o se puede comenzar a crear otra base.

### 7.5.2. Algoritmo de entrenamiento en vivo

1. Se prepara el escenario de la misma manera que para una base de imágenes.
2. Se coloca la cámara en posición fija, y la base giratoria sin el objeto.
3. Se captura la fotografía base de la escena. Esta imagen es nombrada como “workspace” o *espacio de trabajo*.
4. Se coloca el objeto sobre la base, y estase enciende.
5. Se procede a ejecutar el programa de FOA-DA. La diferencia principal que existe en este entrenamiento es que el programa ya no lee la lista de la base de imágenes designada por el archivo *imbd*, en cambio, el programa captura una fotografía al inicio de cada ciclo de la GP. En ese momento se produce la imagen de entrenamiento. La imagen capturada será la única foto que componga la lista *images<sub>tr</sub>* (Véase Algoritmo 1).
6. El resto de ejecución del algoritmo FOA-DA sigue los mismos pasos.
7. Una vez que termine la ejecución, se puede apagar la base giratoria. También se pueden hacer cambios en la escena, ya que con cada nueva ejecución, se captura una nueva imagen de espacio de trabajo.

### 7.6. Ventajas y limitaciones

Como se puede observar en el procedimiento, la automatización no es completa. Aún existen fases en las que el humano debe realizar acciones. Sin embargo, se propone la automatización de actividades que suelen consumir una gran cantidad de tiempo, estas son: hacer una imagen de entrenamiento, construir una base de imágenes, y utilizar sólo una imagen por ciclo de entrenamiento.

Como se hace mención, la imagen de entrenamiento se crea de forma manual. Es

necesario utilizar algún programa de edición de imágenes para crear una figura de color blanco sobre un fondo negro, y que esta figura cubra todos los píxeles que pertenecen al objeto de interés. El procedimiento presentado en este trabajo permite automatizar este proceso; sin embargo, también tiene su limitante: la calidad de la figura. La figura que cubre al objeto, de hecho, es suficientemente exacta. El problema radica en otros dos puntos: el ruido en la imagen, y las sombras.

El ruido en una imagen se representa como pequeños cambios aleatorios en las intensidades de color en cada uno de los componentes del RGB. Dependiendo de la calidad de la cámara, la intensidad y el tipo de iluminación, este ruido puede tener la fuerza suficiente como para que un programa de sustracción de fondo llegue a clasificarlo como un objeto nuevo en la escena. Por supuesto, con el debido cuidado en la iluminación de la escena y una serie de filtros, el ruido puede ser minimizado lo suficiente como para que no genere píxeles blancos sobre otra cosa que no sea el objeto.

El otro problema son las sombras generadas por el nuevo objeto. Una sombra representa un cambio de intensidad mucho más fuerte que el ruido ambiental. Sortear este problema ya no es tan evidente. Sin embargo, ya existen varios algoritmos diseñados para detectar sombras. Para este trabajo de investigación, el algoritmo de sustracción de fondo implementado es muy sencillo, y por tanto, las sombras sí representan un problema, pues son detectadas como objetos nuevos. Sin embargo, es sólo un problema temporal, pues no hay más que implementar alguno de los algoritmos que ya resuelven este tipo de situaciones. Aún en el estado actual del programa, con la debida iluminación, una sombra representa una parte muy pequeña de la figura, y dada la naturaleza de la GP, las soluciones generadas buscarán cubrir el objeto como prioridad, ya que su aptitud depende de ello.

La automatización de la construcción de una base de imágenes es una mejora directa de automatizar la creación de una imagen de entrenamiento. Este tipo de bases no son más que un grupo organizado de imágenes con su respectiva imagen de entrenamiento. Editar cientos o miles de imágenes para crear estas bases es una actividad que requiere de mucho tiempo y esfuerzo, y puede resultar una tarea abrumadora para aquellos que no tienen experiencia en la edición de imágenes. Entonces, un algoritmo que ayude a

editar y organizar de forma automática estos archivos resulta de gran ayuda, y agiliza el proceso para implementar nuevas bases de imágenes.

Por supuesto, el método también tiene sus limitantes. Todas las bases de imágenes creadas así necesitan una imagen de fondo. Esto implica que si se requiere hacer un cambio de ángulo, de acercamiento, de iluminación, etc., es necesario tomar una nueva imagen de fondo. A pesar de ello, la velocidad con la que se pueden crear nuevas bases de imágenes permite incluir todos estos cambios de manera rápida y sencilla. También, probar nuevas ideas que necesiten objetos locales, o que no se encuentren en las bases de imágenes públicas, es mucho más rápido. Esto es una ventaja considerable, aún si las imágenes automáticas tienen algunos defectos.

Finalmente, con el aprendizaje en vivo se explora la capacidad y el comportamiento de algoritmos que simulan una parte del funcionamiento de la corteza visual. Sus ventajas y limitaciones son exploradas en los capítulos restantes.

En el presente capítulo se detallan los algoritmos que se siguieron para automatizar casi por completo el proceso de aprendizaje. Aún existen muchas otras formas e ideas para lograr la automatización completa de esta etapa, lo cual es un objetivo importante a seguir, debido a la inclusión de los sistemas inteligentes y de robótica en la vida diaria del ser humano. Se expone como mayor ventaja, la velocidad de implementación de nuevas bases de imágenes para entrenamiento, mientras que sus limitaciones pueden ser sorteadas con la implementación de algoritmos más robustos de sustracción de fondo.

En el Capítulo 8 se presentan los resultados obtenidos para las propuestas de este trabajo de tesis, comparándose con los resultados del sistema predecesor bajo las mismas condiciones de uso.

## Capítulo 8. Experimentos y resultados

### 8.1. Experimentos realizados

En el trabajo de tesis se produjeron dos algoritmos: FOA-LDA y FOA-HDA. Para poder tener un punto de comparación y observar los cambios en los resultados de los nuevos sistemas, también se realizaron ejecuciones con la versión original: ADS/FOA. Los tres algoritmos se corrieron en dos formas de entrenamiento: con bases de imágenes, y en vivo. Así pues, se realizaron las siguientes ejecuciones:

**Tabla 1: Lista de Experimentos.**

Experimento	Aprendizaje	Ejecuciones	RGB	D	KFold
FOA	En vivo	5	30	0	No
	Base de Imágenes	30	84	0	70/14
FOA-LDA	En vivo	5	30	30	No
	Base de Imágenes	30	84	84	70/14
FOA-HDA	En vivo	5	30	30	No
	Base de Imágenes	30	84	84	70/14

Para los experimentos en vivo, se decidió realizar 5 ejecuciones para cada algoritmo. Esto se debe a que, a diferencia de un experimento con bases de imágenes, el aprendizaje en vivo necesita una cámara en el momento de ejecución. Por ello, no es posible correr más de una ejecución a la vez. Si se quisieran correr más ejecuciones, se necesitaría más cámaras, y otras computadores en donde correr estas ejecuciones.

Para los experimentos con bases de imágenes, se decidió correr una validación cruzada, o K-Fold (Ver Apéndice C). Para esto se usaron 84 pares de imágenes (Color/Profundidad), repartidas en seis grupos, construyendo finalmente seis combinaciones distintas de 70 pares de imágenes de entrenamiento y 14 pares de imágenes de prueba. Se realizaron cinco ejecuciones por combinación. Se debe notar que para los experimentos de FOA, no se usaron las imágenes de profundidad.

Los experimentos fueron realizados en dos modelos de computadoras:

- Una laptop Lenovo<sup>®</sup> G40-70 de arquitectura x86-64, procesador Intel<sup>®</sup> Core™ i3-4005U 1.70 GHz, 4 GB de RAM, tarjeta gráfica Intel<sup>®</sup> HD Graphics 4400. Sistema operativo Linux openSUSE 13.2.

- Cuatro estaciones de trabajo de modelo Dell Precision T7600 de arquitectura x86-64, procesador Intel<sup>®</sup> Xeon<sup>®</sup> E5-2609 2.40 GHz de 8 núcleos, 8 GB de memoria RAM, tarjeta gráfica NVIDIA<sup>®</sup> GF100GL Quadro<sup>®</sup> 4000. Sistema operativo Linux openSUSE 13.1

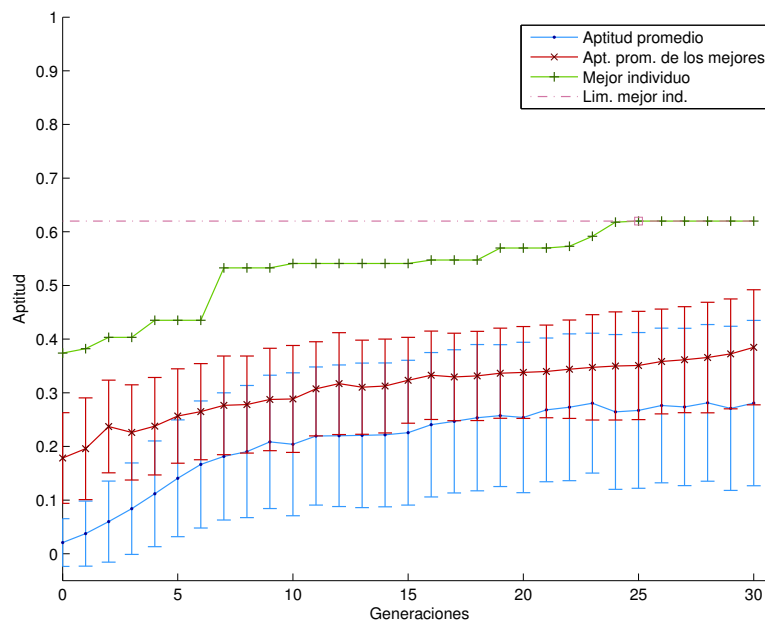
Todas las computadoras trabajaron con MATLAB<sup>®</sup> R2011b.

## 8.2. Resultados de entrenamiento con base de imágenes

En esta sección se muestran las gráficas y tablas de los resultados generales obtenidos.

### 8.2.1. FOA

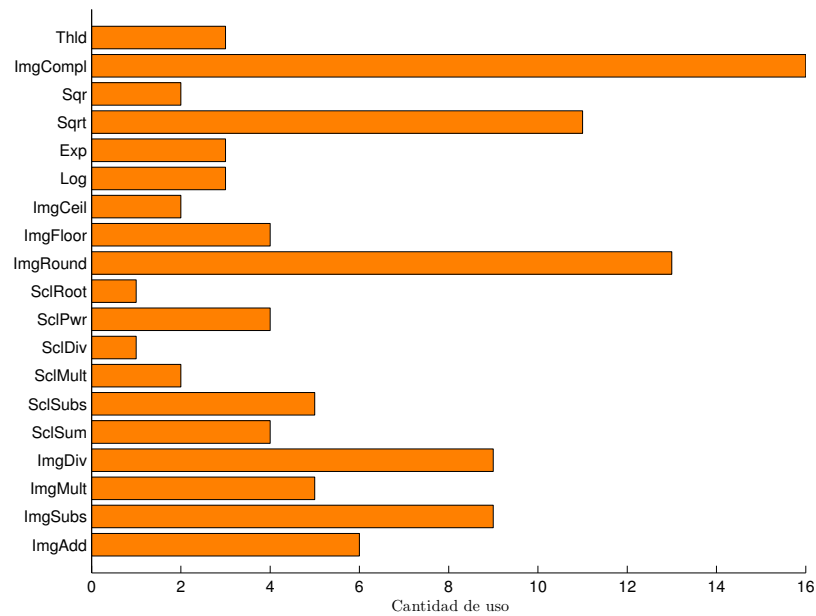
La gráfica de la Figura 19 muestra la evolución promedio de 30 ejecuciones. Los mejores individuos alcanzaron un máximo de 0.6200 en aptitud. Para todos los casos de estos experimentos, la máxima aptitud que puede obtener un individuo es 1.



**Figura 19:** Evolución promedio en 30 ejecuciones para FOA entrenados con base de imágenes.

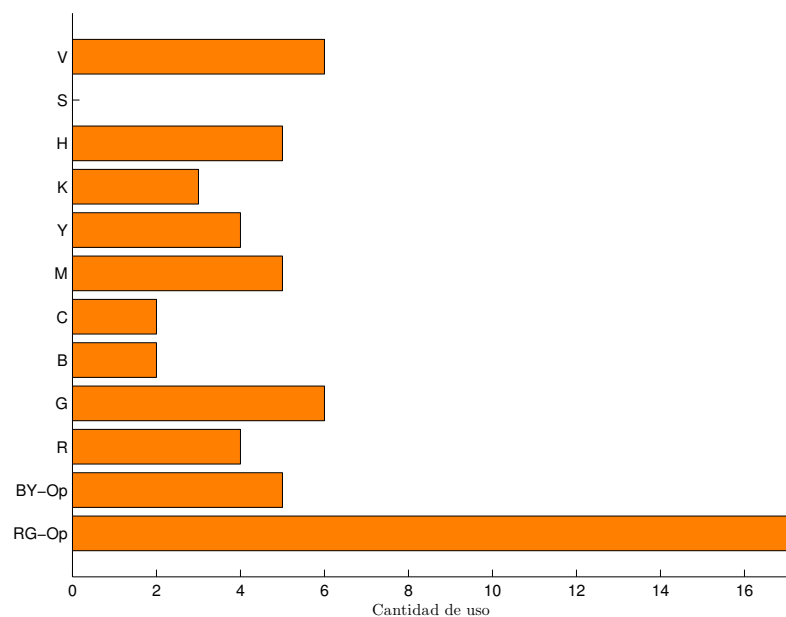
De la Figura 20 a la 25 se muestra la cantidad de uso de las funciones y terminales implementadas para la construcción de los individuos de cada dimensión. Las figuras 26 y 27 muestran, de la misma manera, las funciones y terminales usadas para la función de

integración de características (FI). Nótese que las terminales para FI son los resultados de las funciones de cada dimensión (color, orientación, forma, e intensidad). También se debe recordar que la función de intensidad ya está definida (Sección 3.2.1), por lo que es la única dimensión que no tiene gráfica de uso. El conteo se realiza para el conjunto compuesto por el mejor individuo de cada ejecución, es decir, 30 individuos. El Apéndice E contiene una lista de las definiciones para cada función y terminal.

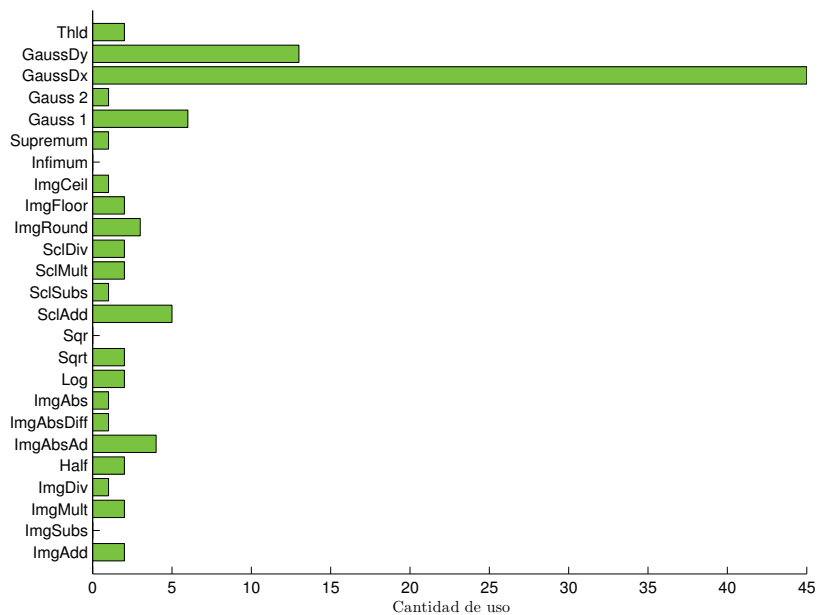


**Figura 20:** Cantidad de uso de funciones de color en 30 individuos de FOA entrenados con base de imágenes.

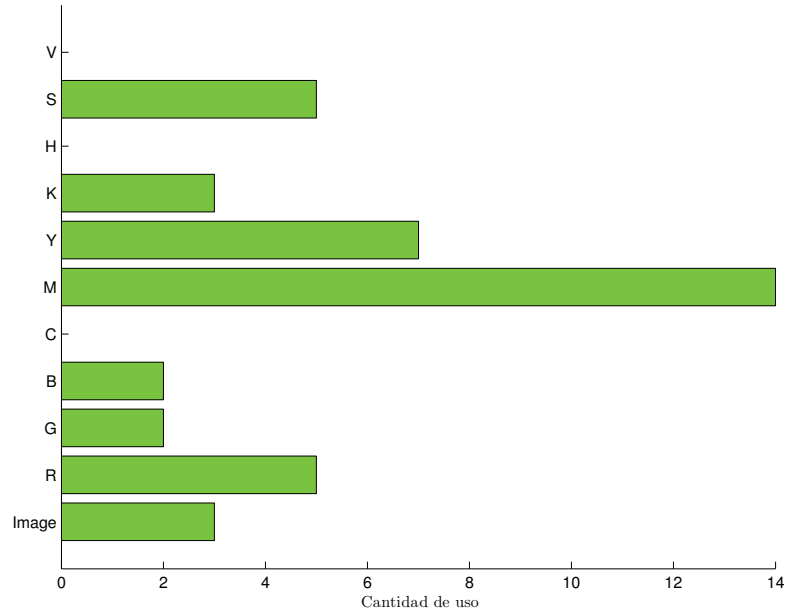




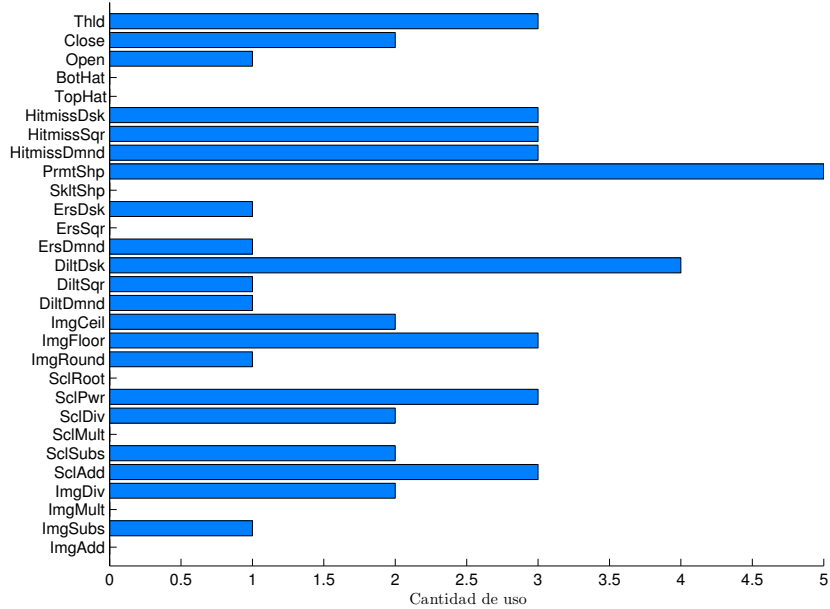
**Figura 21:** Cantidad de uso de terminales de color en 30 individuos de FOA entrenados con base de imágenes.



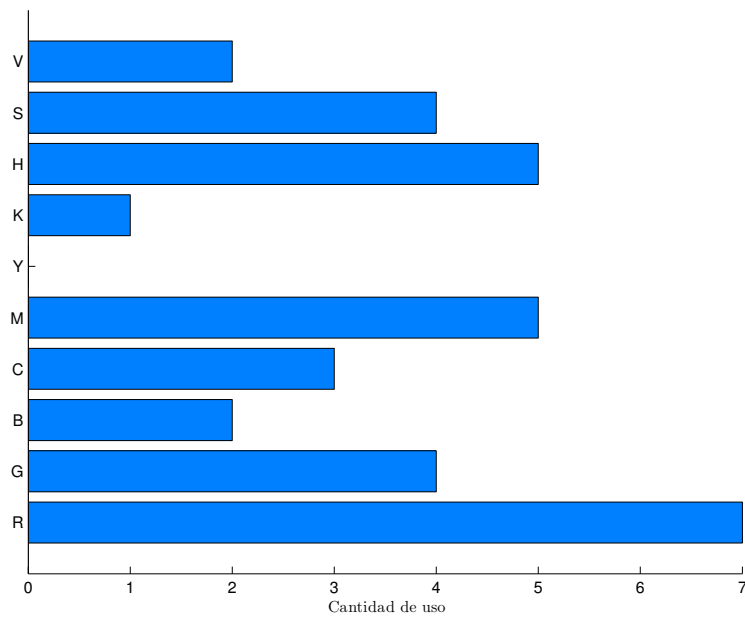
**Figura 22:** Cantidad de uso de funciones de orientación en 30 individuos de FOA entrenados con base de imágenes.



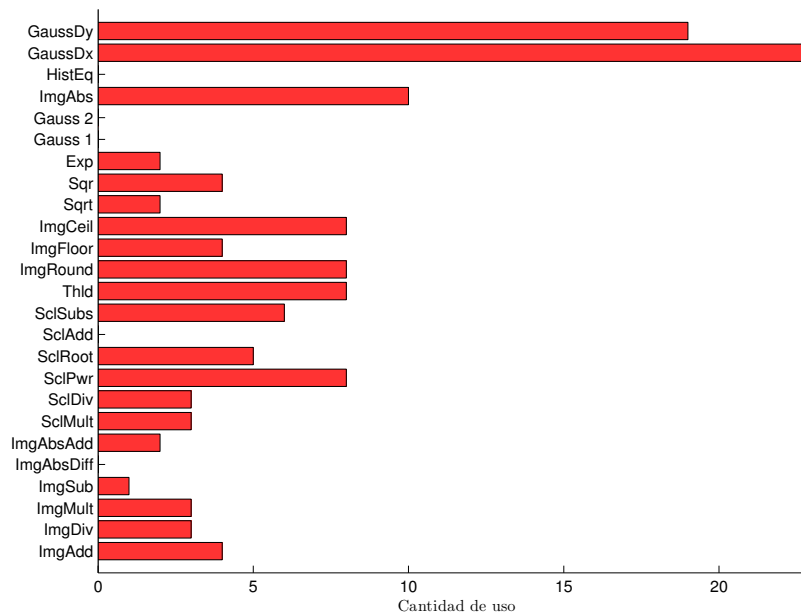
**Figura 23:** Cantidad de uso de terminales de orientación en 30 individuos de FOA entrenados con base de imágenes.



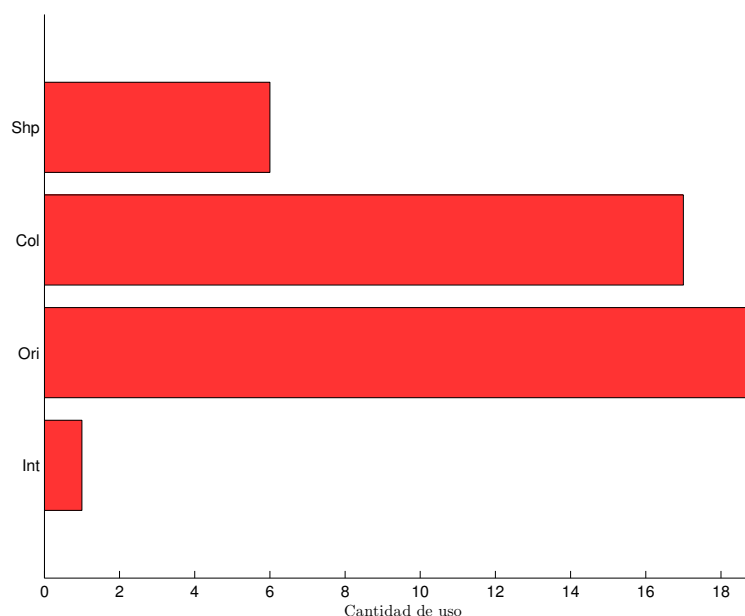
**Figura 24:** Cantidad de uso de funciones de forma en 30 individuos de FOA entrenados con base de imágenes.



**Figura 25:** Cantidad de uso de terminales de forma en 30 individuos de FOA entrenados con base de imágenes.



**Figura 26:** Cantidad de uso de funciones de FI en 30 individuos de FOA entrenados con base de imágenes.



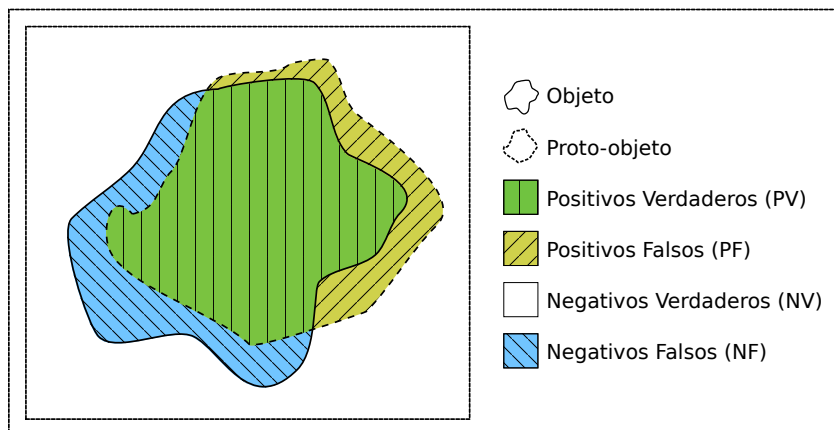
**Figura 27:** Cantidad de uso de terminales de FI en 30 individuos de FOA entrenados con base de imágenes.

Para comprender las gráficas restantes, se debe repasar el concepto de positivos y negativos, que a su vez pueden ser verdaderos o falsos. Estos se usan para medir la certeza del proto-objeto (PO) producido por la solución, en comparación con el proto-objeto de la imagen de entrenamiento (POE). Cada pixel de la imagen producida se clasifica en una de las cuatro clases. Para esta explicación se define un pixel como *marcado* si su valor es mayor a “0”, *i.e.*, que forma parte de la figura del PO.

Todos los pixeles marcados en la imagen de PO son considerados *positivos*. Un *positivo verdadero* es aquel pixel que está marcado en PO y también en POE. Un positivo falso es aquel pixel que está marcado para PO, pero no así para POE. En concepto, un positivo verdadero es cuando la máquina deduce que tal pixel es parte del objeto, y efectivamente lo es (corroborando en POE); un positivo falso es cuando la máquina establece que tal pixel es parte del objeto, cuando en realidad no lo es.

Todos los pixeles que no son marcados en PO son clasificados como *negativos*. Un *negativo verdadero* es aquel pixel que no está marcado en PO y tampoco en POE. Un *negativo falso* es aquel pixel que no está marcado en PO, pero sí está marcado en POE. En concepto, los pixeles negativos verdaderos son los que la máquina clasifica como ajenos al objeto observado, y efectivamente no son parte del objeto. En un negativo falso,

la máquina deduce que tal pixel no es parte del objeto, cuando en realidad sí lo es. En la Figura 28 se puede observar una representación gráfica de las cuatro clasificaciones.



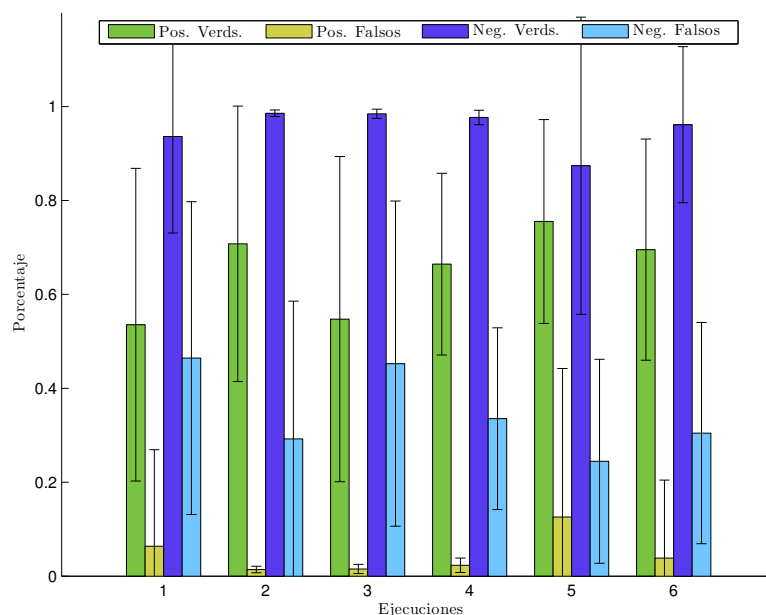
**Figura 28:** Clasificación de píxeles en la imagen del proto-objeto.

