



**Diseño e implementación de una celda colaborativa robotizada mediante robots móviles y  
humanoides para clasificación de objetos**

Escobar Gómez, Diego Andrés

Departamento de Eléctrica, Electrónica y Telecomunicaciones

Carrera de Ingeniería en Electrónica, Automatización y Control

Trabajo de titulación, previo a la obtención del título de Ingeniero en Electrónica, Automatización y  
Control

Ing. Ibarra Jácome, Oswaldo Alexander Mgs.

4 de septiembre 2020



## Urkund Analysis Result

**Analysed Document:** Tesis ESCOBAR DIEGO.pdf (D78629408)  
**Submitted:** 9/4/2020 9:50:00 PM  
**Submitted By:** oaibarra@espe.edu.ec  
**Significance:** 2 %

### Sources included in the report:

Tesis\_Camino\_Rojas\_V\_12\_V1.pdf (D60821952)  
UrkundRSSI-BonillaGaby.pdf (D46196342)  
Tesis\_Camino\_Rojas.pdf (D59610945)  
Tesis Katherine Enriquez.pdf (D54322709)  
<https://rdu.unc.edu.ar/bitstream/handle/11086/12888/Informe.pdf?sequence=1&isAllowed=y>

### Instances where selected sources appear:

15

Firma

A handwritten signature in blue ink, enclosed within a blue oval. The signature is stylized and appears to be "Oswaldo Alexander Ibarra Jácome".

Ing. Oswaldo Alexander Ibarra Jácome MGs.  
Cc. 1719535427



DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES  
CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

#### CERTIFICACIÓN

Certifico que el trabajo de titulación, "Diseño e implementación de una celda colaborativa robotizada mediante robots móviles y humanoides para clasificación de objetos" fue realizado por el señor **Escobar Gómez, Diego Andrés** el cual ha sido revisado y analizado en su totalidad por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos legales, teóricos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 4 de septiembre del 2020

Firma:

Ing. Ibarra Jácome, Oswaldo Alexander MGs.

C.C.: 1719535427



DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES  
CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

#### RESPONSABILIDAD DE AUTORÍA

Yo, **Escobar Gómez, Diego Andrés**, con cédula de ciudadanía n° 1719524611, declaro que el contenido, ideas y criterios del trabajo de titulación: **Diseño e implementación de una celda colaborativa robotizada mediante robots móviles y humanoides para clasificación de objetos** es de mi autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Sangolquí, 4 de septiembre del 2020

Firma:

**Escobar Gómez, Diego Andrés**

C.C.: 1719524611



DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES  
CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

#### AUTORIZACIÓN DE PUBLICACIÓN

Yo, **Escobar Gómez, Diego Andrés**, con cédula de ciudadanía n° 1719524611, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **Diseño e implementación de una celda colaborativa robotizada mediante robots móviles y humanoides para clasificación de objetos** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, 4 de septiembre del 2020

Firma:

**Escobar Gómez, Diego Andrés**

C.C.: 1719524611

### **Dedicatoria**

Este trabajo de titulación se lo dedico especialmente a Ian con mucho amor, y que sepa que con esfuerzo y dedicación todo es posible aun pese a las adversidades que puede darse en la vida al final siempre existirá una meta por alcanzar.

## **Agradecimiento**

Agradezco bastante a Juan y Silvia mis padres y a Mario y Mary mis suegros por el apoyo brindado durante estos años, sin su ayuda no hubiese podido culminar mis estudios.

Agradezco a Karen mi compañera de vida y a Ian mi pequeño hijo por el apoyo que me han sabido brindar para nunca rendirme y seguir adelante, también me han sabido comprender por el tiempo que dedique a mis estudios y no pasar con ellos.

Agradezco a Alejandro y a Inés mis abuelos que desde pequeño los considero como mis segundos padres y siempre han cuidado de mí, me alegra mucho aun tener la dicha de tenerlos y que me vean cumplir un sueño más.

Tampoco puedo olvidar de agradecer a mis compañeros e ingenieros con los que tuve la oportunidad de trabajar, en el que espero que mis aportaciones les haya sido válidas.

Agradezco a mi tutor de titulación por la comprensión, los consejos y el apoyo que me dio durante la realización del trabajo de titulación.

## Índice de Contenidos

<b>Resumen</b> .....	21
<b>Abstract</b> .....	22
<b>Capítulo I Generalidades</b> .....	23
<b>Antecedentes</b> .....	23
<b>Justificación e Importancia</b> .....	27
<b>Alcance</b> .....	28
<b>Objetivos</b> .....	34
<i>Objetivo General</i> .....	34
<i>Objetivos Específicos</i> .....	34
<b>Capítulo II Fundamentación teórica</b> .....	35
<b>Introducción</b> .....	35
<b>Revolución Industrial 4.0</b> .....	37
<b>Robótica Colaborativa</b> .....	38
<i>Sistemas Colaborativos</i> .....	38
<i>Arquitecturas de control</i> .....	40
<b>Robots Móviles</b> .....	42
<i>Robots Móviles de Ruedas</i> .....	43
<b>Robots Humanoides</b> .....	45
<b>Visión por Computador</b> .....	47
<i>Sistema de captura de imágenes</i> .....	47
<i>Procesamiento digital de imágenes</i> .....	50
<b>Capítulo III Desarrollo del sistema clasificatorio por visión</b> .....	56



<b>Descripción del hardware del sistema clasificatorio</b> .....	56
<i>Limitaciones encontradas</i> .....	56
<i>Computador</i> .....	57
<i>Cámara</i> .....	58
<i>Kit Bioid Premium</i> .....	59
<i>Controlador CM-510</i> .....	62
<i>Tarjeta de desarrollo</i> .....	64
<i>Tarjeta de desarrollo para visión</i> .....	66
<i>Actuadores</i> .....	68
<i>Sensores</i> .....	72
<i>Módulos de comunicación</i> .....	75
<i>Baterías</i> .....	79
<i>Robot Humanoide</i> .....	82
<i>Robot Móvil</i> .....	85
<i>Celda de trabajo</i> .....	89
<i>Objetos de color</i> .....	90
<b>Descripción del software del sistema clasificatorio</b> .....	91
<i>Softwares de sistema</i> .....	92
<i>Lenguajes de programación</i> .....	92
<i>Softwares de programación</i> .....	92
<i>Protocolos de comunicación</i> .....	95
<i>Algoritmos del Sistema Multi-Agente</i> .....	103
<b>Capítulo IV Desarrollo del algoritmo clasificatorio</b> .....	115

	10
<b>Interfaz gráfica</b> .....	115
<i>Arquitectura y Navegación</i> .....	116
<i>Distribución</i> .....	116
<i>Color y Texto</i> .....	117
<i>Comandos e Ingreso de Datos</i> .....	117
<i>Pantallas de la interfaz del proyecto</i> .....	118
<b>Visión de máquina</b> .....	120
<b>Ubicación en el espacio de trabajo</b> .....	124
<b>Trabajo colaborativo</b> .....	135
<b>Comunicación inalámbrica</b> .....	143
<b>Capítulo V Pruebas y resultados</b> .....	145
<b>Pruebas</b> .....	145
<i>Prueba 1: Análisis de la iluminación dentro de la celda</i> .....	145
<i>Prueba 2: Error entre el área calculada y el área real</i> .....	146
<i>Prueba 3: Error entre la distancia calculada y la distancia real</i> .....	148
<i>Prueba 4: Análisis del trabajo colaborativo</i> .....	151
<b>Análisis y discusión de resultados</b> .....	157
<i>Prueba 1</i> .....	157
<i>Prueba 2</i> .....	174
<i>Prueba 3</i> .....	182
<i>Prueba 4</i> .....	191
<b>Capítulo VI Conclusiones y Recomendaciones</b> .....	199
<b>Conclusiones</b> .....	199

	11
<b>Recomendaciones .....</b>	<b>201</b>
<b>Trabajos futuros .....</b>	<b>202</b>
<b>Referencias Bibliográficas .....</b>	<b>204</b>
<b>Anexos .....</b>	<b>211</b>

## Índice de Tablas

<b>Tabla 1</b> Tareas de los humanos en las revoluciones industriales .....	35
<b>Tabla 2</b> Sistemas de locomoción en robots móviles de ruedas.....	44
<b>Tabla 3</b> Características de cada uno de los tipos de iluminación artificial. ....	48
<b>Tabla 4</b> Técnicas de iluminación .....	49
<b>Tabla 5</b> Análisis comparativo técnico para determinar el mejor tipo de cámara .....	58
<b>Tabla 6</b> Contenido del kit Bioloid Premium .....	60
<b>Tabla 7</b> Análisis comparativo técnico para determinar la mejor tarjeta de desarrollo.....	65
<b>Tabla 8</b> Análisis comparativo técnico para determinar la mejor tarjeta de desarrollo para visión artificial.....	66
<b>Tabla 9</b> Especificaciones DYNAMIXEL AX-12A .....	69
<b>Tabla 10</b> Especificaciones técnicas del CI 74LS241 .....	71
<b>Tabla 11</b> Análisis comparativo técnico para la selección del mejor sensor de color .....	74
<b>Tabla 12</b> Análisis de carga para la selección de la batería del robot humanoide.....	79
<b>Tabla 13</b> Análisis de carga para la selección de la batería para el robot móvil .....	80
<b>Tabla 14</b> Análisis comparativo técnico para determinar el mejor modelo de baterías LiPo.....	80
<b>Tabla 15</b> Tabla de control del área EEPROM del DYNAMIXEL AX-12A .....	96
<b>Tabla 16</b> Tabla de control del área RAM del DYNAMIXEL AX-12A .....	97
<b>Tabla 17</b> Tipos de instrucciones.....	99
<b>Tabla 18</b> Bits del estado de error.....	100
<b>Tabla 19</b> Modos de recuperación y métodos de aproximación de contorno de la función cv.findContours().....	123
<b>Tabla 20</b> Direcciones IP y puertos utilizados .....	144

<b>Tabla 21</b> Cantidad de objetos detectados con la máscara color amarillo .....	159
<b>Tabla 22</b> Cantidad de objetos detectados con la máscara color azul.....	162
<b>Tabla 23</b> Cantidad de objetos detectados con la máscara color naranja .....	165
<b>Tabla 24</b> Cantidad de objetos detectados con la máscara color rojo.....	169
<b>Tabla 25</b> Cantidad de objetos detectados con la máscara color verde.....	172
<b>Tabla 26</b> Error porcentual del cálculo del área de la marca del robot humanoide .....	177
<b>Tabla 27</b> Resultados obtenidos para la orientación del robot humanoide .....	177
<b>Tabla 28</b> Error porcentual del cálculo del área de la marca del robot móvil .....	180
<b>Tabla 29</b> Resultados obtenidos para la orientación del robot móvil.....	181
<b>Tabla 30</b> Asignación de objetivos en el robot humanoide .....	186
<b>Tabla 31</b> Error porcentual de la distancia calculada con el robot humanoide.....	186
<b>Tabla 32</b> Asignación de objetivos en el robot móvil.....	190
<b>Tabla 33</b> Error porcentual de la distancia calculada con el robot móvil .....	191
<b>Tabla 34</b> Resultados obtenidos en la repetición 1 de la clasificación de objetos de la prueba 4	192
<b>Tabla 35</b> Resultados obtenidos en la repetición 2 de la clasificación de objetos de la prueba 4	193
<b>Tabla 36</b> Resultados obtenidos en la repetición 3 de la clasificación de objetos de la prueba 4	193
<b>Tabla 37</b> Resultados obtenidos en el trabajo colaborativo de la prueba 4 .....	197

## Índice de Figuras

<b>Figura 1</b> Robots a la espera de la orden de inicio .....	29
<b>Figura 2</b> Búsqueda de los objetos de interés dentro del área de trabajo .....	30
<b>Figura 3</b> Detección de objetos de interés .....	30
<b>Figura 4</b> Traslado de los robots hacia los objetos de interés.....	31
<b>Figura 5</b> Agarre de los objetos de interés.....	31
<b>Figura 6</b> Objetos superpuestos dentro del área de trabajo .....	32
<b>Figura 7</b> Robot móvil con varios objetos a su alrededor .....	33
<b>Figura 8</b> Fin del proceso .....	34
<b>Figura 9</b> Robots homogéneos.....	39
<b>Figura 10</b> Diferentes tipos de jerarquía de coordinación (a)Distribuida (b)Centralizado (c) Descentralizado .....	42
<b>Figura 11</b> Tipos de locomoción en un robot móvil (a) Ruedas (b) Patas (c) Orugas.....	43
<b>Figura 12</b> Elektro. El primer robot de la historia .....	46
<b>Figura 13</b> Robots humanoides (de izquierda a derecha): Pepper, NAO, Sophia, Atlas .....	47
<b>Figura 14</b> Representación y movimiento de un kernel al recorrer toda la imagen.....	51
<b>Figura 15</b> Representación en forma de cubo del modelo de color RGB.....	53
<b>Figura 16</b> Representación en forma de cono invertido del modelo de color HSV .....	54
<b>Figura 17</b> Cámara Web Klip Xtreme Xcam 300.....	59
<b>Figura 18</b> Kit Bioloid Premium .....	59
<b>Figura 19</b> Robot humanoide tipo-A .....	61
<b>Figura 20</b> Robot de sondeo.....	61
<b>Figura 21</b> Controlador CM-510 .....	62

<b>Figura 22</b> Conexión en cadena servomotores AX-12A con CM-510 .....	63
<b>Figura 23</b> Distribución de pines del puerto externo CM-510 .....	63
<b>Figura 24</b> Composición de los pines del puerto de comunicación .....	64
<b>Figura 25</b> Placa de desarrollo Teensy 3.6 .....	66
<b>Figura 26</b> Raspberry Pi Zero W .....	67
<b>Figura 27</b> Diagrama electrónico entre RPZW y CM-510 .....	68
<b>Figura 28</b> Servomotor DYNAMIXEL AX-12A .....	69
<b>Figura 29</b> Circuito interno half dúplex UART del controlador CM-510.....	70
<b>Figura 30</b> Diagrama electrónico DYNAMIXEL con otros controladores.....	71
<b>Figura 31</b> Composición de los pines del GS-12 .....	72
<b>Figura 32</b> Raspberry Pi Camera Module V2 .....	73
<b>Figura 33</b> Carcasa RPZW para usarla con Raspberry Pi Camera Module V2 .....	73
<b>Figura 34</b> Sensor de color TCS34725 .....	74
<b>Figura 35</b> Módulos de comunicación ZIG-100 (izq) y ZIG-110A (der) .....	75
<b>Figura 36</b> Distribución de pines ZIG-100 (izq) y ZIG-110A (der) .....	76
<b>Figura 37</b> (a) Módulo ZIG2Serial (arriba), (b) Conector RS232 hembra del módulo ZIG2Serial (abajo) .....	77
<b>Figura 38</b> Diagrama electrónico de conexión ZIG2Serial – USB a Serial.....	78
<b>Figura 39</b> Diseño placa PCB.....	78
<b>Figura 40</b> Implementación final conexión ZIG2Serial – USB a Serial .....	79
<b>Figura 41</b> Baterías LiPo de 11.1V (a) 2200mAh, (b) 1000mAh.....	81
<b>Figura 42</b> Modelo 3D soporte RPZW .....	83
<b>Figura 43</b> Dimensiones del soporte diseñado en AutoCAD para la RPZW .....	84

<b>Figura 44</b> Robot humanoide incorporado con: (1) 20 actuadores AX-12A. (2) 1 sensor GS-12. (3) 1 convertidor de nivel lógico TTL bidireccional. (4) 1 controlador CM-510. (5) 1 tarjeta de desarrollo RPZW. (6) 1 Raspberry Pi Camera Module V2 (7) 1 batería LiPo 11.1V 2200mAh. También se usó las carcasas diseñadas y piezas del kit.....	85
<b>Figura 45</b> Diagrama electrónico del robot móvil .....	86
<b>Figura 46</b> Dimensiones de la estructura del robot móvil .....	87
<b>Figura 47</b> Robot móvil incorporado con: (1) 6 actuadores AX-12A. (2) ZIG-110A. (3) Sensor TCS34725. (4) Batería LiPo 11.1V 1000mAh. (5) Teensy 3.6. (6) Convertidor DC-DC MP1584EN. (7) CI 74LS241. (8) 6 molex de 3-pins. (9) Fusible de 1A. (10) Portafusible. (11) Interruptor. (12) Led. (13) 2 resistencias.....	88
<b>Figura 48</b> Celda de trabajo.....	89
<b>Figura 49</b> Objetos de varios colores .....	90
<b>Figura 50</b> Arquitectura de coordinación centralizada de la celda colaborativa robotizada.....	91
<b>Figura 51</b> Programa R+ Task 3.0 .....	93
<b>Figura 52</b> Estructura de datos (Programación de movimientos).....	94
<b>Figura 53</b> Paquete de instrucciones.....	98
<b>Figura 54</b> Paquete de estado .....	99
<b>Figura 55</b> Paquete de comunicación ZIG-100/110A .....	101
<b>Figura 56</b> Procesos ejecutados en forma paralela comunicación del MAS .....	104
<b>Figura 57</b> Patrones de caminata del robot humanoide.....	105
<b>Figura 58</b> Diagrama de flujo del código embebido del agente 1.....	106
<b>Figura 59</b> Información válida para ejecutar posibles movimientos agente 1.....	107
<b>Figura 60</b> Diagrama de flujo programación por hilos del agente 1 .....	107



<b>Figura 61</b> Diagrama de flujo del código embebido del agente 2.....	108
<b>Figura 62</b> Estructura para adquisición de datos del código embebido .....	109
<b>Figura 63</b> Estructura para la validación de datos del código embebido.....	110
<b>Figura 64</b> Estructura para ejecutar ordenes en el robot móvil.....	110
<b>Figura 65</b> Estructura para ejecutar el censado de validación de color con el robot móvil .....	112
<b>Figura 66</b> Estructura del primer hilo de programación del agente 3 .....	113
<b>Figura 67</b> Estructura del segundo hilo de programación del agente 3.....	114
<b>Figura 68</b> Estructura del tercer hilo de programación del agente 3.....	114
<b>Figura 69</b> Procesos del algoritmo de organización .....	115
<b>Figura 70</b> Distribución de la pantalla Título .....	116
<b>Figura 71</b> Distribución de la pantalla principal .....	117
<b>Figura 72</b> Tipos de entrada de comandos.....	118
<b>Figura 73</b> Pantalla de título del proyecto .....	119
<b>Figura 74</b> Pantalla principal del proyecto .....	119
<b>Figura 75</b> Diagrama de flujo para componente de visión.....	121
<b>Figura 76</b> Estructura para almacenar y clasificar polígonos de acuerdo a su forma y color .....	122
<b>Figura 77</b> Diagrama de flujo para componente de ubicación.....	125
<b>Figura 78</b> Diferentes ubicaciones del objeto a recolectar .....	128
<b>Figura 79</b> Estructura para determinar la orientación de los robots .....	130
<b>Figura 80</b> Detección de coordenadas del triángulo en diferentes rotaciones .....	131
<b>Figura 81</b> Orden considerado de las coordenadas .....	132
<b>Figura 82</b> Ejemplos de la representación de posición y orientación .....	134
<b>Figura 83</b> Diagrama de flujo del componente de trabajo colaborativo .....	135

<b>Figura 84</b> Estructura para la asignación de objetivos .....	136
<b>Figura 85</b> Estructura para determinar cantidad de obstáculos y de objetos de interés .....	137
<b>Figura 86</b> Estructura que permite detectar y evitar colisiones .....	138
<b>Figura 87</b> Estructura para determinar el tipo de movimiento que tiene que ejecutar cada robot.....	141
<b>Figura 88</b> Estructura que permite la realización del trabajo colaborativo .....	143
<b>Figura 89</b> Prueba 1 con una intensidad de iluminación baja (a), intensidad de iluminación media (b) y con una intensidad de iluminación alta (c). .....	146
<b>Figura 90</b> Prueba 2 con el marcador robot humanoide rotado en los cuatro cuadrantes.....	147
<b>Figura 91</b> Prueba 2 con el marcador robot móvil rotado en los cuatro cuadrantes .....	148
<b>Figura 92</b> Prueba 3 con los robots y un objeto.....	149
<b>Figura 93</b> Prueba 3 con los robots y dos objetos.....	150
<b>Figura 94</b> Prueba 3 con los robots y tres objetos .....	150
<b>Figura 95</b> Prueba 3 con los robots y cuatro objetos.....	151
<b>Figura 96</b> Prueba 3 con los robots y cinco objetos .....	151
<b>Figura 97</b> Prueba 4 selección de objetos amarillos. ....	152
<b>Figura 98</b> Prueba 4 selección de objetos azules .....	152
<b>Figura 99</b> Prueba 4 selección de objetos amarillos y azules.....	153
<b>Figura 100</b> Prueba 4 de petición de ayuda colaborativa por objetos superpuestos.....	154
<b>Figura 101</b> Prueba 4 de petición de ayuda colaborativa por colisión del robot móvil con algún objeto .....	155
<b>Figura 102</b> Resultado obtenido para máscara color amarillo con intensidad de iluminación baja.....	157