Los resultados de la etapa de prueba muestran las capacidades de las mejores soluciones ante imágenes que no formaron parte de su entrenamiento. Los entrenamientos se hicieron por medio de un k-fold de seis grupos. Se usaron 84 imágenes, por lo que 70 quedaron para entrenamiento y las otras 14 para la etapa de prueba. Para cada grupo, el conjunto de entrenamiento y prueba fueron diferentes. En el Apéndice C se detalla el procedimiento para construir los grupos de imágenes usados para la validación cruzada. También, para cada grupo se realizaron cinco ejecuciones.

En la Figura 29 se puede observar la gráfica de los promedios de porcentaje de positivos y negativos. Cada grupo de cuatro barras verticales representa los resultados de los mejores cinco individuos (el mejor de cada entrenamiento) con un grupo de imágenes de prueba (cada grupo es diferente a los demás). Una gráfica del *resultado ideal* sería aquella en la que los positivos verdaderos y los negativos verdaderos lleguen al 100% (“1” en la gráfica), mientras que los positivos falsos y negativos falsos estén en 0%.

Para obtener los porcentajes, se hacen las siguientes relaciones: la cantidad de píxeles positivos verdaderos, y de píxeles negativos falsos, se dividen entre el total de píxeles marcados en POE; esto quiere decir que la mancha del objeto definida en POE está compuesta de los positivos verdaderos (que se marcaron correctamente) y los negativos falsos (los que faltaron por marcar). De forma similar, la cantidad de píxeles negativos verda-

deros, y positivos falsos, son divididas entre el total de pixeles que no están marcados en POE; esto quiere decir que el área no marcada en POE está compuesta de los negativos verdaderos (que acertadamente, no fueron marcados) y los positivos falsos (que fueron marcados por error).

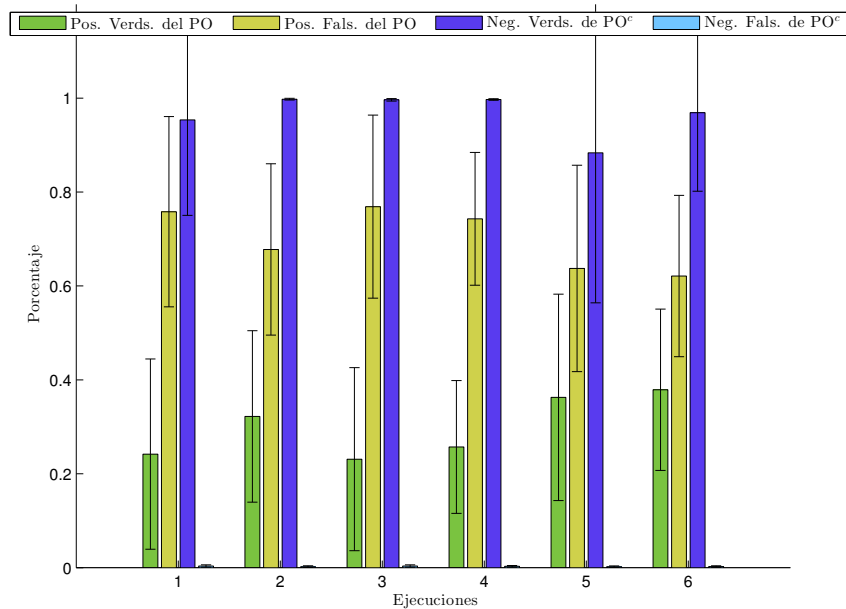


**Figura 29:** Etapa de prueba para FOA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada.

Es importante conocer qué tantos pixeles se marcaron correctamente, pero también es interesante saber la composición del propio proto-objeto. Es decir, qué porción de la mancha es correcta; si cubre más al objeto, o a otras cosas en la imagen. Para esto se construyó la gráfica de la Figura 30. Esta gráfica está organizada de la misma manera que la de la Figura 29, y la fuente de los datos es la misma.

Para conocer los porcentajes, se hacen las siguientes operaciones: tanto la cantidad de positivos verdaderos, como los positivos falsos, son divididos entre la suma de estos mismos. Con esto, se puede saber qué porción del proto-objeto está representado por positivos verdaderos (pixeles marcados correctamente) y positivos falsos (pixeles marcados incorrectamente). Por otro lado, se tiene el complemento del proto-objeto ( $PO^c$ ), que conforma toda el área que no fue marcada en la imagen del PO. De la misma forma que los positivos, los negativos verdaderos y falsos son divididos entre la suma de ellos

mismos, obteniendo así la porción del complemento representada por negativos positivos (pixeles que, acertadamente, no fueron marcados) y negativos falsos (pixeles que no fueron marcados, pero sí eran parte del objeto).

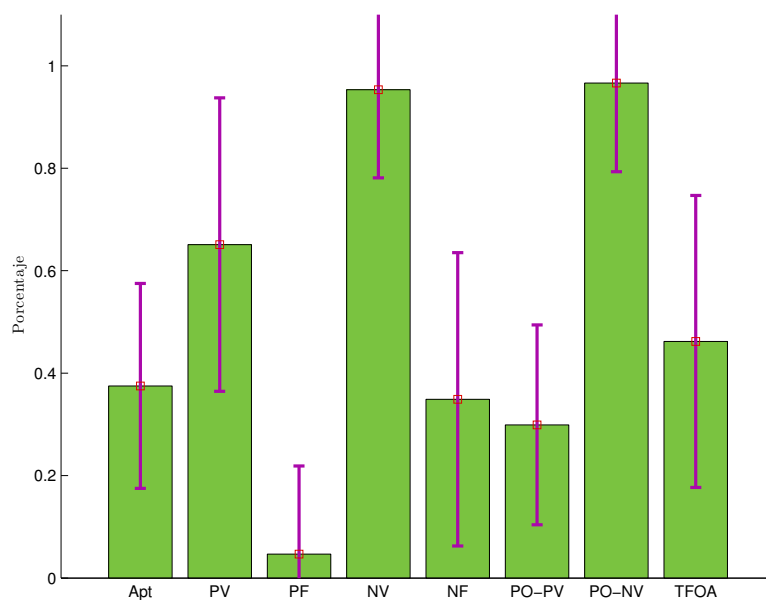


**Figura 30:** Etapa de prueba para FOA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos del proto-objeto para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada.

La gráfica de la Figura 31 presenta un resumen de los promedios de las 30 ejecuciones realizadas. Los parámetros mostrados son: aptitud (Apt), positivos verdaderos (PV), positivos falsos (PF), negativos verdaderos (NV), negativos falsos (NF), positivos verdaderos del proto-objeto (PO-PV), negativos verdaderos del complemento del proto-objeto (PO-NV), y finalmente la *certeza de punto de foco de atención* (TFOA). Este último parámetro se detalla con la Figura 32.

Como se detalla en la Sección 6.1.3, el punto de foco de atención está representado por el pixel más prominente obtenido por la red neuronal WTA. En el algoritmo del FOA, este punto sirve para comparar y seleccionar uno de los mapas, los cuales son producidos durante la ejecución, para el formado del proto-objeto. Su posición con respecto al objeto no es tan relevante, al punto de que este puede encontrarse fuera del área del objeto.

Sin embargo, para FOA-LDA y FOA-HDA, el punto del FOA es la base del algoritmo, pues es el pixel de referencia para formar los vecindarios y calcular la pertenencia de los



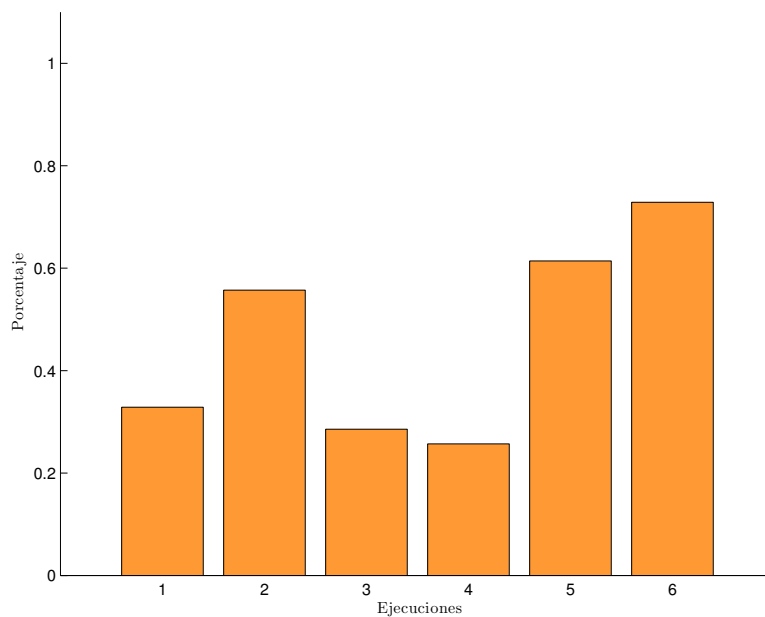
**Figura 31:** Estadística general de la etapa de prueba para 30 individuos de FOA entrenados con base de imágenes.

pixeles, dependiendo de su relación en profundidad con la que posee el punto de foco de atención. Por tanto, una función que logre colocar el punto del FOA dentro del objeto, tiene mejores oportunidades para producir un proto-objeto acertado. Por otro lado, una función que produzca un punto del FOA fuera del objeto, tiene muchas menos posibilidades de localizar dicho objeto. Este comportamiento es congruente con el enfoque de los ojos en la distancia. Por tanto, es interesante verificar que tan certera fue la ubicación de este punto para los tres algoritmos. Así pues, la Figura 32 muestra la gráfica de certeza del FOA.

La Tabla 2 muestra la información representada en las gráficas de esta sección. Las etiquetas y sus significados son: aptitud (Apt), positivos verdaderos (PV), positivos falsos (PF), negativos verdaderos (NV), negativos falsos (NF), positivos verdaderos del proto-objeto (PO:PV), positivos falsos del proto-objeto (PO:PF), negativos verdaderos del complemento del proto-objeto (PO:NV), negativos falsos del complemento del proto-objeto (PO:NF), y certeza del punto de foco de atención (TFOA).

Las descripciones realizadas para las gráficas de esta sección aplican para el resto de los resultados.





**Figura 32:** Etapa de prueba para FOA entrenado con base de imágenes - Certeza de punto de foco de atención para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada.

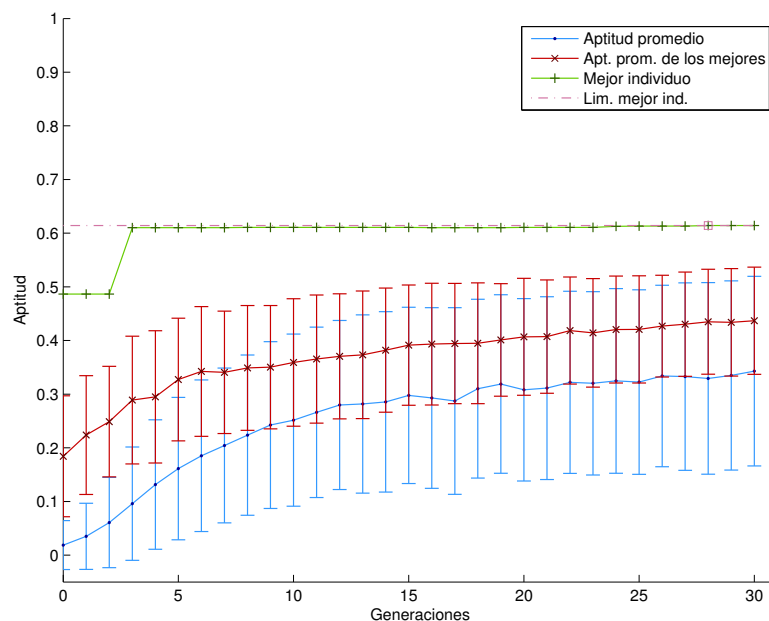
**Tabla 2: Resultados del K-Fold para FOA entrenado con base de imágenes.**

	KF 1	KF 2	KF 3	KF 4	KF 5	KF 6	Prom. Gnrl.
<b>Apt</b>	30.40 ±22.31	41.92 ±18.09	30.63 ±22.29	33.11 ±10.47	42.78 ±21.59	46.19 ±17.08	37.51 ±20.01
<b>PV</b>	53.55 ±33.31	70.77 ±29.33	54.74 ±34.63	66.45 ±19.35	75.54 ±21.71	69.54 ±23.56	65.10 ±28.63
<b>PF</b>	6.36 ±20.56	1.41 ±0.70	1.53 ±0.98	2.31 ±1.54	12.60 ±31.64	3.85 ±16.62	4.68 ±17.21
<b>NV</b>	93.64 ±20.56	98.59 ±0.70	98.47 ±0.98	97.69 ±1.54	87.40 ±31.64	96.15 ±16.62	95.32 ±17.21
<b>NF</b>	46.45 ±33.31	29.23 ±29.33	45.26 ±34.63	33.55 ±19.35	24.46 ±21.71	30.46 ±23.56	34.90 ±28.63
<b>PO:PV</b>	24.19 ±20.27	32.22 ±18.25	23.11 ±19.49	25.70 ±14.14	36.26 ±21.98	37.89 ±17.18	29.90 ±19.51
<b>PO:PF</b>	75.81 ±20.27	67.78 ±18.25	76.89 ±19.49	74.30 ±14.14	63.74 ±21.98	62.11 ±17.18	70.10 ±19.51
<b>PO:NV</b>	95.36 ±20.33	99.78 ±0.19	99.65 ±0.26	99.71 ±0.18	88.37 ±31.97	96.91 ±16.74	96.63 ±17.29
<b>PO:NF</b>	0.35 ±0.25	0.22 ±0.19	0.35 ±0.26	0.29 ±0.18	0.21 ±0.18	0.23 ±0.17	0.28 ±0.22
<b>TFOA</b>	32.86 ±18.63	55.71 ±28.30	28.57 ±20.82	25.71 ±18.63	61.43 ±35.93	72.86 ±15.49	46.19 ±28.49

## 8.2.2. FOA-LDA

Esta sección muestra los resultados obtenidos para el k-fold de FOA-LDA. La descripción de las gráficas y sus significados es descrita en la Sección 8.2.1.

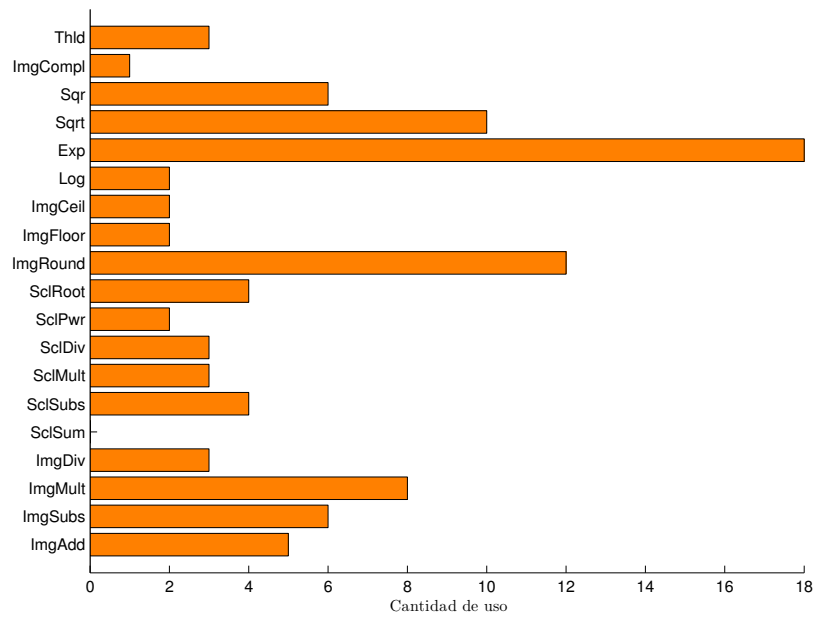
La gráfica de la Figura 33 muestra la evolución promedio de 30 ejecuciones. Los mejores individuos alcanzaron un máximo de 0.6141 en aptitud.



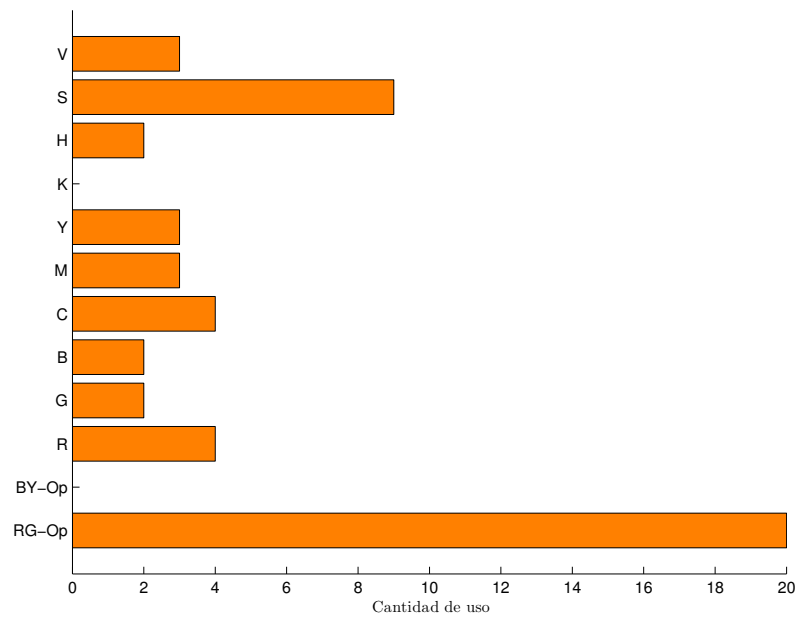
**Figura 33:** Evolución promedio en 30 ejecuciones para FOA-LDA entrenados con base de imágenes.

**Tabla 3: Resultados del K-Fold para FOA-LDA entrenado con base de imágenes.**

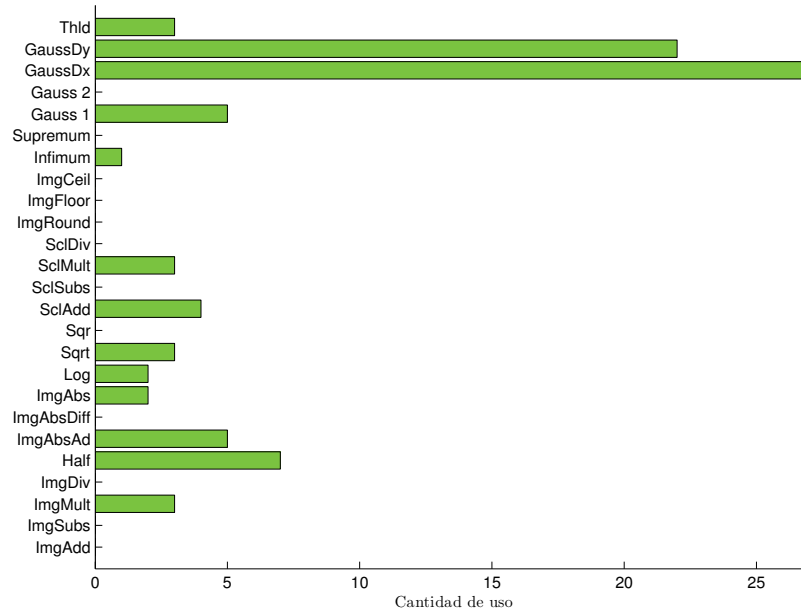
	KF 1	KF 2	KF 3	KF 4	KF 5	KF 6	Prom. Gnrl.
<b>Apt</b>	49.91 ±17.84	37.90 ±17.06	42.87 ±20.84	46.92 ±16.14	40.37 ±21.36	52.08 ±18.80	45.01 ±19.33
<b>PV</b>	74.90 ±21.07	67.34 ±25.90	59.43 ±24.75	69.91 ±21.51	68.95 ±29.40	74.00 ±22.19	69.09 ±24.70
<b>PF</b>	1.36 ±2.02	2.14 ±2.86	1.53 ±2.27	1.36 ±1.91	2.18 ±2.70	1.38 ±2.42	1.66 ±2.40
<b>NV</b>	98.64 ±2.02	97.86 ±2.86	98.47 ±2.27	98.64 ±1.91	97.82 ±2.70	98.62 ±2.42	98.34 ±2.40
<b>NF</b>	25.10 ±21.07	32.66 ±25.90	40.57 ±24.75	30.09 ±21.51	31.05 ±29.40	26.00 ±22.19	30.91 ±24.70
<b>PO:PV</b>	41.41 ±19.57	28.18 ±14.36	36.85 ±21.32	37.47 ±15.01	31.09 ±17.93	44.03 ±19.43	36.51 ±18.82
<b>PO:PF</b>	58.59 ±19.57	71.82 ±14.36	63.15 ±21.32	62.53 ±15.01	68.91 ±17.93	55.97 ±19.43	63.49 ±18.82
<b>PO:NV</b>	99.80 ±0.15	99.75 ±0.17	99.69 ±0.16	99.75 ±0.18	99.75 ±0.22	99.79 ±0.18	99.75 ±0.18
<b>PO:NF</b>	0.20 ±0.15	0.25 ±0.17	0.31 ±0.16	0.25 ±0.18	0.25 ±0.22	0.21 ±0.18	0.25 ±0.18
<b>TFOA</b>	78.57 ±28.57	45.71 ±23.47	67.14 ±10.83	84.29 ±12.78	75.71 ±17.20	84.29 ±17.05	72.62 ±22.20



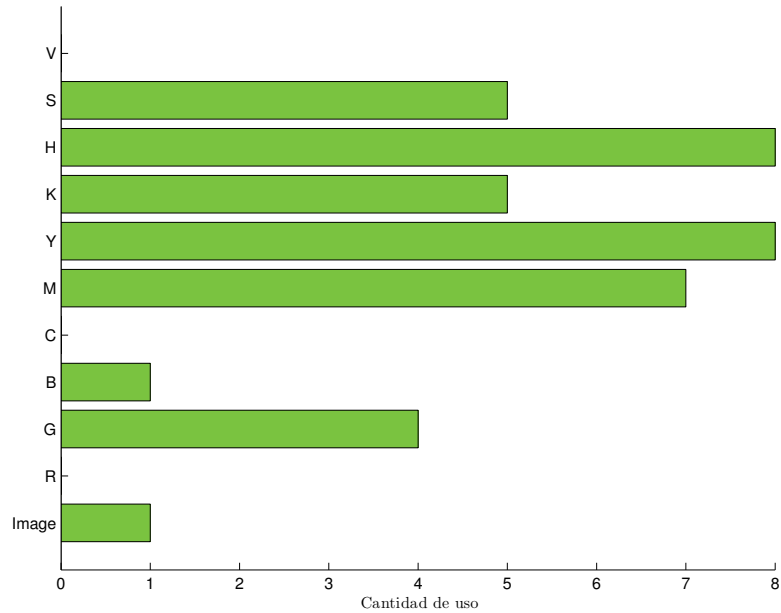
**Figura 34:** Cantidad de uso de funciones de color en 30 individuos de FOA-LDA entrenados con base de imágenes.



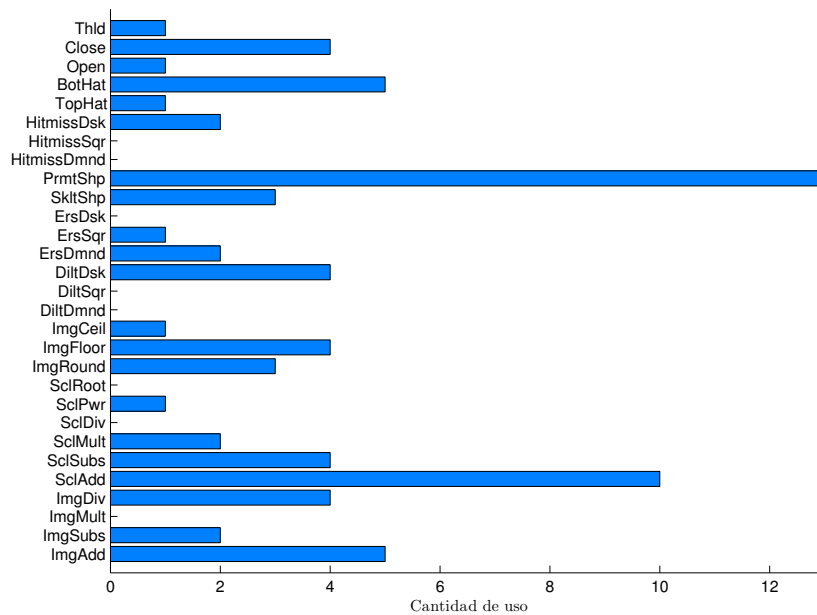
**Figura 35:** Cantidad de uso de terminales de color en 30 individuos de FOA-LDA entrenados con base de imágenes.



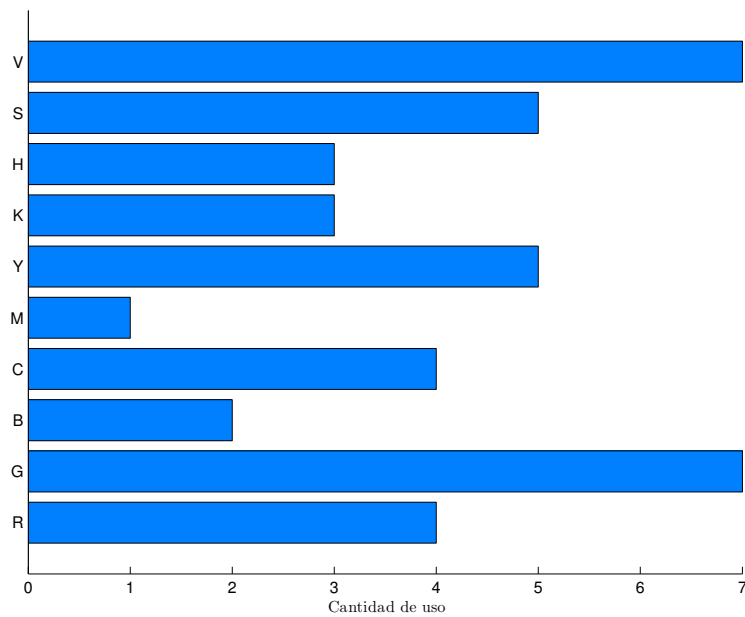
**Figura 36:** Cantidad de uso de funciones de orientación en 30 individuos de FOA-LDA entrenados con base de imágenes.



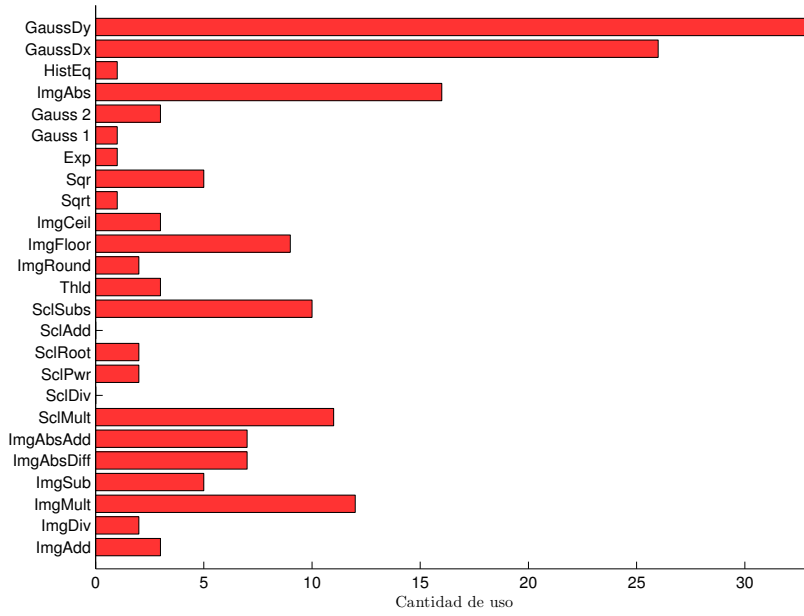
**Figura 37:** Cantidad de uso de terminales de orientación en 30 individuos de FOA-LDA entrenados con base de imágenes.