<b>Figura 103</b> Resultado obtenido para máscara color amarillo con intensidad de iluminación media.....	158
<b>Figura 104</b> Resultado obtenido para máscara color amarillo con intensidad de iluminación alta .....	158
<b>Figura 105</b> Resultado obtenido para máscara color azul con intensidad de iluminación baja ...	160
<b>Figura 106</b> Resultado obtenido para máscara color azul con intensidad de iluminación media	161
<b>Figura 107</b> Resultado obtenido para máscara color azul con intensidad de iluminación alta ....	162
<b>Figura 108</b> Resultado obtenido para máscara color naranja con intensidad de iluminación baja .....	163
<b>Figura 109</b> Resultado obtenido para máscara color naranja con intensidad de iluminación media.....	164
<b>Figura 110</b> Resultado obtenido para máscara color naranja con intensidad de iluminación alta .....	165
<b>Figura 111</b> Resultado obtenido para máscara color rojo con intensidad de iluminación baja ...	167
<b>Figura 112</b> Resultado obtenido para máscara color rojo con intensidad de iluminación media	168
<b>Figura 113</b> Resultado obtenido para máscara color rojo con intensidad de iluminación alta ....	168
<b>Figura 114</b> Resultado obtenido para máscara color verde con intensidad de iluminación baja	170
<b>Figura 115</b> Resultado obtenido para máscara color verde con intensidad de iluminación media.....	171
<b>Figura 116</b> Resultado obtenido para máscara color verde con intensidad de iluminación alta .	172
<b>Figura 117</b> Resultados obtenidos para el robot humanoide en el cuadrante I .....	175
<b>Figura 118</b> Resultados obtenidos para el robot humanoide en el cuadrante II .....	175
<b>Figura 119</b> Resultados obtenidos para el robot humanoide en el cuadrante III .....	176

<b>Figura 120</b> Resultados obtenidos para el robot humanoide en el cuadrante IV.....	176
<b>Figura 121</b> Resultados obtenidos para el robot móvil en el cuadrante I.....	178
<b>Figura 122</b> Resultados obtenidos para el robot móvil en el cuadrante II.....	179
<b>Figura 123</b> Resultados obtenidos para el robot móvil en el cuadrante III.....	179
<b>Figura 124</b> Resultados obtenidos para el robot móvil en el cuadrante IV .....	180
<b>Figura 125</b> Resultados obtenidos para el robot humanoide con un objeto.....	182
<b>Figura 126</b> Resultados obtenidos para el robot humanoide con dos objetos.....	183
<b>Figura 127</b> Resultados obtenidos para el robot humanoide con tres objetos .....	183
<b>Figura 128</b> Resultados obtenidos para el robot humanoide con cuatro objetos .....	184
<b>Figura 129</b> Resultados obtenidos para el robot humanoide con cinco objetos .....	184
<b>Figura 130</b> Resultados obtenidos de la asignación del objeto más cercano al robot humanoide .....	185
<b>Figura 131</b> Resultados obtenidos para el robot móvil con un objeto .....	187
<b>Figura 132</b> Resultados obtenidos para el robot móvil con dos objetos .....	187
<b>Figura 133</b> Resultados obtenidos para el robot móvil con tres objetos.....	188
<b>Figura 134</b> Resultados obtenidos para el robot móvil con cuatro objetos .....	188
<b>Figura 135</b> Resultados obtenidos para el robot móvil con cinco objetos .....	189
<b>Figura 136</b> Resultados obtenidos de la asignación del objeto más cercano al robot móvil.....	190

## Resumen

En décadas pasadas se utilizaban robots gruesos y rígidos, actualmente las personas se están moviendo hacia el uso de robots colaborativos. En el presente documento se presenta el desarrollo de una celda colaborativa robotizada utilizando robots humanoides y móviles para la clasificación de objetos de acuerdo al color, el proyecto se divide en 5 etapas: la arquitectura de control, en donde se realiza el diseño de un sistema multi-agente, que permitirá la comunicación entre todos los componentes de la celda colaborativa; la detección de objetos, en donde mediante técnicas de procesamiento de imágenes se busca los objetos de interés de acuerdo a su color, para lo cual se usa una cámara dentro de la celda, cubriendo un área de 120cm x 90cm; la asignación de objetivos, en la que se determina el objeto de interés más cercano a cada robot y este proceda a desplazarse a su ubicación, empleando algoritmos matemáticos que permiten la realización de los cálculos necesarios; la manipulación, encargada de la recolección y traslado de los objetos de interés en cada uno de los robots y finalmente la etapa colaborativa, la cual se encuentra validando constantemente peticiones de ayuda, ya sea por colisión de los robots con algún objeto o por no poder acceder al objeto deseado. El desarrollo del proyecto se lo realizó con lenguajes de programación como Python y C++, utilizando los softwares de Python IDLE, Arduino, Thonny y R+ Task 3.0.

Palabras clave:

- **ROBÓTICA COLABORATIVA**
- **BIOLOID**
- **DETECCION DE OBJETOS**
- **SISTEMA MULTI AGENTE**

### **Abstract**

In past decades thick and rigid robots were used, today people are moving towards the use of collaborative robots. This document presents the development of a collaborative robotic cell using humanoid and mobile robots for the classification of objects according to color, the project is divided into 5 stages: the control architecture, where the design of a multi-agent system is carried out, which allows communication between all the components of the collaborative cell; the detection of objects, where through image processing techniques objects of interest are searched according to their color, for which a camera is used inside the cell, covering an area of 1.25cm x 94cm; the assignment of objectives, in which the object of interest closest to each robot is determined and it proceeds to move to its location, using mathematical algorithms that allow the necessary calculations to be carried out; the manipulation, in charge of the collection and transfer of the objects of interest in each of the robots and finally the collaborative stage, which is constantly validating the requests for help, either due to collision of the robots with an object or due to not being able to access to the desired object. The development of the project was carried out with programming languages such as Python and C ++, using the Python IDLE, Arduino, Thonny and R + Task 3.0 softwares.

Keywords:

- **COLLABORATIVE ROBOTICS**
- **BIOLOID**
- **OBJECT DETECTION**
- **MULTI AGENT SYSTEM**

## Capítulo I Generalidades

### Antecedentes

Las revoluciones industriales se han dado por la convergencia de las nuevas tecnologías aplicadas en antiguos sistemas productivos, a través de la historia. La Industria 4.0 fue un término propuesto por el Gobierno alemán para describir la digitalización de sistemas y procesos industriales, y su interconexión mediante IoT y así llegar a una nueva visión de la fábrica del futuro o fábrica inteligente. (Joyanes, 2017)

Pelegri (2019) menciona que los primeros robots colaborativos de Universal Robots se instalaron en la industria hace unos diez años, pero cada vez son más los robots instalados en las fábricas, y además se menciona que los robots colaborativos son la pieza clave para el desarrollo de la Industria 4.0. El avance de la tecnología ha permitido que los robots cumplan con todos los estándares de calidad necesarios para la aplicación de la robótica colaborativa en la alimentación y en la agricultura, revolucionando la forma de trabajar.

Uno de los casos internacionales donde se ha incluido robots colaborativos es en la multinacional Continental Automotive Spain, es el fabricante español más robotizado en el negocio y el primero en liderar el movimiento hacia la industria 4.0. En el 2016, la empresa hizo la adquisición de algunos cobots de UR10 para automatizar la fabricación y manipulación de placas PCB, y redujo los tiempos de cambio en un 50%, comparada con realización de la tarea de forma manual. (Pelegri, 2019)

En Ecuador, existen empresas que usan robots dentro de sus procesos de fabricación, una de ellas es la planta de General Motors en Quito que tiene sistemas de software y big data

para manejar la información de todos los procesos de ensamblaje de autos, la otra es la empresa Mexichem Plastigama posee un robot que regula y controla el caudal del agua y la temperatura para la fabricación de tuberías. (EL UNIVERSO, 2018)

Brenes (2016) presentó el diseño, implementación y prueba de una interfaz de programación de aplicaciones para la locomoción de robots humanoides bioloid con el uso de un sistema integrado BeagleBone Black. Donde la principal contribución de este trabajo fue el desarrollo de una novedosa integración de tecnología para controlar los robots bioloid. El funcionamiento general del sistema lo validaron en la competición de robots humanoides CEABOT, en la prueba de subir y bajar escalones. Los resultados obtenidos en la locomoción del robot Bioloid utilizando un sistema empotrado BeagleBone Black, fueron exitosos.

Los robots humanoides más avanzados en la actualidad corresponden a esfuerzos de empresas u organizaciones gubernamentales que los utilizan para futuros desarrollos comerciales y hasta para exhibiciones de mercadotecnia. (Brenes, 2016)

El trabajo de Paladines (2015) describe el modelo cinemático directo e inverso para las piernas del robot humanoide Bioloid Premium tipo A. La importancia de este trabajo es la obtención del modelo cinemático el cual serviría como base para la obtención del modelo dinámico del robot para generar nuevas aplicaciones y estrategias de control. Los resultados obtenidos en este trabajo no se pudieron validar debido a que no poseen el robot físicamente.

Aguirre & Moyano (2019) mencionan que en la Universidad de las Fuerzas Armadas han existido pocos trabajos anteriores sobre robótica colaborativa.

En la Universidad de las Fuerzas Armadas – ESPE, en el año 2014 fue presentado el trabajo de titulación, previo a la obtención del título de Ingeniero en Mecatrónica del



Departamento de Ciencias de la Energía y Mecánica con el Tema: “Desarrollo de un sistema de robótica colectiva con procesamiento centralizado entre dos robots humanoides Bioloid Premium” por parte del alumno: Tabango Obaco Ricardo Daniel, bajo la dirección del Ing. Chacón Alejandro.

En este trabajo se utilizaron dos robots humanoides Bioloid Premium para que realicen un trabajo colectivo por medio de una unidad de procesamiento central que se basa en la visión artificial mediante el uso del programa LabVIEW y una cámara web de alta definición Logitech C310 y datos obtenidos de los sensores incorporados en los robots. Además, se dio solución a la cinemática del robot humanoide de 18 grados de libertad. Se utilizó un control retroalimentado por medio de visión artificial, el cual permitió detectar la pelota, así como determinar la posición y la orientación de los robots mediante el procesamiento digital, para que los dos robots se pasen una pelota de un lado a otro en un área de trabajo. (Tabango, 2014)

Una conclusión del trabajo de titulación de Tabango (2014) es:

A través de la visión artificial se logró determinar un sistema de distancias y ubicación que satisfizo las necesidades del proyecto. Este sistema determinó la posición de los robots y la pelota además de la distancia entre estos.

Mientras que una recomendación del trabajo de titulación de Tabango (2014) es:

Para un mejor control es recomendable usar robots humanoides con cámaras montadas en cada robot, de esta manera se puede analizar mejor el entorno y el objetivo.

En el año 2018 fue presentado el trabajo de titulación, previo a la obtención del título de Ingeniero en Mecatrónica del Departamento de Ciencias de la Energía y Mecánica con el

Tema: “Diseño e implementación de una arquitectura IoT para robótica colaborativa” por parte de los alumnos: González David y Verdugo Alexandra, bajo la dirección del Ing. Escobar Luis.

Este trabajo consistió en el diseño e implementación de una arquitectura IoT para robótica colaborativa y tuvo tres componentes principales, el primero fue un sistema multi-robot compuesto por tres robots móviles controlados por la tarjeta NI MyRIO, el segundo la arquitectura IoT basada en el protocolo MQTT y el tercero el planificador automático encargado de la recepción de datos de la interfaz de usuario y el control colaborativo centralizado. El sistema consistió en la clasificación de piezas en función de su color. Las pruebas fueron enfocadas en el tiempo de respuesta, el costo computacional, el sistema multi-robot y el planificador automático. (González & Verdugo, 2018)

Una conclusión del trabajo de titulación de González & Verdugo (2018) fue:

Se desarrolló un planificador automático clásico que permitió la canalización de información del usuario hacia los robots, la detección de las unidades disponibles en el sistema y el control colaborativo centralizado.

Mientras que una recomendación del trabajo de titulación de González & Verdugo (2018) fue:

El uso de un sistema híbrido de odometría y navegación inercial para el posicionamiento de robots móviles se recomienda cuando la estimación en la posición no sea crítica, especialmente para desarrollos de bajo costo y sobre plataforma con escasa capacidad de procesamiento.

## **Justificación e Importancia**

En noviembre del 2017 El Gobierno Ecuatoriano estableció el Plan Nacional de Desarrollo 2017-2021, donde se pretende garantizar el acceso progresivo de las personas a sus derechos durante todo el ciclo de vida, a través de políticas públicas y programas para el desarrollo social y humano. Este documento, además, orienta las acciones del Gobierno actual para los próximos cuatro años.

El Plan Nacional de Desarrollo para el periodo 2017-2021 se organiza en tres Ejes Programáticos y nueve Objetivos Nacionales de Desarrollo. El objetivo 5: Impulsar la productividad y competitividad para el crecimiento económico sostenible de manera redistributiva y solidaria, en este se menciona “Promover la investigación, la formación, la capacitación, el desarrollo y la transferencia tecnológica, la innovación y el emprendimiento, la protección de la propiedad intelectual, para impulsar el cambio de la matriz productiva mediante la vinculación entre el sector público, productivo y las universidades” (SENPLADES, 2017)

El presente proyecto, se justifica con el objetivo 5 del Plan de Desarrollo 2017-2021, al ser un proyecto de investigación y desarrollo, de estrategias de control aplicadas a robótica; tema que pretende promover la transferencia tecnológica dentro de la universidad.

Las celdas de automatización y robótica integradas se han convertido en una parte importante dentro de la industria. Es interdisciplinaria al ofrecer la unión de diversos campos como la ingeniería de modelado matemático, robótica, control de múltiples variables, visión

artificial. Las aplicaciones de las celdas robotizadas son totalmente diferentes unas de otras debido a la utilización de diferentes robots.

La importancia de esta celda robotizada para la universidad a nivel de pregrado, se fundamenta en la relación que tiene con las materias de Sistemas de Control, Procesamiento Digital de Señales, Microcontroladores, Robótica, Control Digital y Control Inteligente, en donde se podrían estudiar por separado lo de cada materia o el funcionamiento en conjunto de todo para lograr diferentes actividades.

Las diferentes teorías de control han sido fundamentales en el avance de la ingeniería y la ciencia. Estos se han convertido en una parte importante e integral en los sistemas robóticos, en los procesos modernos de fabricación y en cualquier operación industrial. Es deseable que la mayoría de los ingenieros y científicos estén familiarizados con la teoría y la práctica del control. (Ogata, 2013)

El presente trabajo de titulación servirá como base para futuras investigaciones dentro del campo de robótica colaborativo a nivel de pregrado y postgrado.

### **Alcance**

El presente trabajo de titulación pretende realizar el desarrollo de una celda colaborativa robotizada para clasificación de objetos por detección de color dentro de un área aproximadamente de 2x2 metros utilizando visión artificial con una cámara ubicada en la parte superior de la celda robotizada y conectada a una computadora. La clasificación de los objetos lo van a realizar dos robots contruidos con los kits Bioloid Premium disponibles en el Departamento de Eléctrica, Electrónica y Telecomunicaciones: un robot móvil diferencial con

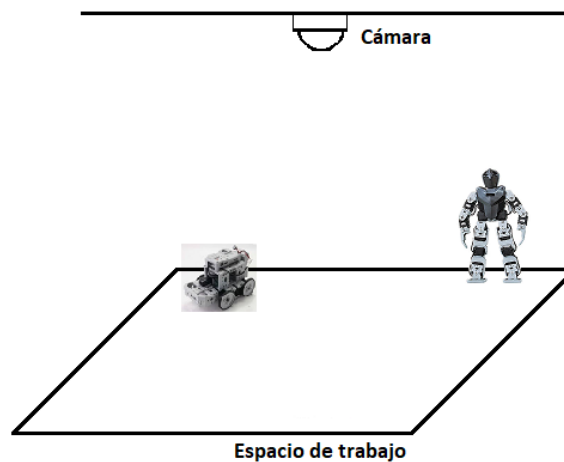
gripper en la parte frontal y un robot humanoide con cámara montada la cual permitirá analizar mejor el entorno y el objetivo.

La celda colaborativa robotizada operará de la siguiente manera:

1. Inicialmente los robots se ubicarán dentro del área de trabajo esperando la orden de inicio, como se muestra en la Figura 1.
2. En la parte superior de la celda se encuentra ubicada una cámara conectada a una computadora, en la cual se realizará una interfaz gráfica para la interacción con el usuario.

### Figura 1

*Robots a la espera de la orden de inicio*

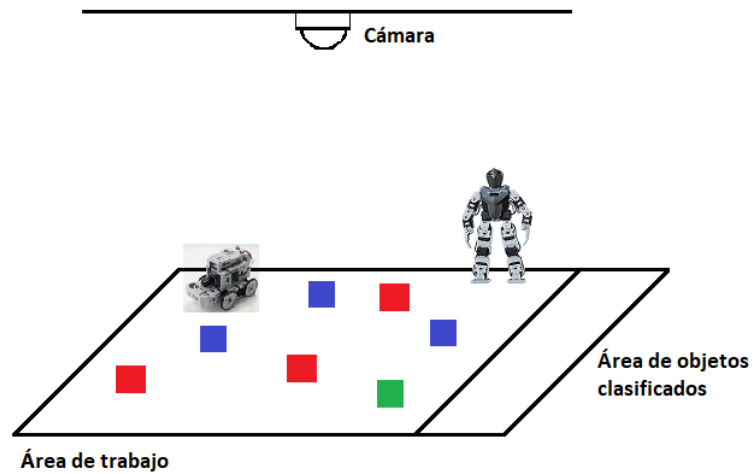


*Nota:* La imagen muestra la ubicación de los robots dentro del espacio de trabajo.

3. El usuario será el encargado de realizar la petición de los objetos a clasificar de acuerdo al color y la cantidad.
4. Una vez enviada la petición al sistema, la computadora con la cámara superior en la celda realizará la detección de los objetos de interés dentro del área de trabajo.

**Figura 2**

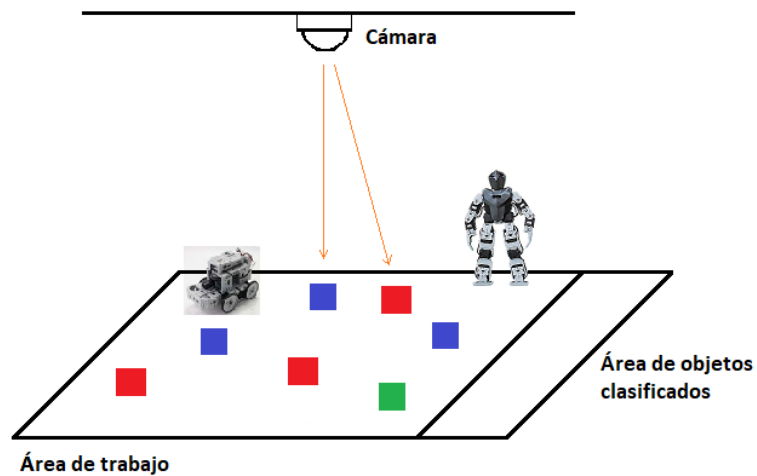
*Búsqueda de los objetos de interés dentro del área de trabajo*



*Nota:* La ubicación de los objetos y de los robots dentro del área de trabajo será de manera aleatoria.

**Figura 3**

*Detección de objetos de interés*



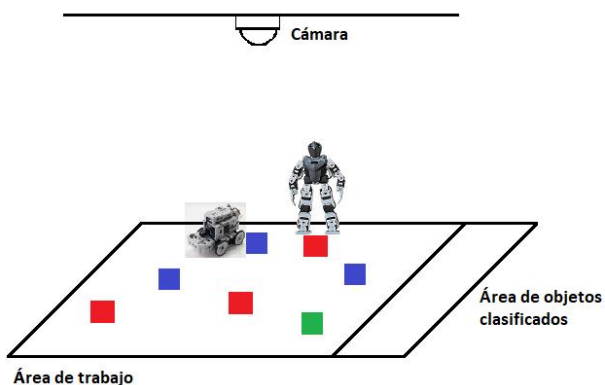
*Nota:* La computadora a través de la cámara detecta los objetos según su color.

5. Una vez que se ha detectado los objetos de interés del usuario Figura 3, se mandarían órdenes a los robots para que se trasladen hacia ellos Figura 4. Para la validación de los

objetos el robot humanoide lo realizará a través de la cámara montada, mientras que el robot móvil lo realizará con un sensor de color.

**Figura 4**

*Traslado de los robots hacia los objetos de interés*

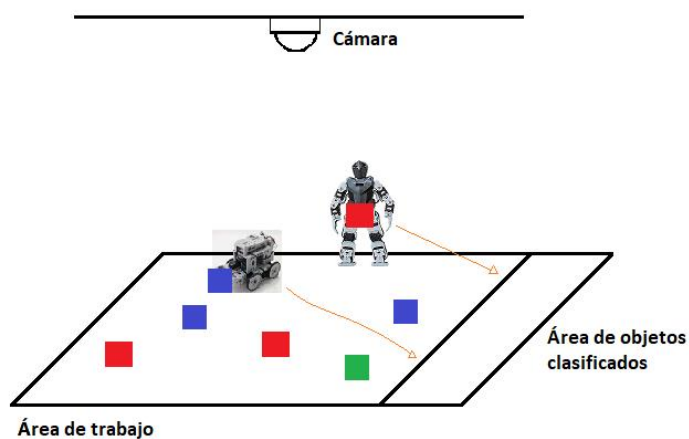


*Nota:* Cuando al robot se le asigna un objeto este se traslada a su ubicación.

6. Si los robots detectan que es el mismo color de la orden enviada desde la computadora, estos agarran al objeto y lo llevan hacia el área de objetos clasificados Figura 5.

**Figura 5**

*Agarre de los objetos de interés*



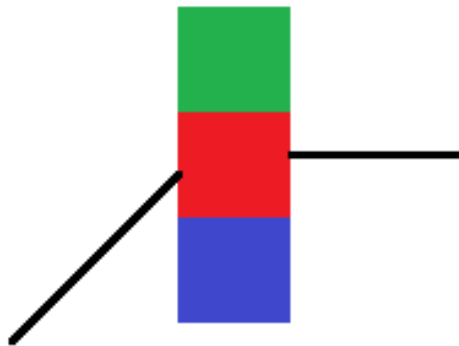
*Nota:* Una vez que el robot recoge el objeto de interés procederá a llevarlo a otra área.

7. El algoritmo clasificatorio permitirá que los robots hagan esta clasificación sin que los mismos se choquen entre ellos.
8. Una vez colocado los objetos dentro del área de objetos clasificados, los robots recibirán nuevas órdenes para los siguientes objetos de interés.
9. Se pueden tener los siguientes casos:

El primer caso se da cuando el robot móvil detecta un color diferente al de la orden, esto se da a que hay dos o más objetos superpuestos (en otras palabras, uno encima de otro), en este caso el robot móvil enviará una petición de ayuda al robot humanoide.

#### **Figura 6**

*Objetos superpuestos dentro del área de trabajo*



*Nota:* Al estar objetos superpuestos la cámara no logra detectar el objeto que se encuentra en la parte de abajo.

El robot humanoide analizará la petición de ayuda, donde se analizará si ese objeto es de prioridad urgente o normal. Si el objeto es de prioridad urgente dejará de realizar su actividad y se trasladará hacia la ubicación de la petición, bajará el objeto de interés para que el robot móvil lo pueda coger y continuará con su actividad. Si el objeto es de prioridad normal

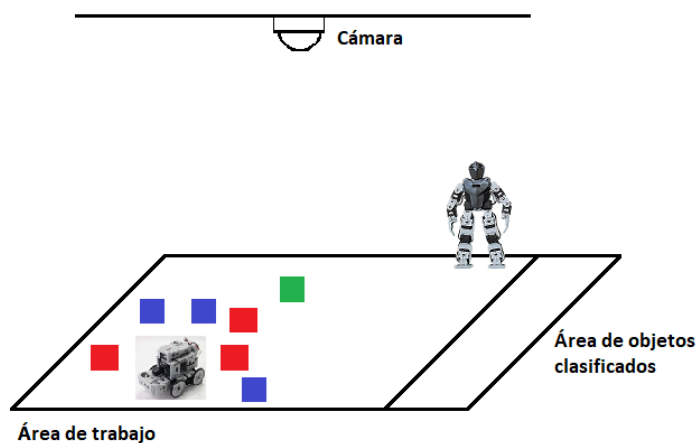


acabará primero con su actividad y luego se trasladará hacia la ubicación de la petición, bajará el objeto de interés y se pondrá a la espera de nuevas órdenes.

El segundo y el tercer caso se da cuando el robot móvil/robot humanoide se encuentra con muchos objetos a su alrededor (Figura 7) y los mismos le dificultan el paso para ir por el objeto de interés, en ese momento envía una petición de ayuda al otro robot para que le ayude a despejar el camino. El otro robot analizará si la petición de ayuda es de prioridad urgente o normal para atenderla.

**Figura 7**

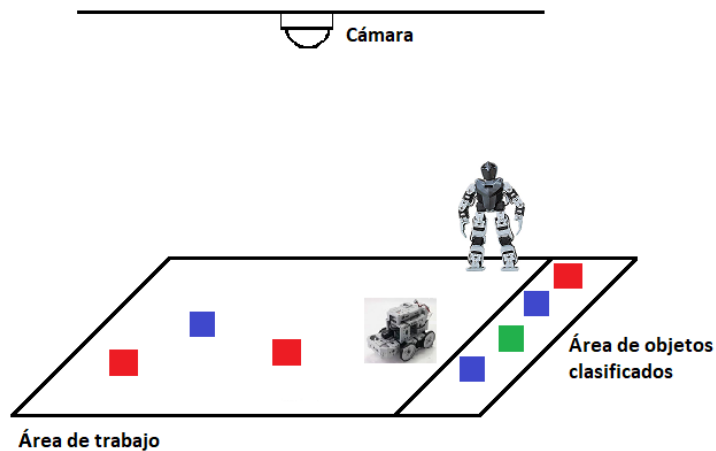
*Robot móvil con varios objetos a su alrededor*



*Nota:* Cuando un robot detecta una posible colisión hace una petición de ayuda.

10. El proceso terminará cuando se haya cumplido con la petición ingresada por el usuario

Figura 8.

**Figura 8***Fin del proceso*

*Nota:* Cuando los objetos seleccionados se encuentren dentro del área de objetos clasificados el proceso termina y los robots se detienen.

## Objetivos

### **Objetivo General**

Desarrollar una celda robotizada mediante un robot móvil y un robot humanoide para clasificación de diversos objetos dentro del área de trabajo.

### **Objetivos Específicos**

- Implementar un algoritmo de control que permita la realización de trabajo colaborativo entre dos o más robots de características diferentes.
- Buscar y ejecutar un algoritmo de visión artificial que permita la clasificación de objetos mediante la detección del color de los objetos de interés.
- Determinar la posición de los robots y de objetos dentro del área de trabajo mediante el uso de técnicas de visión.









## Capítulo II Fundamentación teórica

### Introducción

Antes de la primera revolución industrial, los humanos, los animales, el viento y el agua fueron las fuentes de energía más importantes. Durante la primera revolución industrial (Tabla 1), se implementan las máquinas de vapor para la producción de energía. Durante la segunda revolución industrial se utiliza la producción de flujo, el esfuerzo para el operador se reduce aún más. En 1885 se da la interacción física directa con la máquina, por primera vez la electrificación permite una operación simple usando interruptores y la visualización de estados usando luz eléctrica (Trenkle & Furmans, 2015).

**Tabla 1**

*Tareas de los humanos en las revoluciones industriales*

	Industria 1.0	Industria 2.0	Industria 3.0	Industria 4.0
<b>Comienzo</b>	1784	1870	1969	2004
<b>Logro</b>				
	Telar mecánico	Línea de montaje	Automatización	Fabrica digital
<b>Tipos de tareas de los humanos sobre la maquina</b>				
	Se necesita menos fuerza muscular. La máquina toma el mando	Casi no se necesita fuerza muscular. La máquina hace algunas tareas.	La máquina realiza casi todas las tareas.	La máquina adquiere conocimiento. El ser humano solo es un “conductor” que marca tendencia

*Nota:* Recuperado de (Trenkle & Furmans, 2015)

Con la automatización en la tercera revolución industrial, las maquinas asumen más tareas. Los sistemas de seguridad automatizados son más confiables que los humanos. Con la Industria 4.0, las máquinas están conectadas en red y son autónomas. Se reduce el esfuerzo ejercido por los humanos y se reducen los riesgos durante la operación (Trenkle & Furmans, 2015).

Universal Robots de Rossum del escritor checo Karel Capek en la década de 1920 introdujo la palabra robot para describir criaturas que parecen humanas pero que se usan solo para trabajos tediosos (Fukuda et al., 2017). La palabra robot proviene de una palabra checa, *Robota*, que significa trabajo forzado; por definición robot es, una maquina programable por una computadora, capaz de realizar una serie de acciones automáticamente (LUCA ROBOTICS, 2020).

Habitualmente, los robots se mantienen dentro de jaulas de seguridad para la seguridad tanto de los robots como de los humanos. En décadas pasadas se usaba robots gruesos y rígidos, en la actualidad las personas se están moviendo hacia el uso de robots colaborativos. Ya sea en la fábrica de un fabricante de automóviles o dentro de una oficina o una casa, la tendencia es sacar a los robots de sus zonas de seguridad y coexistir en el mismo entorno con humanos y otros robots (Sandakalum et al., 2018).

En el área de la robótica existen problemas que pueden ser resueltos por un solo robot sin embargo también existen tareas que necesitan de dos o más robots para cumplir un objetivo en común. Para ellos es necesario crear Sistemas Multi-Robot en los cuales cada miembro tiene bien definido su rol dentro del mismo (López, 2009).

Un objetivo frecuente al hacer uso de Sistemas Multi-Agente (MAS) en ingeniería es aprovechar la interacción entre los agentes para realizar tareas avanzadas mediante cooperación, independientemente del tipo de tarea que tengan que realizar. Se ha adoptado en gran medida los MAS, dado que la aplicación de sus herramientas garantiza que el software final emulará un comportamiento organizacional (Vargas & Castillo, 2015).

#### **Revolución Industrial 4.0**

El termino Industria 4.0 representa la cuarta revolución industrial, una nueva etapa en la organización y el control de toda la cadena de valor durante el ciclo de vida de los productos. Este ciclo se basa en los deseos del cliente que cada vez son más individualizados y van desde el pedido, el desarrollo, la fabricación y la entrega del producto final al cliente, hasta el reciclaje (Wegener, 2016). Las redes deben estar completamente integradas, sin interrupciones, haciendo uso de sistemas de software incompatibles (Bauer et al., 2015).

Flores et al. (2018) destacan que los elementos tecnológicos con lo que cuenta esta revolución son: el big data y análisis de datos, los robots autónomos, la simulación de procesos en computadora, sistemas de integración, internet de las cosas, nanotecnología, inteligencia artificial, ciberseguridad, almacenamiento de información en la nube, impresión 3D o manufactura aditiva y la realidad aumentada.

Si bien a la industria 4.0 se la puede referir como economía 4.0, la cual es impulsada por cambios en la formación de mano de obra calificada. Flores et al. (2019) mencionan que la industria 4.0 es una revolución tecnológica, de producción y creación de fábricas inteligentes, también es una revolución educativa, cultural, social, estética, de comunicación y valores. Se

produce un ambiente en el que predomina la automatización e interacción de sistemas ciberfísicos, que plantean condiciones profundas de desarrollo de la nueva educación 4.0.

### **Robótica Colaborativa**

La robótica colaborativa es el uso de múltiples agentes robóticos que puedan trabajar colectivamente para el logro eficiente de tareas complejas y arduas. Usar múltiples robots también es más ventajoso que los sistemas de un solo robot. Los robots colaboradores pueden realizar una tarea en mucho menos tiempo que un solo robot (Bano et al., 2010) .

A medida que los sistemas robóticos y de manufactura aumentan su complejidad en cuanto al número de componente, los Sistemas Multi-agentes (MAS) también necesitan de una planificación y programación de tareas que permita la coordinación de los componentes individuales, para que al trabajar colectivamente puedan desarrollar las actividades con más eficiencia que si estas fueran desarrolladas por un solo agente (Castillo et al., 2013).

En un escenario colaborativo es necesario que se den tres elementos claves: igual participación, responsabilidad e independencia de cada uno de los agentes que conforman el sistema colaborativo. Según Zambrano (2018) hay dos tipos de ambientes, el primero el cooperativo en el cual se da búsqueda de una meta común mediante la división de tareas, y el segundo el colaborativo donde el logro del objetivo se da compartiendo tareas.

### ***Sistemas Colaborativos***

**Sistema Multi-Agente.** Un sistema multi-agente es una agrupación de agentes robóticos con funciones similares que actúan entre sí para el desarrollo de una tarea determinada (Zambrano, 2018). Este tipo de sistema es la base de la robótica colaborativa y en esta debe existir una

estructura que permita la interconexión entre los distintos agentes robóticos. Mediante la planeación y programación se determinan las tareas que cada agente robótico puede ejecutar, generando una coordinación general cuya finalidad es el cumplimiento de diferentes objetivos individuales y el objetivo principal. (Aguirre & Moyano, 2019)

**Sistema homogéneo.** Este sistema se conforma de robots con iguales características tanto en software como en hardware, lo que permite un relevo dentro del mismo, debido a que pueden ser reemplazados por otro agente del sistema. Los robots modulares forman parte de los sistemas homogéneos, los cuales son reconfigurables, comparten información y llevan a cabo tareas específicas, sus características físicas permiten que puedan ser organizados de diferentes maneras para un tipo de necesidad. En la Figura 9 se puede observar un ejemplo de robots homogéneos (Zambrano, 2018).

**Figura 9**

*Robots homogéneos*



*Nota:* Recuperado de (López, 2009)

**Sistema heterogéneo.** Este sistema se encuentra conformado por robots de diferentes tipos y características tanto en software como en hardware. Los robots tienen diferentes comportamientos y cada uno puede cumplir una tarea específica que se encuentra determinado

por sus características físicas. Por ejemplo, el transportar objetos por robots móviles, mientras que la selección de objetos por robots manipuladores (Zambrano, 2018).

### ***Arquitecturas de control***

Una arquitectura de control se utiliza para resolver la tarea general asignada al MAS, también se encarga de fusionar la información obtenida por varios robots, lidiar con la comunicación, asignar las tareas a los agentes, especificar un conjunto de reglas que permitan interactuar a los agentes de modo individual, además de superar las interferencias entre los robots de modo que la coordinación se pueda lograr de manera eficiente.

A medida que el número de agentes aumenta en el MAS, el control del sistema se vuelve más complejo (Bonilla, 2019). La coordinación de tareas en el MAS se ha dividido en tres categorías según la arquitectura de control del equipo, estas pueden ser: distribuida, centralizada y descentralizada.

**Arquitectura de coordinación distribuida.** En esta arquitectura, no existe un robot de control central. Los agentes del MAS son completamente autónomos, cada uno se encarga del cálculo de la tarea y de la toma de decisiones, ya que se encuentran provistos de varios sensores para la adquisición de datos del entorno, un controlador propio y tecnología de comunicación. Los robots se coordinan en función de la estructura grafica que se muestra en la Figura 10-(a), en la que cada nodo representa un agente y entre nodos existe el intercambio de información para construir un modelo completo del entorno.

**Arquitectura de coordinación centralizada.** En la arquitectura de coordinación centralizada, hay un agente de control central denominado como líder, el cual tiene la capacidad de comunicarse con todos los demás agentes (Figura 10-(b)). El agente central se encarga de todas



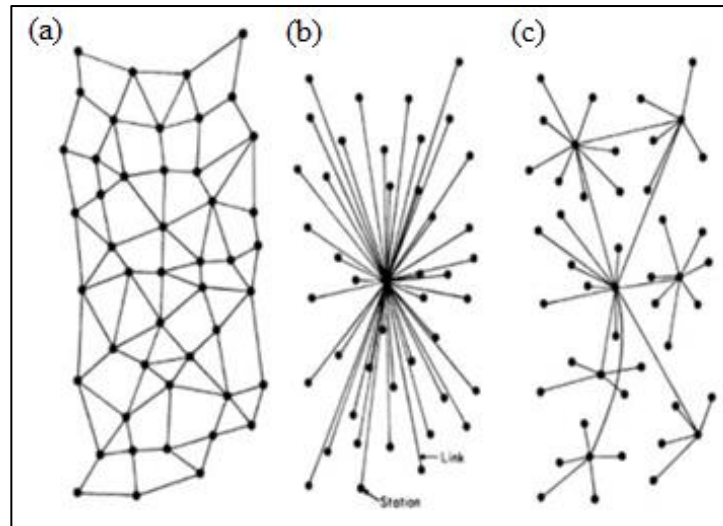
las funciones de decisión y planificación, buscando soluciones óptimas para el cumplimiento de la tarea general. Los demás agentes de esta arquitectura presentan diseños más sencillos que la anterior arquitectura mencionada (Bonilla, 2019).

Esta arquitectura funciona bien para una pequeña cantidad de agentes y funciona más rápido que la coordinación descentralizada, pero se vuelve ineficiente para una gran cantidad de robots debido a la pérdida de información y la sobrecarga de comunicación (Mohamed et al., 2018). Un fallo en la comunicación con el agente central provoca que la arquitectura falle completamente, debido a que toda la información debe salir y llegar a este agente (Aguirre & Moyano, 2019).

**Arquitectura de coordinación descentralizada.** Esta arquitectura es un intermediario entre la arquitectura centralizada y la arquitectura distribuida los agentes se basan en la estructura que se muestra en la Figura 10-(c). El proceso de control se logra utilizando más de un agente de control central local, lo que conlleva a una organización de agentes en grupos donde cada grupo es responsable del cumplimiento de las sub-tareas individualmente de manera centralizada (Mohamed et al., 2018). Esta arquitectura descentralizada proporciona soluciones más sólidas y capaces de influir en las acciones de todo el equipo, aquí la caída de un agente central local no conlleva a la falla completa de la arquitectura si no solo de un grupo.

**Figura 10**

*Diferentes tipos de jerarquía de coordinación (a) Distribuida (b) Centralizado (c) Descentralizado*



*Nota:* Recuperado de (Mohamed et al., 2018)

### **Robots Móviles**

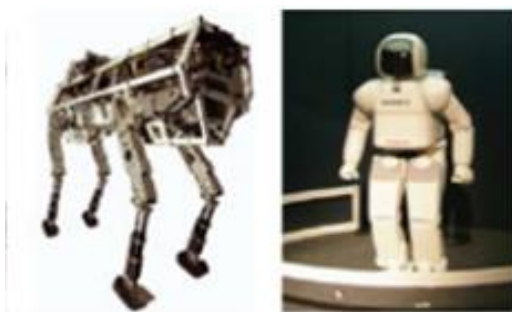
Un robot móvil es un sistema electromecánico capaz de desplazarse de manera autónoma sin estar sujeto físicamente a un solo punto. Posee sensores que permiten monitorear a cada momento su posición relativa a su punto de origen y a su punto de destino (Barrientos et al., 2007). Se encuentran provistos de algún tipo de sistema de locomoción que les permite desplazarse, tales como se puede observar en la Figura 11.

**Figura 11**

*Tipos de locomoción en un robot móvil (a) Ruedas (b) Patas (c) Orugas*



(a)



(b)



(c)

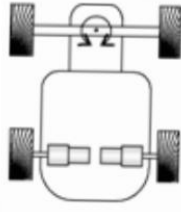

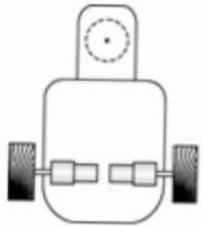

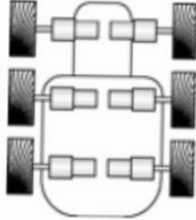



*Nota:* Recuperado de (Barrientos et al., 2007)

### ***Robots Móviles de Ruedas***

Los robots móviles emplean diferentes tipos de locomoción mediante ruedas que les confiere características y propiedades diferentes (Bañó, 2003). En la Tabla 2 se detalla las principales características de los sistemas de locomoción más comunes en los robots móviles de ruedas.

Tabla 2

*Sistemas de locomoción en robots móviles de ruedas*

Sistema	Características	Esquema	Ejemplo
<b>Ackerman</b>	Se basa en dos ruedas traseras tractoras, mientras que las ruedas delanteras son del tipo direccionamiento, y se utilizan para seguir la trayectoria del robot.		 RB-CAR
<b>Tracción diferencial</b>	Tiene dos ruedas montadas, pero son propulsadas y controladas independientemente. El direccionamiento viene dado por la diferencia de velocidades en las ruedas. Adicionalmente, existe una o más ruedas para soporte.		 RB-2 BASE
<b>Skid steer</b>	Este sistema dispone de varias ruedas en cada lado del vehículo que actúan de forma simultánea. El movimiento es el resultado de combinar las velocidades de las ruedas de la izquierda con las de la derecha.		 SUMMIT-XL HL
<b>Tracción omnidireccional</b>	Este sistema se basa en la utilización de tres ruedas directrices y motrices. Posee tres grados de libertad, por lo que puede realizar cualquier movimiento, y posicionarse en cualquier posición en cualquier orientación.		 SUMMIT-XL STEEL

*Nota:* Recuperado de (Barrientos et al., 2007), (Gómez, 2011), (Robotnik Automation S.L.L., 2020)

## **Robots Humanoides**

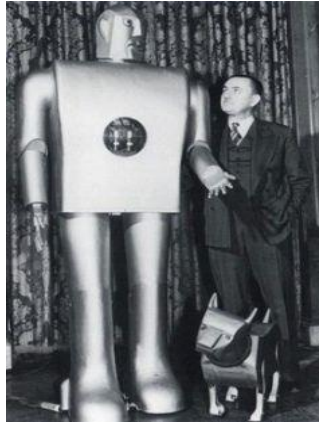
La robótica humanoide es una rama importante de la robótica y no solo está asociada con las disciplinas de la ciencia y la ingeniería, sino que también está profundamente conectada con los dominios sociales, legales y éticos (Fukuda et al., 2017). Los primeros diseños subestimaron significativamente los desafíos asociados; sin embargo, el surgimiento de nuevas teorías y tecnologías han permitido realizar mejores diseños de robots humanoides.