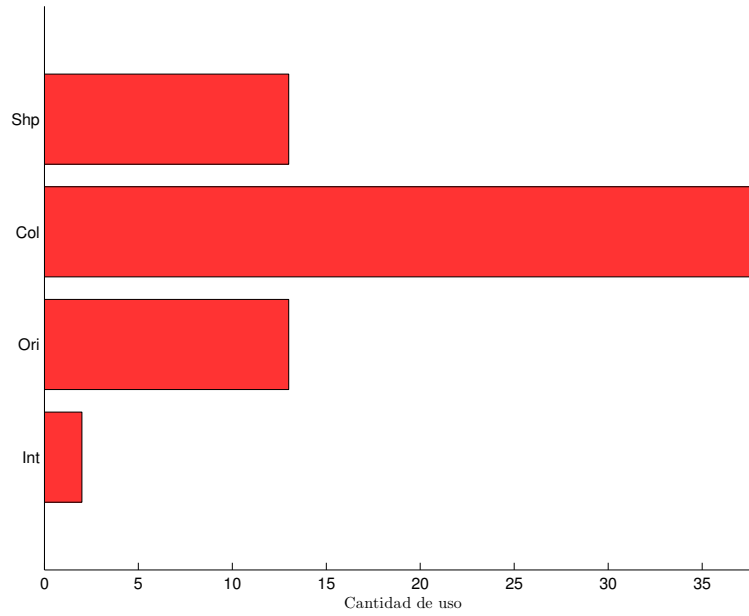
**Figura 38:** Cantidad de uso de funciones de forma en 30 individuos de FOA-LDA entrenados con base de imágenes.



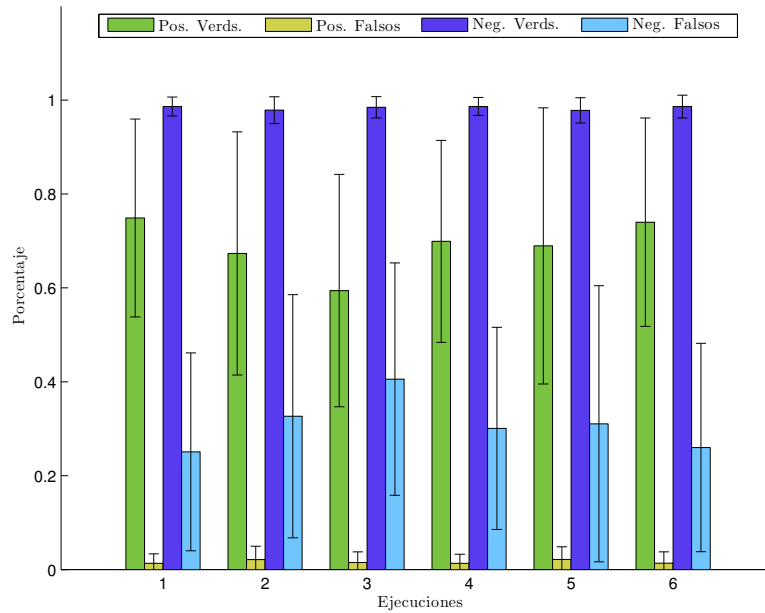
**Figura 39:** Cantidad de uso de terminales de forma en 30 individuos de FOA-LDA entrenados con base de imágenes.



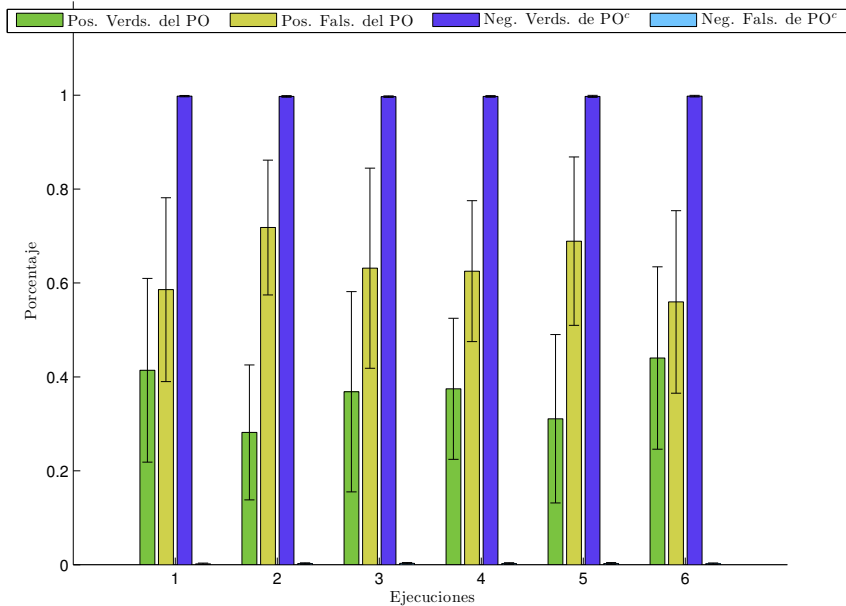
**Figura 40:** Cantidad de uso de funciones de FI en 30 individuos de FOA-LDA entrenados con base de imágenes.



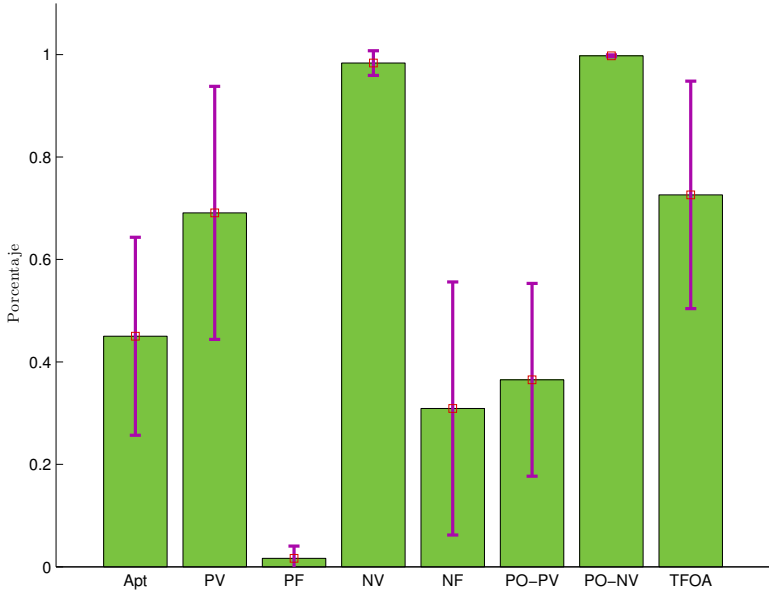
**Figura 41:** Cantidad de uso de terminales de FI en 30 individuos de FOA-LDA entrenados con base de imágenes.



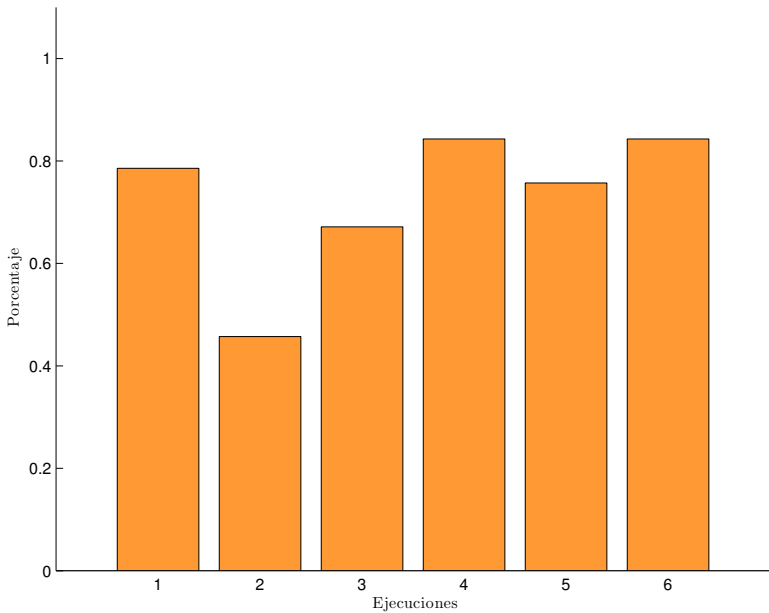
**Figura 42:** Etapa de prueba para FOA-LDA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada.



**Figura 43:** Etapa de prueba para FOA-LDA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos del proto-objeto para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada.



**Figura 44:** Estadística general de la etapa de prueba para 30 individuos de FOA-LDA entrenados con base de imágenes.



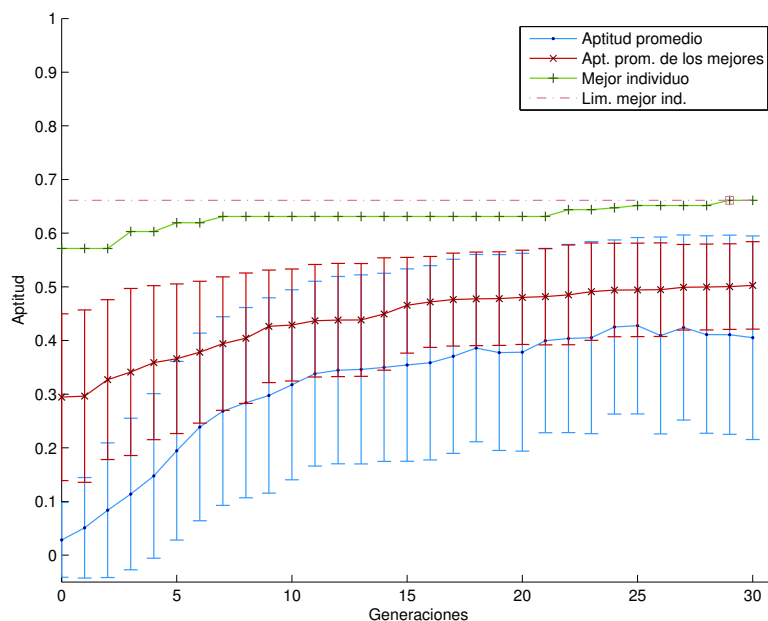
**Figura 45:** Etapa de prueba para FOA-LDA entrenado con base de imágenes - Certeza de punto de foco de atención para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada.



### 8.2.3. FOA-HDA

Esta sección muestra los resultados obtenidos para el k-fold de FOA-HDA. La descripción de las gráficas y sus significados es descrita en la Sección 8.2.1.

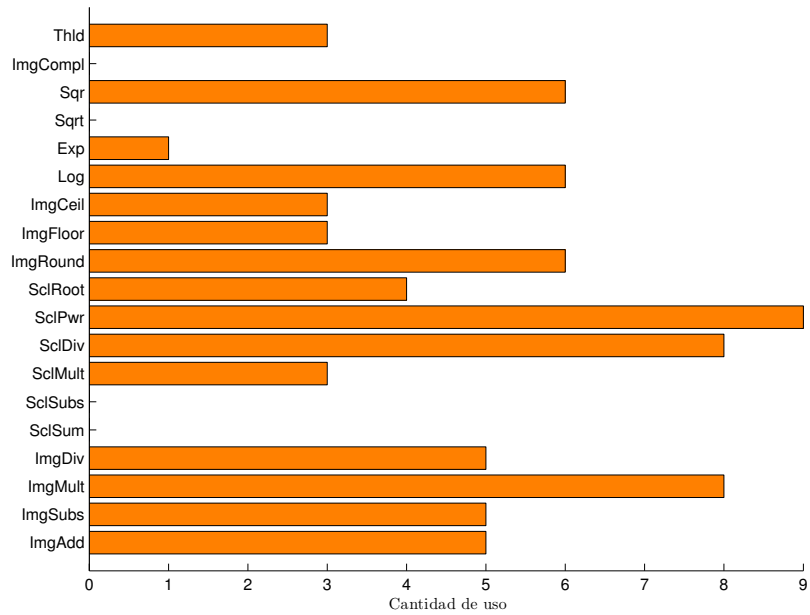
La gráfica de la Figura 46 muestra la evolución promedio de 30 ejecuciones. Los mejores individuos alcanzaron un máximo de 0.6611 en aptitud.



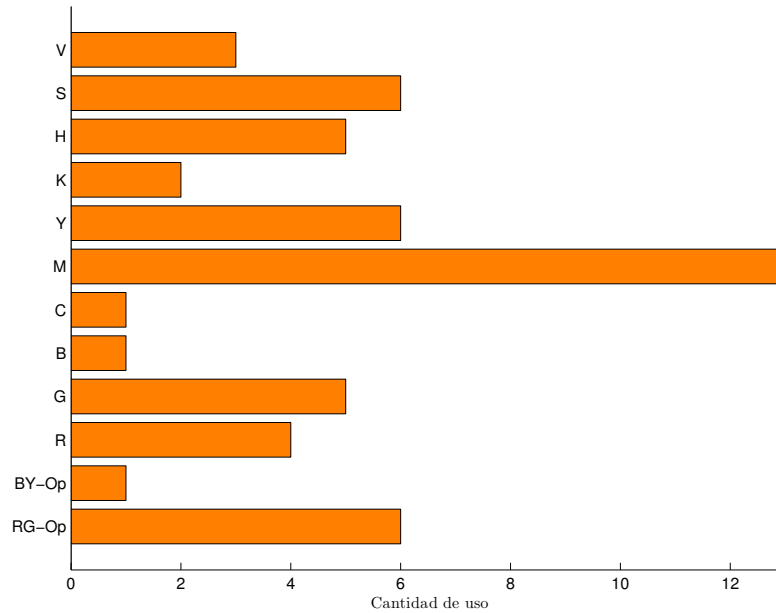
**Figura 46:** Evolución promedio en 30 ejecuciones para FOA-HDA entrenados con base de imágenes.

**Tabla 4: Resultados del K-Fold para FOA-HDA entrenado con base de imágenes.**

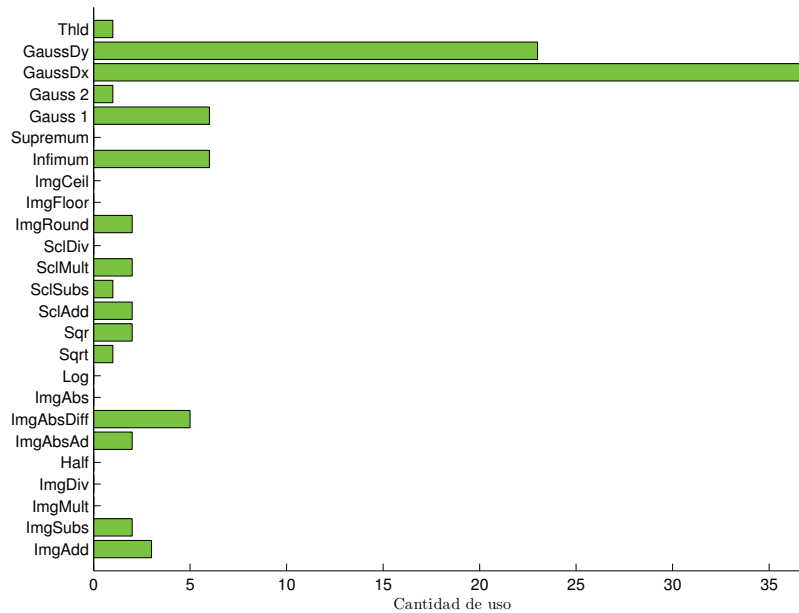
	KF 1	KF 2	KF 3	KF 4	KF 5	KF 6	Prom. Gnrl.
<b>Apt</b>	46.96 ±27.38	48.77 ±26.28	48.79 ±22.69	52.38 ±21.43	44.84 ±29.99	51.17 ±23.32	48.82 ±25.33
<b>PV</b>	74.60 ±23.48	69.03 ±24.50	72.25 ±19.27	74.15 ±22.04	79.64 ±15.87	74.54 ±12.58	74.03 ±20.21
<b>PF</b>	15.26 ±33.74	11.23 ±29.35	9.94 ±27.55	7.56 ±23.92	28.10 ±42.39	15.02 ±33.64	14.52 ±32.79
<b>NV</b>	84.74 ±33.74	88.77 ±29.35	90.06 ±27.55	92.44 ±23.92	71.90 ±42.39	84.98 ±33.64	85.48 ±32.79
<b>NF</b>	25.40 ±23.48	30.97 ±24.50	27.75 ±19.27	25.85 ±22.04	20.36 ±15.87	25.46 ±12.58	25.97 ±20.21
<b>PO:PV</b>	40.93 ±26.44	42.63 ±24.62	41.89 ±23.13	44.48 ±19.82	40.23 ±28.44	45.36 ±23.30	42.59 ±24.36
<b>PO:PF</b>	59.07 ±26.44	57.37 ±24.62	58.11 ±23.13	55.52 ±19.82	59.77 ±28.44	54.64 ±23.30	57.41 ±24.36
<b>PO:NV</b>	99.77 ±0.17	99.64 ±0.48	99.59 ±0.65	99.76 ±0.28	99.60 ±0.56	99.51 ±0.76	99.65 ±0.53
<b>PO:NF</b>	0.23 ±0.17	0.36 ±0.48	0.41 ±0.65	0.24 ±0.28	0.40 ±0.56	0.49 ±0.76	0.35 ±0.53
<b>TFOA</b>	88.57 ±3.91	91.43 ±7.82	90.00 ±6.39	85.71 ±16.75	95.71 ±3.91	88.57 ±13.92	90.00 ±9.67



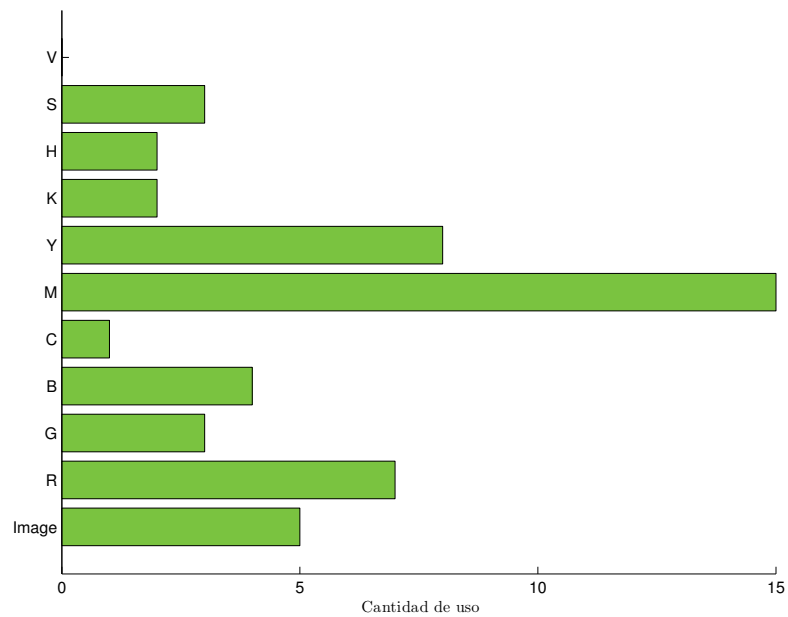
**Figura 47:** Cantidad de uso de funciones de color en 30 individuos de FOA-HDA entrenados con base de imágenes.



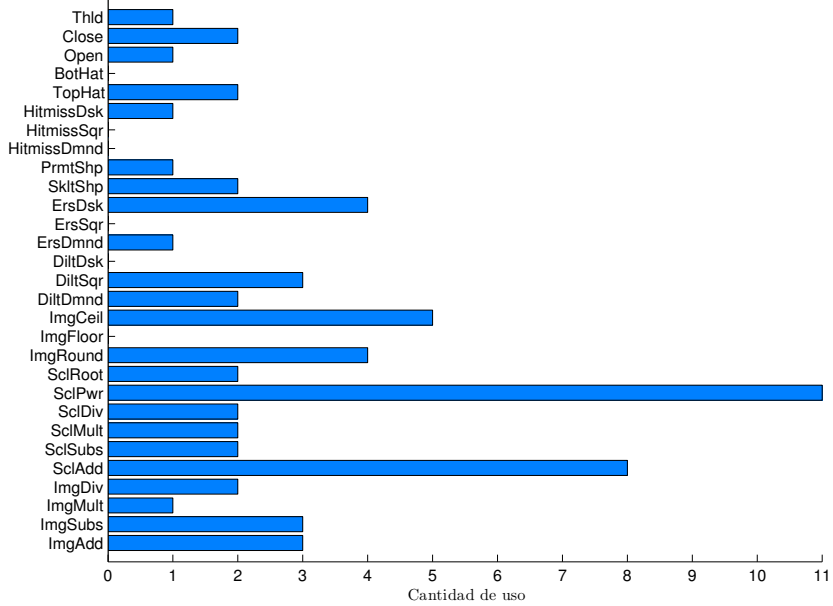
**Figura 48:** Cantidad de uso de terminales de color en 30 individuos de FOA-HDA entrenados con base de imágenes.



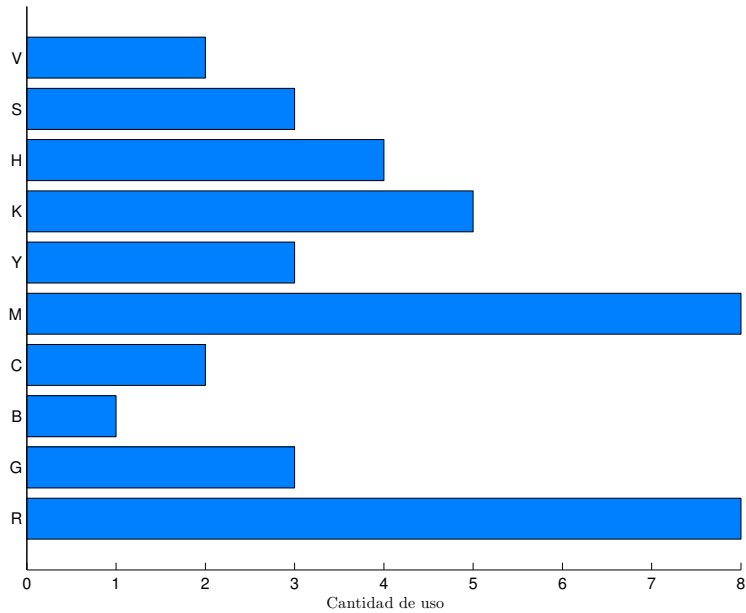
**Figura 49:** Cantidad de uso de funciones de orientación en 30 individuos de FOA-HDA entrenados con base de imágenes.



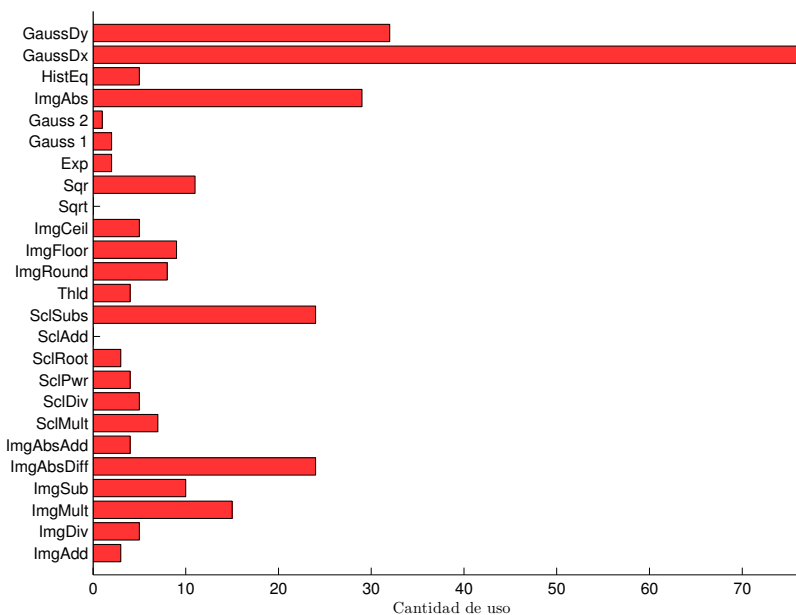
**Figura 50:** Cantidad de uso de terminales de orientación en 30 individuos de FOA-HDA entrenados con base de imágenes.



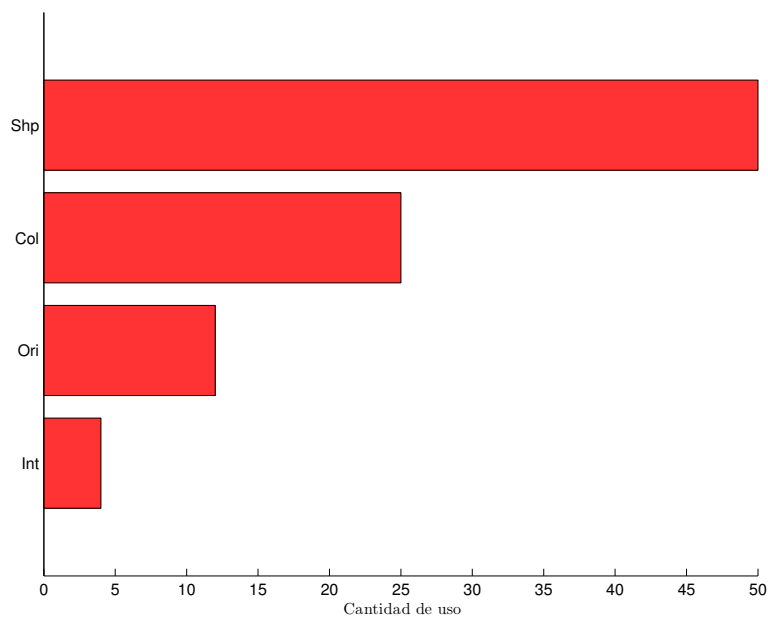
**Figura 51:** Cantidad de uso de funciones de forma en 30 individuos de FOA-HDA entrenados con base de imágenes.



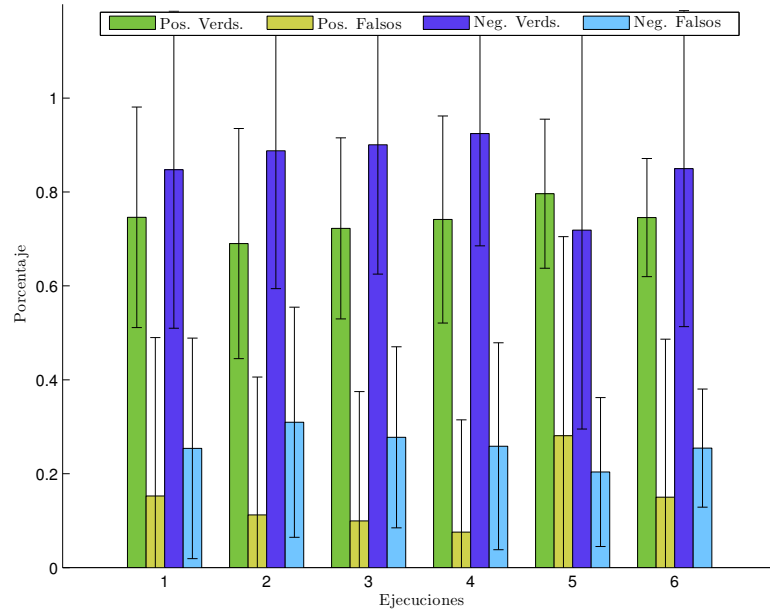
**Figura 52:** Cantidad de uso de terminales de forma en 30 individuos de FOA-HDA entrenados con base de imágenes.



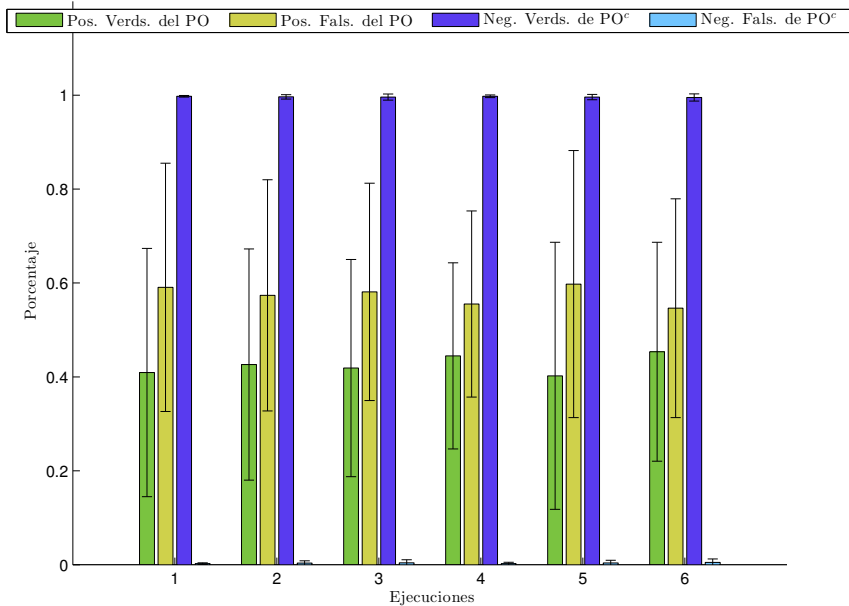
**Figura 53:** Cantidad de uso de funciones de FI en 30 individuos de FOA-HDA entrenados con base de imágenes.