Se puede definir a un robot humanoide como un sistema robotizado que es desarrollado para simular la constitución y los movimientos realizados por los humanos. En general, los robots humanoides tienen un torso con una cabeza, dos brazos y dos piernas, sin embargo, muchos robots humanoides modernos solo se basan en el cuerpo humano de la cintura para arriba (LUCA ROBOTICS, 2020).

Un robot humanoide debe poder percibir su propio estado y el estado de su entorno para poder actuar con éxito (Behnke, 2008), razón por la cual es importante que tenga capacidades sensorio motoras incluidas en sus diseños. En la búsqueda de robots humanoides, algunos logros loables incluyen robots y maquinas con locomoción bípeda, capacidades de aprendizaje, arquitecturas predictivas y funciones bioinspiradas (Fukuda et al., 2017).

**Figura 12**

*Elektro. El primer robot de la historia*



*Nota:* Recuperado de (Wikipedia, 2020)

El primer humanoide real fue creado por la empresa Westinghouse Electric and Manufacturing Co. en 1937, fue denominado “Elektro the Moto-Man”, construido de aluminio y acero. Elektro (Figura 12) era capaz de realizar 26 rutinas diferentes, como despertarse, hablar, contar y fumar (Revista de Robots, 2020).

En las últimas décadas se ha llevado acabo muchas realizaciones robóticas en formas antropomórficas reconocibles que integran el movimiento, la percepción y la interacción para recrear las funciones físicas, cognitivas y sociales de los humanos (Fukuda et al., 2017). Pero aun así las capacidades de los robots humanoides actuales son bastante limitadas, en comparación con los humanos (Behnke, 2008).

A continuación, en la Figura 13 se muestran algunos ejemplos de los robots humanoides actuales, que se han vuelto famosos en el mundo, debido a su aplicación en diversos campos.

**Figura 13**

*Robots humanoides (de izquierda a derecha): Pepper, NAO, Sophia, Atlas*



*Nota:* Recuperado de (Revista de Robots, 2020)

### **Visión por Computador**

Visión Artificial o Visión por Computador es el conjunto de técnicas de adquisición, procesamiento, análisis y comprensión de las imágenes del mundo real (Moreno & Ramírez, 2017).

#### ***Sistema de captura de imágenes***

Sánchez & Ricolfe (2016) describen que una imagen es una representación bidimensional de una escena del mundo tridimensional, resultado de la adquisición de una señal proporcionada por un sensor. Una imagen digital se define como una matriz de dimensiones  $N \times M$ , los valores de cada elemento de la imagen representan desde niveles oscuros de luminosidad hasta valores claros.

El proceso de formación y captura de una imagen digital depende de un conjunto de elementos cada uno de los cuales cumple una misión determinada, dentro de este conjunto de elementos se encuentran las fuentes de iluminación, el medio de la escena, los objetos de la escena y las ópticas.

**Iluminación.** En las aplicaciones de visión, los sistemas de iluminación cobran una importancia especial y que puede llegar a estar subestimados. Un sistema de iluminación según Sánchez & Ricolfe (2016) si está bien diseñado proporciona luz a la escena de forma que la imagen que se obtiene favorece al posterior proceso sobre la misma, mejorando la información necesaria para la detección y extracción de los objetos y las características de interés. En la Tabla 3 se clasifican los tipos de iluminación.

**Tabla 3**

*Características de cada uno de los tipos de iluminación artificial.*

<b>Tipo de iluminación</b>	<b>Ventajas</b>	<b>Inconvenientes</b>
Incandescente/ Halógena	Bajo coste y fáciles de utilizar. Permite ajustar la intensidad de luz. Oscila 50 veces por segundo.	Desprende gran cantidad de calor. Su espectro se centra en el rojo siendo deficiente para azules, verdes o amarillos.
Fluorescentes	Se calienta menos que el incandescente. Su espectro se centra en los colores del ojo humano. La duración está estimada en torno a 10.000 horas.	La longitud de onda de la luz cambia con el uso. Para que sean válidos en aplicaciones industriales tienen que trabajar a una frecuencia del orden de 25 KHz.
Led	Gran durabilidad (100.000 horas). Posibilidad de encender y apagar solamente en el tiempo de captura de la imagen. Fácil elección de la longitud de onda de la fuente de luz dentro del espectro visible e infrarrojo.	Precio




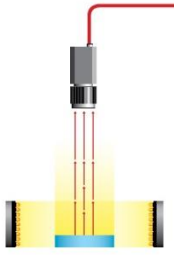
Tipo de iluminación	Ventajas	Inconvenientes
Láser	Se utilizan para generar luz estructurada con forma diversas tales como líneas, líneas paralelas, líneas cruzadas, reticulares, puntos y matriz de puntos. Están disponibles en multitud de la longitud de ondas desde el visible al infrarrojo cercano.	Precio
Fibra óptica	Se utiliza para llevar la luz a cualquier punto distante de la fuente de luz. Permite iluminar pequeñas áreas.	Precio. Solo sirve para iluminar pequeñas áreas.

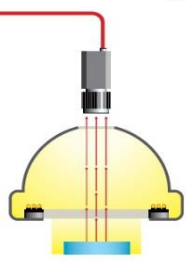
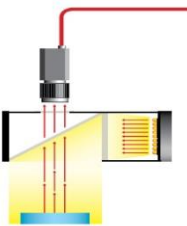
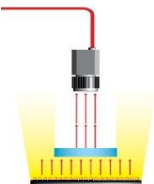
*Nota:* Recuperado de (Sánchez & Rricolfe, 2016)

Para mejorar notablemente la eficiencia del sistema de visión es necesario utilizar la técnica de iluminación adecuada (Sánchez & Rricolfe, 2016). Existen diferentes técnicas de iluminación que se utilizan para resaltar diferentes aspectos que dependen de las posibles aplicaciones.

**Tabla 4**

*Técnicas de iluminación*

Técnica	Descripción	Esquema	Pros	Contras
Direccional	La iluminación de la escena se realiza con uno o varios puntos de luz en la que el ángulo de incidencia no es ni paralelo ni perpendicular al eje de la cámara.		Flexible. Adaptable. Barata.	Produce brillos. Genera sombras.
Lateral o darkfield	Se utiliza luz direccional en la que el ángulo de incidencia es paralelo a la superficie a inspeccionar y perpendicular al eje de la cámara.		Resalta la textura de la superficie del objeto. Descubre grietas.	Aparecen zonas quemadas y sombras. Poco contraste del borde.

Técnica	Descripción	Esquema	Pros	Contras
Difusa	Iluminando la escena de forma indirecta se consigue una luz suave.		Reduce brillos. Reduce sombras. Iluminación suave.	Sistema de iluminación de gran tamaño. Dificultad para encajar en pequeños espacios.
Difusa axial	Luz difusa alineada con el eje óptico de la cámara. Se utiliza un cristal polarizado.		No existen sombras. Iluminación suave.	Poca intensidad.
Contraluz	El objeto a inspeccionar se sitúa entre la fuente de luz y la cámara.		Se obtiene una imagen del borde bien definida	Elimina los detalles de la superficie.

*Nota:* Recuperado de (INFAIMON, 2020), (Sánchez & Rricolfe, 2016)

### **Procesamiento digital de imágenes**

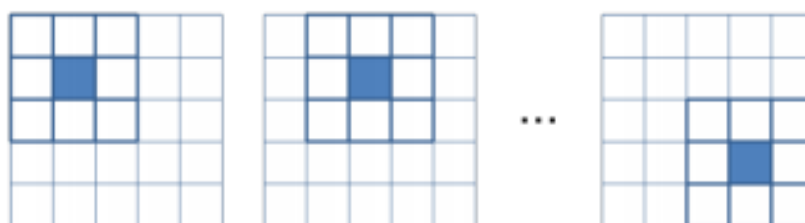
El procesamiento digital de imágenes como lo detalla (Domínguez, 1996) se lo denomina al conjuntos de técnicas y procesos para descubrir o hacer resaltar información contenida en una imagen usando como herramienta principal una computadora. Las principales técnicas de procesamiento digital de imágenes mejoran y realzan sus características, las operaciones que son llevadas a cabo por este tipo de técnicas implican el procesamiento del valor de sus píxeles (Espinosa et al., 2016).

**Filtrado en el dominio del espacio.** El filtrado de imágenes en el dominio del espacio consiste en operaciones que se llevan a cabo sobre los píxeles de la imagen. Los filtros que se usan en el dominio del espacio son lineales y no lineales, en donde los filtros lineales se encuentran

basados en los llamados kernels o máscaras de convolución. Los filtros basados en kernels se representan como una ventana móvil de coeficientes que recorre toda la imagen centrada en cada uno de sus píxeles. El nuevo valor de cada píxel se obtiene en función de los valores del kernel, del propio píxel y de sus vecinos (Figura 14). (Espinosa et al., 2016)

#### Figura 14

*Representación y movimiento de un kernel al recorrer toda la imagen*



*Nota:* Recuperado de (Espinosa et al., 2016)

Los filtros pueden clasificarse como pasa bajo o pasa alto. Los filtros paso bajo realizan un suavizado de la imagen, este tipo de filtros elimina ruido y otros pequeños detalles que no son de interés. Entre los filtros de suavizado destacan los de media, media ponderada, mediana, adaptativo y Gaussiano. Los filtros paso alto realizan un realzado de los detalles de la imagen, este tipo de filtro atenúa las zonas uniformes y enfatiza los detalles de interés. Existen dos tipos de procedimientos: basados en la sustracción de la media y basados en derivadas. (Espinosa et al., 2016)

En el filtro gaussiano, el kernel utilizado contiene como coeficientes valores que siguen una distribución Gaussiana bivalente. El valor máximo aparece en el píxel de referencia y disminuye en función de la varianza establecida. En la práctica, este tipo de filtros es utilizada como etapa de preprocesado para la detección de bordes u otras operaciones. (Espinosa et al., 2016)

**Realce y detección de bordes.** Los métodos para realce de bordes más utilizados son el Canny y el Laplaciano, los cuales son filtros espaciales. Los filtros laplacianos están basados en la segunda derivada. La siguiente ecuación muestra la máscara más común.

$$M = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Este tipo de filtros es muy sensible al ruido. Al igual que los bordes, el ruido también se ve realzado tras aplicarlos.

El método Canny utiliza el gradiente de la imagen para realzar los bordes. Ya que el gradiente es sensible al ruido, este método utiliza dos umbrales para detectar bordes. Los bordes con valores por debajo del umbral más pequeño son descartados, los que tienen valores entre ambos umbrales son marcados como bordes débiles y los que tienen valores mayores que el umbral mayor son marcados como bordes fuertes. Tanto el método de Canny como el Laplaciano en la práctica son aplicados después de un suavizado Gaussiano. (Espinosa et al., 2016)

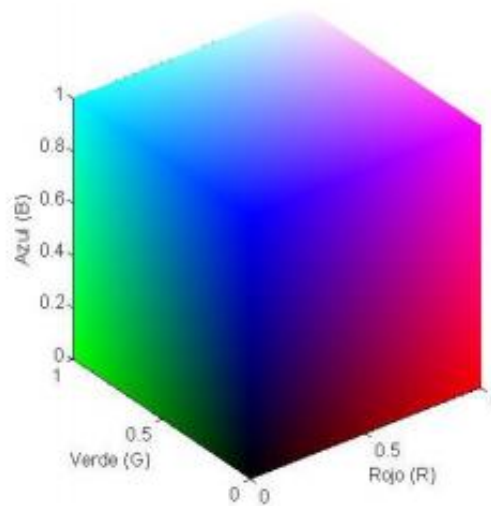
**Modelos de color.** Un modelo de color es un modelo matemático que describe como los colores se pueden representar mediante tuplas de números (Skrede, 2017). Los espacios de color son organizaciones específicas que permiten describir colores y a su vez descomponerlos en distintos canales (Martínet et al., 2016). Existen diferentes formas para representar los colores, los más utilizados son:

- RGB: periféricos como pantallas, cámaras y escáneres.
- CMYK: impresoras.
- HVS: selección intuitiva de colores

En el modelo RGB al variar las cantidades de los colores primarios (rojo, verde y azul), se pueden obtener numerosos colores. Los colores rojo, verde y azul representan las tres componentes elementales del sistema de coordenadas cartesianas y definen el espacio de color básico, donde cada color es un punto en tres dimensiones.

### Figura 15

*Representación en forma de cubo del modelo de color RGB*

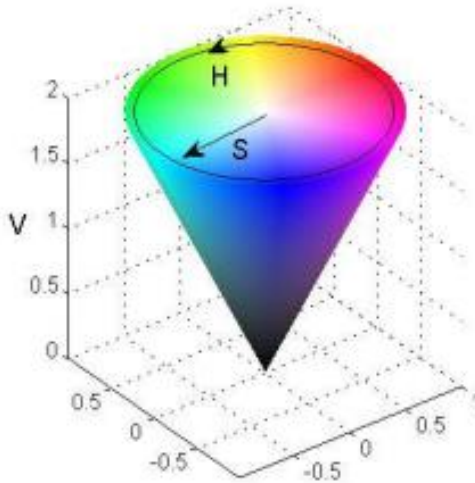


*Nota:* Recuperado de (Martín et al., 2016)

El modelo de color HSV según Barranco & Salgado (2015) es una descripción de color en términos de tres componentes: Matiz, Saturación y Valor (Hue, Saturation y Value). Este modelo describe de manera circular la frecuencia de cada color en el espectro visible en la componente H, la pureza del color en la componente S, mientras que la componente V la cercanía del pixel al negro o al blanco.

**Figura 16**

*Representación en forma de cono invertido del modelo de color HSV*



*Nota:* Recuperado de (Martín et al., 2016)

**Procesamiento morfológico.** El procesamiento morfológico es una técnica de la visión por computador que analiza las imágenes basándose en la forma, son empleados tanto a imágenes binarias, como a imágenes en niveles de gris y de color. Esta técnica suele emplearse tanto para el pre-procesamiento, con operaciones para la eliminación de ruido o el realce de objetos, como para el análisis de imágenes, extrayendo características tales como el esqueleto, definición de contornos o búsqueda de patrones (Cancelas et al., 2016).

Las transformaciones morfológicas se basan en el empleo de las operaciones de conjuntos entre una imagen y un elemento estructurante, estas consisten en trasladar el elemento estructurante a lo largo de la imagen, de manera semejante a las convoluciones en el filtrado lineal. Las principales transformaciones morfológicas son:

- Dilatación (dilation): esta operación es también conocida como adición de Minkowski, es una operación de crecimiento progresivo. Al dilatar la imagen aumenta su tamaño, pero puede perderse detalles de la forma.
- Erosión (erosion): de manera análoga esta operación es conocida con resta de Minkowski. La erosión reducirá el tamaño de los objetos, puede emplearse para suprimir de una imagen los elementos más pequeños dejando solo los grandes, también permite separar objetos rompiendo contactos débiles entre ellos.
- Apertura (opening): esta operación se realiza mediante una erosión seguida de una dilatación empleando el mismo elemento estructurante.
- Cierre (closing): en esta operación primero se hará una operación de dilatación, para seguidamente, manteniendo el mismo elemento estructurante, hacer una erosión.

### **Capítulo III Desarrollo del sistema clasificatorio por visión**

En este capítulo, se va a hacer una breve explicación de cómo se encuentra conformada la celda colaborativa robotizada del presente trabajo de titulación. Primero se detallará los componentes de hardware de los que se encuentra compuesta cada parte de la celda colaborativa, posterior se detallará como se encuentra conformado la parte del software para el funcionamiento de la celda colaborativa.

#### **Descripción del hardware del sistema clasificatorio**

El hardware del sistema clasificatorio se encuentra conformado por cuatro componentes: un robot humanoide, un robot móvil de ruedas, una celda de trabajo y objetos de color. En primer lugar, se explicarán algunas limitaciones imprevistas con las que no se contaban en la realización del proyecto, después se describen los elementos de hardware utilizados en el sistema clasificatorio, para finalmente explicar cómo se encuentran conformados cada uno de los componentes del sistema clasificatorio.

#### ***Limitaciones encontradas***

Para la realización del trabajo de titulación se encontraron algunas limitaciones, razones por las cuales se tuvieron que adaptar ciertas cosas para cumplir con los objetivos del mismo.

Cada kit Bioloid Premium viene con un módulo ZIG-100/110A, el módulo ZIG-110A cuenta con un conector molex<sup>1</sup> específico para este módulo. La limitación encontrada fue que solo se cuenta con un conector para los dos módulos ZIG-110A, otra limitación fue que para

---

<sup>1</sup> Molex - Termino que expresa la interconexión de un zócalo de dos piezas.



conectar el módulo ZIG-100 directo a la computadora se necesita el dispositivo USB2DYNAMIXEL el cual no viene incluido en el kit, además al ser un kit robótico antiguo no es fácil conseguir este dispositivo en el país a más que la empresa fabricante ya lo dejó fuera de producción.

Los kits Bioloid Premium viene con 18 actuadores cada uno por lo que se pensaba tener un total de 36 actuadores, los cuales hubiesen servido para armar dos robots humanoides de 18 GDL<sup>2</sup> cada uno. La limitación que se encontró fue que de los 36 actuadores con los que se quería contar apenas 26 actuadores son funcionales.

Las baterías LiPo<sup>3</sup> que vienen incluidas en cada uno de los kits ya son útiles, debido a que ya no se las había usado un tiempo considerable. Por lo tanto, se tuvo que hacer la adquisición de nuevas baterías, pero al no encontrar baterías de la misma marca del kit se presentó una limitación con respecto a las carcasas.

### ***Computador***

En el proyecto es necesario un computador encargado del procesamiento central de toda la información, los requerimientos del computador son mínimos por lo que se puede usar cualquier tipo de computadora. Para el proyecto se va utilizar un computador portátil marca Lenovo modelo ideapad 310-15ISK con procesador Intel Core i7-6500U 2.50GHz y memoria RAM de 12 GB.

---

<sup>2</sup> GDL - Grados De Libertad

<sup>3</sup> LiPo - Polímero de litio.

### **Cámara**

El objetivo de utilizar una cámara es para incorporar un sistema de visión externa, que permitirá obtener una vista de planta superior del entorno de trabajo dentro de la celda colaborativa. En consecuencia, se procederá a realizar la selección del tipo de cámara óptima para el proyecto mediante la siguiente tabla de análisis comparativo.

**Tabla 5**

*Análisis comparativo técnico para determinar el mejor tipo de cámara*

<b>Tipo de cámara</b>	<b>Precio</b>	<b>Acceso Externo</b>	<b>Seguridad</b>	<b>Robustez</b>	<b>Forma de programación</b>	<b>Total</b>
IP	2	3	1	2	2	10
Web	4	5	4	5	4	22
Lineal	3	2	1	2	2	10
Matricial	1	1	1	2	1	6

*Nota:* En la tabla la ponderación de 5 es de mayor accesibilidad y 1 de menor accesibilidad.

Como se aprecia en la Tabla 5 el tipo de cámara a utilizar en el proyecto es una cámara web. No se tuvo que hacer la adquisición de una cámara web debido a que en el Departamento de Eléctrica, Electrónica y Telecomunicaciones se cuenta con la cámara Klip Xtreme modelo Xcam 300, cuyas características son suficientes para la realización del proyecto.

Las características principales con la que cuenta esta cámara son: interfaz USB 2.0, resolución de 640x480 píxeles, micrófono incorporado, además posee un balance de blancos automáticos, rotación omnidireccional. La cámara puede enfocar a una distancia mínima de 3cm y una máxima de 200cm, cubriendo un área máxima de aproximadamente 120x160cm. El espacio de trabajo propuesto para el proyecto es de 200x200cm, pero debido a las características con las que cuenta la cámara elegida para el proyecto se modifica el espacio de trabajo.

**Figura 17**

*Cámara Web Klip Xtreme Xcam 300*



*Nota:* La figura muestra la cámara utilizada en el proyecto de titulación.

***Kit Bioloid Premium***

El kit Bioloid Premium es un kit de robot desarrollado para aficionados y para el ámbito educativo, es fabricado por la empresa de robots coreana ROBOTIS. El kit de la Figura 18 consta de componentes modulares que permite construir robots en 29 configuraciones diferentes, desde un robot de 1 GDL hasta a un humanoide de 18 GDL. Los kits Bioloid Premium disponibles en el Departamento de Eléctrica, Electrónica y Telecomunicaciones cuenta con los siguientes elementos que se describen en la Tabla 6.

**Figura 18**

*Kit Bioloid Premium*



*Nota:* Recuperado de (ROBOTIS, s.f.)

**Tabla 6***Contenido del kit Bioloid Premium*

<b>Cantidad</b>	<b>Artículo</b>
1	Controlador CM-510
18	Servos DYNAMIXEL AX-12A
1	Conjunto de armazón de plástico ligero
1	Conjunto de cables de conexión
1	Conjunto de piel exclusivo
1	Sensor Gyro
1	Sensor de medición de distancia Sharp
2	Sensor IR
1	Módulo de comunicación inalámbrica (ZIG-100/110A)
1	Batería de polímero de litio de 11.1V a 1000mAh
1	Cargador exclusivo de batería
1	Destornillador
1	Guía de programación RoboPlus
1	Manual de ensamble (29 robots)

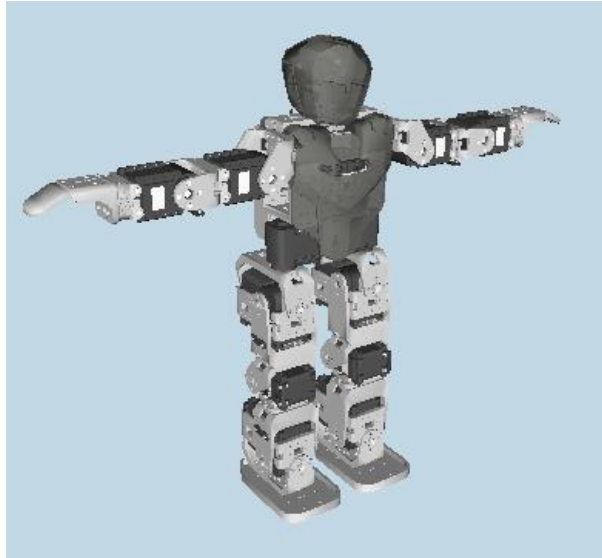
*Nota:* La tabla muestra los elementos del kit Bioloid Premium.

El kit permite ensamblar tres tipos de robots humanoides: tipo-A, tipo-B y tipo-C. La diferencia entre estos modelos son los GDL, el tipo-A tiene 18 GDL mientras que los otros dos tienen 16 GDL, esto implica que para ensamblar un humanoide tipo-A se requieren de 18 actuadores y para los otros modelos apenas 16 actuadores. También el kit permite ensamblar cuatro variaciones de robots móviles: robot de sondeo, robot excavador, robot auto inteligente y robot trazador de línea.

Para la realización del proyecto se tomó como referencia al robot humanoide tipo-A (Figura 19) y al robot de sondeo (Figura 20), luego se hicieron variaciones en sus diseños implementando y quitando elementos que permitirán cumplir con los objetivos individuales de cada uno.

**Figura 19**

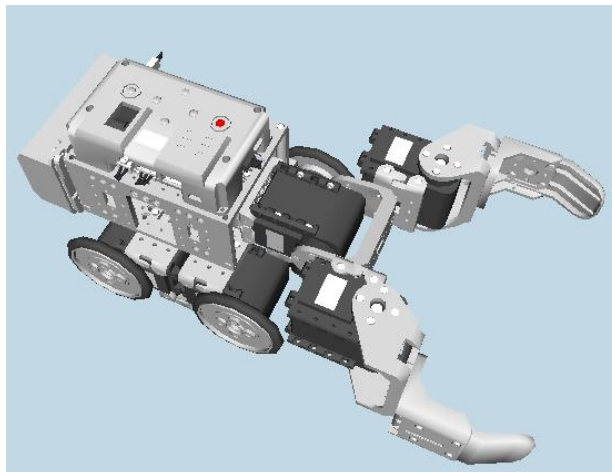
*Robot humanoide tipo-A*



*Nota:* La figura fue recuperado del Software R+Design

**Figura 20**

*Robot de sondeo*



*Nota:* La figura fue recuperado del Software R+Design

### **Controlador CM-510**

Cada kit Bioloid Premium viene equipado con un controlador CM-510 como se puede observar en la Figura 21. El controlador CM-510 tiene un procesador ATmega2561 de la familia AVR RISC Microchip de 8 bits, la tensión recomendada de trabajo es de 11.1V la misma que es proporcionada por una batería LiPo de 3 celdas. Para la realización del presente trabajo de titulación solo se utilizó 1 de los 2 controladores disponibles, el mismo se encarga del control de los actuadores y la adquisición de datos de los sensores con los que cuenta el robot humanoide.

**Figura 21**

*Controlador CM-510*

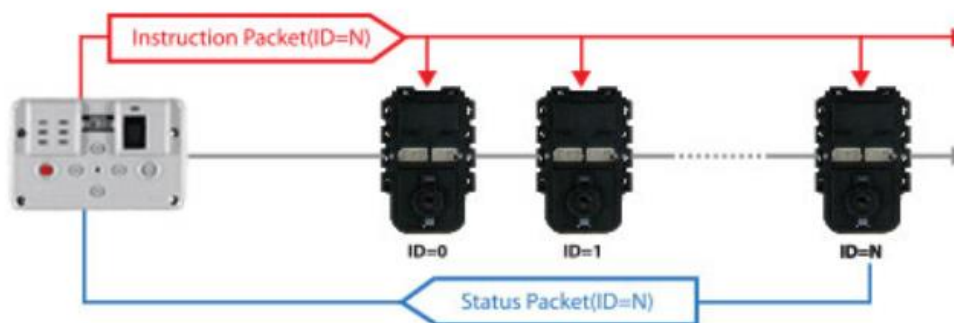


*Nota: Recuperado de (ROBOTIS, s.f.)*

El controlador CM-510 cuenta con 6 puertos auxiliares de 5 pines compatibles para la conexión de sensores tales como: sensor de distancia Sharp, sensor de tacto, sensor de IR, etc.; además posee un puerto opcional para comunicación inalámbrica Zigbee, este puerto también puede ser utilizado para comunicarse con cualquier otro dispositivo externo; también tiene 5 puertos de bus de servicio AX que se utiliza para conectar el AX DYNAMIXEL en un método de conexión en cadena como se muestra en la Figura 22.

**Figura 22**

*Conexión en cadena servomotores AX-12A con CM-510*



*Nota:* Recuperado de (ROBOTIS, s.f.)

La distribución de pines del puerto externo de 5 pines en el controlador CM-510 se tiene como en la Figura 23 donde: el pin 1-OUT (5V-par posible), pin 2-VDD (5V), pin 3-ADC (lectura de señales analógicas de sensores), pin 4-GND y el pin 5-NC (no utilizado). Para el proyecto se decidió usar uno de estos puertos para suministrar energía a la RPZW<sup>4</sup>.

**Figura 23**

*Distribución de pines del puerto externo CM-510*



*Nota:* Recuperado de (ROBOTIS, s.f.)

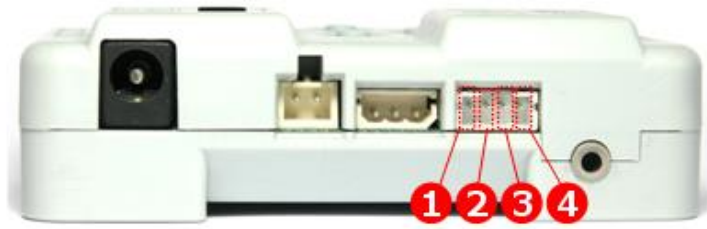
---

<sup>4</sup> RPZW – abreviación de Raspberry Pi Zero W

El puerto de conexión del dispositivo de comunicación en el CM-510 tiene la siguiente distribución de pines Figura 24, donde: pin 1-GND, pin 2-VDD (2.7 ~ 3.6V), pin 3-RXD y pin 4-TXD. Con esta distribución de pines en el proyecto se logró establecer con la RPZW comunicación serial.

#### **Figura 24**

*Composición de los pines del puerto de comunicación*



*Nota:* Recuperado de (ROBOTIS, s.f.)

#### **Tarjeta de desarrollo**

Una tarjeta de desarrollo es un dispositivo electrónico para el diseño y prototipado rápido de sistemas digitales o analógicos. Las tarjetas de desarrollo vienen incorporadas con un microcontrolador, memoria, módulos de comunicación, etc., además posee un entorno de desarrollo para la programación de aplicaciones de control, procesamiento de datos, entre otras.

A continuación, se procederá a realizar la selección de la tarjeta de desarrollo óptima para el proyecto mediante la siguiente tabla de análisis comparativo.



**Tabla 7**

*Análisis comparativo técnico para determinar la mejor tarjeta de desarrollo*

<b>Tecnología</b>	<b>Memoria</b>	<b>Puertos E/S</b>	<b>Precio</b>	<b>Existencia</b>	<b>Robustez</b>	<b>Forma de programación</b>	<b>Total</b>
Teensy	5	4	4	4	5	4	26
Arduino	3	3	3	5	3	5	22
PIC	1	2	1	2	2	3	11
ESP32	3	1	3	4	4	2	17

*Nota:* En la tabla la ponderación de 5 es de mayor accesibilidad y 1 de menor accesibilidad.

Como se aprecia en la Tabla 7 la tecnología a utilizar es la tarjeta de desarrollo Teensy.

La placa de desarrollo USB de Teensy que se va a usar en el proyecto es la versión 3.6 (Figura 25), se encuentra basada en el procesador ARM Cortex-M4 de 32 bits con un cristal que le permite trabajar a 180MHz. Tiene 58 pines digitales de entrada/salida todos con interrupción, el voltaje de salida/entrada es de 3.3V a una corriente de 10mA; además tiene 25 pines analógicos de entrada que funciona con 2 convertidores de 13 bits de resolución y 2 pines analógicos de salida con 12 bits de resolución. También cuenta con 6 puertos seriales de los cuales 2 son FIFO<sup>5</sup>.

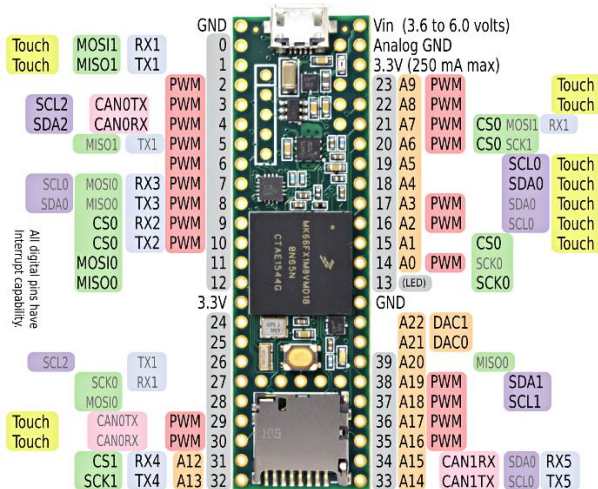
Para el proyecto se decidió usar esta tarjeta de desarrollo como controlador principal del robot móvil, en el cual se encarga del control de los actuadores, la comunicación inalámbrica, y la gestión del sensor de color.

---

<sup>5</sup> FIFO - del inglés de First Input-First Output (Primero en Entrar-Primero en Salir)

Figura 25

Placa de desarrollo Teensy 3.6



Nota: Recuperado de ([https://www.pjrc.com/store/teensy36\\_pins.html](https://www.pjrc.com/store/teensy36_pins.html))

### Tarjeta de desarrollo para visión

Un requerimiento del proyecto es dotar de visión artificial al robot humanoide, para lo cual es necesario una tarjeta de desarrollo que permita cumplir con este requerimiento. A continuación, se procederá a realizar la selección de la tarjeta de desarrollo adecuada para el proyecto mediante la siguiente tabla de análisis comparativo.

Tabla 8

Análisis comparativo técnico para determinar la mejor tarjeta de desarrollo para visión artificial

Tecnología	Raspberry Pi 1 B+	Raspberry Pi 2 B	Raspberry Pi 3 B	Raspberry Pi Zero	Raspberry Pi Zero W
Precio	2	1	1	5	4
N° Cores	2	5	5	2	2
CPU Clock	2	3	5	4	4
RAM	4	5	5	4	4
USB	3	5	5	2	2
Wi-Fi	0	0	5	0	5
Bluetooth	0	0	5	0	5
GPIO	2	3	3	3	3

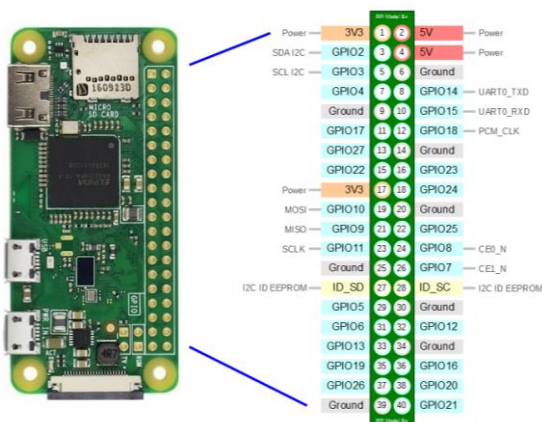
Tecnología	Raspberry Pi 1 B+	Raspberry Pi 2 B	Raspberry Pi 3 B	Raspberry Pi Zero	Raspberry Pi Zero W
UART	5	5	5	5	5
Cámara	5	5	5	5	5
Tamaño	2	3	3	5	5
Peso	3	3	3	5	5
Consumo	3	2	1	5	5
<b>Total</b>	<b>33</b>	<b>40</b>	<b>51</b>	<b>45</b>	<b>54</b>

*Nota:* En la tabla la ponderación de 5 es de mayor accesibilidad y 1 de menor accesibilidad.

Como se observa en la Tabla 8 la tarjeta a utilizar en el proyecto es la Raspberry Pi Zero W, esta tarjeta es un mini computador que posee las siguientes características: una CPU de un solo núcleo a 1 GHz para procesar diferentes operaciones, memoria RAM de 512MB compartida con la GPU, dos puertos USB OTG uno de alimentación y otro de datos, cabezal de 40 pines, conector de cámara CSI, chip inalámbrico Cy.press CYW43438 que proporciona conectividad LAN inalámbrica 802.11n en la banda de 2.4GHz y conectividad Bluetooth 4.0 que le permite conectarse a internet o a otros dispositivos inalámbricos fácilmente. El tamaño y peso de esta tarjeta son óptimos para no interferir en la estabilidad del robot humanoide.

**Figura 26**

*Raspberry Pi Zero W*

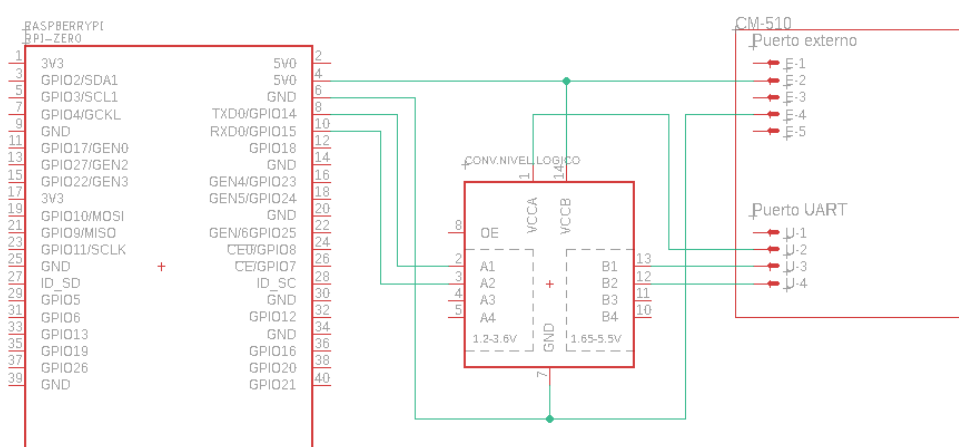


*Nota:* Recuperado de (<https://www.raspberrypi.org/>)

El diagrama electrónico para la comunicación entre la RPZW y el controlador CM-510 se muestra Figura 27. En el circuito se utilizó un convertidor de nivel lógico TTL bidireccional como dispositivo de protección, debido a que los pines de comunicación de la RPZW son a 3.3V y los pines de comunicación del CM-510 son a 5V.

**Figura 27**

*Diagrama electrónico entre RPZW y CM-510*



*Nota:* La figura muestra la forma de conexión entre la RPZW y CM-510

### **Actuadores**

Los actuadores utilizados en este proyecto son los servomotores DYNAMIXEL AX-12A incluidos en el kit Bioloid (Figura 28), estos son actuadores inteligentes que incorporan una caja reductora, un motor DC y un microcontrolador Atmega8 de la familiar AVR RISC Microchip. Tiene la capacidad de hacer el censado de la velocidad, temperatura, posición, voltaje y carga, además el algoritmo de control que utiliza le permite ajustar individualmente la posición en cada uno de los servomotores. El microcontrolador es el encargado de gestionar los sensores internos y el control de posición y velocidad, entiende 50 comandos tanto de lectura como de escritura que definen su funcionamiento.

**Figura 28**

*Servomotor DYNAMIXEL AX-12A*



*Nota:* Recuperado de (ROBOTIS, s.f.)

**Tabla 9**

*Especificaciones DYNAMIXEL AX-12A*

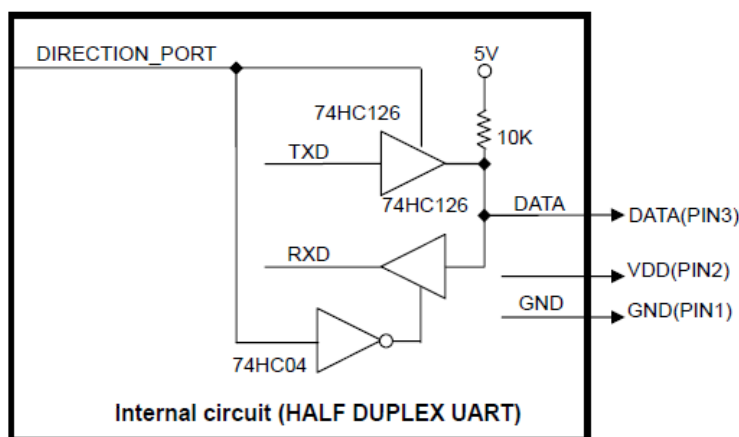
<b>Artículo</b>	<b>Especificaciones</b>
Velocidad de transmisión	7843 bps ~ 1Mbps
Resolución	0.29°
Grado de carrera	0° ~ 300°, giro sin fin
Peso	54.6 g
Relación de transmisión	254:1
Par de parada	1.5 N*m (a 12V, 1.5A)
Velocidad sin carga	59 rpm (12V)
Voltaje de entrada	9.0 ~ 12.0V
Señal de comando	Paquete digital
Tipo de protocolo	Comunicación serie asíncrona half duplex
Conexión física	Nivel TTL Multi Drop Bus
Carnet de identidad	0 ~ 253
Material	Platico de ingeniería

*Nota:* La tabla muestra las principales características de los actuadores AX-12A.

El DYNAMIXEL usa un protocolo de comunicación half dúplex UART para comunicarse con el controlador CM-510, la comunicación half dúplex es un protocolo de comunicación en serie en el que TxD y RxD no se pueden usar al mismo tiempo debido a que forman un solo bus de datos.

**Figura 29**

*Circuito interno half dúplex UART del controlador CM-510.*



*Nota:* Recuperado de (ROBOTIS, s.f.)

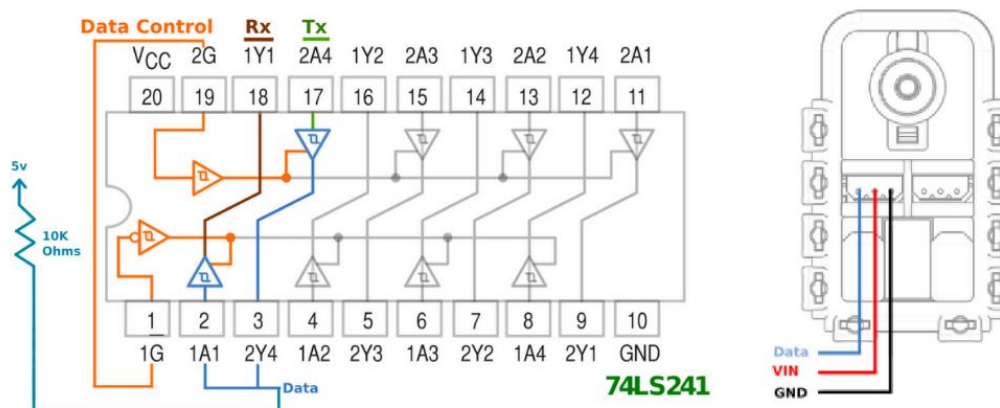
También existe la posibilidad de establecer la comunicación entre el DYNAMIXEL y la placa de desarrollo Tensy 3.6, haciendo uso de un buffer tri-estado, este se lo puede hacer con dos CI<sup>6</sup> el 74HC04 y el 74HC126 tal como recomienda ROBOTIS (2020) y los utiliza en sus controladores como se puede observar en la Figura 29, o se puede usar un solo circuito integrado 74LS241 que permite una conexión y utilización más sencilla, en la Figura 30 el Rx y Tx son los pines de comunicación UART de la Teensy 3.6, mientras Data Control es un pin digital que permite la activación de transmisión o recepción, y en color azul se muestra el bus de datos que va conectado a uno o varios DYNAMIXELS.

---

<sup>6</sup> CI – Circuito Integrado

Figura 30

Diagrama electrónico DYNAMIXEL con otros controladores



Nota: Recuperado de (<https://savageelectronics.com/blog/arduino-biblioteca-dynamixel>)

El CI 74LS241 es un buffer octal y controlador de línea diseñado para ser empleado como controlador de dirección de memorias, controladores de reloj y transmisores/receptores orientados al bus de datos de una placa de PC mejorada (MOTOROLA). En el Tabla 10 se tiene los rangos de operación óptimo de este CI.