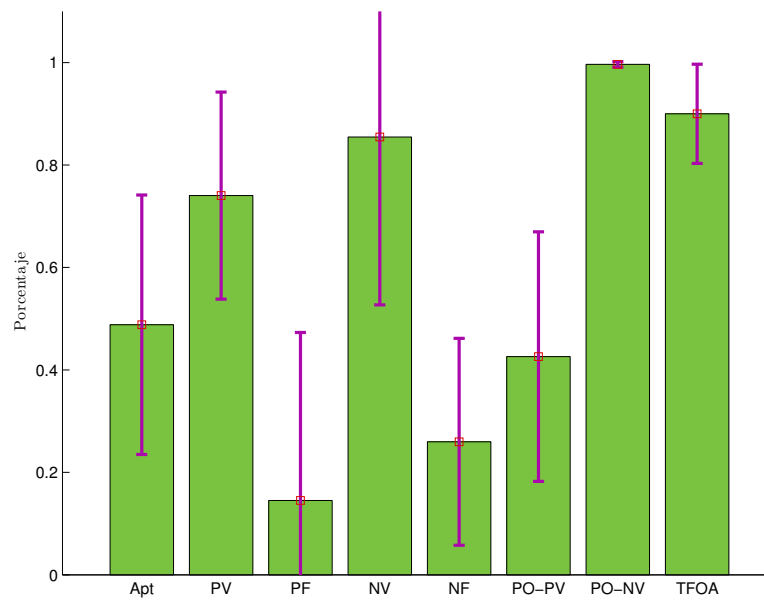
**Figura 54:** Cantidad de uso de terminales de FI en 30 individuos de FOA-HDA entrenados con base de imágenes.



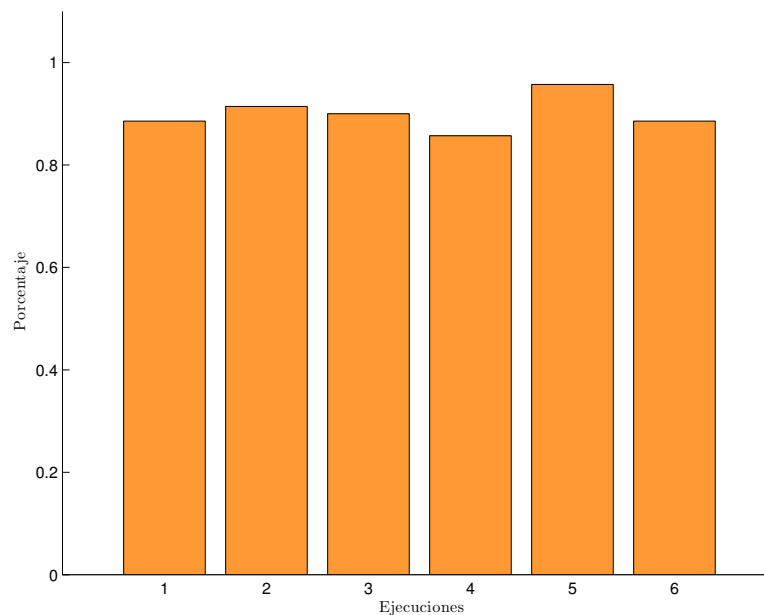
**Figura 55:** Etapa de prueba para FOA-HDA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada.



**Figura 56:** Etapa de prueba para FOA-HDA entrenado con base de imágenes - Promedio de porcentaje de positivos y negativos del proto-objeto para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada.



**Figura 57:** Estadística general de la etapa de prueba para 30 individuos de FOA-HDA entrenados con base de imágenes.



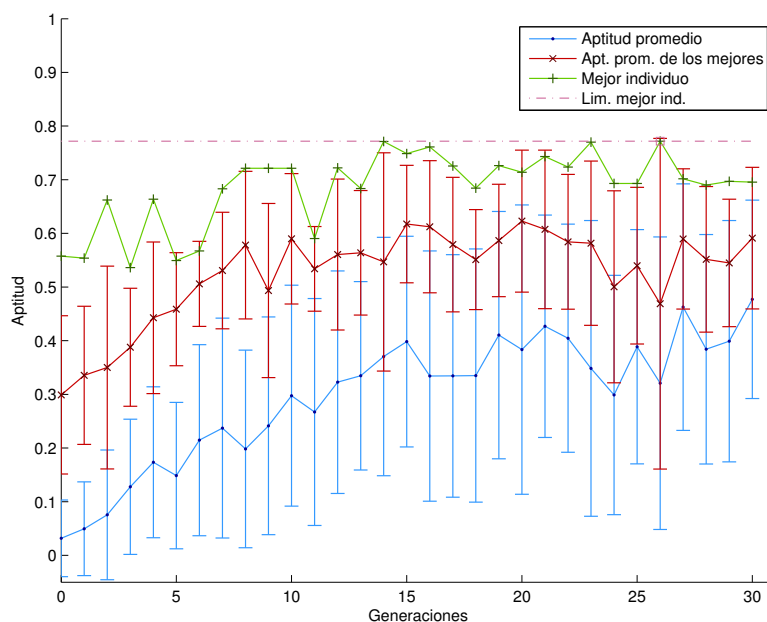
**Figura 58:** Etapa de prueba para FOA-HDA entrenado con base de imágenes - Certeza de punto de foco de atención para seis grupos de ejecuciones. Cada grupo promedia 5 ejecuciones que utilizan el mismo grupo imágenes de prueba, resultado de la validación cruzada.

### 8.3. Resultados de entrenamiento en vivo

#### 8.3.1. FOA en vivo

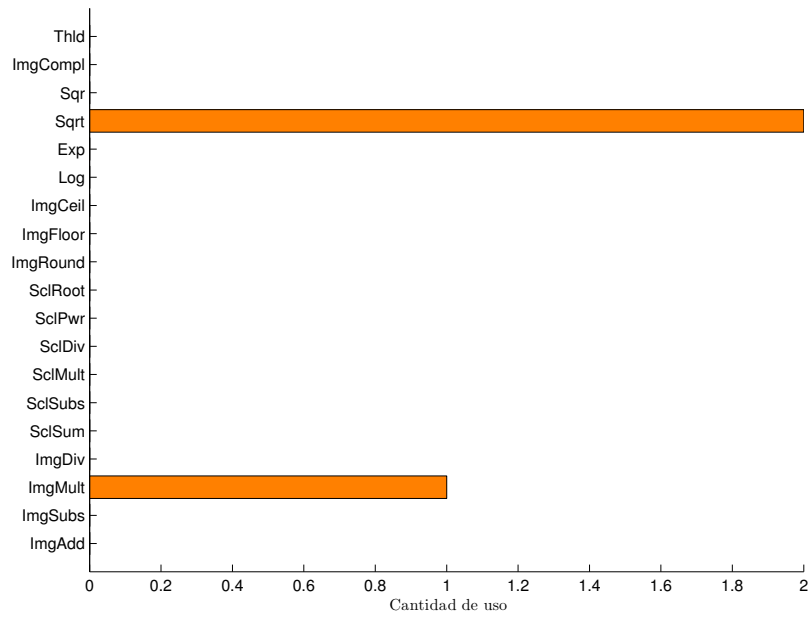
Esta sección muestra los resultados obtenidos para cinco ejecuciones de FOA en vivo. Los individuos usados para la etapa de prueba usan el mismo grupo de imágenes, y cada individuo usado es el mejor individuo de su respectiva ejecución. La descripción de las gráficas y sus significados es descrita en la Sección 8.2.1.

La gráfica de la Figura 59 muestra la evolución promedio de cinco ejecuciones. Los mejores individuos alcanzaron un máximo de 0.7718 en aptitud.

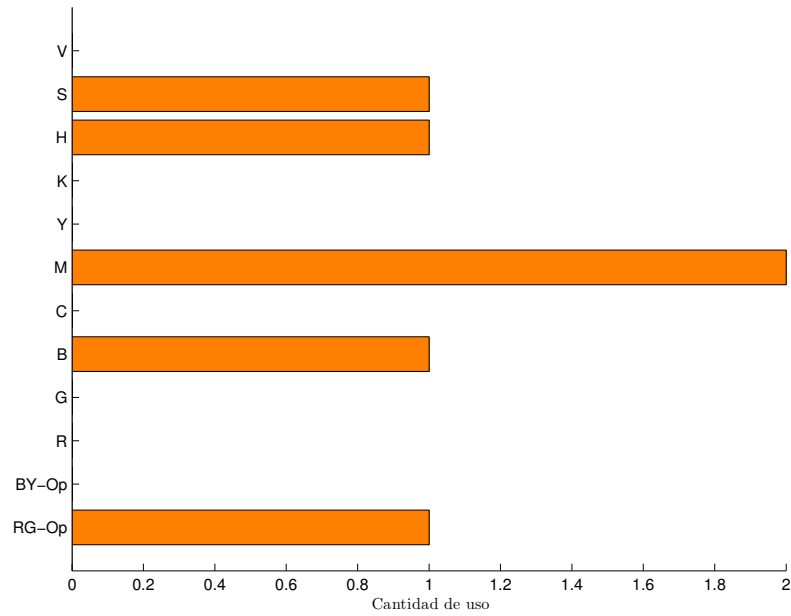


**Figura 59:** Evolución promedio en 5 ejecuciones para FOA entrenados en vivo.

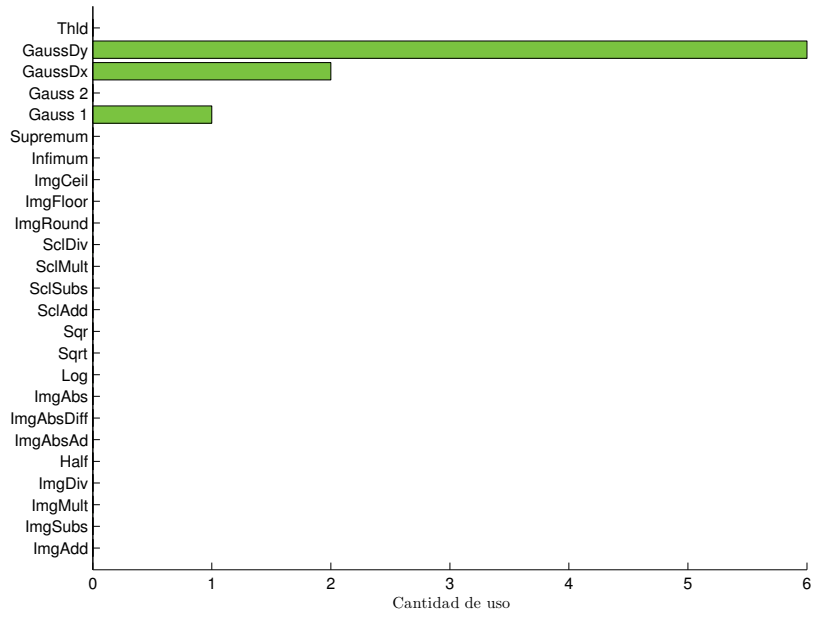




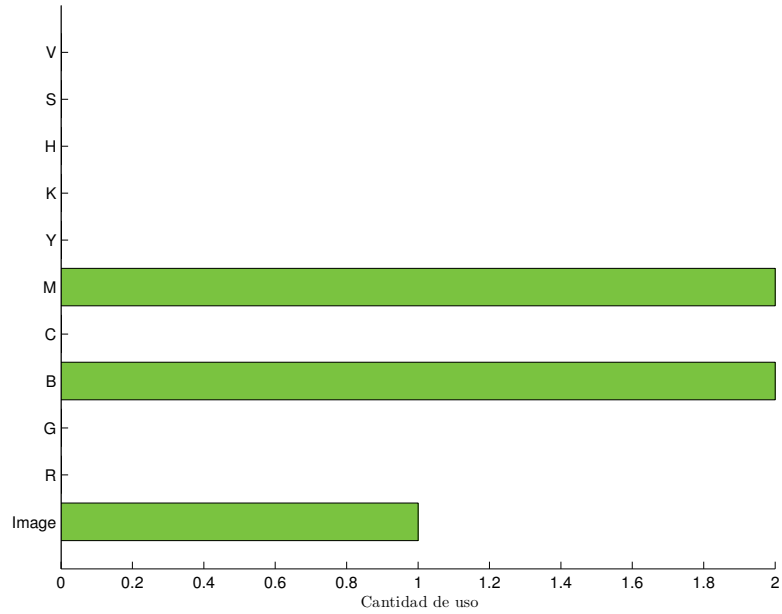
**Figura 60:** Cantidad de uso de funciones de color en 5 individuos de FOA entrenados en vivo.



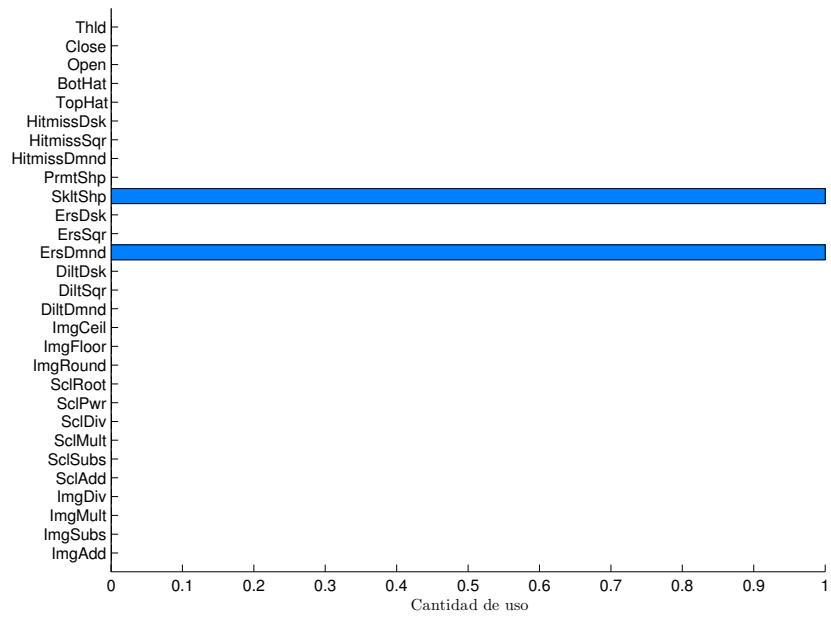
**Figura 61:** Cantidad de uso de terminales de color en 5 individuos de FOA entrenados en vivo.



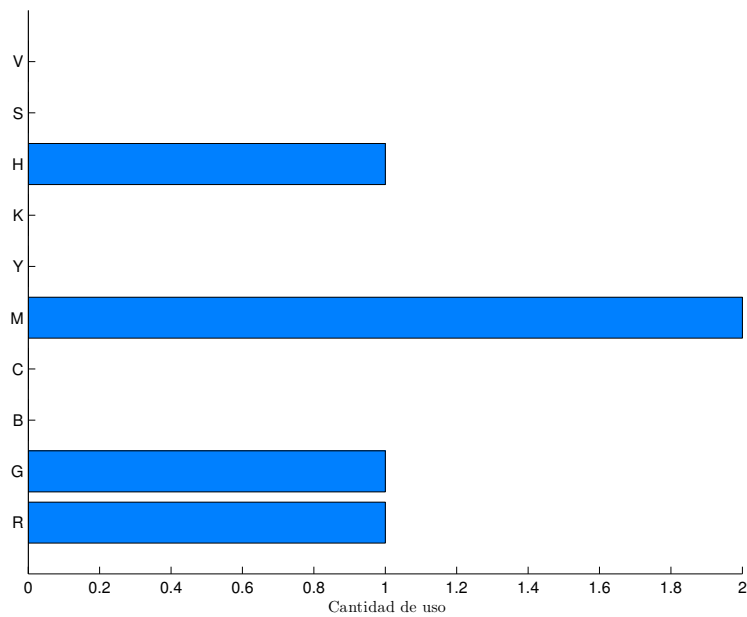
**Figura 62:** Cantidad de uso de funciones de orientación en 5 individuos de FOA entrenados en vivo.



**Figura 63:** Cantidad de uso de terminales de orientación en 5 individuos de FOA entrenados en vivo.



**Figura 64:** Cantidad de uso de funciones de forma en 5 individuos de FOA entrenados en vivo.



**Figura 65:** Cantidad de uso de terminales de forma en 5 individuos de FOA entrenados en vivo.

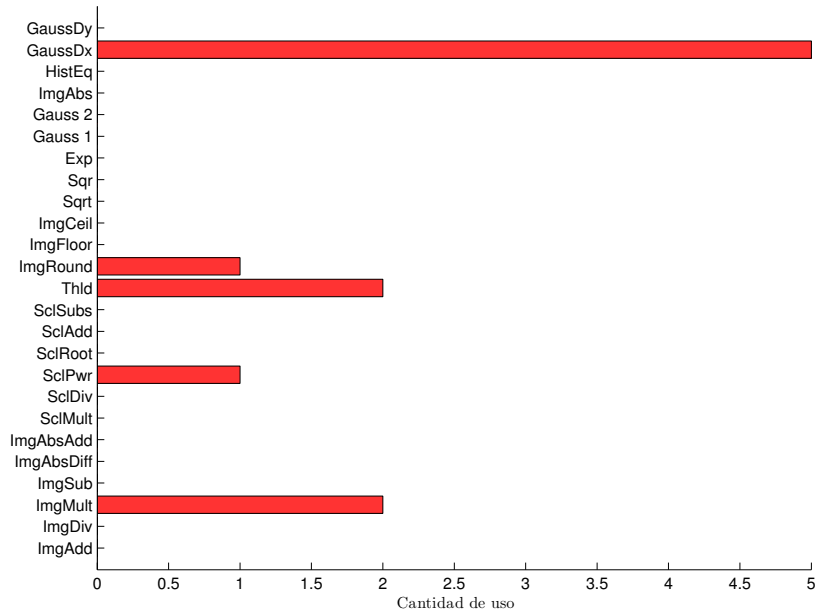


Figura 66: Cantidad de uso de funciones de FI en 5 individuos de FOA entrenados en vivo.

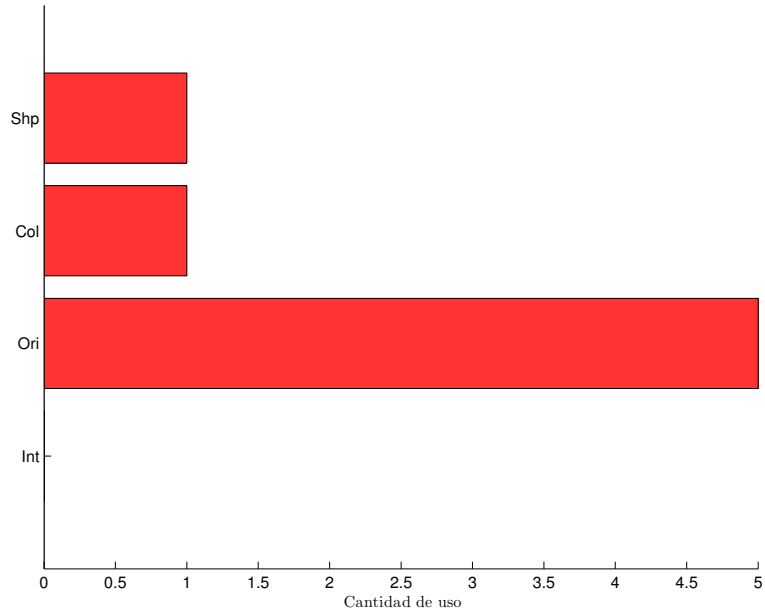
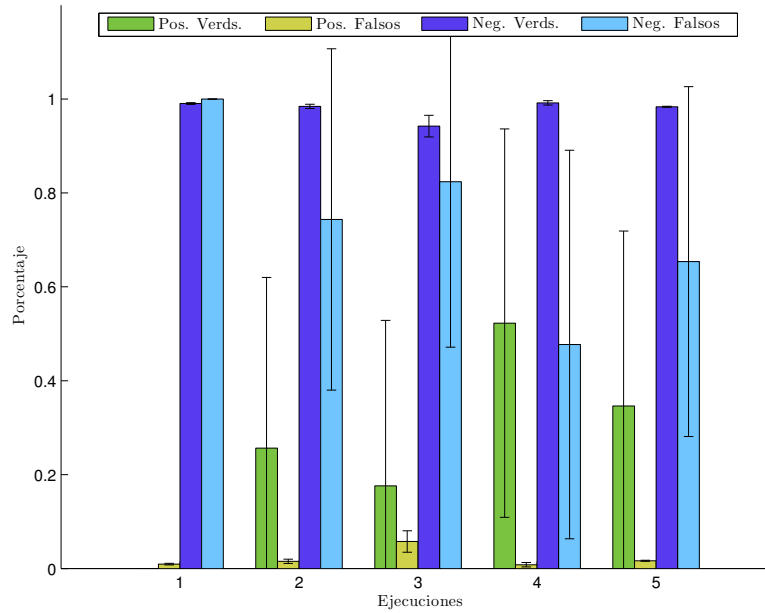
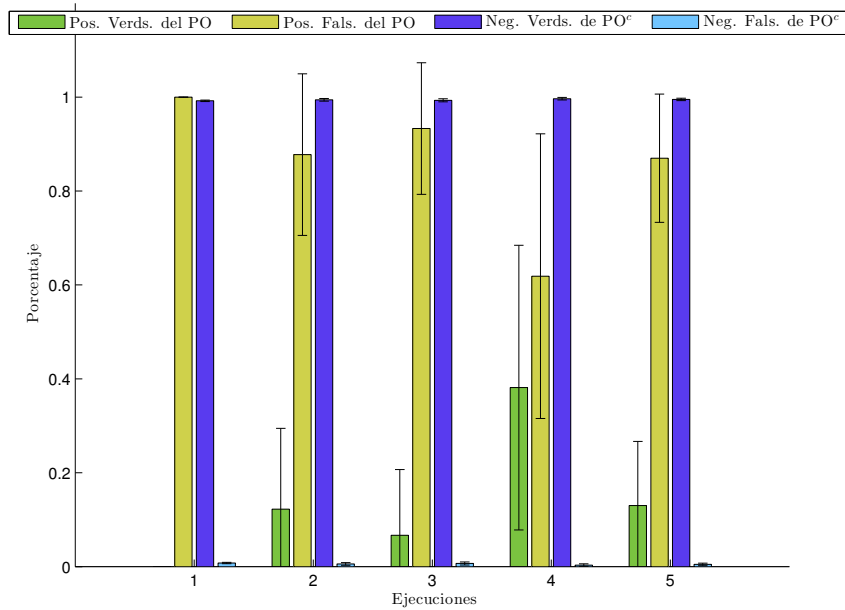


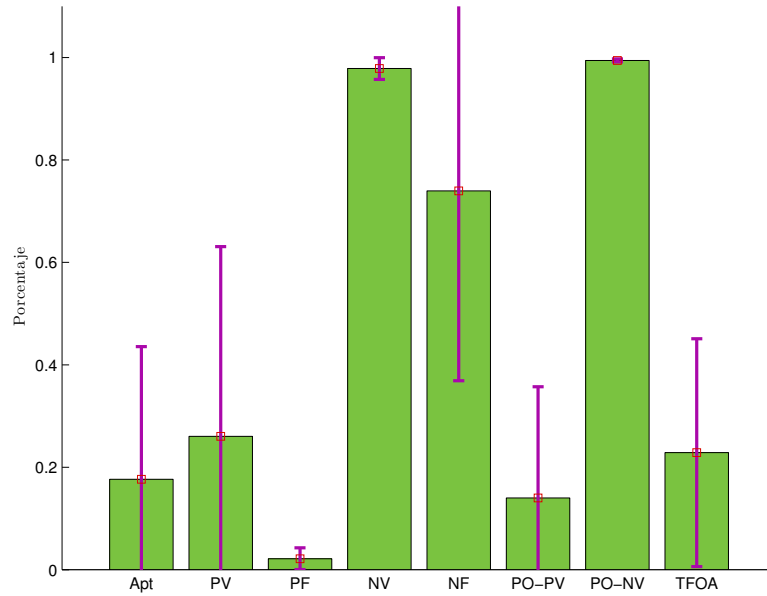
Figura 67: Cantidad de uso de terminales de FI en 5 individuos de FOA entrenados en vivo.



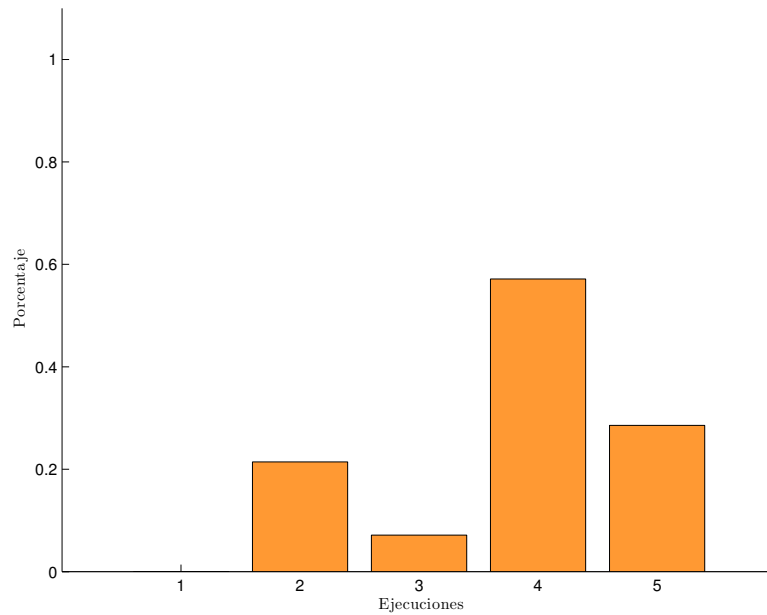
**Figura 68:** Etapa de prueba para FOA entrenado en vivo - Promedio de porcentaje de positivos y negativos para 5 individuos que utilizan el mismo grupo imágenes de prueba.



**Figura 69:** Etapa de prueba para FOA entrenado en vivo - Promedio de porcentaje de positivos y negativos del proto-objeto para 5 individuos que utilizan el mismo grupo imágenes de prueba.



**Figura 70:** Estadística general de la etapa de prueba para 5 individuos de FOA entrenados en vivo.



**Figura 71:** Etapa de prueba para FOA entrenado en vivo - Certeza de punto de foco de atención para 5 individuos que utilizan el mismo grupo imágenes de prueba.