Tabla 10

Especificaciones técnicas del CI 74LS241

Símbolo	Parámetro	Mínimo	Típico	Máximo	Unidad
$V_{CC}$	Voltaje de suministro	4.75	5	5.25	V
$T_A$	Rango de temperatura ambiente de funcionamiento	0	25	70	°C
$I_{OH}$	Corriente de salida – alta			15	mA
$I_{OL}$	Corriente de salida – bajo			24	mA

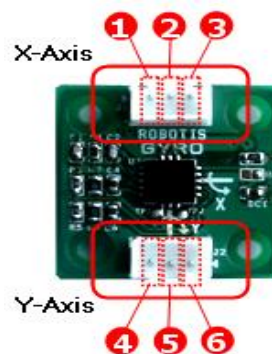
Nota: La tabla muestra los requerimientos eléctricos para su correcto funcionamiento.

## Sensores

**GS-12.** El sensor giroscopio GS-12 incluido en el kit sirve para medir la velocidad angular en dos ejes, permite determinar de qué lado está inclinado el robot o si está obteniendo más potencia, puede usarse para mantener el equilibrio del robot humanoide u otras aplicaciones de movimiento. El rango de cálculo de velocidad angular es de  $-300^\circ/\text{s}$  a  $300^\circ/\text{s}$ , con una sensibilidad de  $3.33\text{mV}/\text{dps}$  a un voltaje de suministro de 4.5 a 5.5V. La distribución de pines se muestra en la Figura 31, donde los pines 1 y 6 son las señales analógicas de la velocidad angular en el eje X e Y respectivamente, los pines 2 y 5 son GND y los pines 3 y 4 son VCC.

**Figura 31**

*Composición de los pines del GS-12*



*Nota:* Recuperado de (ROBOTIS, s.f.)

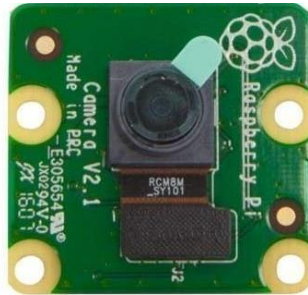
**Raspberry Pi Camera Module V2.** En el presente trabajo de titulación se decidió usar la Raspberry Pi Camera Module V2, se lo puede utilizar para tomar videos de alta definición, así como fotografías fijas. Este módulo tiene incorporado con un sensor de imagen Sony IMX219 de alta calidad con resolución fija de 8 megapíxeles y con un lente de enfoque fijo, que se conecta a través de uno de los pequeños enchufes en la superficie superior de la placa, utilizando la



interfaz CSI, la resolución del sensor es de 3280x2464 píxeles, el campo de visión horizontal es de 62.2° y el vertical de 48.8°.

### Figura 32

*Raspberry Pi Camera Module V2*



*Nota:* Recuperado de (<https://www.raspberrypi.org/>)

Para el presente trabajo de titulación se utilizó la carcasa propia de la RPZW la cual viene con tres tapas diferentes y una de ellas permite sujetar la Raspberry Pi Camera Module V2 como se observa en la Figura 33, brindando la protección adecuada a la cámara para no sufrir daños y se mantenga fija durante su funcionamiento.

### Figura 33

*Carcasa RPZW para usarla con Raspberry Pi Camera Module V2*



*Nota:* Recuperado de (<https://www.raspberrypi.org/>)

**Sensor de color.** La utilización de un sensor de color en el proyecto, es para que el robot móvil pueda realizar la validación de color del objeto de interés. Razón por la cual se realizó la selección del sensor de color que más se ajusta con las necesidades del proyecto, en base a la siguiente tabla de análisis comparativo.

**Tabla 11**

*Análisis comparativo técnico para la selección del mejor sensor de color*

Tecnología	Precio	Tamaño	Robustez	Interface de comunicación	Forma de programación	Total
VEML6040	3	5	3	4	4	19
TCS34725	3	4	5	4	5	21
TCS3200	5	2	1	2	3	13

*Nota:* En la tabla la ponderación de 5 es de mayor accesibilidad y 1 de menor accesibilidad.

En la Tabla 11 se puede observar que el sensor de color a utilizar es el TCS34725 (Figura 34), este sensor es un sensor de color digital tiene incluido sensores RGB y de luz blanca que le permite medir los valor de color de un objeto o luz. También cuenta con un filtro bloqueador de infrarrojo integrado, además viene incluido con un regulador de voltaje que le permite ser energizado ya se con 3.3V o con 5V. Este sensor es compatible con cualquier microcontrolador con pines I2C.

**Figura 34**

*Sensor de color TCS34725*



*Nota:* Recuperado de (www.google.com)

### **Módulos de comunicación**

**ZIG-100/110A.** Los módulos de comunicación inalámbrica incluidos en el kit son el ZIG-100 y el ZIG-110A, estos son módulos de comunicación inalámbrica de corta distancia que utilizan la capa física de Zigbee. Estos módulos proporcionan funciones iguales, están diseñados para reemplazar la comunicación por cable y la única diferencia entre los mismos es el dispositivo con el que va a interactuar mediante comunicación serial UART. El ZIG-100 puede conectarse al CM-5, ZIG2Serial, RC-100 mientras que el ZIG-110A puede conectarse al CM-100, CM-510, CM-530, CM-700 u otras tarjetas de desarrollo.

### **Figura 35**

*Módulos de comunicación ZIG-100 (izq) y ZIG-110A (der)*



*Nota:* Recuperado de (ROBOTIS, s.f.)

La frecuencia de operación de ambos módulos es de 2.4GHz con una banda ancha máxima de 250 kbps a un suministro de voltaje entre 2.7V y 3.6V, la velocidad de transmisión predeterminada en ambos es de 57600 bps.

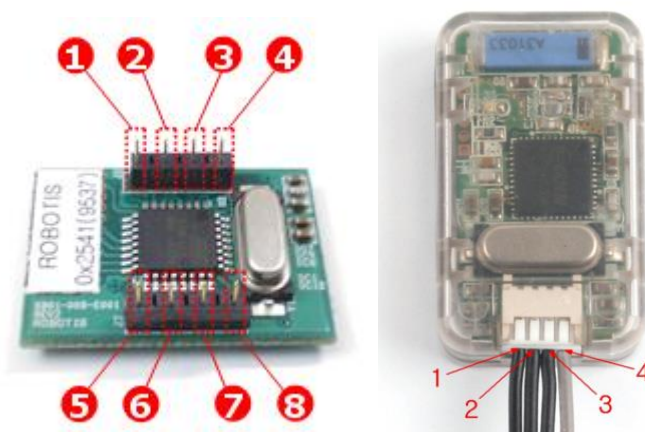
Dependiendo de la configuración del ZIG-100 y ZIG-110A, se puede realizar comunicación 1:1, comunicación 1: N o N: N. Para el presente trabajo de titulación se realizará

una comunicación 1:1 entre el módulo ZIG-100 y ZIG-110A, para esto se configuró la ID inalámbrica del uno en el otro y viceversa.

En la Figura 36 se muestra la distribución de pines de los módulos ZIG-100/110A, en el módulo ZIG-100 el pin 1-GND, pin 2-VCC (2.7 ~ 3.6V), pin 3-LED (estado de conexión), pin 4-RESET, pin 5-RXD, pin 6-TXD, pin 7 y pin 8 son utilizados para seleccionar el canal de transmisión; en el módulo ZIG-110A el pin 1-RXD, pin 2-TXD, pin 3-VCC (2.7 ~ 3.6V) y el pin 4-GND.

**Figura 36**

*Distribución de pines ZIG-100 (izq) y ZIG-110A (der)*



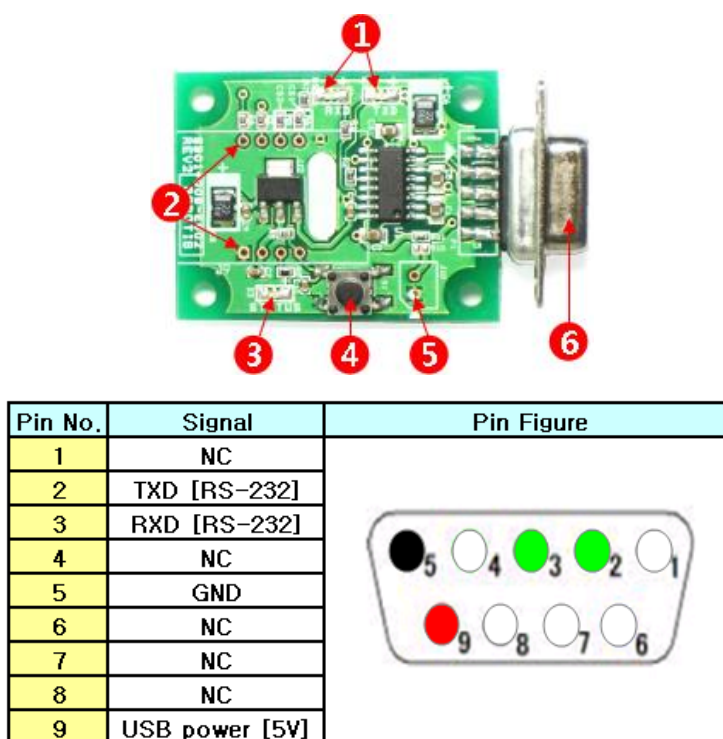
*Nota:* Recuperado de (ROBOTIS, s.f.)

**ZIG2Serial.** Otro módulo incluido en el kit es el ZIG2Serial que permite conectar un ZIG-100 a la computadora a través de señales RS232, ya sea a los puertos serie o USB mediante un dispositivo USB2Serial o USB2DYNAMIXEL. Este módulo se encuentra compuesto por los elementos que se muestran en la Figura 37-a, (1) hay dos leds que muestran el estado de comunicación RXD y TXD, (2) espacio para conectar ZIG-100, (3) led que muestra el estado de conexión con otro dispositivo, donde si parpadea significa que no ha encontrado otro dispositivo para la conexión y si permanece encendido significa que encontró otro dispositivo y que la

conexión esta lista, (4) botón de reinicio, (5) conector de alimentación externo a 5V, (6) conector RS232 hembra para conectar a la PC y su distribución de pines se muestra en la Figura 37-b.

**Figura 37**

(a) Módulo ZIG2Serial (arriba), (b) Conector RS232 hembra del módulo ZIG2Serial (abajo)



*Nota:* Recuperado de (ROBOTIS, s.f.)

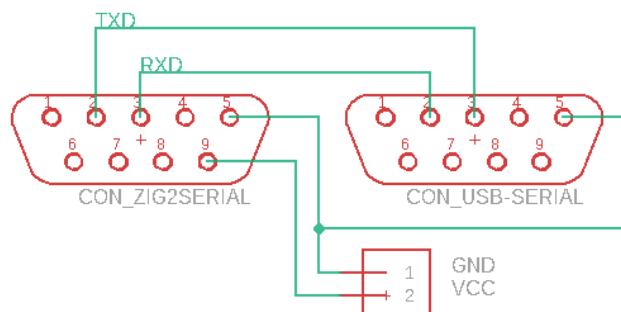
En el presente trabajo de titulación se presentó una limitación al momento de usar el ZIG2Serial debido a que no se tiene un USB2DYNAMIXEL para realizar la conexión directa hacia la PC. Como se sabía las conexiones que son necesarias para que funcione el ZIG2Serial lo cual se ha descrito anteriormente, se realizó el diseño de un pequeño circuito.

El diseño del circuito se desarrolló en Autodesk Eagle, este permite la conexión entre el módulo ZIG2Serial con conector DB9 RS-232 hembra y un cable convertidor de USB macho a

serial DB9 RS-232 hembra, usando 2 conectores DB-9 macho y una fuente de alimentación externa de 5V, el esquemático de conexión se detalla en la Figura 38 y la PCB<sup>7</sup> implementada en la Figura 39.

**Figura 38**

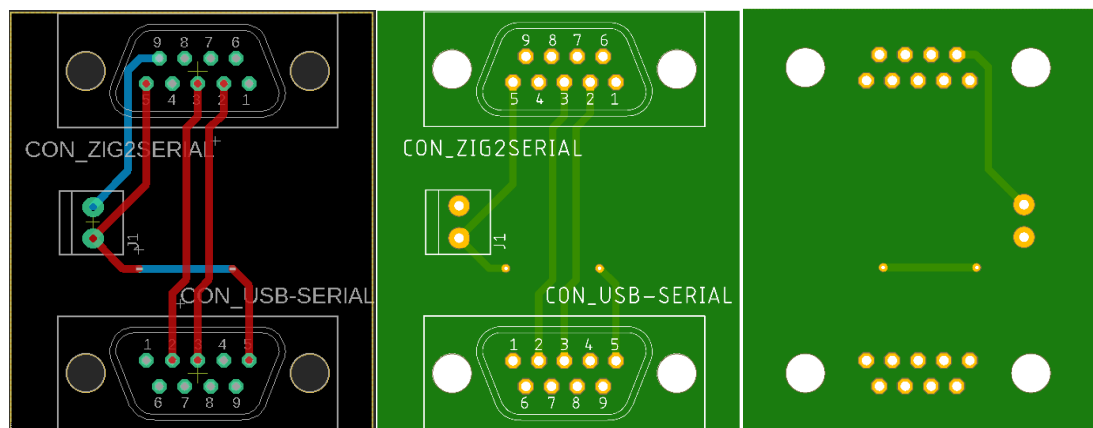
*Diagrama electrónico de conexión ZIG2Serial – USB a Serial*



*Nota:* La figura muestra la conexión del ZIG2Serial para la programación del CM-510.

**Figura 39**

*Diseño placa PCB*



*Nota:* La figura muestra una representación real del circuito final en placa PCB.

<sup>7</sup> PCB – Placa de circuito impreso (del inglés, “Printed Circuit Board”)

La implementación final entre el módulo ZIG2Serial y el cable convertidor USB a Serial que se utilizó en el proyecto, se puede observar en la Figura 40.

**Figura 40**

*Implementación final conexión ZIG2Serial – USB a Serial*



*Nota:* Esta implementación se la debió realizar debido a los inconvenientes encontrados.

### **Baterías**

La adquisición de las nuevas baterías se hizo en base a un análisis de carga de cada uno de los elementos implementados en los robots, considerando los máximos valores de consumo de corriente. Luego tras un análisis comparativo técnico se selecciona el modelo adecuado de baterías disponibles en el mercado.

**Tabla 12**

*Análisis de carga para la selección de la batería del robot humanoide*

<b>Cantidad</b>	<b>Elemento</b>	<b>Voltaje de operación</b>	<b>Consumo de corriente individual</b>	<b>Consumo de corriente total</b>
20	DYNAMIXEL AX-12A	9 – 12 V	70mA	1400mA
1	Sensor GS-12	4.5 – 5.5 V	25mA	7mA
1	Raspberry Pi Zero W	5 V	250mA	350mA
1	Raspberry Pi Camera Module V2	3.3 V	200mA	200mA
1	Convertidor de nivel lógico bidireccional TTL	2.2 – 12 V	15mA	8mA
1	Controlador CM-510	6.5 – 15 V	100mA	100mA

*Nota:* En la tabla se muestra las especificaciones de operación encontradas de cada elemento.

**Tabla 13**

*Análisis de carga para la selección de la batería para el robot móvil*

<b>Cantidad</b>	<b>Elemento</b>	<b>Voltaje de operación</b>	<b>Consumo de corriente individual</b>	<b>Consumo de corriente total</b>
6	DYNAMIXEL AX-12A	9 – 12 V	70mA	420mA
1	Teensy 3.6	3.6 – 6 V	250mA	250mA
1	Módulo ZIG-110A	2.7 – 3.6 V	30mA	30mA
1	CI 74LS241	4.75 – 5.25	24mA	24mA
1	Convertidor DC-DC MP1584EN	4.5 – 28 V	30mA	30mA

*Nota:* En la tabla se muestra las especificaciones de operación encontradas de cada elemento.

**Tabla 14**

*Análisis comparativo técnico para determinar el mejor modelo de baterías LiPo*

<b>Modelo</b>	<b>Voltaje</b>	<b>Capacidad</b>	<b>Tasa de descarga</b>	<b>Precio</b>	<b>Conector</b>	<b>Existencia</b>	<b>Total</b>
Turnigy	5	3	5	5	5	4	27
ZIPPY	1	1	1	3	2	1	9
Turnigy nano-tech	5	5	5	4	5	5	29
Turnigy Graphene	1	2	3	1	1	2	10

*Nota:* En la tabla la ponderación de 5 es de mayor accesibilidad y 1 de menor accesibilidad.

Como se aprecia en la Tabla 14 las baterías LiPo a utilizar en el proyecto son del modelo Turnigy Nano-Tech. Procediendo con el análisis de la Tabla 12 se deduce que, el consumo de corriente total requerido en el robot humanoide es aproximadamente 2065mA con un voltaje máximo requerido de 15V. También con el análisis de la Tabla 13 se deduce que el consumo de corriente total requerido en el robot móvil es aproximadamente 754mA con un voltaje máximo requerido de 12V.



Así para el robot humanoide se eligió la batería LiPo Turnigy nano-tech con capacidad de 2200mAh, voltaje de 11.1V en 3 celdas, tasa de descarga de 25C constante / 50C interrumpida y conector Xt-60. En cuanto al robot móvil se eligió la batería LiPo Turnigy nano-tech con capacidad de 1000mAh, voltaje de 11.1V, tasa de descarga 25-50C y conector Xt-60.

**Figura 41**

*Baterías LiPo de 11.1V (a) 2200mAh, (b) 1000mAh*



*Nota:* El uso de baterías permite que los robots se muevan de manera independiente.

Utilizando la siguiente ecuación se estima el tiempo de autonomía de las baterías, para determinar el tiempo de operación de cada uno de los robots dentro del MAS y así tener una mejor planificación del sistema clasificatorio.

$$\text{Tiempo de autonomía} = \frac{\text{Capacidad de la batería}}{\text{Consumo total del sistema}} \quad (1)$$

Reemplazando los datos obtenidos para el robot humanoide se tiene:

$$\text{Tiempo de autonomía} = \frac{2200\text{mAh}}{2065\text{mA}} = 1.07\text{h} = 64 \text{ min} \quad (2)$$

Reemplazando los datos obtenidos para el robot móvil se tiene:

$$\text{Tiempo de autonomía} = \frac{1000\text{mAh}}{754\text{mA}} = 1.33\text{h} = 80 \text{ min} \quad (3)$$

Con los resultados obtenidos cada robot tiene una autonomía de operación de por lo menos 60 minutos dentro del MAS.

Debido a que el tamaño de las baterías adquiridas es distinto al de las baterías incluidas en los kits, la limitación que se tuvo es que las nuevas baterías no alcanzaron en las carcasas que vienen en los kits, por lo que toco adecuarlas en cada uno de los robots. En el robot humanoide se diseñó una carcasa protectora en acrílico negro de 2mm de espesor con dimensiones de 118x47x32mm el cual se ubicó en la parte posterior del robot humanoide, mientras que en el robot móvil se la ubicó en la parte interna de la carcasa del robot.

### ***Robot Humanoide***

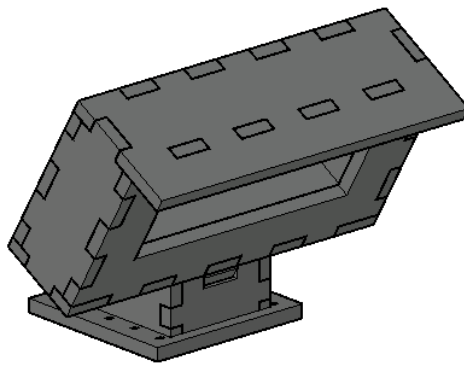
El robot humanoide como requerimiento para el proyecto es que pueda sostener objetos y como se puede observar en el robot de referencia de la Figura 19 este modelo no consta de un sistema que le permita sostener objetos, por tal motivo se agregó un sistema de agarre en las extremidades superiores del robot. ROBOTIS (2020) muestra a los usuarios los materiales necesarios para ensamblar pinzas y agregar el sistema de agarre a los robots humanoides, pero detallan que para usar las pinzas se deben ensamblar un humanoide tipo-B o tipo-C debido a que se necesitan dos actuadores para ensamblar las pinzas.

No se optó por cambiar el modelo del robot humanoide para agregar las pinzas requeridas, debido a que se cuenta con 8 actuadores útiles aparte de los 18 actuadores necesarios para el ensamblaje del humanoide y dado a que solo se necesitan 6 actuadores para el ensamblaje del robot móvil quedan 2 actuadores libre, los mismo que se van a utilizar para ensamblar las pinzas formando así un robot humanoide de 20 GDL.

La modificación realizada en el ensamblaje del robot humanoide tipo-A es en la parte de la cabeza del robot, se decidió no poner las piezas en forma de cabeza y lo que se hizo es un diseño en madera MDF<sup>8</sup> de 3mm de un soporte para la RPZW con las dimensiones que se observa en la Figura 43.

#### **Figura 42**

*Modelo 3D soporte RPZW*



*Nota:* La inclinación de soporte es para observar los objetos que se encuentra debajo del robot.