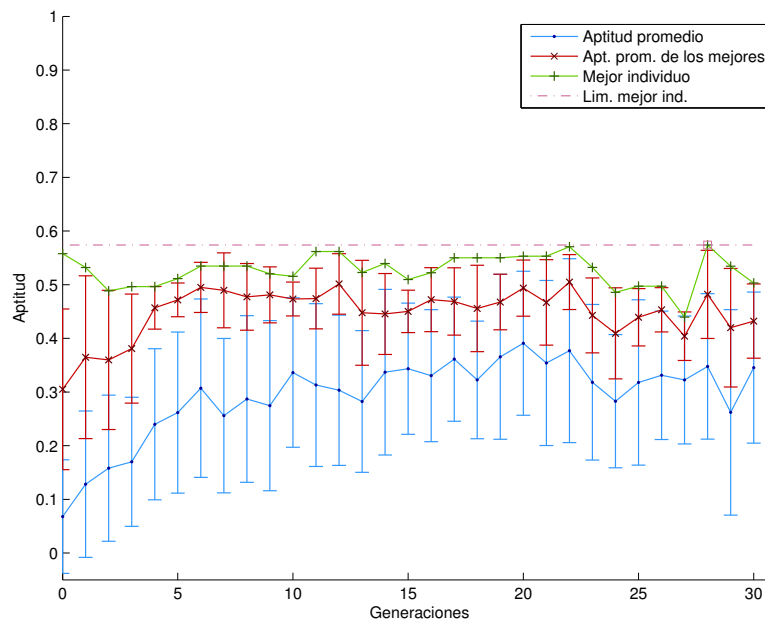
**Tabla 5: Resultados de cinco ejecuciones para FOA en vivo.**

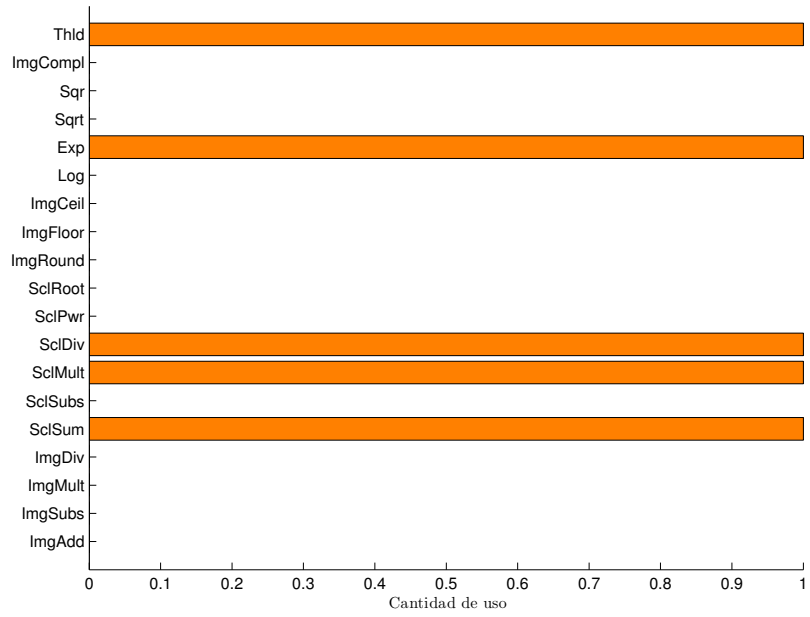
	E 1	E 2	E 3	E 4	E 5	Prom. Gnrl.
<b>Apt</b>	0.00 ±0.00	16.49 ±23.12	9.51 ±19.43	43.37 ±33.68	18.87 ±19.91	17.65 ±25.92
<b>PV</b>	0.00 ±0.00	25.65 ±36.34	17.62 ±35.25	52.27 ±41.37	34.63 ±37.25	26.03 ±37.06
<b>PF</b>	0.95 ±0.18	1.56 ±0.45	5.77 ±2.30	0.82 ±0.47	1.65 ±0.11	2.15 ±2.12
<b>NV</b>	99.05 ±0.18	98.44 ±0.45	94.23 ±2.30	99.18 ±0.47	98.35 ±0.11	97.85 ±2.12
<b>NF</b>	100.00 ±0.00	74.35 ±36.34	82.38 ±35.25	47.73 ±41.37	65.37 ±37.25	73.97 ±37.06
<b>PO:PV</b>	0.00 ±0.00	12.25 ±17.21	6.68 ±14.01	38.14 ±30.31	13.01 ±13.66	14.02 ±21.70
<b>PO:PF</b>	100.00 ±0.00	87.75 ±17.21	93.32 ±14.01	61.86 ±30.31	86.99 ±13.66	85.98 ±21.70
<b>PO:NV</b>	99.21 ±0.14	99.43 ±0.28	99.32 ±0.33	99.66 ±0.27	99.51 ±0.26	99.43 ±0.30
<b>PO:NF</b>	0.79 ±0.14	0.57 ±0.28	0.68 ±0.33	0.34 ±0.27	0.49 ±0.26	0.57 ±0.30
<b>TFOA</b>	0.00	21.43	7.14	57.14	28.57	22.86 ±22.25

### 8.3.2. FOA-LDA en vivo

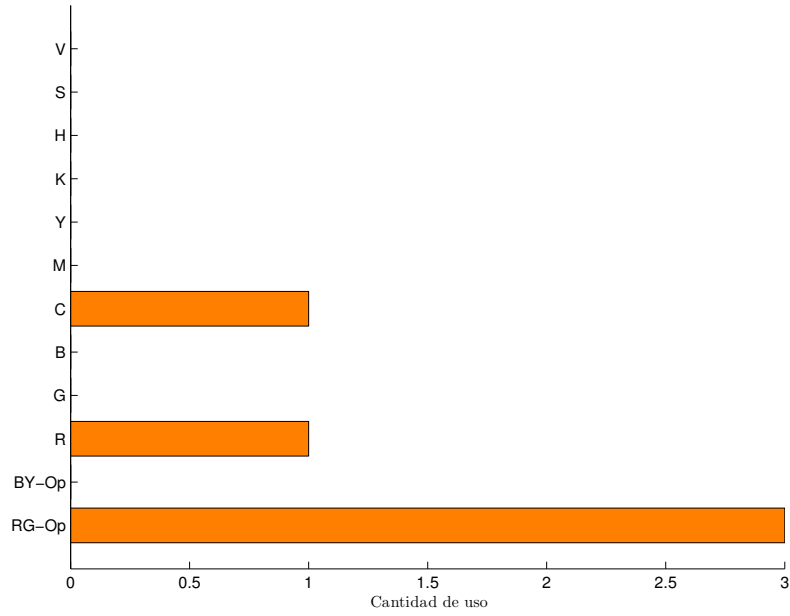
Esta sección muestra los resultados obtenidos para cinco ejecuciones de FOA-LDA en vivo. La descripción de las gráficas y sus significados es descrita en la Sección 8.2.1.

La gráfica de la Figura 72 muestra la evolución promedio de cinco ejecuciones. Los mejores individuos alcanzaron un máximo de 0.5739 en aptitud.

**Figura 72:** Evolución promedio en 5 ejecuciones para FOA-LDA entrenados en vivo.

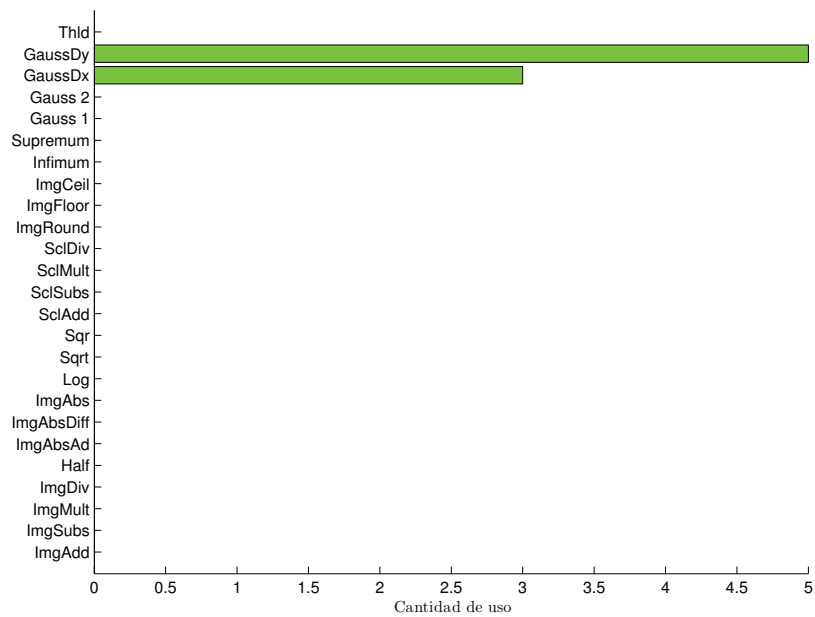


**Figura 73:** Cantidad de uso de funciones de color en 5 individuos de FOA-LDA entrenados en vivo.

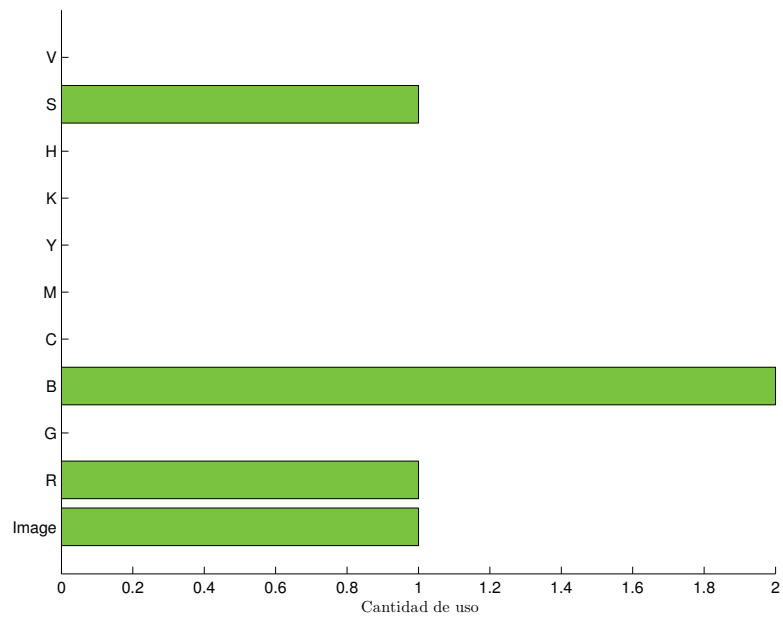


**Figura 74:** Cantidad de uso de terminales de color en 5 individuos de FOA-LDA entrenados en vivo.





**Figura 75:** Cantidad de uso de funciones de orientación en 5 individuos de FOA-LDA entrenados en vivo.



**Figura 76:** Cantidad de uso de terminales de orientación en 5 individuos de FOA-LDA entrenados en vivo.

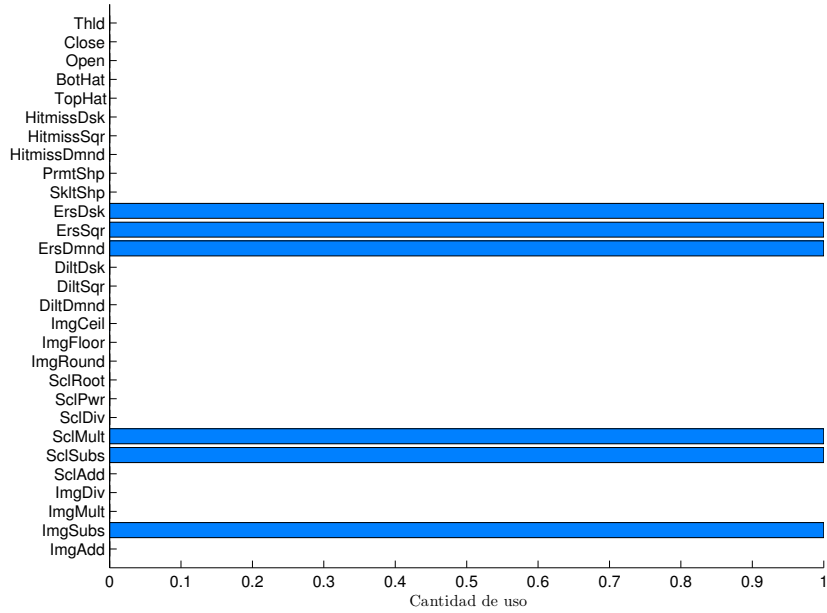


Figura 77: Cantidad de uso de funciones de forma en 5 individuos de FOA-LDA entrenados en vivo.

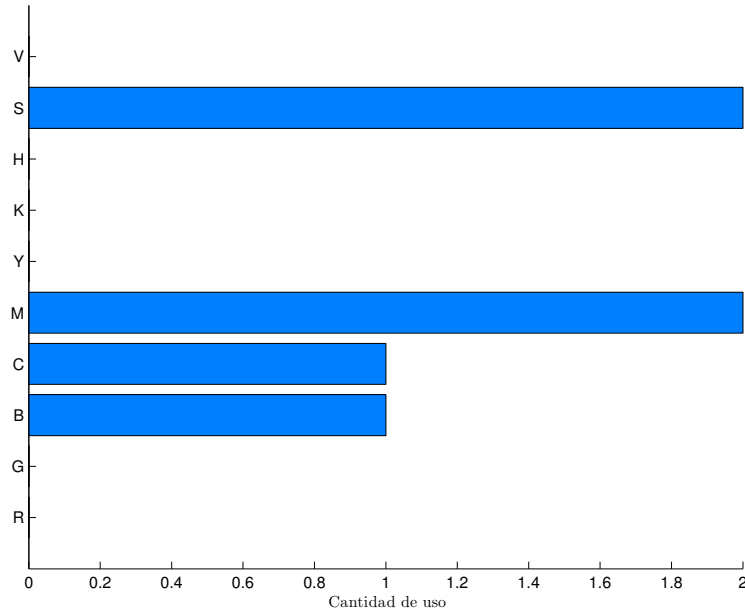
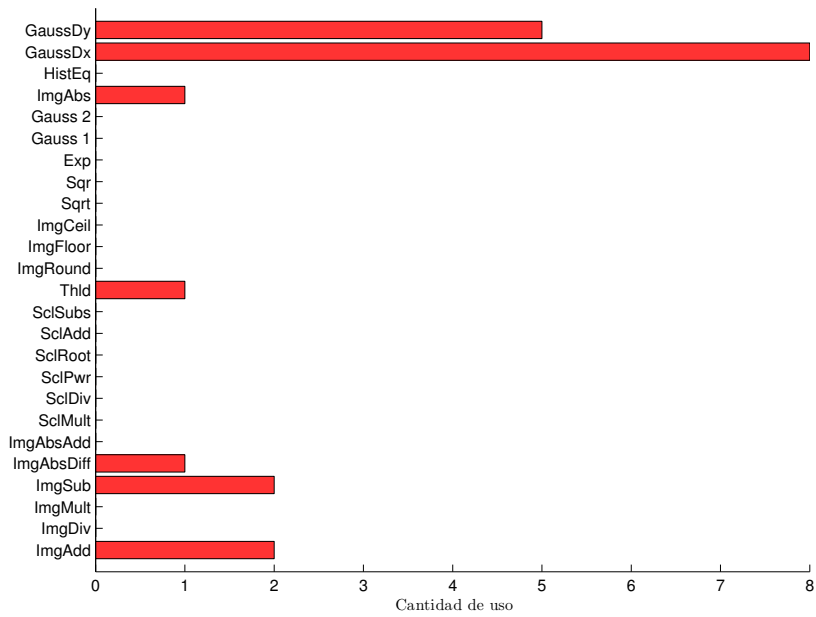
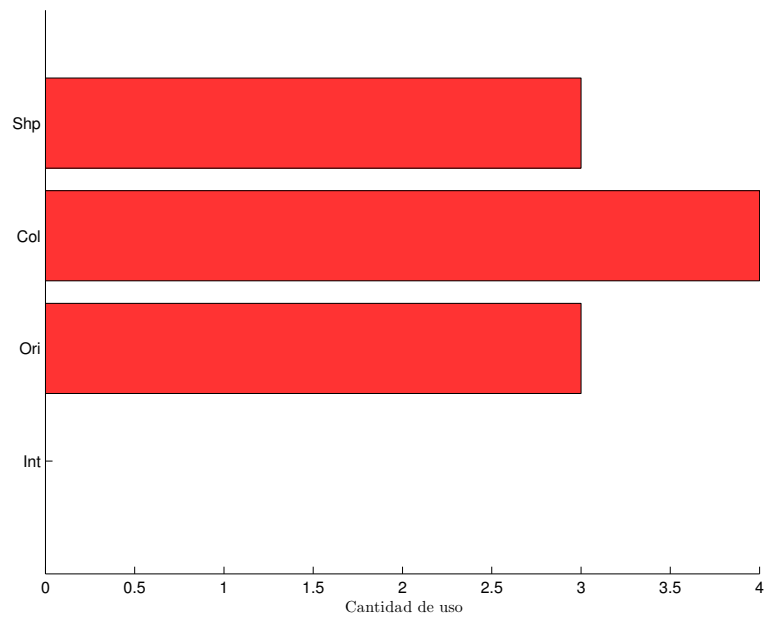


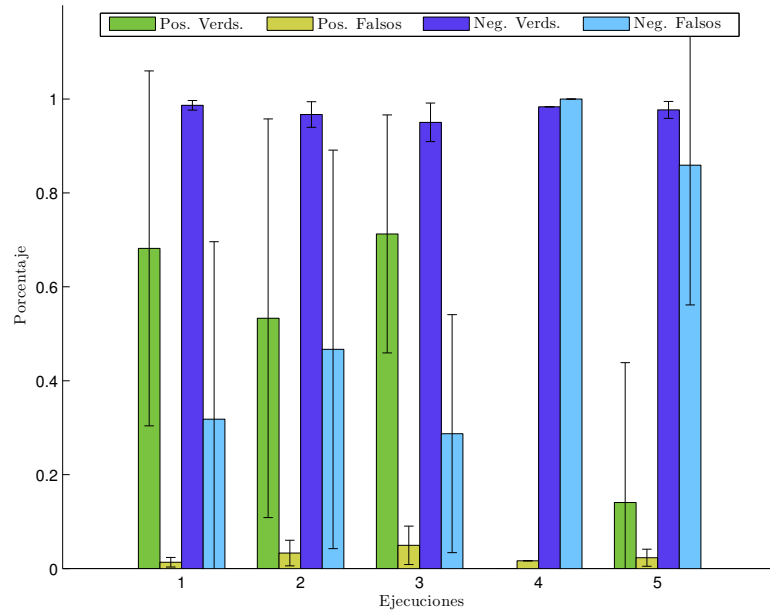
Figura 78: Cantidad de uso de terminales de forma en 5 individuos de FOA-LDA entrenados en vivo.



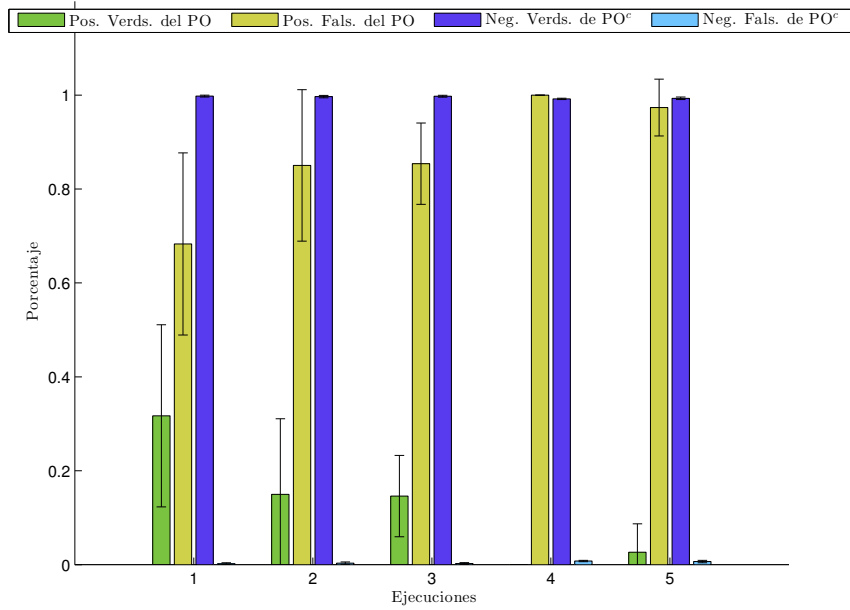
**Figura 79:** Cantidad de uso de funciones de FI en 5 individuos de FOA-LDA entrenados en vivo.



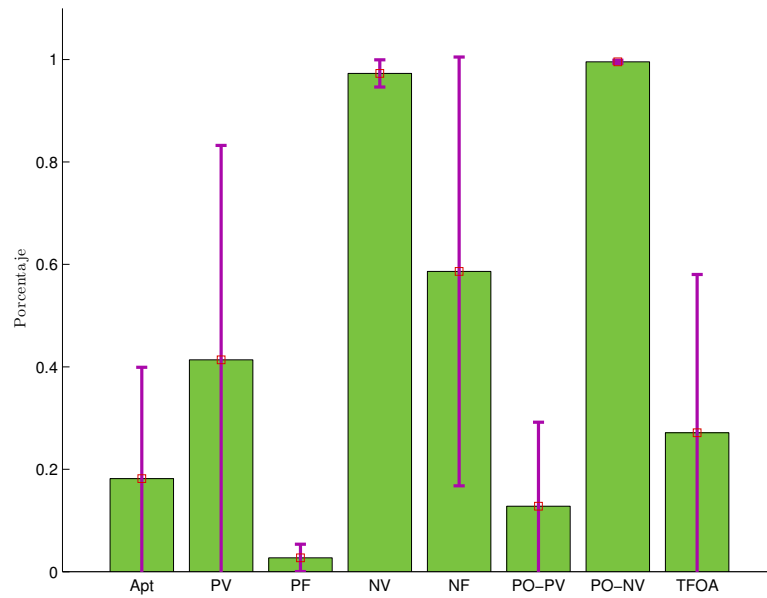
**Figura 80:** Cantidad de uso de terminales de FI en 5 individuos de FOA-LDA entrenados en vivo.



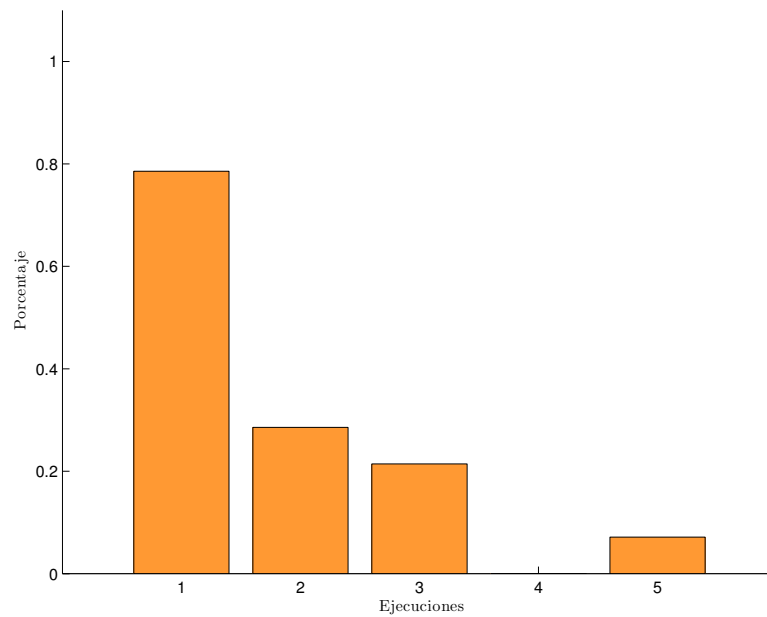
**Figura 81:** Etapa de prueba para FOA-LDA entrenado en vivo - Promedio de porcentaje de positivos y negativos para 5 individuos que utilizan el mismo grupo imágenes de prueba.



**Figura 82:** Etapa de prueba para FOA-LDA entrenado en vivo - Promedio de porcentaje de positivos y negativos del proto-objeto para 5 individuos que utilizan el mismo grupo imágenes de prueba.



**Figura 83:** Estadística general de la etapa de prueba para 5 individuos de FOA-LDA entrenados en vivo.



**Figura 84:** Etapa de prueba para FOA-LDA entrenado en vivo - Certeza de punto de foco de atención para 5 individuos que utilizan el mismo grupo imágenes de prueba.

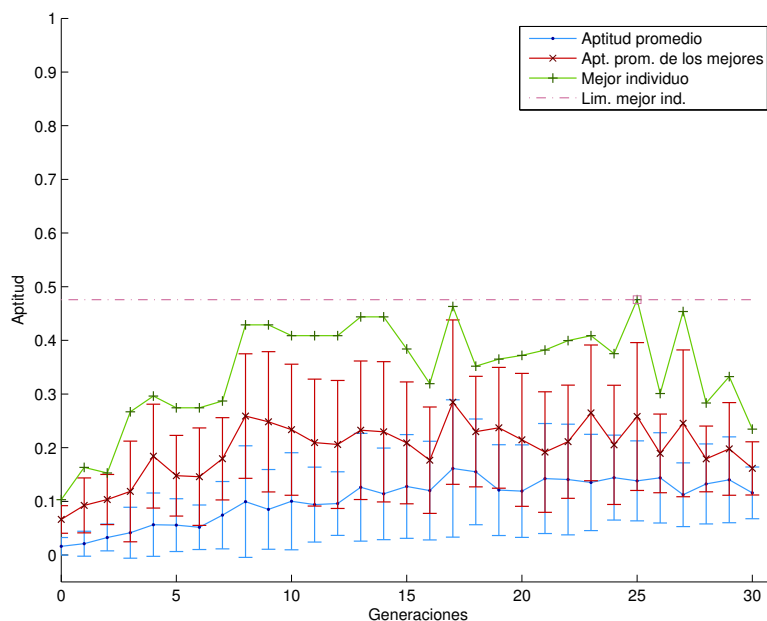
**Tabla 6: Resultados de cinco ejecuciones para FOA-LDA en vivo.**

	E 1	E 2	E 3	E 4	E 5	Prom. Gnrl.
<b>Apt</b>	42.29 ±24.84	21.91 ±21.26	22.36 ±11.29	0.00 ±0.00	4.32 ±9.56	18.18 ±21.75
<b>PV</b>	68.19 ±37.81	53.31 ±42.44	71.27 ±25.34	0.00 ±0.00	14.08 ±29.77	41.37 ±41.85
<b>PF</b>	1.34 ±1.02	3.30 ±2.73	4.96 ±4.09	1.66 ±0.00	2.32 ±1.81	2.72 ±2.67
<b>NV</b>	98.66 ±1.02	96.70 ±2.73	95.04 ±4.09	98.34 ±0.00	97.68 ±1.81	97.28 ±2.67
<b>NF</b>	31.81 ±37.81	46.69 ±42.44	28.73 ±25.34	100.00 ±0.00	85.92 ±29.77	58.63 ±41.85
<b>PO:PV</b>	31.70 ±19.39	14.97 ±16.12	14.61 ±8.67	0.00 ±0.00	2.65 ±6.05	12.79 ±16.40
<b>PO:PF</b>	68.30 ±19.39	85.03 ±16.12	85.39 ±8.67	100.00 ±0.00	97.35 ±6.05	87.21 ±16.40
<b>PO:NV</b>	99.79 ±0.20	99.68 ±0.25	99.76 ±0.20	99.21 ±0.15	99.33 ±0.26	99.55 ±0.32
<b>PO:NF</b>	0.21 ±0.20	0.32 ±0.25	0.24 ±0.20	0.79 ±0.15	0.67 ±0.26	0.45 ±0.32
<b>TFOA</b>	78.57	28.57	21.43	0.00	7.14	27.14 ±30.89

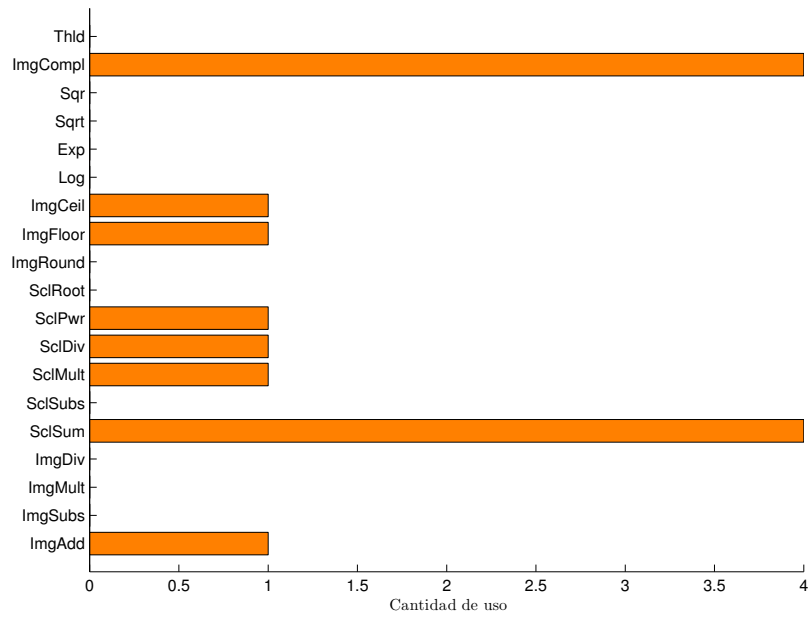
### 8.3.3. FOA-HDA en vivo

Esta sección muestra los resultados obtenidos para cinco ejecuciones de FOA-HDA en vivo. La descripción de las gráficas y sus significados es descrita en la Sección 8.2.1.

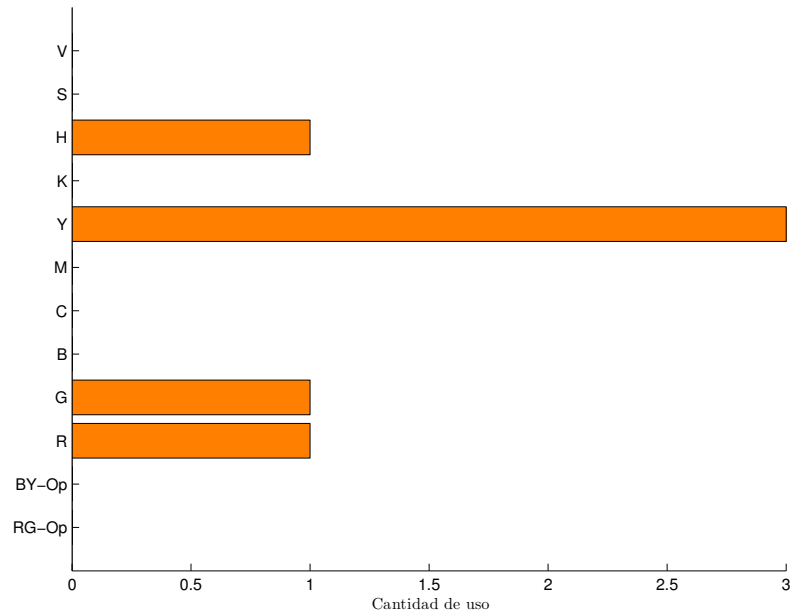
La gráfica de la Figura 85 muestra la evolución promedio de cinco ejecuciones. Los mejores individuos alcanzaron un máximo de 0.4758 en aptitud.