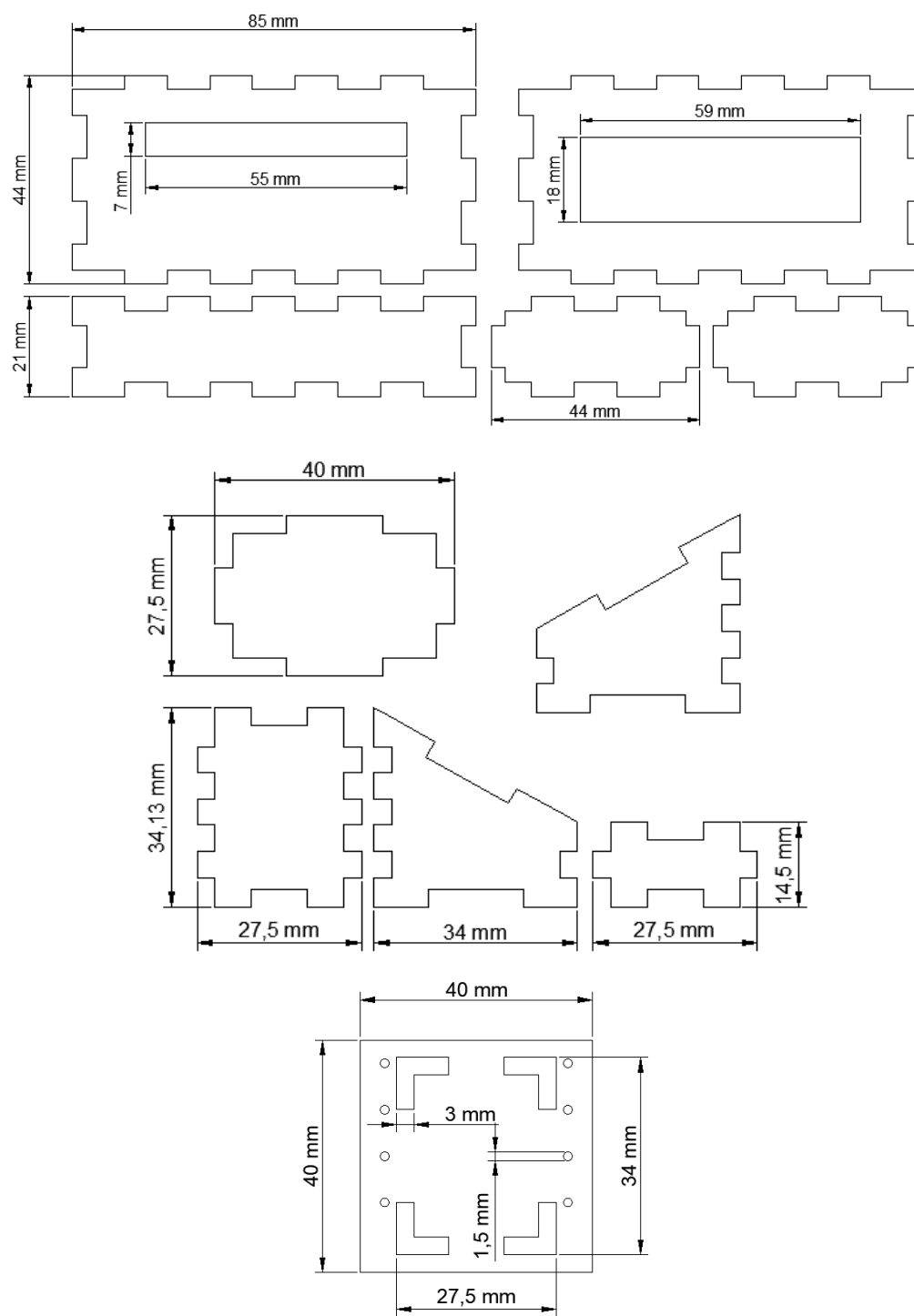
En la Figura 42 se muestra la representación en 3D del soporte diseñado para el robot humanoide. Finalmente en la Figura 44 se muestra el robot humanoide con las modificaciones y los elementos adicionales implementados, que permiten que el robot humanoide cumpla con los requerimientos para el proyecto.

---

<sup>8</sup> Medium Density Fibreboard (Tablero de fibras de densidad media)

Figura 43

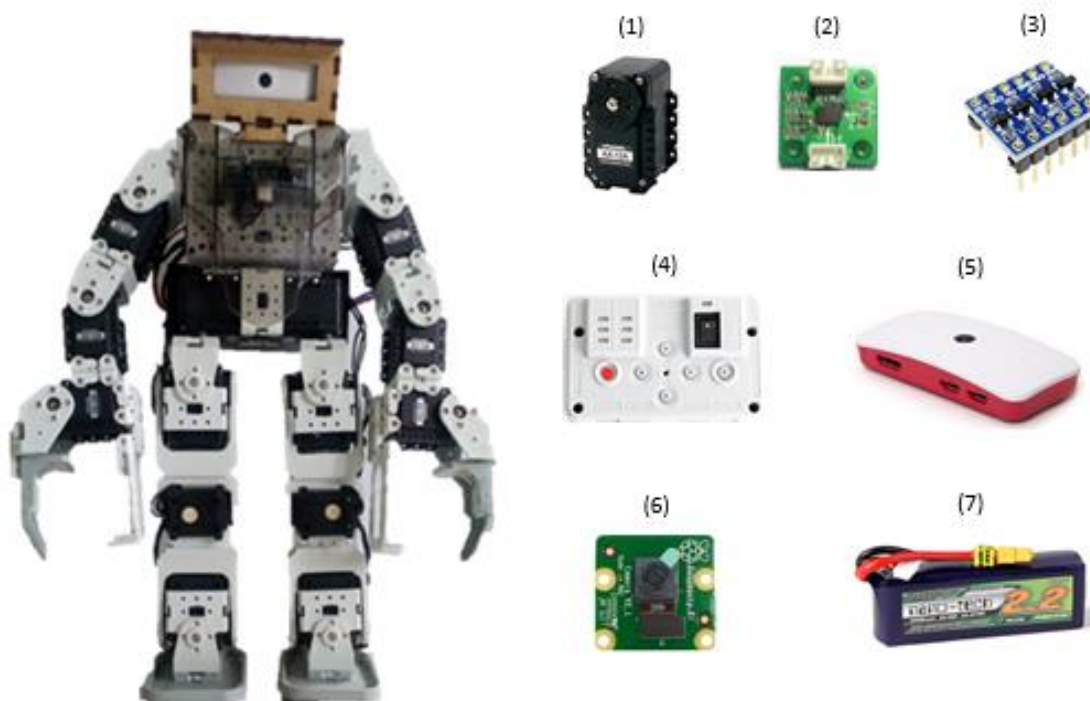
Dimensiones del soporte diseñado en AutoCAD para la RPZW



Nota: Diseño realizado por el autor.

**Figura 44**

*Robot humanoide incorporado con: (1) 20 actuadores AX-12A. (2) 1 sensor GS-12. (3) 1 convertidor de nivel lógico TTL bidireccional. (4) 1 controlador CM-510. (5) 1 tarjeta de desarrollo RPZW. (6) 1 Raspberry Pi Camera Module V2 (7) 1 batería LiPo 11.1V 2200mAh. También se usó las carcasas diseñadas y piezas del kit.*



*Nota:* Realizado por el autor.

### **Robot Móvil**

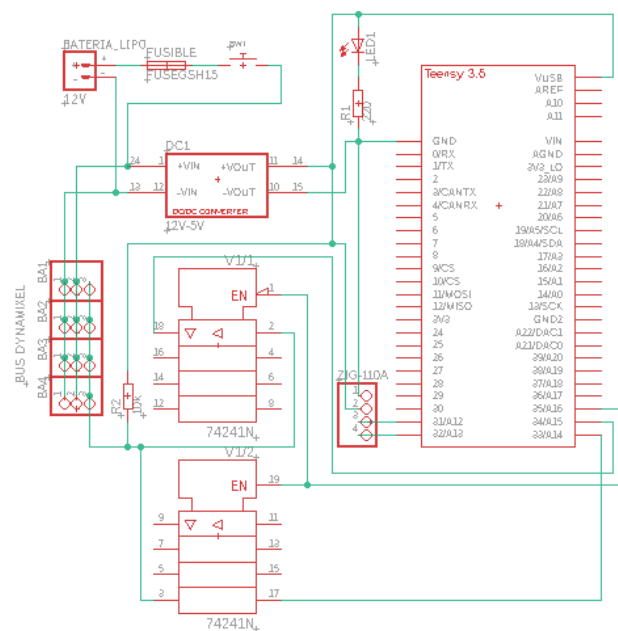
Como se mencionó anteriormente el robot móvil se encuentra conformado por 6 actuadores, los mismo que se encuentran distribuidos de la siguiente manera: 4 actuadores para construir un robot móvil con sistema de locomoción tipo skid steer y los 2 actuadores restantes para añadir al robot móvil una pinza en la parte delantera, lo que le permitirá sujetar, trasladar y dejar cosas de un lugar a otro. El controlador usado para el robot móvil es la Teensy 3.6, la

razón por la que se decidió cambiar el controlador CM-510 en el robot móvil, es que este controlador posee tecnología antigua y con este cambio se pretende repotenciar el uso de nuevas tecnologías para el desarrollo de diferentes aplicaciones.

En la Figura 45 se muestra el diagrama electrónico diseñado para el funcionamiento del robot móvil, el diseño se realizó con elementos de protección tanto para la parte de control como para la parte de potencia, haciendo uso de un fusible, un interruptor y un MP1584EN que es un convertidor de voltaje DC-DC Step-Down de 3A.

**Figura 45**

*Diagrama electrónico del robot móvil*



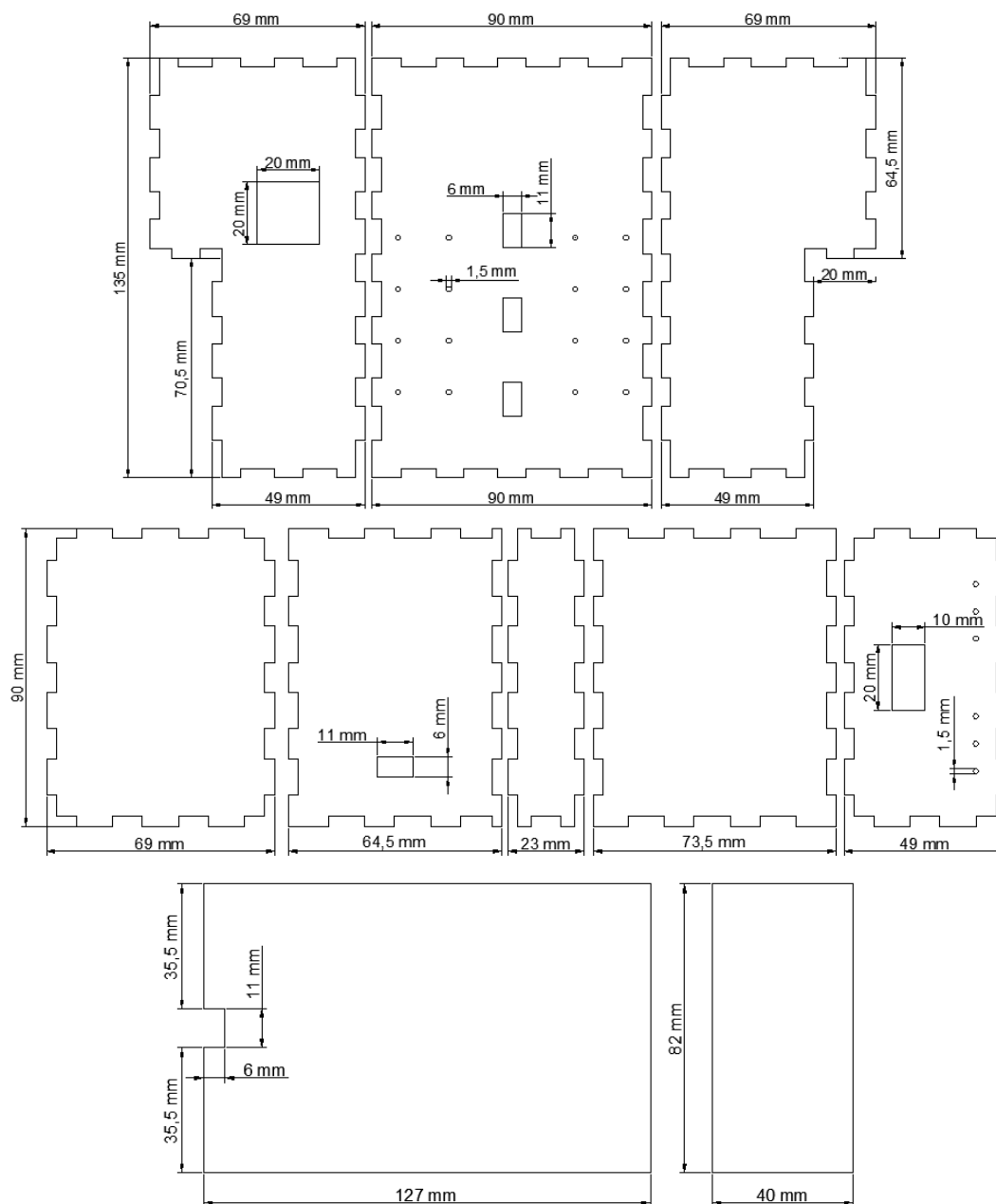
*Nota:* Realizado por el autor.

Para la estructura del robot móvil se hizo un diseño en madera MDF de 3mm de espesor con las dimensiones que se detallan en la Figura 46. Para el ensamblaje del robot móvil se tomó como referencia ciertas partes del robot de la Figura 20 y se las acopló a la estructura diseñada.

También se realizó la impresión en 3D del conector de los actuadores AX-12A, como elemento de protección.

**Figura 46**

*Dimensiones de la estructura del robot móvil*

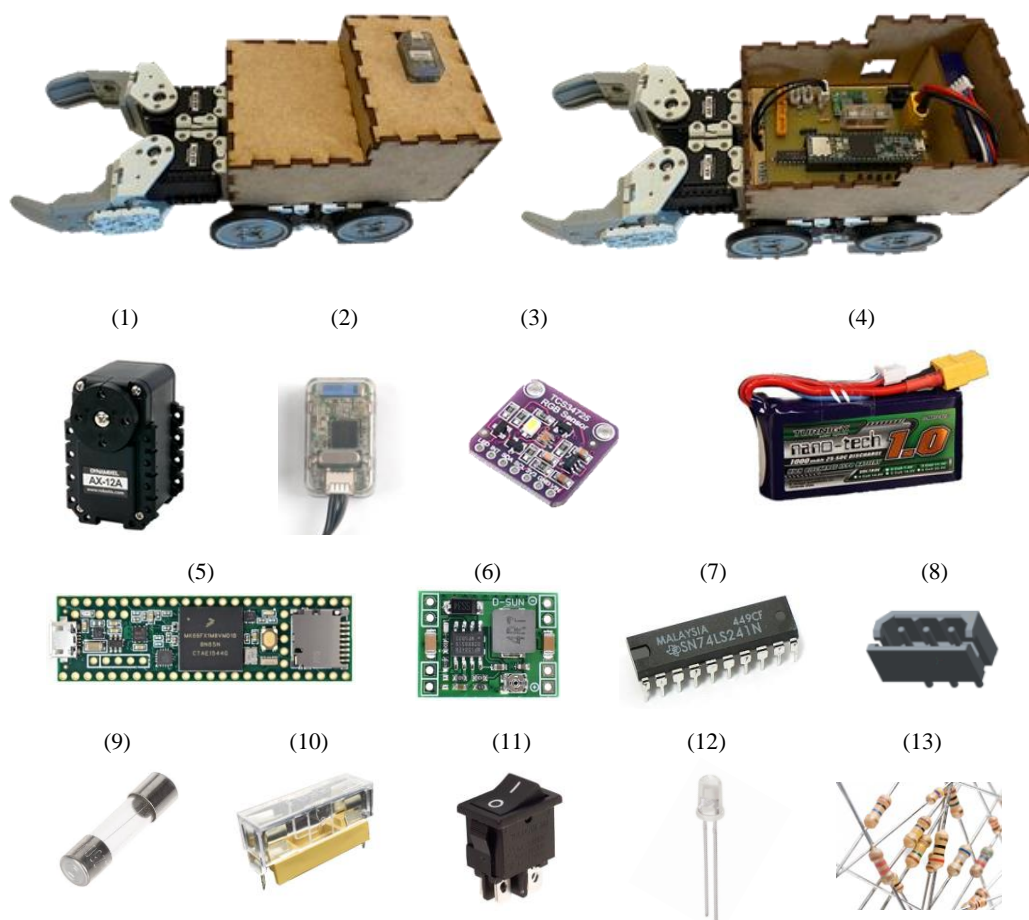


*Nota:* Realizado por el autor.

Finalmente en la Figura 47 se muestra el robot móvil y los elementos con los que encuentra implementado, y que le permitirá cumplir con los requerimientos dentro del proyecto.

#### Figura 47

Robot móvil incorporado con: (1) 6 actuadores AX-12A. (2) ZIG-110A. (3) Sensor TCS34725. (4) Batería LiPo 11.1V 1000mAh. (5) Teensy 3.6. (6) Convertidor DC-DC MP1584EN. (7) CI 74LS241. (8) 6 molex de 3-pins. (9) Fusible de 1A. (10) Portafusible. (11) Interruptor. (12) Led. (13) 2 resistencias.



*Nota:* Realizado por el autor.



### ***Celda de trabajo***

La estructura de la celda es de aproximadamente de 1.2 metros cuadrado, espacio necesario para que los robots puedan circular con normalidad. El material elegido para la fabricación del piso y las paredes de la celda es madera triplex de 5mm de espesor, con una base metálica que brinda la resistencia necesaria para no sufrir ningún tipo de daño producto de las posibles caídas que puedan darse en el robot humanoide. El techo de la celda cuenta con una estructura metálica desarmable, que permite un fácil transporte. La computadora se ubicará en una mesa a lado de la celda.

Las dimensiones del piso son de 120x100 cm, las paredes largas de 120x10 cm y las paredes pequeñas de 100x10 cm. La base metálica le da a la celda una altura de 10 cm desde el suelo con la finalidad de nivelar la celda, mientras que el techo de la celda se encuentra a una altura de 170cm desde el piso de la celda.

### **Figura 48**

#### *Celda de trabajo*



- 1) Estructura
- 2) Objetos de color
- 3) Sistema de iluminación
- 4) Cámara
- 5) Computadora
- 6) Modulo ZIG2SERIAL

*Nota:* Realizado por el autor.

La altura del techo de la celda viene dada por la cámara, ya que a esa altura el campo de visión de la cámara provee una vista de planta superior de toda la celda de trabajo. También se tomó en cuenta el sistema de iluminación dentro de la celda, para lo cual se va a iluminar artificialmente usando focos led ubicados en las esquinas del techo, aplicando la técnica direccional de iluminación.

### ***Objetos de color***

Los objetos que se van a utilizar para la clasificación son cilindros de cartón prensado con dimensiones de 8cm de diámetro y 9cm de altura, el material y las dimensiones de las cajas fueron elegidas para que tanto el robot humanoide como el robot móvil puedan sujetarlas y trasladarlas de un lugar a otro con facilidad. Además, se encuentran recubiertas de fomix que no genera brillo y permite que la detección con la cámara sea adecuada. Los colores elegidos para los objetos de interés a clasificar son el amarillo y el azul, mientras que el color para los obstáculos el naranja.

### **Figura 49**

*Objetos de varios colores*



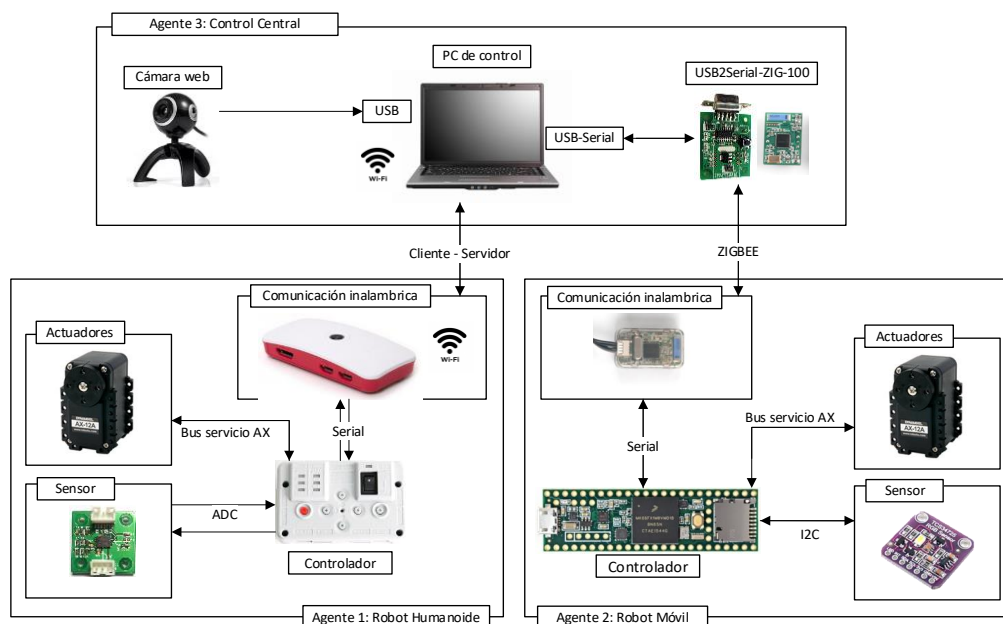
*Nota:* Realizado por el autor.