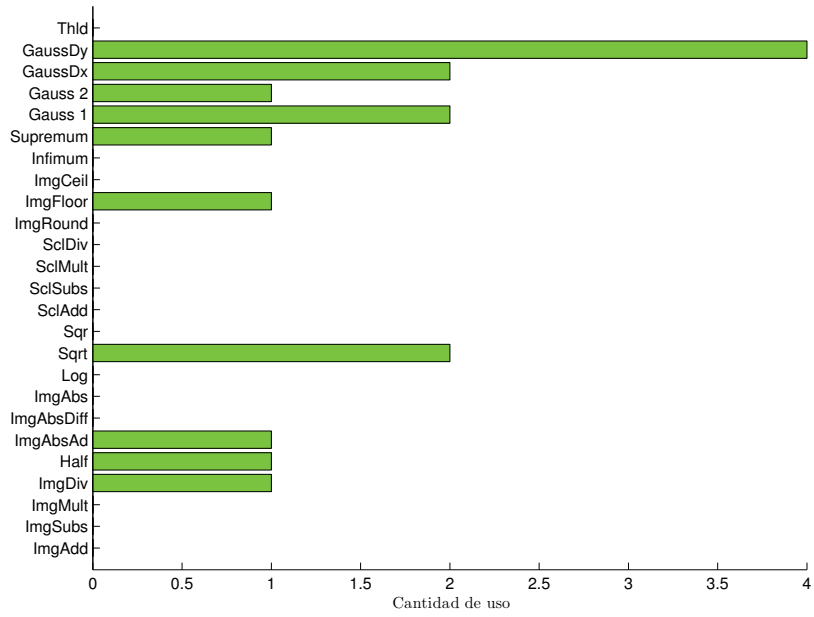
**Figura 85:** Evolución promedio en 5 ejecuciones para FOA-HDA entrenados en vivo.



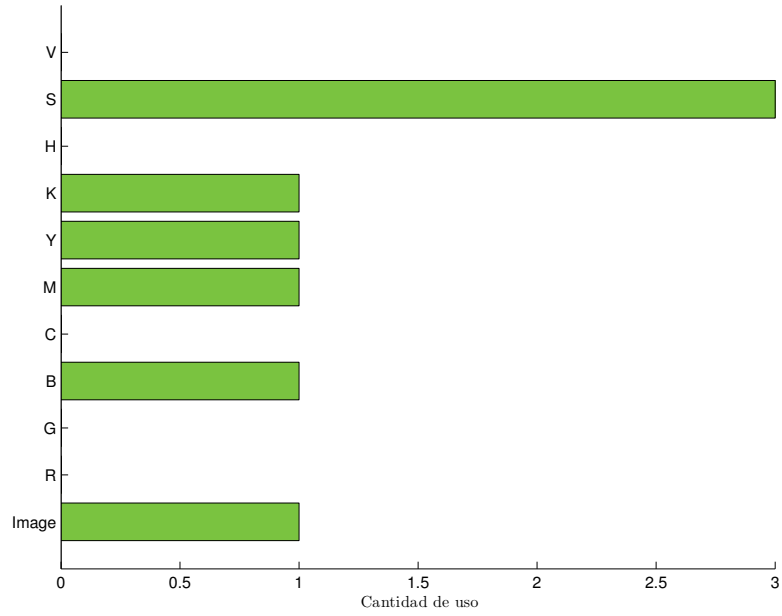
**Figura 86:** Cantidad de uso de funciones de color en 5 individuos de FOA-HDA entrenados en vivo.



**Figura 87:** Cantidad de uso de terminales de color en 5 individuos de FOA-HDA entrenados en vivo.

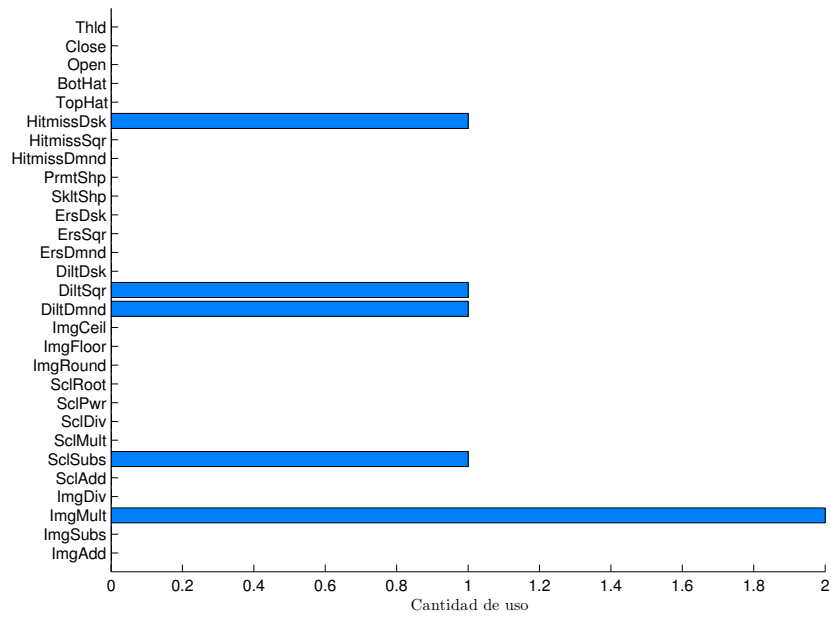


**Figura 88:** Cantidad de uso de funciones de orientación en 5 individuos de FOA-HDA entrenados en vivo.

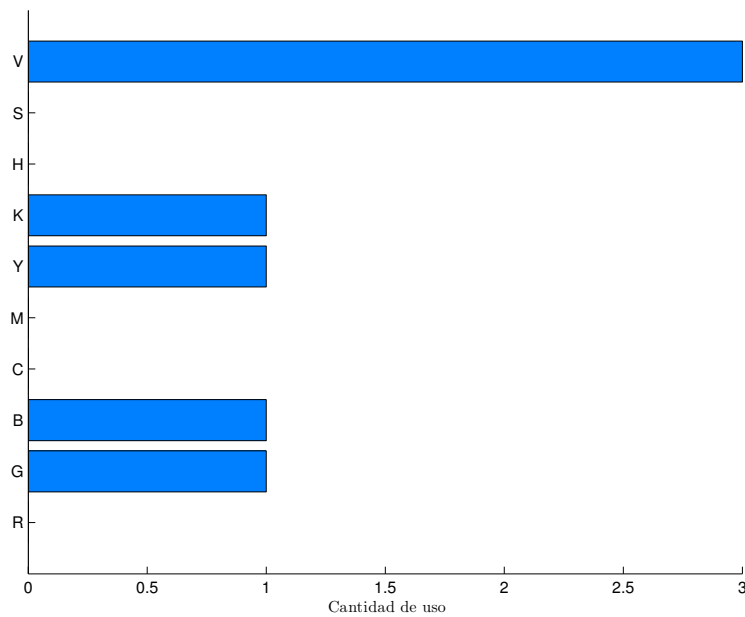


**Figura 89:** Cantidad de uso de terminales de orientación en 5 individuos de FOA-HDA entrenados en vivo.

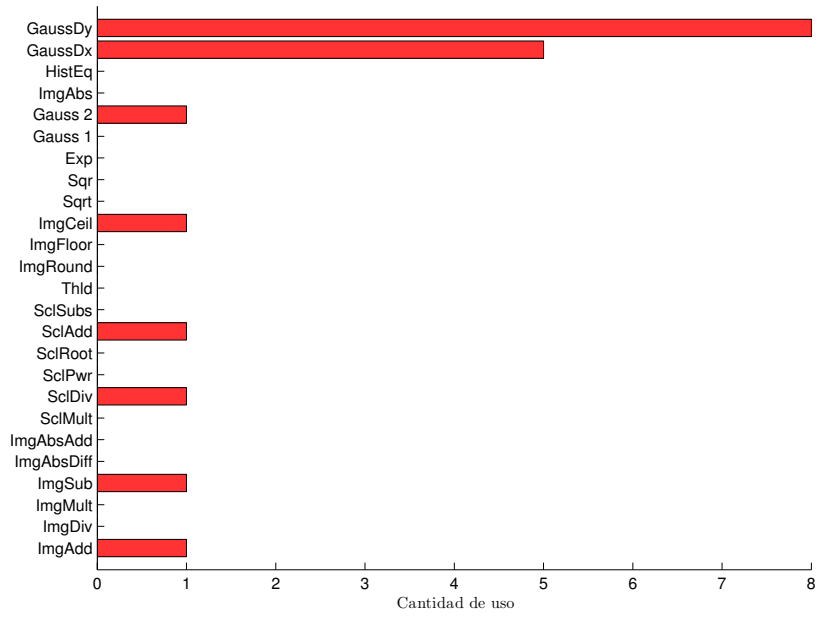




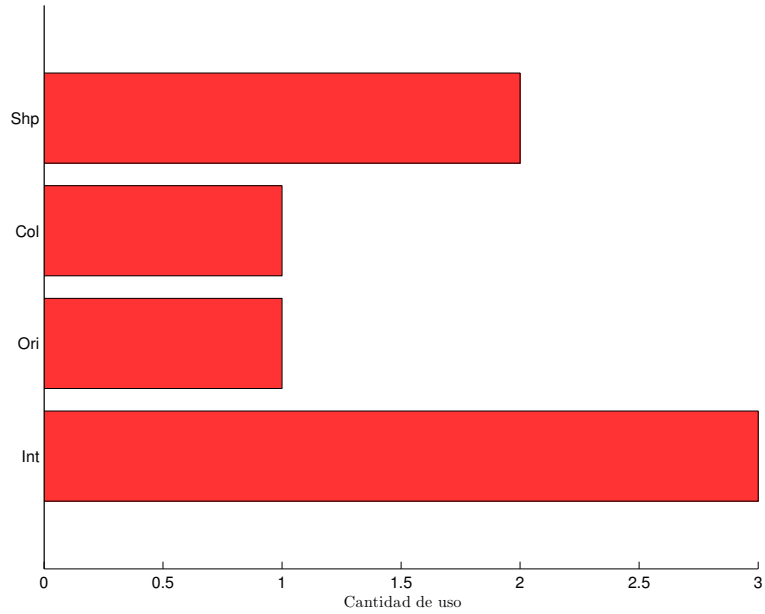
**Figura 90:** Cantidad de uso de funciones de forma en 5 individuos de FOA-HDA entrenados en vivo.



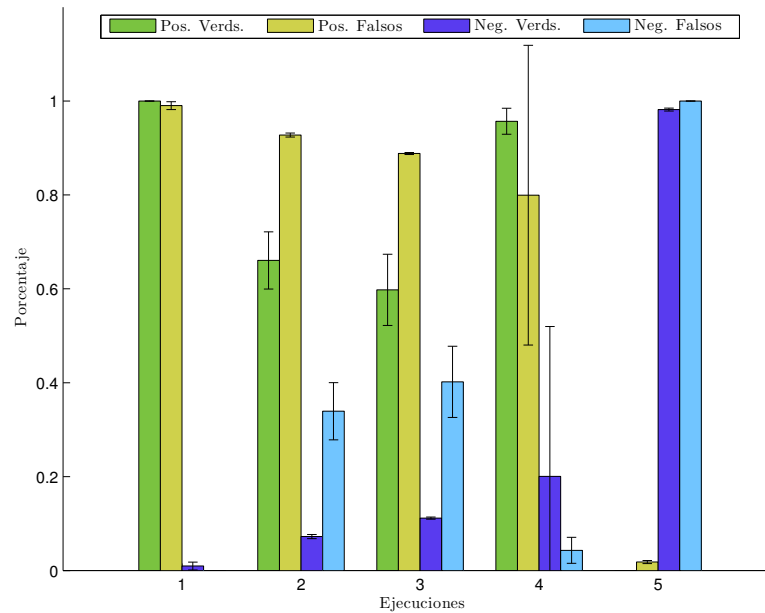
**Figura 91:** Cantidad de uso de terminales de forma en 5 individuos de FOA-HDA entrenados en vivo.



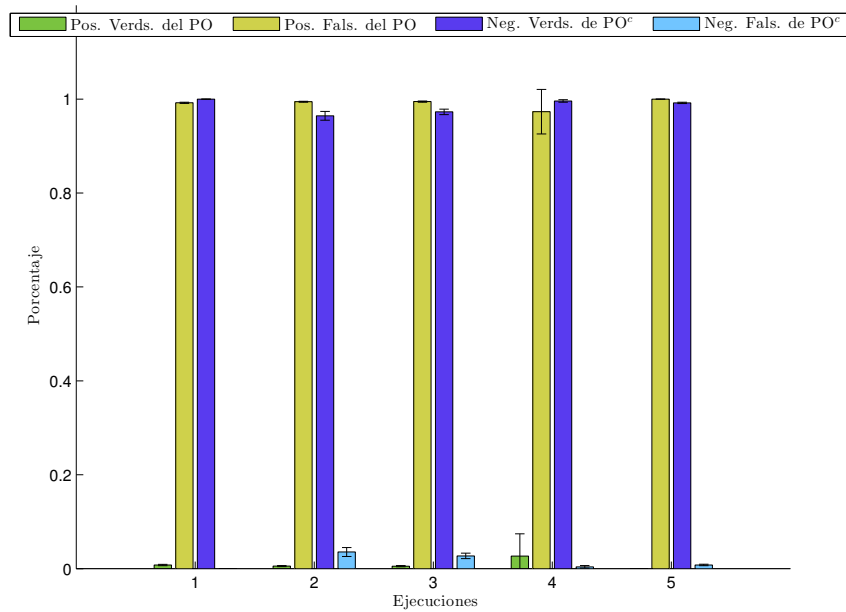
**Figura 92:** Cantidad de uso de funciones de FI en 5 individuos de FOA-HDA entrenados en vivo.



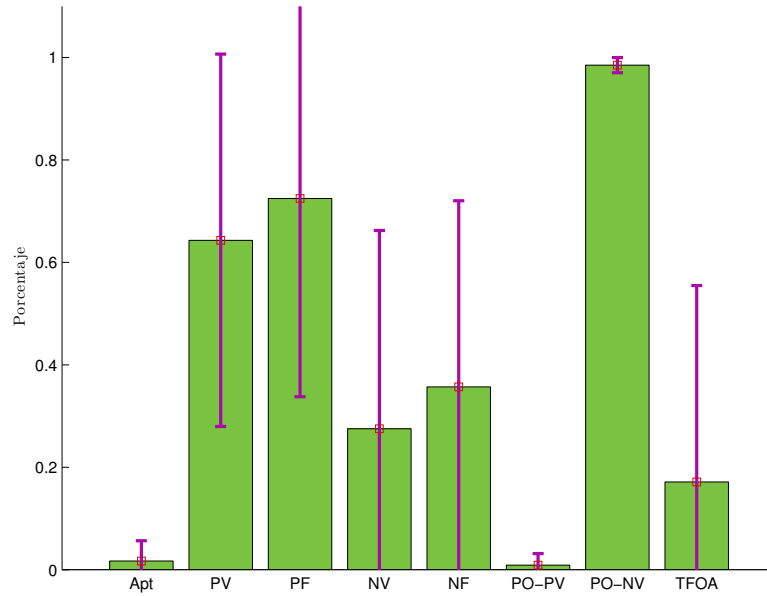
**Figura 93:** Cantidad de uso de terminales de FI en 5 individuos de FOA-HDA entrenados en vivo.



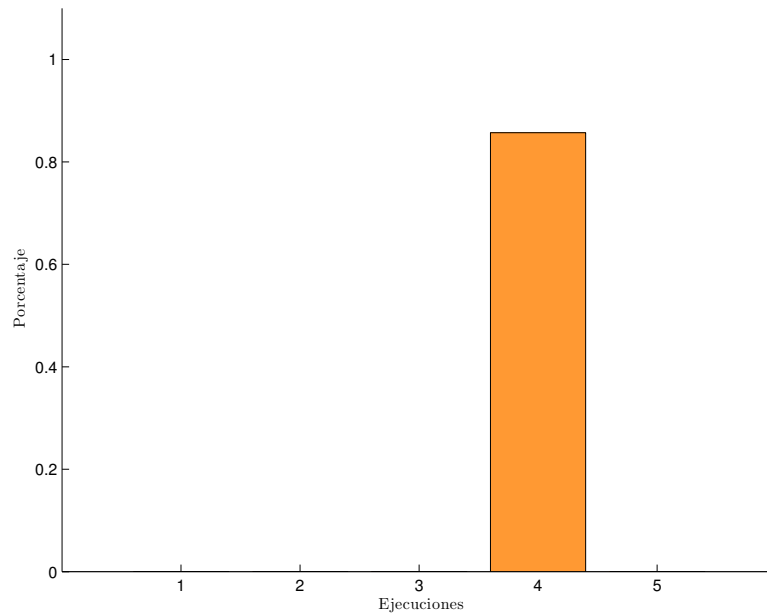
**Figura 94:** Etapa de prueba para FOA-HDA entrenado en vivo - Promedio de porcentaje de positivos y negativos para 5 individuos que utilizan el mismo grupo imágenes de prueba.



**Figura 95:** Etapa de prueba para FOA-HDA entrenado en vivo - Promedio de porcentaje de positivos y negativos del proto-objeto para 5 individuos que utilizan el mismo grupo imágenes de prueba.



**Figura 96:** Estadística general de la etapa de prueba para 5 individuos de FOA-HDA entrenados en vivo.



**Figura 97:** Etapa de prueba para FOA-HDA entrenado en vivo - Certeza de punto de foco de atención para 5 individuos que utilizan el mismo grupo imágenes de prueba.

**Tabla 7: Resultados de cinco ejecuciones para FOA-HDA en vivo.**

	E 1	E 2	E 3	E 4	E 5	Prom. Gnrl.
<b>Apt</b>	1.56 ±0.28	1.10 ±0.22	1.05 ±0.26	4.84 ±8.24	0.00 ±0.00	1.71 ±3.95
<b>PV</b>	100.00 ±0.00	66.06 ±6.09	59.80 ±7.59	95.69 ±2.76	0.00 ±0.00	64.31 ±36.35
<b>PF</b>	99.03 ±0.84	92.75 ±0.44	88.82 ±0.22	79.95 ±31.92	1.82 ±0.32	72.47 ±38.69
<b>NV</b>	0.97 ±0.84	7.25 ±0.44	11.18 ±0.22	20.05 ±31.92	98.18 ±0.32	27.53 ±38.69
<b>NF</b>	0.00 ±0.00	33.94 ±6.09	40.20 ±7.59	4.31 ±2.76	100.00 ±0.00	35.69 ±36.35
<b>PO:PV</b>	0.79 ±0.14	0.55 ±0.11	0.53 ±0.13	2.67 ±4.75	0.00 ±0.00	0.91 ±2.26
<b>PO:PF</b>	99.21 ±0.14	99.45 ±0.11	99.47 ±0.13	97.33 ±4.75	100.00 ±0.00	99.09 ±2.26
<b>PO:NV</b>	100.00 ±0.00	96.45 ±0.95	97.28 ±0.58	99.61 ±0.27	99.21 ±0.15	98.51 ±1.49
<b>PO:NF</b>	0.00 ±0.00	3.55 ±0.95	2.72 ±0.58	0.39 ±0.27	0.79 ±0.15	1.49 ±1.49
<b>TFOA</b>	0.00	0.00	0.00	85.71	0.00	17.14 ±38.33

## 8.4. Comparación de resultados

### 8.4.1. Con base de imágenes

#### 8.4.1.1. Progreso de la aptitud

En la Tabla 8 se comparan los resultados promedio de la aptitud en la evolución y la aptitud obtenida en la etapa de prueba. En comparación con FOA, se puede observar una mejora en FOA-LDA y FOA-HDA, de +6.23 y +12.45 en la aptitud promedio final del entrenamiento. De manera congruente, el algoritmo de FOA-HDA obtiene mayor aptitud que FOA-LDA.

Comparando sólo los mejores individuos obtenidos de cada algoritmo, se puede notar que FOA y FOA-LDA obtuvieron resultados muy similares, siendo el individuo de FOA el de más aptitud con una diferencia de +0.59. Por otro lado, FOA-HDA produjo individuos con un poco más de aptitud que los otros dos, con +4.11 de diferencia con respecto a FOA. .

En la etapa de prueba, nuevamente FOA-LDA y FOA-HDA obtienen mejores resultados a comparación de FOA, con diferencias de +7.5 y +11.31 respectivamente. FOA-HDA obtiene resultados un poco mejores que FOA-LDA.

La mejora en aptitud era de esperarse, ya que los nuevos algoritmos utilizan la información de profundidad. Dos pixeles con prominencia similar pueden ser diferenciados por

**Tabla 8: Comparación de aptitud en la etapa de entrenamiento (promedio inicial, promedio final, y mejor individuo) y la etapa de prueba para el entrenamiento con base de imágenes.**

	Prom. inicial	Prom. final	Mejor individuo	Prom. en pruebas
<b>FOA</b>	2.09 ±4.44	28.07 ±15.40	62.00	37.51 ±20.01
<b>FOA-LDA</b>	1.87 ±4.56	34.30 ±17.65	61.41	45.01 ±19.33
<b>FOA-HDA</b>	2.85 ±6.96	40.52 ±18.96	66.11	48.82 ±25.33

medio de sus distancias (entre sí, y con respecto al punto de foco de atención). Entonces, un pixel con suficiente prominencia puede ser eliminado si su distancia no es congruente con la del punto de foco de atención. De la misma forma, si hay una parte del objeto que genera muy poca prominencia, puede ser incluida si su distancia muestra congruencia con la distancia del objeto.

#### 8.4.1.2. Uso de las dimensiones

Es interesante analizar la selección de dimensiones que la GP utiliza, ya que muestran pistas sobre qué dimensiones producen mejor información o la necesaria para resolver los problemas de visión que se le presentan. Para todos los experimentos realizados, se usó el mismo objeto; la diferencia radica en el tipo de algoritmo implementado, y cómo los cambios realizados en los nuevos algoritmos influyeron en el uso de las distintas funciones y terminales a disposición de la GP. El Apéndice E contiene un glosario de todas las funciones y terminales usadas.

Las gráficas de las figuras 27, 41 y 54, muestran el uso de las cuatro dimensiones (forma, color, orientación e intensidad) para FOA, FOA-LDA y FOA-HDA respectivamente. En cuanto a similitud, se puede observar que la dimensión de intensidad fue la menos usada. Esto es congruente con los resultados de Dozal *et al.* (2014), en los que se muestra que la función de integración tiende a utilizar la información de intensidad en menor cantidad. En cuanto a las demás dimensiones, hubo cambios para los tres algoritmos. Es importante notar que, como mostraron Dozal *et al.* (2014) en sus resultados, el uso de las dimensiones en el algoritmo de FOA siguen siendo orientación, color, forma e intensidad, en ese orden descendente. Esto es interesante porque muestra que al integrar la dimensión de distancia, los nuevos algoritmos cambiaron la estrategia para el uso de dimensiones.

Como se menciona, las dimensiones más usadas en FOA son orientación, color, y forma, en orden descendente. Para FOA-LDA, color fue la dimensión más usada, mientras que forma y orientación se usó en las mismas cantidades. Finalmente, FOA-HDA usó más forma, luego color y orientación.

En cuanto a las funciones (véase las figuras 26, 40 y 53) usadas para construir la función de integración, se puede observar que las 3 funciones de mayor uso para los tres algoritmos fueron las derivadas de Gaussianas (GaussDx, GaussDy) y la función `ImgAbs`, la cual retorna una matriz donde a cada valor se le aplica una operación de absoluto.

Las gráficas de las funciones usadas en dimensión de color se muestran en las figuras 20, 34, y 47. Se puede observar variedad en el uso de funciones, sin embargo, las tres que más destacan son: para FOA, `ImgCompl` (que produce el complemento de una imagen), `Sqrt` (que obtiene la raíz cuadrada de los elementos de la imagen), e `ImgRound` (hace un redondeo al entero más cercano). Para FOA-LDA fueron `Exp` (exponencial de cada elemento de la imagen), `ImgRound` y `Sqrt`. En FOA-HDA, los más usados fueron `SciPwr` (que eleva cada valor de la imagen a la potencia  $1/k$ ), `SciDiv` (que multiplica los valores de la imagen por  $k$ ), e `ImgMult` (que multiplica elemento por elemento dos matrices del mismo tamaño). A diferencia de FOA y FOA-LDA, FOA-HDA casi no usó `ImgCompl`, `Sqrt` o `Exp`.

Los terminales más usados para color (véase las figuras 21, 35 y 48) en FOA fueron `RG-Op` (oponencia de color rojo-verde), `V` (valor) y `G` (verde). FOA-LDA usó mayormente `RG-op`, `S` (saturación), y `G` (verde) en misma cantidad que `C` (cyan). FOA-HDA prefirió `M` (magenta), `S`, y `Y` (amarillo) en misma cantidad que `RG-Op`.

A diferencia de la dimensión de color, la cual tuvo cierta variedad en el uso de funciones, la dimensión de orientación muestra una fuerte tendencia a usar las funciones de derivadas de Gaussianas (véase las figuras 22, 36, y 49). Los tres algoritmos usaron las funciones `GaussDx` y `GaussDy` en cantidades considerablemente mayores que el resto de funciones a disposición. Después de estas, la tercera función más usada para FOA y FOA-HDA fue "Gauss 1", y para FOA-LDA la función `Half` (donde cada elemento se multiplica por 0.05). Las derivadas de Gaussianas son usadas en filtros que se especializan

en detectar cambios bruscos en la imagen, es decir, detectan los bordes. Independientemente de que en la dimensión de orientación sea natural y lógico usar derivadas de gaussianas, estas funciones han mostrado ser de gran utilidad para la GP, pues también se puede observar que las funciones de integración de características implementan estas funciones mucho más que el resto que tienen a disposición.

En las gráficas de las figuras 23, 37, y 50 se pueden observar los terminales usadas en la dimensión de orientación. Para FOA, los terminales más usados fueron M, Y, R (rojo), y S. Para FOA-LDA fueron Y, H (matiz), y M. En cuanto a FOA-HDA, M, Y, y R fueron los de mayor uso.

La figuras 24, 38, y 51 muestran las gráficas de uso para las funciones de la dimensión de forma. En FOA, las más usadas fueron PrmtShp (un método para obtener los pixeles que pertenecen a los perímetros de las formas en la imagen), DiltDsk (una modalidad de la operación morfológica de dilatación), y en igual medida Thld (función de umbral), las tres modalidades de la función hitmiss (una operación morfológica llamada *acierto-fallo*), ImgFloor (operación de piso), SclPwr, y SclAdd (donde se suma  $1/k$  a cada elemento de la imagen). Para FOA-LDA las que más se usaron fueron PrmtShp, SclAdd, y en misma medida BottomHat (la transformación *sombrero de copa negro*) y ImgAdd (que suma los valores de dos imágenes). De manera interesante, FOA-HDA no tuvo operaciones morfológicas entre las tres más usadas, las cuales fueron SclPwr, SclAdd e ImgCeil (operación de techo).

En cuanto a los terminales más usados (véase las figuras 25, 39, y 52), con FOA quedaron R, H, y M. Para FOA-LDA los más usados fueron V, G, S, Y. Finalmente, en FOA-HDA fueron R, M, y K (negro).

Bajo estos resultados, se pueden tener varias hipótesis sobre la razón de las decisiones de la GP para el uso de las dimensiones. En FOA, la orientación es la dimensión de mayor uso, sin embargo lo sigue muy de cerca la dimensión de color. El uso de color era de esperarse para el tipo de objeto usado en este experimento; el color rojo predominante en la mayoría del objeto genera prominencia, y resalta de un fondo con predominantes colores de tonos blancos, verdes y cafés.



Considerando que la GP podría producir funciones muy dependientes de la dimensión de color, se agregó otro objeto de un color muy similar en otra parte de la escena para que sirviera de *contrapeso* visual. Esto obligaba al programa a que otro tipo de características fueran necesarias para lograr diferenciar un objeto del otro. En la Figura 98 se puede observar una de las fotos usadas en el experimento. Como se menciona, las funciones de derivadas de gaussianas parecen generar información de gran utilidad para la GP y esta puede ser una de las razones por las que la información de la dimensión de orientación pasa a ser más usada que la de color. Por otro lado, la dimensión de forma no es tan usada como las otras dos, aunque no podemos descartar su utilidad.



**Figura 98:** Se fuerza al programa a buscar una manera de diferenciar correctamente un objeto de otro a pesar de la similitud en color.

Para FOA-LDA, la dimensión de color tomó más uso que orientación y forma. Hay que considerar que de cierta manera, la información de profundidad establece indirectamente la forma del objeto, es decir, al establecer un vecindario (y posteriormente la máscara de profundidad), se está definiendo qué parte de la imagen es parte del objeto, y por tanto queda resaltada una forma. Al ser la distancia la que entonces genera cierta delimitación de la figura del objeto, el algoritmo da prioridad al uso del color, aún habiendo un objeto de color similar en la escena.

Finalmente, FOA-HDA da un giro y toma a la forma como la de mayor uso, después a color, y finalmente orientación. FOA-HDA se caracteriza por filtrar toda información que no pertenezca al área de interés, e incluye el funcionamiento de FOA-LDA. Una idea de lo que puede estar sucediendo en esta situación puede ser debido al filtrado de FOA-HDA. Al no poder realizar recortes en una imagen, lo que se hace es igualar a cero todos los valores de píxeles que no corresponden al área de interés, y como resultado, la imagen que queda puede presentar varios bloques o manchas de color negro. Esto puede generar una gran cantidad de bordes ¿Es posible que estos bordes pudieran generar información que distraiga al foco de atención? Esto pudiera hacer que su uso generara funciones de menor aptitud.

También es interesante notar que en los experimentos, el objeto con color similar quedó fuera del área de interés, por lo que el sistema tenía la posibilidad de establecer la dimensión de color como la solución definitiva. Sin embargo no fue así, y se puede ver que de hecho, la dimensión de forma es usada el doble que la de color.

En cuanto a las funciones usadas para cada dimensión en los tres algoritmos, se puede observar más constancia con las funciones de mayor uso. También para los terminales, donde se usaron aquellos que tenían mayor información sobre el objeto como lo son el magenta, rojo, las oponencias de colores rojo-verde, y el valor de saturación.

Las diferencias mostradas en los resultados de los tres algoritmos impulsan a realizar un análisis mucho más profundo de cómo se construyeron los mejores individuos y detallar el procedimiento que siguen para producir sus proto-objetos. De esta manera se podrán comprobar las hipótesis mencionadas. Este es un trabajo que queda a futuro.

#### **8.4.1.3. Desempeño de detección**

El desempeño en detección está medido con los resultados de la etapa de prueba. Las gráficas de las figuras 29, 42 y 55 muestran los resultados, así como las tablas 2, 3 y 4.

Se debe recordar que la detección está representada por una imagen binaria, donde una sola mancha de color blanco representa el área que pertenece al objeto, mientras

que el color negro representa el resto. Los píxeles que construyen la detección pueden clasificarse como positivos verdaderos y falsos, y negativos verdaderos y falsos. Esta clasificación se detalla en la Sección 8.2.1.

Idealmente, la imagen del proto-objeto perfecto es aquella que está compuesta sólo de positivos verdaderos y negativos verdaderos. Por supuesto, es muy difícil obtener este tipo de resultados, por ello, se busca tener soluciones que se acerquen lo más posible a tal situación.

Las mejores soluciones de FOA crearon proto-objetos con un promedio de 65.10 % ( $\pm 28.63$  %) de positivos verdaderos. De forma complementaria, el promedio de negativos falsos fue 34.9 % ( $\pm 28.63$  %). En cuanto a positivos falsos, abarcaron un 4.68 % ( $\pm 17.21$  %). Correspondiente a esto, sus negativos verdaderos quedaron en 95.32 % ( $\pm 17.21$  %).

Con FOA-LDA se detectó un promedio de 69.09 % ( $\pm 24.7$  %) de positivos verdaderos, por lo que los negativos falsos quedaron en 30.91 % ( $\pm 24.7$  %). En cuanto a positivos falsos, detectó un 1.66 % ( $\pm 2.4$  %). Correspondiente a esto, los negativos verdaderos quedaron en un 98.34 % ( $\pm 2.4$  %).

Por último, con FOA-HDA los positivos verdaderos quedaron 74.03 % ( $\pm 20.21$  %), y de forma correspondiente 25.97 % ( $\pm 20.21$  %) para los negativos falsos. En cuanto a positivos falsos, detectó un 14.52 % ( $\pm 32.79$  %). Entonces, los negativos verdaderos quedaron en un 85.48 % ( $\pm 32.79$  %).

Se puede observar una mejoría en la detección de positivos verdaderos por parte de FOA-LDA y FOA-HDA en comparación con FOA. También los negativos verdaderos de FOA-LDA mejoraron con respecto a FOA; no así para FOA-HDA, ya que aumentó la detección de positivos falsos.

#### **8.4.1.4. Composición del proto-objeto**

Ya que las gráficas de positivos y negativos están en porcentajes, es difícil imaginar el cómo está compuesto el proto-objeto. Las gráficas de las figuras 29, 42 y 55 muestran la parte del proto-objeto de entrenamiento que fue cubierta por el proto-objeto producido

por las soluciones de los tres algoritmos. Las gráficas de las figuras 30, 43 y 56 muestran entonces sólo la composición del proto-objeto producido por los algoritmos.

La composición promedio del proto-objeto de FOA quedó así: Los positivos verdaderos sólo componen un 29.9% ( $\pm 19.51\%$ ), siendo el resto del proto-objeto, 70.1% ( $\pm 19.51\%$ ) positivos falsos. Quiere decir que más de la mitad de la detección no pertenecía al objeto.

Para FOA-LDA, la composición promedio del proto-objeto quedó así: los positivos verdaderos representaron un 36.51% ( $\pm 18.82\%$ ) de la mancha, siendo el resto del proto-objeto, 63.49% ( $\pm 18.82\%$ ), positivos falsos. Quiere decir que la más de la mitad del proto-objeto sigue siendo detección falsa. Sin embargo, ha mejorado con respecto a FOA.

Para FOA-HDA, la composición promedio del proto-objeto quedó así: los positivos verdaderos representaron un 42.59% ( $\pm 24.36\%$ ) de la mancha, siendo el resto del proto-objeto, 57.41% ( $\pm 24.36\%$ ), positivos falsos. Se debe notar que la gráfica de la Figura 55 muestra que la detección falsa aumentó en comparación con los otros dos algoritmos, sin embargo, en esta gráfica podemos ver que a pesar de ello la composición del proto-objeto sigue siendo mejor que FOA y FOA-LDA.

#### **8.4.1.5. Punto de FOA**

Se debe recordar que el punto de FOA es el punto de mayor prominencia, resultado de la red neuronal WTA (ver Sección 6.1.3); la mancha del proto-objeto se crea en base al punto ganador del mapa de prominencia.

Las gráficas de las figuras 32, 45, y 58 muestran los promedios de certeza del punto de FOA por cada grupo de ejecuciones. Las tablas 2, 3 y 4 muestran el resultado promedio. Se puede ver que para el algoritmo de FOA, un 46.19% ( $\pm 28.49\%$ ) de puntos de FOA fueron detectados dentro de el objeto. Mientras tanto, para FOA-LDA fueron un 72.62% ( $\pm 22.20\%$ ), y para FOA-HDA fueron un 90% ( $\pm 9.67\%$ ). Es de notar que no sólo el promedio de certeza aumentó para los nuevos algoritmos, sino que la desviación estándar también disminuyó.

Este comportamiento tiene mucho sentido debido a lo siguiente. Para FOA, el que el punto de FOA no se encuentre sobre el objeto no significa mucho problema, mientras la

mancha que produzca cubra al objeto, está haciendo bien su trabajo. Sin embargo, para FOA-LDA y FOA-HDA, este punto toma una importancia crucial; el punto de foco de atención ahora tiene una posición en el espacio tridimensional. Esto quiere decir que si el punto estuviera fuera del vecindario del objeto, es muy probable que, aunque originalmente pudiera estar cubriendo una parte del objeto, esta ya no sea tomada en cuenta porque el punto de FOA se encuentra en otro nivel de profundidad.

El punto de foco de atención representa el punto central de donde se está ubicando la fovea. Por tanto, físicamente no es posible estar concentrándose en un punto que está atrás o adelante del objeto que se quiere observar. Para FOA-LDA y FOA-HDA este punto debe ser congruente con el objeto que se está observando.

#### 8.4.2. Entrenamiento en vivo

Los resultados de aptitud del entrenamiento en vivo se pueden observar en la Tabla 9.

**Tabla 9: Comparación de aptitud en la etapa de entrenamiento (promedio inicial, promedio final, y mejor individuo) y la etapa de prueba para el entrenamiento en vivo.**

	Prom. inicial	Prom. final	Mejor individuo	Prom. en pruebas
<b>FOA</b>	3.18 ±7.14	47.71 ±18.49	77.18	17.65 ±25.92
<b>FOA-LDA</b>	6.78 ±10.58	34.55 ±14.07	57.39	18.18 ±21.75
<b>FOA-HDA</b>	1.62 ±1.62	11.58 ±4.82	47.58	1.71 ±3.95

Con el entrenamiento en vivo se pueden observar resultados muy diferentes, en comparación con el entrenamiento con bases de imágenes. Para empezar, las gráficas de la aptitud en el proceso evolutivo para FOA (Figura 59) muestran que el valor de aptitud del mejor individuo fue mayor que con la base de imágenes. Las gráficas de las figuras 72 y 85 muestran que, en los nuevos algoritmos, el mejor individuo disminuyó en el máximo de aptitud alcanzada. También, contrario al entrenamiento con base de imágenes, el algoritmo de FOA alcanzó mayor aptitud, después FOA-LDA, y finalmente FOA-HDA con la menor aptitud.

Sin embargo, los resultados de la etapa de prueba muestran lo contrario a los resultados de aptitud en la evolución (véase las tablas 5, 6 y 7). La aptitud de las soluciones encontradas por FOA bajó hasta 17.65 % ( $\pm 25.92$ ), los positivos verdaderos quedaron en 26.03 % ( $\pm 37.06$  %), y estos sólo componen el 14.02 % ( $\pm 21.70$  %) del proto-objeto.

Los nuevos algoritmos obtuvieron mejores resultados en la etapa de prueba, pero no mejores que los entrenados con bases de imágenes. Las soluciones de FOA-LDA alcanzaron una aptitud de 18.18 % ( $\pm 21.75$  %), con un 41.37 % ( $\pm 41.85$  %) de positivos verdaderos, los cuales formaron el 12.79 % ( $\pm 16.40$  %) del proto-objeto. Para FOA-HDA la aptitud bajó hasta 1.71 % ( $\pm 3.95$  %); logró cubrir un 64.31 % ( $\pm 36.35$  %) de positivos verdaderos, pero estos sólo representaron 0.91 % ( $\pm 2.26$  %) del proto-objeto.

Los resultados obtenidos muestran sólo una cosa: el método de entrenamiento no puede ser utilizado con el algoritmo que se siguió para la modalidad “en vivo”. En esta sección se explica porqué y qué es lo que falta para lograr que el entrenamiento en vivo funcione.

Se debe recordar que el algoritmo de FOA produce soluciones a partir del entrenamiento realizado con la programación genética. Este método utiliza un conjunto de imágenes de entrenamiento para comprobar la capacidad de las soluciones que construye. Esta capacidad se mide por medio de una función de aptitud, y este valor de aptitud queda asignado a su individuo para después ser el valor que guía al método de selección.

En un entrenamiento, la base de imágenes no cambia, y por tanto el valor de aptitud que tiene un individuo es válido para tal grupo de imágenes aunque se trate de una nueva generación. Es aquí donde surgen los problemas para la modalidad “en vivo”.

Suponga que se tienen un conjunto de individuos de una generación  $N$ , a los cuales se les calcula su valor de aptitud en base a las imágenes de entrenamiento designadas para el experimento. Se guarda el mejor individuo, y para la siguiente generación se realiza una selección por algún método (e.g., selección por ruleta). Entonces la generación  $N + 1$  está compuesta de nuevos individuos y el mejor de la generación anterior  $N$ , designado como *best\_sol*. Como la base de imágenes sigue siendo la misma, el individuo *best\_sol* es tan bueno en la nueva generación como lo era en la anterior. Esto ya no es válido para los entrenamientos “en vivo”.

En un entrenamiento “en vivo”, ya no se garantiza que el individuo *best\_sol* sea tan bueno en la generación  $N + 1$  como lo fue en la generación  $N$ . Esto quiere decir que existe el peligro de que al final del proceso evolutivo, el mejor individuo no sea aquel que

ha mejorado con el tiempo en base al mismo grupo de imágenes, sino algún individuo que casualmente fue mejor que los demás para la última generación, nada más. Así como no hay garantía de que el mejor individuo de la generación  $N$  sea igual de bueno para  $N + 1$ , tampoco se puede asegurar que el mejor individuo de la generación  $N + 1$  haya sido bueno en las generaciones anteriores. Esto explica el comportamiento de altas y bajas en las gráficas de la evolución de aptitud para los experimentos de entrenamiento “en vivo” (Véase las figuras 59, 72 y 85) en comparación con el entrenamiento con base de imágenes (véase las figuras 19, 33 Y 46).

Es evidente entonces que el método de evolución debe ser modificado para que pueda funcionar con una base de imágenes que cambia en cada generación, con la única garantía de que las nuevas imágenes de entrenamiento siguen siendo del mismo objeto. El cálculo de aptitud debe ser diferente; ya no puede depender sólo de la aptitud del momento. Un individuo que lo hizo mal en la nueva generación aún puede ser útil; por otro lado, un individuo que lo hizo muy bien en la generación presente, puede en realidad no ser tan bueno para las demás generaciones. Adicionalmente, puede que se necesite más de una fotografía por generación, para ayudar a que el cálculo de la aptitud tenga más resultados para trabajar.

Gracias a los resultados obtenidos, se puede seguir trabajando para construir un método de entrenamiento “en vivo”, buscando soluciones para los problemas que han sido expuestos por estos experimentos.

### **8.4.3. Tiempos de ejecución**

Debido a los cambios realizados para los nuevos algoritmos, es importante conocer sus tiempos de ejecución. Para esto se obtuvo el tiempo promedio de la ejecución de los 30 individuos de cada algoritmo que se usaron para los resultados. Cada individuo realizó 14 ejecuciones usando las imágenes de la etapa de prueba.

El algoritmo de FOA tuvo un tiempo promedio de ejecución de  $0.8627(\pm 0.2458)$  segundos. El algoritmo de FOA-LDA tuvo un tiempo promedio de ejecución de  $1.0328(\pm 0.2725)$  segundos. Finalmente, el tiempo promedio de ejecución del algoritmo de FOA-HDA fue de  $10.7577(\pm 3.9327)$  segundos.

Aunque FOA-HDA es el algoritmo que muestra mejores resultados, tiene la desventaja en el tiempo de ejecución. Sin embargo, se debe considerar que los algoritmos de FOA-LDA no están optimizados, por lo que aún queda la oportunidad de mejorar estos tiempos, especialmente si se modifican para trabajar en paralelo.

Este capítulo muestra todos los resultados obtenidos de los seis experimentos realizados: FOA, FOA-LDA y FOA-HDA, cada uno con entrenamiento con bases de imágenes y “en vivo”. En el capítulo 9 se dan a conocer las conclusiones formadas gracias a la realización del trabajo de tesis. Finalmente, en el Capítulo 10 se da a conocer el trabajo a futuro.





## Capítulo 9. Conclusiones

---

Se ha propuesto un método para integrar la dimensión de profundidad en el proceso de la ruta dorsal artificial (ADS), funcionando bajo el paradigma de las potencialidades en función de la distancia. Este método ha mostrado tener un mejor desempeño en detección comparándose con su versión anterior ADS/FOA. El porcentaje de positivos verdaderos en la detección aumentó. La composición promedio de los proto-objetos también mejoró, pues FOA-DA construyó proto-objetos con menos positivos falsos, y de forma equivalente, aumentaron los positivos verdaderos.

Además, la ubicación del punto de foco de atención dentro del objeto detectado es mucho mejor para FOA-DA. Esto último es importante, pues permite tener mejor certeza para la ubicación del objeto detectado en el espacio. Se debe recordar que la ruta dorsal realiza tareas visomotoras, y para ello, es crucial identificar la posición del objeto en el espacio. FOA genera un proto-objeto que cubre parte del objeto observado. Sin embargo, una parte considerable del proto-objeto producido señala otras cosas aparte del objeto. Si se quiere saber la posición de este, utilizar las distancias de los píxeles de todo el proto-objeto puede resultar en un problema, pues se necesitaría distinguir entre los píxeles que realmente pertenecen al objeto para saber cuáles distancias se pueden tomar en cuenta. Sin embargo, si se puede dar cierta certeza de que el punto principal y más prominente del proto-objeto sí se encuentra dentro del mismo, entonces se puede usar este punto como referencia para la distancia del objeto observado. La certeza de este punto es mucho mejor en FOA-DA.

Por ello, los resultados muestran que la dimensión de profundidad aporta de forma positiva en el proceso de la atención visual. Tal conclusión concuerda con las de otros experimentos donde también integra la dimensión de profundidad a un modelo de atención visual (Penaloza *et al.*, 2012; Ogawa *et al.*, 2014).

El método propuesto funciona en dos modalidades: FOA-LDA (Foco de Atención con Potencialidades por Profundidad de Bajo Nivel) que se usa cuando no se tiene información previa sobre las potencialidades que ofrecen los espacios de un entorno; y FOA-HDA (Foco de Atención con Potencialidades por Profundidad de Alto Nivel) el cual usa la in-

formación de las potencialidades basadas en los espacios de la escena para guiar la atención. Estas dos modalidades son similares en concepto a los dos procesos de atención visual: de abajo hacia arriba (Bottom-Up) y de arriba hacia abajo (Top-Down).

A pesar de estar basado en el algoritmo de la ADS y conceptos de la psicología ecológica, el sistema resultante no es totalmente fiel a la biología del funcionamiento del cerebro humano. Además de haber información limitada sobre cómo el cerebro integra o utiliza la información de profundidad, el método de entrada para la información visual es diferente al que usa el ser humano. En lugar de tener un sistema de cámara estéreo, se usa una cámara Kinect, la cual entrega directamente la información de profundidad de la escena, medida en milímetros. Con los debidos cambios, es posible que este sistema pueda interpretar la información de disparidad que proporciona una cámara estéreo. Aún así, el uso de la cámara Kinect ofrece ventajas como simplicidad para la adquisición de la información de profundidad, el soporte para los sistemas operativos Windows y Linux, y el precio. Esto es importante pues permitirá que el sistema pueda ser probado por una gran cantidad de usuarios sin que necesiten gastar muchos recursos.

Se propuso el paradigma de potencialidades en función de la distancia, y su justificación. Este concepto está directamente basado en la idea de potencialidades de Gibson (2014). Creemos que este paradigma propone un concepto interesante sobre cómo los espacios pueden representar potencialidades, dependiendo del observante, y cómo estas potencialidades pueden guiar a la atención visual. El sistema actual es una representación funcional aunque incompleta de este paradigma, y queda para trabajo a futuro completar la representación total de este.

También se realizaron propuestas para automatizar el proceso de aprendizaje. El algoritmo para crear bases de imágenes de manera automática tiene grandes ventajas en velocidad. También disminuye la intervención humana en el proceso, pero no la elimina por completo. El procedimiento tiene sus limitantes: es necesaria una base giratoria para que cambie la posición del objeto, se debe cuidar de que no haya cambios en el fondo, y es necesaria una buena iluminación para disminuir el ruido en la imagen. Con respecto a esto, también se han propuesto algunas mejoras para disminuir o eliminar ciertas desventajas, como mejoras en los filtros, y métodos para poder distinguir entre objetos nuevos y

las sombras.

Aparte de la automatización para crear bases de imágenes de entrenamiento, se experimentó con una forma de entrenamiento que simulara la manera en la que el ser humano es expuesto a los objetos día con día para su aprendizaje. Con esto, el modelo de aprendizaje ya no tendría acceso a una base de imágenes, sino a una sola imagen capturada en el momento. A este se le nombró como entrenamiento “en vivo”.

Sin embargo, los resultados de las soluciones de aprendizaje en vivo mostraron un desempeño mucho más bajo que las soluciones obtenidas con entrenamiento con bases de imágenes. Esto se debe a que se cometió un error al sobrestimar la capacidad de la GP bajo las condiciones que se impusieron. Para lograr corregir el problema, se debe modificar la manera en que la GP calcula la aptitud de los individuos. La forma en que lo hace cuando se tiene una base de imágenes funciona correctamente. Sin embargo, en un aprendizaje en vivo la imagen de entrenamiento cambia en cada generación, y esto hace que el sistema de aprendizaje tenga fallas. El mejor individuo de la generación presente ya no garantiza tener la misma aptitud con la siguiente generación, pues se trata de una imagen diferente. Esta es la razón por la que el algoritmo de FOA con entrenamiento en vivo aparenta producir individuos con mejor aptitud que los de su versión con base de datos, sin embargo su desempeño en detección en la etapa de prueba termina siendo mucho más bajo.

Cabe mencionar que, aunque el proceso de entrenamiento en vivo no resultó tan bueno, los modelos que usaron la profundidad demostraron tener más casos en los que lograron mucha mayor detección que FOA, aunque su aptitud en general también fue muy baja. Es importante hacer notar que FOA-LDA pudo generar mayor detección gracias a su referencia en la distancia y a la necesidad de centrar el punto de FOA dentro del objeto, lo que le dejaba en mejor posición al momento de tratar de detectar el área de este.

A pesar de los resultados, gracias a los experimentos realizados con el entrenamiento en vivo, se han identificado los cambios que se necesitan aplicar al método de entrenamiento para que este pueda funcionar como se espera.

Ahora que el proceso de la ADS integra la información de distancia, se continua en

el proceso de simular por completo el funcionamiento de la ruta dorsal. Se debe recordar que la ruta dorsal realiza las funciones visomotoras, y para realizarlas, necesita la información de profundidad de los objetos en la escena. Con FOA-DA los robots humanoides como NAO, o los brazos de robots con cámaras de RGB-D montadas, tienen las herramientas para realizar este tipo de tareas.

Con todo lo hecho en este trabajo de tesis, aún quedan más objetivos por cumplir e ideas por desarrollar. En el Capítulo 10, se presenta una lista los planes sobre trabajo a futuro.

## Capítulo 10. Trabajo a futuro

---

Existen aún muchas partes del Sistema FOA-DA que necesitan afinarse o desarrollarse. La lista presentada muestra los puntos en los que se estará trabajando.

- Usar un mejor método para detectar las siluetas. El algoritmo que se usa para formar las siluetas de entrenamiento es muy básico, lo que hace que sea sensible a los ruidos de luz ambiental y a las sombras. Un método que podría resolver este problema es el de Horprasert *et al.* (1999), quienes desarrollaron un algoritmo estadístico para detectar, en base al modelo estadístico de una escena, los objetos nuevos (sustracción de fondo) y la sombra que producen. Esto es exactamente lo que se necesita en el algoritmo que automatiza la creación de imágenes de entrenamiento. El modelo necesita al menos 100 muestras del fondo, pero es algo que no le lleva a la computadora más de unos minutos procesar.
- Agregar la parte de integración de profundidad a la programación cerebral. Las funciones de vecindarios y profundidad general al momento de integrar la distancia, se basan en valores estáticos, que aunque funcionan bien para el objeto que se usó en este experimento, es posible que otros objetos necesiten otros valores, para estos casos, sería interesante agregar estos dos valores a la búsqueda realizada por la GP. De esta manera, los vecindarios y la profundidad general serían adaptables a las necesidades del objeto.
- Adaptar la GP para poder evolucionar con imágenes en vivo. Es evidente que la GP necesita calcular la aptitud de los individuos de otra manera si se quiere lograr un aprendizaje en vivo.
- Desarrollar más el concepto de HDA. Para fines de demostrar el concepto de potencialidades en profundidad de alto nivel, se da por sentado que el sistema ya tiene conocimiento de estas potencialidades. Pero queda una pregunta en el aire: ¿cómo forma estos mapas? Es evidente que nadie nace conociéndolos, y que conforme avanza la experiencia personal, estas potencialidades van tomando forma. El funcionamiento ideal sería aquel en el que los mapas de potencialidades se van formando conforme a la experiencia del sistema inteligente.
- Comenzar a formar un modelo de conocimiento. Si deseamos comenzar a imple-

mentar todo este aprendizaje en una aplicación (robots), necesitamos algo que le de forma a todo el conocimiento, producto del entrenamiento. Una de las propuestas que se consideran es la implementación de ontologías, que puedan clasificar y relacionar todo el conocimiento. Las ontologías son una de las formas en que se podría representar las potencialidades de alto nivel.

- Integrar FOA-DA a la AVC. Ahora que ya se ha visto el funcionamiento de FOA con profundidad, se debe averiguar como hacerlo funcionar junto con la AVS para formar una AVC que implemente los conceptos de FOA-DA. La información que produce la ADS es utilizada por la AVS, entonces se espera que las mejoras que produjo la integración de distancia en la ADS sirvan también a la AVS.

## Lista de referencias bibliográficas

- Berg, A. (2011). Kinect to matlab on osx and linux via mex. Recuperado el 12 de abril de 2016: <http://acberg.com/kinect/>.
- Chun, M. M. y Wolfe, J. M. (2008). Visual attention. En: *The Blackwell Handbook of Sensation and Perception*. Blackwell Publishing Ltd., pp. 272–310.
- Clemente, E., Olague, G., Dozal, L., y Mancilla, M. (2012). Object recognition with an optimized ventral stream model using genetic programming. En: *Applications of Evolutionary Computation*. Springer, pp. 315–325.
- Clemente Torres, E. H. (2015). *Reconocimiento de objetos en una escena bajo el paradigma del cómputo evolutivo y la corteza visual*. Tesis de doctorado. Centro de Investigación Científica y Educación Superior de Ensenada. 199 p.
- Crock, N. (2011). Kinect depth vs. actual distance. Recuperado el 12 de abril de 2016: <http://mathnathan.com/2011/02/depthvsdistance/>.
- Dickens, C., Drake, D., Hjelm, N., Erdfelt, J., de Goede, H., Rousseau, L., Batard, P., Chen, X., Niedermann, H. U., y Gray, T. (2014). libusb - a cross-platform library that gives apps easy access to usb devices. Recuperado el 12 de abril de 2016: <https://sourceforge.net/projects/libusb/files/libusb-1.0/libusb-1.0.18/>.
- Dozal, L., Olague, G., Clemente, E., y Hernández, D. E. (2014). Brain programming for the evolution of an artificial dorsal stream. *Cognitive Computation*, **6**(3): 528–557.
- Dozal García, L. F. (2014). *Evolución de una ruta dorsal artificial utilizando programación cerebral para la atención visual*. Tesis de doctorado. Centro de Investigación Científica y Educación Superior de Ensenada. 102 p.
- Explorable (2008). Método de muestreo estratificado. Recuperado el 12 de abril de 2016: <https://explorable.com/es/muestreo-estratificado/>.
- Farivar, R. (2009). Dorsal–ventral integration in object recognition. *Brain Research Reviews*, **61**(2): 144–153.
- Gibson, J. J. (2014). *The ecological approach to visual perception: classic edition*. Psychology Press. pp. 119–136.
- Giesel, M. y Zaidi, Q. (2014). Rapid sensing of material affordances. En: *First Workshop on Affordances: Affordances in Vision for Cognitive Robotics (in conjunction with RSS 2014)*, Berkeley, USA.
- Goddard, E., Mannion, D. J., McDonald, J. S., Solomon, S. G., y Clifford, C. W. (2011). Color responsiveness argues against a dorsal component of human v4. *Journal of vision*, **11**(4): 3–3.
- Gonzalez, R. C. y Woods, R. E. (2002). *Digital Image Processing*. SL: Prentice Hall, segunda edición. 793 p.
- Goodale, M. A. y Milner, A. D. (1992). Separate visual pathways for perception and action. *Trends in neurosciences*, **15**(1): 20–25.



- Grabner, H., Gall, J., y Van Gool, L. (2011). What makes a chair a chair? En: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, pp. 1529–1536.
- Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, **43**(5): 907–928.
- Hecht, E. (2002). *Optics*. Addison Wesley, cuarta edición. 680 p.
- Horprasert, T., Harwood, D., y Davis, L. S. (1999). A statistical approach for real-time robust background subtraction and shadow detection. En: *IEEE ICCV*. Vol. 99, pp. 1–19.
- Itti, L. y Koch, C. (2001a). Computational modelling of visual attention. *Nature reviews neuroscience*, **2**(3): 194–203.
- Itti, L. y Koch, C. (2001b). Feature combination strategies for saliency-based visual attention systems. *Journal of Electronic Imaging*, **10**(1): 161–169.
- Itti, L., Koch, C., y Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, **20**(11): 1254–1259.
- Johnston, B., Yang, F., Mendoza, R., Chen, X., y Williams, M.-A. (2008). Ontology based object categorization for robots. En: *Practical Aspects of Knowledge Management*. Springer, pp. 219–231.
- Kjellström, H., Romero, J., y Kragić, D. (2011). Visual object-action recognition: Inferring object affordances from human demonstration. *Computer Vision and Image Understanding*, **115**(1): 81–90.
- Koch, C. y Ullman, S. (1987). Shifts in selective visual attention: towards the underlying neural circuitry. En: *Matters of intelligence*. Springer, pp. 115–141.
- Maillot, N., Thonnat, M., y Boucher, A. (2004). Towards ontology-based cognitive vision. *Machine Vision and Applications*, **16**(1): 33–40.
- Mishkin, M., Ungerleider, L. G., y Macko, K. A. (1983). Object vision and spatial vision: two cortical pathways. *Trends in neurosciences*, **6**: 414–417.
- Mizoguchi, R. (2003). Part 1: introduction to ontological engineering. *New Generation Computing*, **21**(4): 365–384.
- Montero, D. (2011). Openkinect in opensuse. Recuperado el 12 de abril de 2016: <http://www.david-montero.es/openkinect-in-opensuse/>.
- Myers, A., Kanazawa, A., Fermuller, C., y Aloimonos, Y. (2014). Affordance of object parts from geometric features. En: *Workshop on Vision meets Cognition, CVPR*, Boston, MA, USA.

- Ogawa, T., Ozeki, M., y Oka, N. (2014). A visual attention model using depth information from the point of gaze. En: *International Workshop on Advanced Image Technology*, Bangkok, Thailand.
- Olague, G., Clemente, E., Dozal, L., y Hernández, D. E. (2014). Evolving an artificial visual cortex for object recognition with brain programming. En: *EVOLVE-A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation III*. Springer, pp. 97–119.
- OpenKinect Community (2015). Getting started - openkinect. Recuperado el 12 de abril de 2016: [https://openkinect.org/wiki/Getting\\_Started/](https://openkinect.org/wiki/Getting_Started/).
- Penaloza, C. I., Mae, Y., Ohara, K., y Arai, T. (2012). Using depth to increase robot visual attention accuracy during tutoring. En: *Proceedings of IEEE International Conference on Humanoid Robots-Workshop of Developmental Robotics*. pp. 14–19.
- Penta, A., Picariello, A., y Tanca, L. (2007). Towards a definition of an image ontology. En: *Database and Expert Systems Applications, 2007. DEXA'07. 18th International Workshop on*. IEEE, pp. 74–78.
- The BLFS Development Team (2003). Beyond linux from scratch - version 7.7. chapter 9. general libraries. Recuperado el 12 de abril de 2016: <http://www.linuxfromscratch.org/blfs/view/7.7/general/libusb.html>.
- Treisman, A. M. y Gelade, G. (1980). A feature-integration theory of attention. *Cognitive psychology*, **12**(1): 97–136.
- Tünnermann, J. y Mertsching, B. (2014). Saliency and affordance in artificial visual attention. Berkeley, USA. RSS.
- Varadarajan, K. M. y Vincze, M. (2012). Afrob: The affordance network ontology for robots. En: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, pp. 1343–1350.
- Varwani, H., Choithwani, H., Sahatiya, K., Gangan, S., Gyanchandani, T., y Mane, D. (2013). Understanding various techniques for background subtraction and implementation of shadow detection. *International Journal of Computer Technology and Applications*, **4**(5): 822.
- Wolfe, J. M. (1994). Guided search 2.0 a revised model of visual search. *Psychonomic bulletin & review*, **1**(2): 202–238.



# Apéndices

## A. Biblioteca libfreenect

Para que sea posible usar la cámara de Kinect por medio de la PC es necesario instalar la API de libfreenect. En esta sección se listan los pasos básicos para su instalación en una computadora con sistema operativo Linux openSUSE 13.2. Para ver la información completa o seguir una instalación más sencilla en sistemas Linux Ubuntu, se puede ver el instructivo en la página de la comunidad de OpenKinect (OpenKinect Community, 2015).

Primero se instalan las dependencias:

```
$ sudo zypper in cmake libusb-1_0-0 libusb-1_0-devel pkg-config freeglut freeglut-devel
```

También pueden instalarse por medio del administrador de software de YaST.

Después se introducen los siguientes comandos para obtener una copia del proyecto git de libfreenect, compilarlo y finalmente instalarlo:

```
$ clone git://github.com/OpenKinect/libfreenect.git
$ cd libfreenect
$ mkdir build
$ cd build
$ cmake ..
$ make
$ sudo make install
```

El comando `cmake` servirá para configurar las direcciones de instalación. Es recomendable que cambie la dirección de instalación `"/usr/local/"` por `"/usr/local/libfreenect/"`.

Es posible que el paquete instalado de `libusb` esté en una versión anterior a la necesaria para el `libfreenect`. Si ese es el caso, saldrá un error como este al momento de la compilación:

```
../lib/libfreenect.so.0.5.0: undefined reference to 'libusb_get_parent'
collect2: error: ld returned 1 exit status
```

Para este caso, es necesario instalar `libusb` manualmente. La versión usada para este proyecto es la 1.0.18. El código fuente se puede obtener de la página de SourceForge (Dickens *et al.*, 2014). El instructivo se puede consultar en la página de LinuxFromScratch (The BLFS Development Team, 2003). Los pasos básicos son los siguientes, ejecutando-se dentro de la carpeta del código fuente de `libusb`:

```
$ ./configure --prefix=/usr --disable-static
$ make
$ make install
```

Con esto la biblioteca libfreenect ya está instalada. Ahora se necesita hacer una configuración para que el dispositivo Kinect pueda ser usado sin la necesidad del superusuario.

Para esto se debe crear un archivo de reglas:

```
$ cd /etc/udev/rules.d
$ sudo vi 66-kinect.rules
```

El contenido del archivo debe ser el siguiente:

```
#Rules for Kinect
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02b0", MODE:="0666"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02ad", MODE:="0666"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02ae", MODE:="0666"
#END
```

Si se está usando el editor “vi”, una vez escrito el archivo, se presiona la tecla ESC y se introduce el comando para guardar el archivo y cerrar el editor:

```
.wq
```

Para probar su funcionamiento, es necesario conectar el dispositivo Kinect a la computadora, y correr el siguiente comando:

```
$ /usr/local/libfreenect/bin/freenect-glfw
```

Si todo se encuentra bien configurado, se podrá ver en pantalla la secuencia de video, en color y profundidad, que esté capturando la cámara en el momento. Para más información sobre la instalación de libfreenect en openSUSE se puede consultar la página de Montero (2011).

## B. Funcionamiento entre Kinect y MATLAB

La API de libfreenect está escrita en lenguaje C/C++, por ello, es necesario integrar una interfaz que permita utilizar esta biblioteca junto con los programas del proyecto de tesis, los cuales están escritos para MATLAB. Afortunadamente, MATLAB puede ejecutar código en C y C++ usando los archivos MEX.

Alex Berg ya proporciona el código necesario para esta interfaz con archivos MEX, y se puede obtener de su página (Berg, 2011). Sin embargo, es posible que el código contenga un error, y tenga unas líneas de código faltantes, lo que haría que no funcione. Para corregir esto, sólo se deben agregar las siguientes dos líneas de código al archivo llamado "kinect\_mex.cc":

```
freenect_set_log_level(f_ctx, FREENECT_LOG_DEBUG);  
freenect_select_subdevices(f_ctx, (freenect_device_flags)(FREENECT_DEVICE_MOTOR  
— FREENECT_DEVICE_CAMERA));
```

Justo antes de estas dos líneas de código:

```
int nr_devices = freenect_num_devices (f_ctx);  
printf ("Number of devices found: %d\n", nr_devices);
```

Junto con el archivo kinect\_mex.cc, viene otro archivo llamado "test.m", el cual tiene un código de ejemplo para llamar a la función de inicio del Kinect y manejar el dispositivo por medio de la consola de comando de MATLAB.



## C. Proceso de validación

### C.1. Validación cruzada

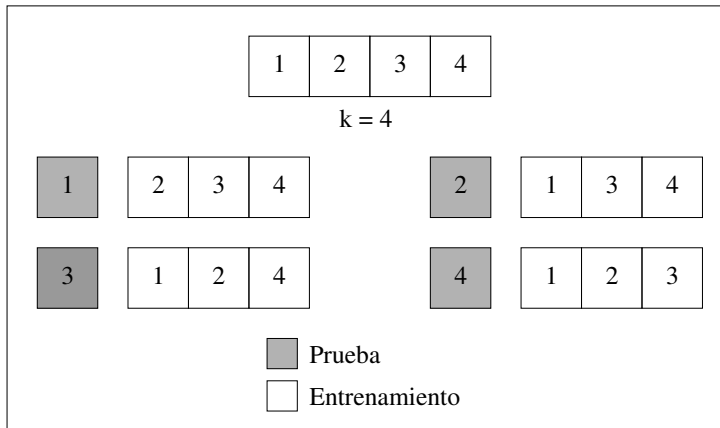
Se le llama *validación de  $k$  iteraciones* (también conocida en Inglés como “K-fold cross-validation”) a la técnica usada para evaluar los resultados de un análisis estadístico, y garantizar que estos resultados son independientes de los datos que conforman al conjunto de entrenamiento y de prueba. Esta técnica es usada ampliamente en inteligencia artificial para evaluar el desempeño de los distintos modelos que se crean.

Para realizar este tipo de validación, se divide el conjunto de datos que se usará para el experimento en  $k$  grupos del mismo tamaño. En el caso del presente trabajo de tesis se formaron seis grupos ( $k = 6$ ). Los datos son las ternarias de fotos: la imagen color (RGB), la de profundidad (D), y la de entrenamiento (imagen binaria). Se dispuso usar un total de 84 ternarias para los experimentos. Estas imágenes se seleccionaron por un muestreo de un conjunto de 390 ternarias. En el Apéndice C.2 se especifica el procedimiento de este muestreo.

Una vez formados los grupos, se ordenan y etiquetan de 1 a  $k$ . De esta manera se crean  $k$  conjuntos, de forma que cada conjunto  $n$  ( $1 \leq n \leq k$ ) asigna al grupo  $n$  como datos de prueba, y el resto de grupos como datos de entrenamiento. En la Figura 99 se puede ver un ejemplo de cómo se forman los grupos de entrenamiento y prueba para una validación cruzada con  $k = 4$ .

Una vez organizados los conjuntos, el mismo experimento se corre un número de  $x$  veces para cada conjunto. Como sus nombres lo indican, el grupo de entrenamiento es con el que el programa de IA creará sus modelos; el grupo de prueba medirá el desempeño del modelo creado con el correspondiente grupo de entrenamiento. Como paso final de la validación, se obtiene la media de los valores de desempeño de cada experimento.





**Figura 99:** Validación cruzada. Construcción de conjuntos de entrenamiento y prueba a partir de un conjunto de datos.

## C.2. Tipo de muestreo

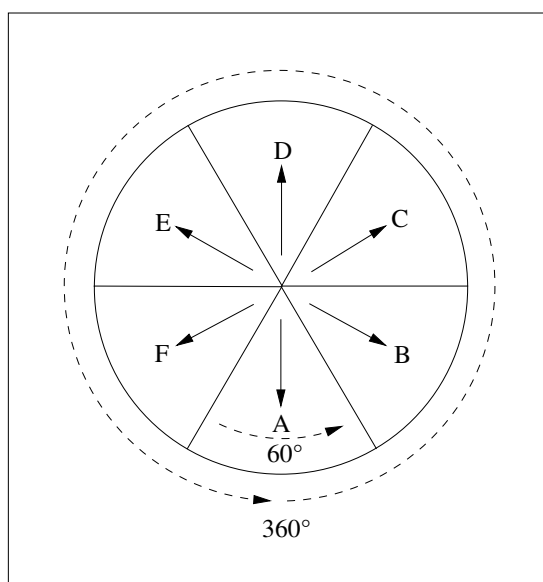
Las imágenes usadas para este experimento son una serie de fotografías tomadas a un objeto, de manera que sólo el objeto se mueve, y este movimiento es un giro de  $360^\circ$  a velocidad constante. Las fotografías fueron tomadas a intervalos constantes. Es importante para el experimento (y en general, los experimentos de aprendizaje visual de objetos) tener fotografías que representen la generalidad del objeto, *i.e.*, que con el conjunto de imágenes del objeto se puedan ver todas las partes que lo conforman. Para el caso de este trabajo de tesis, sólo es posible tener fotografías del objeto girando a lo largo de su eje vertical. Por tanto, es importante que las imágenes que componen el grupo de entrenamiento y prueba representen la generalidad del giro del objeto.

El muestreo aleatorio sería ineficaz en este caso, ya que existe la posibilidad de que unos grupos de entrenamiento queden sin una buena representación sobre ciertos ángulos del objeto. Por ejemplo, un grupo de entrenamiento podría quedarse sin imágenes del frente del objeto; estas imágenes quedarían en el grupo de prueba. Esto haría que el modelo no tuviera la oportunidad de entrenar correctamente, y obtendría malos resultados en la etapa de prueba. Otro caso podría ser que la muestra inicial obtenga porciones desequilibradas de las posiciones del objeto; es deseable tener una variedad equilibrada de los ángulos del objeto.

Como existe una serie de ángulos que se desea ver representados en los grupos y en

la muestra inicial, el tipo de muestreo que se adapta a estas necesidades es el *muestreo aleatorio estratificado*. Este tipo de muestreo divide el conjunto de datos en estratos. Cada estrato representa una característica que se desea ver en la muestra. Estos estratos no deben superponerse, es decir, cada fotografía puede pertenecer solamente a un estrato. Una vez organizados los datos, se realiza un muestreo aleatorio simple para cada estrato.

Para formar los estratos de nuestros experimentos, se dividió el giro del objeto en seis partes. De esta manera se cubren los ángulos de frente, detrás, y de tres cuartos hacia adelante y hacia atrás. En la Figura 100 se puede observar la división de los estratos conforme al giro del objeto.



**Figura 100:** El objeto se encuentra encima de una plataforma que lo hace girar. Las flechas que provienen del centro muestran hacia donde mira la misma cara del objeto en cada estrato. A cada estrato se le asigna una letra que servirá para ordenar las imágenes posteriormente.

Se tomaron seis series de fotografías al objeto a partir de la misma posición inicial. Un giro completo del objeto se representó en 390 imágenes, por lo que cada estrato se compone de 65 fotografías. Como se requieren 84 imágenes para el experimento, se deben extraer 14 imágenes de cada estrato.

Para seleccionar una imagen del estrato, primero se obtiene de forma aleatoria la serie de fotografías (recuérdese que se tomaron seis series), y después se elige de forma aleatoria una de las imágenes del estrato designado.

Con esto se tienen 14 imágenes por estrato, y cada una de ellas fue elegida al azar, pero en conjunto representan el giro completo del objeto de forma equilibrada. Sin embargo, aún falta ordenarlas de forma correcta para los grupos de validación cruzada.

Para ordenar las imágenes de manera que estén repartidas de forma equilibrada, se sigue este procedimiento. Para este ejemplo cada estrato se etiqueta con las letras *A* hasta *F*. Se toma una imagen de forma aleatoria del estrato *A* y se asigna en la primera posición de la lista del conjunto. Después se toma otra imagen aleatoria del estrato *B* y se asigna con la segunda posición. Se sigue de esta manera hasta el estrato *F*. Entonces se repite el proceso. Al final, se ordena el conjunto con imágenes seleccionadas de manera aleatoria, teniendo una serie de la siguiente forma:

A B C D E F A B C D E F A B C D ... F

Para la validación cruzada se divide esta serie en  $k$  grupos. Para este caso, donde  $k = 6$ , cada grupo está conformado de 14 imágenes. Entonces la serie quedaría así:

Grupo 1: A B C D E F A B C D E F A B

Grupo 2: C D E F A B C D E F A B C D

:

Grupo 6: E F A B C D E F A B C D E F

De esta forma, se garantiza que la generalidad de objeto está representada lo más equilibrada posible bajo los parámetros presentes, y con una selección aleatoria de imágenes.

## D. Ontologías

El concepto de *ontologías* es tomado de la filosofía. Como especifican Penta *et al.* (2007), trata la teoría de la existencia. Se intenta explicar qué es lo que es, y cómo el mundo está configurado introduciendo un sistema de *categorías críticas* para considerar las cosas y sus relaciones intrínsecas. Gruber (1995), introdujo este concepto a las ciencias computacionales: una ontología es una especificación explícita de una conceptualización; esta conceptualización es un punto de vista simplificado y abstracto del mundo que se desea representar para algún propósito, *i.e.*, los objetos, conceptos, y otras entidades que son asumidas que existen en algún área de interés, y las relaciones que mantienen entre ellos.

Desde entonces, el concepto de ontologías es usado como un enfoque para resolver múltiples problemas de inteligencia artificial, entre ellos, los de visión artificial.

De acuerdo a Mizoguchi (2003), el rol de una ontología se resume en los siguientes puntos:

- Permite definir un vocabulario común.
- Provee de una estructura apropiada a los datos para la descripción de la información y el intercambio de conocimiento.
- Ofrece una explicación profunda de lo que se deja implícito. Esto favorece principalmente la reutilización e intercambio del conocimiento.
- Interoperabilidad semántica, debido a que se puede diseñar una ontología que defina los significados de las etiquetas y los valores usados.
- Explicación de la razón de ser del diseño, ya que contribuye a la explicación de las suposiciones, precondiciones implícitas requeridas por el problema a resolver, así como la conceptualización del objetivo que refleja esas suposiciones. Contribuye a la sistematización del conocimiento.
- Especifica los modelos a construir, proveyendo de las guías y limitaciones que deben ser satisfechas, *i.e.*, nos provee de conceptos y sus relaciones que son usadas como los bloques de construcción del modelo.

Las ontologías son comúnmente (y erróneamente) conceptualizadas meramente como un conjunto de términos, jerarquía de conceptos, o representación de conocimiento. Sin embargo, las ontologías van más allá de estos tres conceptos, pues aunque los tres son parte de lo que conforma una ontología, el objetivo final de esta es “proveer de guías para modelar el mundo” (Mizoguchi, 2003). El problema de la ontología, entonces, “se trata un problema de contenido” (Mizoguchi, 2003).

Implementar correctamente ontologías a una tarea de reconocimiento de objetos no es sencillo, pero si es realizable (Penta *et al.*, 2007; Johnston *et al.*, 2008; Maillot *et al.*, 2004).

## E. Glosario de funciones y terminales

En esta sección se presenta un glosario de todas las funciones y terminales usadas por el algoritmo de la ADS para la construcción de los operadores visuales evolutivos y la función de integración de características.

### Funciones

**BotHat:** Bottom hat/Transformación de sombrero de copa negro. Función morfológica que devuelve una imagen que contiene los objetos o elementos que son más oscuros que sus alrededores y que son más pequeños que el elemento estructurante. Se define como la diferencia entre el cierre y la imagen de entrada.

**Close:** Cierre. Es una función morfológica. Sea  $A$  una imagen binaria, y  $B$  el elemento estructurante, el cierre de  $A$  por  $B$  se obtiene por la dilatación de  $A$  por  $B$ , seguida por la erosión de la imagen resultante por  $B$ .

**DiltDmnd:** Diamond dilation/Dilatación de diamante. Es una función morfológica. Se describe como el crecimiento de píxeles. El elemento estructurante  $B$  hace un recorrido por todos los píxeles de la imagen  $A$ , si la estructura de  $B$  toca un píxel de la región de  $A$ , al píxel señalado por el elemento estructurante  $B$  se le asigna el valor de "1". Se llama Dilatación Diamante porque la forma del elemento estructurante es de un diamante.

**DiltDsk:** Disk dilation/Dilatación de disco. Función morfológica de dilatación donde el elemento estructurante tiene la forma de un disco.

**DiltSqr:** Square dilation/Dilatación de cuadro. Función morfológica de dilatación donde el elemento estructurante tiene la forma de un cuadrado.

**ErsDmnd:** Diamond erosion/Erosión de diamante. Función morfológica, se describe como la reducción de píxeles. Sea  $A$  una imagen binaria, y  $B$  el elemento estructurante, la erosión de la imagen  $A$  es el conjunto de todos los píxeles de  $A$  en los que siendo colocado sobre ellos el origen de  $B$ , la estructura de  $B$  quedó completamente contenida en  $A$ . Se le llama erosión de diamante porque el elemento estructurante tiene forma de diamante.

**ErsDsk:** Disk erosion/Erosión de disco. Función morfológica de erosión donde el elemento estructurante tiene forma de disco.

**ErsSqr:** Square erosion/Erosión de cuadro. Función morfológica de erosión donde el elemento estructurante tiene forma de cuadrado.

**Exp:** Exponencial. Calcula el exponencial  $e^x$  de cada elemento de la entrada

**Gauss 1:** Se aplica un filtro Gaussiano a la imagen de entrada, con el valor de  $\sigma = 1$ .

**Gauss 2:** Se aplica un filtro Gaussiano a la imagen de entrada, con el valor de  $\sigma = 2$ .

**GaussDx:** Gradiente usando la derivada de Gaussianas de primer orden con respecto a  $X$ .

**GaussDy:** Gradiente usando la derivada de Gaussianas de primer orden con respecto a  $Y$ .

- Half:** Cada elemento de la imagen de entrada se multiplica por el valor de 0.05.
- HistEq:** Histogram Equalization/Ecualización del histograma. Proceso para ajustar los valores de intensidad de la imagen de entrada.
- HitmissDmnd:** Diamond hit-and-miss transform/Transformada “al azar” de diamante. Función morfológica. Sea A una imagen binaria y B el elemento estructurante, la transformada al azar de A por B es el conjunto de pixeles de A que al hacerlos coincidir con el origen B, la estructura coincide perfectamente en A (para los unos y ceros que definen la estructura de B). Se le llama de diamante porque la estructura de B tiene esta forma.
- HitmissDsk:** Disk hit-and-miss transform/Transformada al azar de disco. Función morfológica de transformada al azar donde el elemento estructurante tiene forma de disco.
- HitmissSqr:** Square hit-and-miss transform/Transformada al azar de cuadro. Función morfológica de transformada al azar donde el elemento estructurante tiene forma de cuadrado.
- Infimum:** Ínfimo. Recibe dos matrices A y B, y retorna los valores de A que hayan sido menores que los valores de B en cada fila.
- ImgAbs:** Se devuelve el valor absoluto de cada elemento de la imagen de entrada
- ImgAbsAd:** Images absolute add/Suma absoluta de imágenes. Se suman dos matrices A y B, resultando en la matriz C, donde cada elemento  $C_{ij}$  es la suma de los valores absolutos de los elementos  $A_{ij}$  y  $B_{ij}$ .
- ImgAbsDiff:** Images absolute difference/Diferencia absoluta de imágenes. Se resta la matriz B a la matriz A, resultando en la matriz C, de manera que cada elemento  $C_{ij}$  es el absoluto de la resta del elemento  $B_{ij}$  al elemento  $A_{ij}$ .
- ImgAdd:** Images add/Suma de imágenes. Se suman las matrices A y B, resultando en la matriz C, de manera que cada elemento  $C_{ij}$  es la suma de los elementos  $A_{ij}$  y  $B_{ij}$ .
- ImgCeil:** Image ceil/Techo de imagen. Se hace una operación de techo para cada uno de los elementos de la matriz de entrada.
- ImgCompl:** Image complement/Complemento de imagen. Se obtiene el complemento de la imagen de entrada.
- ImgDiv:** Images division/División de imágenes. Divide la matriz A por B, resultando en la matriz C, de manera que cada elemento  $C_{ij}$  es la división del elemento  $A_{ij}$  por  $B_{ij}$ .
- ImgFloor:** Image floor/Piso de imagen. Se hace una operación de piso para cada uno de los elementos de la matriz de entrada.
- ImgMult:** Images multiplication/Multiplicación de imágenes. Se calcula el producto de Hadamard de las matrices de entrada A y B. Cada elemento  $C_{ij}$  es la multiplicación del elemento  $A_{ij}$  por  $B_{ij}$ .
- ImgSubs:** Images Subtraction/Sustracción de imágenes. Realiza la misma función que **ImgAbsDiff**.
- Log:** Calcula el logaritmo natural  $\ln(x)$  de cada elemento de la matriz de entrada.
- PrmtShp:** Perimeter shape/Forma del perímetro. Esta función morfológica obtiene la forma de las orillas que componen el perímetro de los objetos.
- Open:** Apertura. Es una función morfológica. Sea A una imagen binaria, y B el elemento estructurante, la apertura de A por B se obtiene por la erosión de A por B, seguida por la dilatación de la imagen resultante por B.

- ScIDiv**: Scalar division/División por escalar. Cada elemento de la matriz entrante es multiplicado por un número  $k$ .
- ScIMult**: Scalar multiplication/Multiplicación por escalar. Cada elemento de la matriz entrante es multiplicado por un número  $1/k$ .
- ScIPwr**: Scalar power/Potencia por escalar. Cada elemento de la matriz se eleva a la potencia del número  $1/k$ .
- ScIRoot**: Scalar root/Raíz escalar. Cada elemento de la matriz se eleva a la potencia del número  $k$ .
- ScISum**: Scalar sum/Suma escalar. A cada elemento de la matriz se le suma el número  $1/k$ .
- ScISubs**: Scalar subtraction/Sustracción escalar. A cada elemento de la matriz se le resta el número  $1/k$ .
- SkItShp**: Skeleton shape/Función de esqueleto. Función morfológica para obtener una imagen del esqueleto de los objetos.
- Sqr**: Square/Cuadrado. Cada elemento de la matriz de entrada es elevado al cuadrado.
- Sqrt**: Square root/Raíz cuadrada. Se obtiene la raíz cuadrada absoluta de cada elemento de la matriz entrante.
- Supremum**: Supremo. Recibe dos matrices A y B, y retorna los valores de A que hayan sido mayores que los valores de B en cada fila.
- Thld**: Threshold/Umbral. Calcula el valor del umbral que puede ser usado para convertir una imagen de intensidad a una imagen binaria. Este valor queda entre 0 y 1. El cálculo usa el método de Otsu, el cual escoge un umbral para minimizar la varianza de intraclasses de los píxeles negros y blancos.
- TopHat**: Top hat/Transformación de sombrero de copa blanco. Función morfológica que devuelve una imagen que contiene los objetos o elementos que son más brillantes que sus alrededores y que son más pequeños que el elemento estructurante. Se define como la diferencia entre la apertura y la imagen de entrada.

## Terminales

- B**: Blue/Azul. Matriz que contiene los valores del componente de color azul de la imagen RGB.
- BY-Op**: Matriz que contiene los valores de la oponencia de color entre los componentes azul y amarillo.
- C**: Cyan/Cian. Matriz que contiene los valores del componente de color cian de la imagen CMYK.
- Col**: Mapa visual de la dimensión de color.
- G**: Green/Verde. Matriz que contiene los valores del componente de color verde de la imagen RGB.
- H**: Hue/Matiz. Matriz que contiene los valores del componente de color de matiz de la imagen HSV.
- Image**: Todos los componentes de color de la imagen.



**Int:** Mapa visual de la dimensión de intensidad.

**K:** Key/Negro. Matriz que contiene los valores del componente de color negro de la imagen CMYK.

**M:** Magenta. Matriz que contiene los valores del componente de color magenta de la imagen CMYK

**Ori:** Mapa visual de la dimensión de orientación.

**R:** Red/Rojo. Matriz que contiene los valores del componente de color rojo de la imagen RGB.

**RG-Op:** Matriz que contiene los valores de la oponencia de color entre los componentes rojo y verde.

**S:** Saturation/Saturación. Matriz que contiene los valores del componente de color de saturación de la imagen HSV.

**Shp:** Mapa visual de la dimensión de forma.

**V:** Value/Valor. Matriz que contiene los valores del componente de color de valor de la imagen HSV.

**Y:** Yellow/Amarillo. Matriz que contiene los valores del componente de color amarillo de la imagen CMYK.