## Descripción del software del sistema clasificatorio

El software del sistema clasificatorio se enfoca en el diseño e implementación de un Sistema Multi-Agente, el mismo que se encuentra conformado por un sistema de robots heterogéneos: un robot móvil y un robot humanoide. Cada robot es considerado como un agente del MAS. El computador se encarga de procesar el stream de video de la cámara y se convierte en el tercer agente del MAS. La arquitectura de control con la que va a ser diseñada es la arquitectura de coordinación centralizada, en donde el agente 3 es el nodo central y se encarga del control y transmisión de datos a los demás agentes tal como se puede observar en la Figura 50. Primero se va a explicar los softwares, lenguajes de programación y protocolos utilizados y luego se explicará detalladamente los algoritmos utilizados para el funcionamiento del MAS.

**Figura 50**

*Arquitectura de coordinación centralizada de la celda colaborativa robotizada*



*Nota:* Arquitectura realizada por el autor.

### ***Softwares de sistema***

En el presente trabajo de titulación se utilizan dos sistemas operativos Microsoft Windows 10 y Raspbian, el primero es el sistema operativo con el que cuenta la computadora portátil y el segundo es el sistema operativo oficial para todos los modelos de Raspberry Pi, razón por la cual se la descargó e instaló en la RPZW.

### ***Lenguajes de programación***

Para el desarrollo de los algoritmos del MAS se utilizaron dos lenguajes de programación: el primero es C++ que es un lenguaje de programación que permite la manipulación de objetos, este surgió como la continuación y ampliación del C; y el segundo es Python que es un lenguaje de programación multiparadigma, ya que soporta la programación orientada a objetos, programación imperativa y programación funcional, lo que lo hace un lenguaje de programación dinámico y multiplataforma.

### ***Softwares de programación***

**R+ Task 3.0.** El controlador CM-510 necesita dos tipos de programación: la primera es una programación de los valores de posición y velocidad de cada uno de los actuadores, en otras palabras, todos estos datos se convierten en un movimiento válido el cual puede realizar el robot y la segunda es una programación de actividades relevantes para cumplir una tarea, teniendo en cuenta los datos recibidos del usuario y los datos propios del robot.

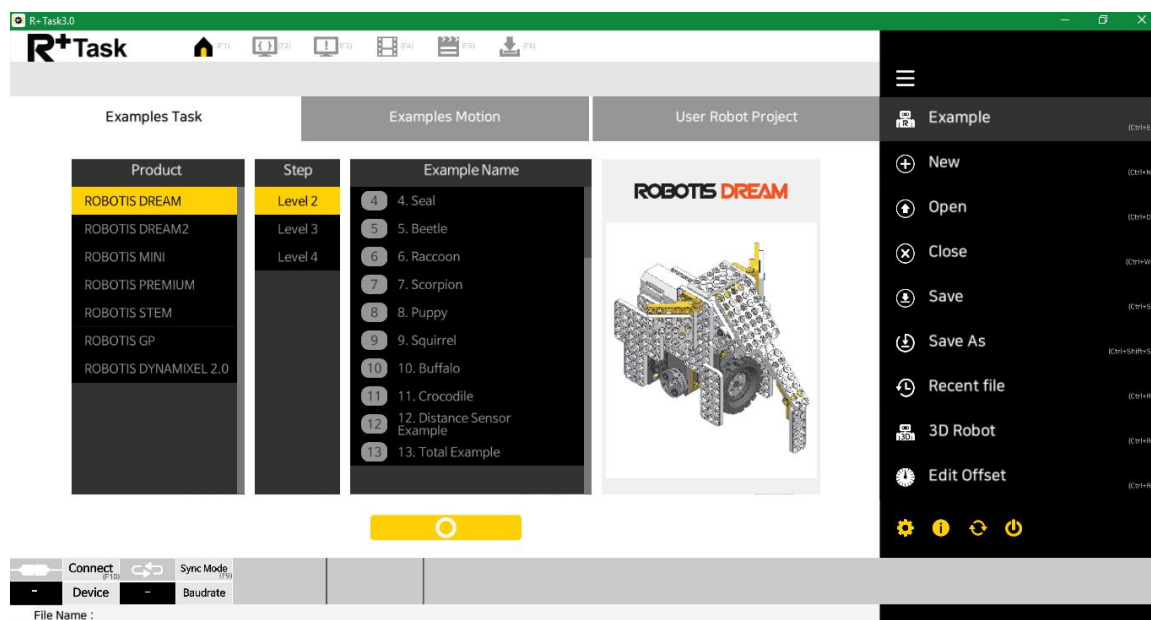
En el presente trabajo de titulación para realizar la programación del controlador CM-510 se optó por usar el software de la compañía fabricante, el cual ha desarrollado y mejorado el software que permite programarlo, la última versión de este software y sobre el cual se

trabajo es R+ Task 3.0, que presenta una interfaz amigable para la gestión y programación de controladores de robots como se observa en la Figura 51.

Este software cuenta con su propio lenguaje de programación de tareas, por lo que se optó por aprender el funcionamiento del conjunto de instrucciones y variables propias del controlador CM-510 que se encuentran en el software.

**Figura 51**

*Programa R+ Task 3.0*

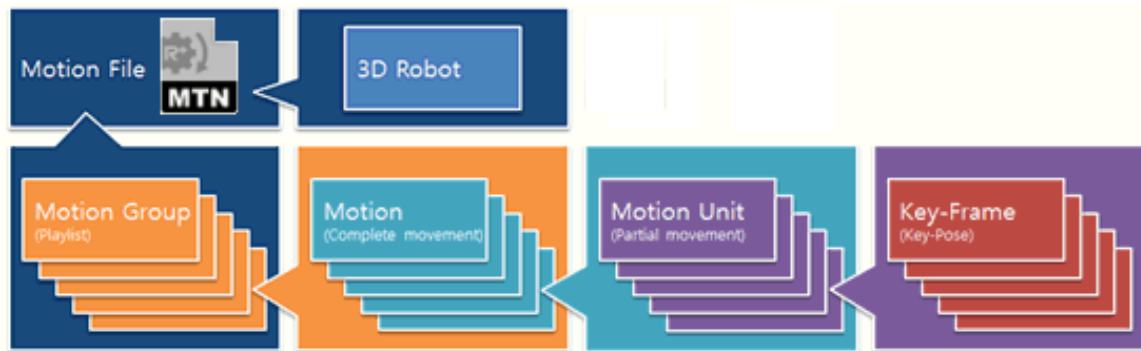


*Nota:* Recuperado del software R+ Task3.0.

La programación de movimientos es un archivo que incluye datos de posición de cada articulación para mover el robot, la estructura de datos que tiene estos archivos se muestra en la Figura 52.

**Figura 52**

*Estructura de datos (Programación de movimientos)*



*Nota:* Recuperado de (ROBOTIS, s.f.)

- Key-Frame: es la combinación de la posición que se quiere mostrar y el tiempo en el que se debe mostrar a lo largo de la línea de tiempo, los parámetros que se puede modificar son la posición, el nivel de flexibilidad y el nivel de fuerza de inercia de cada actuador.
- Motion Unit: es un grupo de Key-Frames ubicados a lo largo de la línea de tiempo que su función es mostrar un movimiento continuo, aquí se determina la velocidad de cambio de los actuadores entre una y otra posición.
- Motion: es un grupo de Motion Units, se genera secuencias de Motion Units para representar movimientos largos, se puede modificar la velocidad de reproducción de cada Motion y la cantidad de veces que se repite un Motion Unit.
- El Motion Group: son todos los Motions realizados y listos para ser descargados al robot, aquí se verifica el espacio disponible en el controlador para realizar esta acción de descarga.

Aparte este software cuenta con un simulador del Robot en 3D, permitiendo verificar y reproducir los movimientos que se están creando.

**Arduino IDE.** El controlador Teensy 3.6 se encuentra basado en un microcontrolador en el que se puede grabar instrucciones, las mismas que se escriben con el lenguaje de programación que ofrece el entorno Arduino IDE<sup>9</sup>. El entorno Arduino IDE es una plataforma de desarrollo basada en una tarjeta electrónica de hardware libre, las instrucciones se encuentran basadas en el lenguaje C++ que permiten la creación de programas para interactuar con los circuitos de la placa. Este software puede ser instalado y ejecutado en cualquier sistema operativo, pero el mismo no viene con soporte para Teensy, por tal razón se tiene que instalar un complemento llamado Teensyduino, que agrega los archivos necesarios de Teensy al software Arduino.

**IDLE (Python GUI).** IDLE es un entorno gráfico de desarrollo elemental que permite editar y ejecutar programas en Python. En Windows, se distribuye junto con el intérprete de Python, es decir, al momento de la instalación también se instala IDLE. Mientras que, en Linux, IDLE se distribuye como una aplicación separada que se puede instalar desde los repositorios de cada distribución, en la distribución de Raspbian para editar y ejecutar programas en Python se usa el entorno gráfico de Thonny, en la instalación de sistema operativo Raspbian ya viene preinstalado Thonny, el mismo que viene integrado con Python.

### ***Protocolos de comunicación***

**Protocolo 1.0 DYNAMIXEL.** Este es un protocolo que permite la comunicación bidireccional entre el controlador principal y el DYNAMIXEL, enviando y recibiendo paquetes de datos. Los paquetes son de dos tipos: paquete de instrucciones que envía el controlador principal al

---

<sup>9</sup> IDE (Entornos de Desarrollo Integrados)

DYNAMIXEL para controlarlo, y paquete de estado que es la respuesta del DYNAMIXEL al controlador principal.

El DYNAMIXEL AX-12A tiene implementada una tabla de control de estructura de datos, esta tabla consta de múltiples campos de datos. Los paquetes de instrucciones pueden ser tipo lectura o escritura, ya sea para verificar el estado del dispositivo o para controlarlo. Para leer o escribir datos se debe designar una dirección específica en el paquete de instrucciones.

La tabla de control se divide en dos áreas: RAM y EEPROM. Los datos en la RAM se restablecen a los valores por defecto cuando se restablece la alimentación (volátil), mientras que los datos en la EEPROM se mantienen (no volátil). El acceso a la tabla de control tiene dos propiedades "RW" que significa permiso de acceso de lectura y escritura y "R" que significa permiso de acceso solo de lectura.

**Tabla 15**

*Tabla de control del área EEPROM del DYNAMIXEL AX-12A*

Dirección	Tamaño (Byte)	Tipo de dato	Descripción	Acceso	Valor Inicial
0	2	Número de modelo	Número de modelo	R	12
2	1	Versión de firmware	Versión de firmware	R	-
3	1	Carne de identidad	ID del DYNAMIXEL	RW	1
4	1	Velocidad de transmisión	Velocidad de comunicación	RW	1
5	1	Tiempo de retraso de retorno	Tiempo de retraso de respuesta	RW	250
6	2	Límite de ángulo CW	Límite de ángulo de sentido horario	RW	0
8	2	Límite de ángulo CCW	Límite de ángulo de sentido anti horario	RW	1023
11	1	Límite de temperatura	Límite máximo de temperatura interna	RW	70
12	1	Límite voltaje mínimo	Límite mínimo de voltaje de entrada	RW	60



Dirección	Tamaño (Byte)	Tipo de dato	Descripción	Acceso	Valor Inicial
13	1	Límite voltaje máximo	Límite máximo de voltaje de entrada	RW	140
14	2	Par máximo	Par máximo	RW	1023
16	1	Nivel de retorno de estado	Seleccionar tipo de estado	RW	2
17	1	LED de alarma	LED para alarma	RW	36
18	1	Apagar	Información de error de apagado	RW	36

*Nota:* Recuperado de (ROBOTIS, s.f.)

**Tabla 16**

*Tabla de control del área RAM del DYNAMIXEL AX-12A*

Dirección	Tamaño (Byte)	Tipo de dato	Descripción	Acceso	Valor Inicial
24	1	Habilitar par	Par motor encendido/apagado	RW	0
25	1	LED	LED de estado encendido/apagado	RW	0
26	1	Margen de cumplimiento de CW	Margen de cumplimiento de CW	RW	1
27	1	Margen de cumplimiento de CCW	Margen de cumplimiento de CCW	RW	1
28	1	Pendiente de cumplimiento de CW	Pendiente de cumplimiento de CW	RW	32
29	1	Pendiente de cumplimiento de CCW	Pendiente de cumplimiento de CCW	RW	32
30	2	Posición del objetivo	Posición de objetivo	RW	-
32	2	Velocidad de movimiento	Velocidad de movimiento	RW	-
34	2	Límite de par	Límite de par (par de objetivo)	RW	ADD14&15
36	2	Posición actual	Posición actual	R	-
38	2	Velocidad actual	Velocidad actual	R	-
40	2	Carga actual	Carga actual	R	-
42	1	Voltaje presente	Voltaje presente	R	-
43	1	Temperatura actual	Temperatura actual	R	-
44	1	Registrado	Si la instrucción está registrada	R	0

Dirección	Tamaño (Byte)	Tipo de dato	Descripción	Acceso	Valor Inicial
46	1	Moviente	Estado del movimiento	R	0
47	1	Bloquear	Bloque de EEPROM	RW	0
48	2	Punch	Umbral de corriente mínimo	RW	32

*Nota:* Recuperado de (ROBOTIS, s.f.)

El paquete de instrucciones son los datos de comando enviados al dispositivo y se encuentra compuesto por (Figura 53): el encabezado que indica el inicio del paquete; el carné de identidad (ID) del dispositivo valor entre 0 ~ 253 o el 254 como broadcast ID, en otras palabras se puede controlar hasta 254 dispositivos por separado o todos los dispositivos al mismo tiempo; la longitud del paquete que es igual al número de parámetros más dos; la instrucción que se desee ejecutar (las instrucciones disponibles se detallan en la Tabla 17); los parámetros que se utilizan cuando se requieren datos adicionales para una instrucción; y finalmente la suma de comprobación de instrucciones, que se utiliza para verificar que el paquete de datos no se haya dañado durante la comunicación. La suma de comprobación de instrucciones se calcula con la siguiente fórmula  $Instruction\ Cheksum = \sim(ID + Longitud + Instrucción + Parametro\ 1 + \dots + Parametro\ N)$ .

### Figura 53

*Paquete de instrucciones*

Encabezado1	Encabezado2	CARNÉ DE IDENTIDAD	Longitud	Instrucción	Param 1	...	Param N	Suma de comprobación
0xFF	0xFF	CARNÉ DE IDENTIDAD	Longitud	Instrucción	Param 1	...	Param N	CHKSUM

*Nota:* Recuperado de (ROBOTIS, s.f.)

**Tabla 17***Tipos de instrucciones*

Valor	Instrucción	Descripción
0x01	Silbido	Verifica si el paquete ha llegado a un dispositivo
0x02	Leer	Leer datos del positivo
0x03	Escribir	Escribir datos en el dispositivo
0x04	Escritura reg	Registra el paquete de instrucciones en un estado de espera para una futura ejecución.
0x05	Acción	Instrucción que ejecuta el paquete que se registró previamente
0x06	Restablecimiento de fabrica	Restablece la tabla de control a su configuración inicial
0x08	Reiniciar	Instrucción que reinicia el DYNAMIXEL
0x83	Sincronizar escritura	Para múltiples dispositivos, instrucciones para escritura
0x92	Lectura masiva	Para múltiples dispositivos, instrucciones para lectura

*Nota:* Recuperado de (ROBOTIS, s.f.)

El controlador envía un paquete de instrucciones al DYNAMIXEL este después de recibirlo envía un paquete de estado o paquete de devolución (Figura 54), el cual tiene una estructura parecida al paquete de instrucciones, pero en vez de enviar una instrucción este envía un estado de error ocurrido durante la operación del DYNAMIXEL (los bits de error se detallan en la Tabla 18).

**Figura 54***Paquete de estado*

Encabezado1	Encabezado2	CARNÉ DE IDENTIDAD	Longitud	Error	Param 1	...	Param N	Suma de comprobación
0xFF	0xFF	CARNÉ DE IDENTIDAD	Longitud	Error	Param 1	...	Param N	CHKSUM

*Nota:* Recuperado de (ROBOTIS, s.f.)

**Tabla 18***Bits del estado de error*

<b>Bit</b>	<b>Error</b>	<b>Descripción</b>
Bit 7	0	-
Bit 6	Error de instrucción	En caso de enviar una instrucción indefinida.
Bit 5	Error de sobrecarga	Cuando la carga actual no puede controlarse mediante el par establecido.
Bit 4	Error de suma de control	Cuando la suma de verificación del paquete de instrucciones transmitido es incorrecta.
Bit 3	Error de rango	Cuando una instrucción esta fuera del rango de uso.
Bit 2	Error de sobrecalentamiento	Cuando la temperatura interna de DYNAMIXEL está fuera del rango de temperatura de funcionamiento establecido en la tabla de control.
Bit 1	Error de límite de ángulo	Cuando la posición del objetivo se escribe fuera del rango entre el límite de ángulo de CW y CCW.
Bit 0	Error de voltaje de entrada	Cuando el voltaje de alimentación esta fuera del rango de voltaje de funcionamiento establecido en la tabla de control.

*Nota:* Recuperado de (ROBOTIS, s.f.)

En (ROBOTIS, s.f.) se puede encontrar un SDK que proporciona funciones de control del DYNAMIXEL implementando el protocolo 1.0 DYNAMIXEL, este SDK es compatible con los sistemas operativo de Windows y Linux, además admite varios lenguajes de programación como C, C++, C#, Python, Java, MATLAB y LabVIEW. En el presente trabajo de titulación se utilizó este SDK para el controlador Teensy 3.6, que anteriormente se mencionó que se usara el entorno de Arduino IDE para su programación.

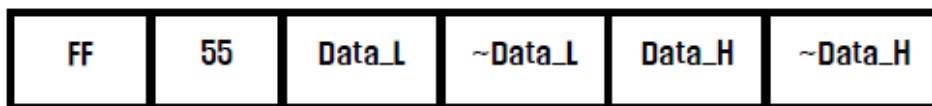
En el software de Arduino IDE existe una librería que hace uso del SDK DYNAMIXEL, pero esta librería solo es compatible para el controlador OpenCM9.04 que es uno de los controladores más actuales de la compañía de Robotis. Por tal motivo se buscó y encontró en el foro que ofrece Arduino una librería compatible con el controlador CM-510, pero la librería encontrada solo permitía un buen control de un actuador DYNAMIXEL, y al tratar de controlar

más actuadores no se obtuvieron los resultados esperados, puesto que por los tiempos de ejecución de las instrucciones de control los actuadores se activaron a destiempo, para solucionar este inconveniente se modificó la librería en base a la tabla de control del DYNAMIXEL AX-12A.

**Zigbee.** La comunicación Zigbee es un conjunto de protocolos de alto nivel de comunicación inalámbrica se lo utiliza para radiodifusión digital de datos con un bajo consumo de energía, esta tecnología se encuentra basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (wireless personal area network, WPAN), opera en las bandas libres ISM<sup>10</sup> de 2.4GHz. Su velocidad de transmisión es baja 250kbps y el rango de cobertura esta entre los 10 y 75 metros, su desempeño no se ve afectado por otro tipo de tecnologías como Wifi o Bluetooth. El paquete de comunicación que usan los módulos ZIG-100/110A bajo este protocolo tiene la siguiente estructura.

#### Figura 55

*Paquete de comunicación ZIG-100/110A*



*Nota:* Recuperado de (ROBOTIS, s.f.)

Este protocolo de igual manera se implementó en el controlador Teensy 3.6 utilizando software de Arduino IDE, para manejar el módulo de comunicación inalámbrica ZIG-110A; también se lo implementó en la computadora portátil utilizando Python IDLE para manejar el

---

<sup>10</sup> ISM – Industrial, Scientific & Medical

módulo de comunicación inalámbrica ZIG2Serial – ZIG-100. Además, se usó este protocolo para lograr la comunicación entre la RPZW y el CM-510.

**TCP/IP.** Son un grupo de protocolos de red que hacen posible la transferencia de datos en redes, entre varios dispositivos y el uso de internet. El protocolo de control de transmisión (TCP) permite establecer una conexión y el intercambio de datos entre dos anfitriones, proporciona un transporte fiable de datos, mientras que el protocolo de internet (IP) utiliza la dirección IP de los dispositivos, para llevar los datos a otros dispositivos dentro de la misma red.

**La arquitectura de comunicación Cliente/Servidor.** Las comunicaciones inalámbricas utilizan el modelo cliente/servidor usando TCP/IP como transporte. Donde un servidor es una aplicación que ofrece algún tipo de servicio a los clientes; y un cliente es el que solicita ese servicio. El servidor se mantiene a la espera de las solicitudes en un puerto conocido de tal manera que los clientes sepan a qué puerto IP deben realizar las peticiones. La familia de protocolos de internet TCP/IP usan una interfaz de programación de aplicaciones denominado como socket, que es un concepto abstracto por el cual dos programas ubicados en diferentes computadoras pueden intercambiar cualquier flujo de datos, de manera fiable y ordenada.

En Picamera (2020) existe una sección de tutoriales que involucran técnicas avanzadas de programación en Python, donde se implementa la librería picamera y otras librerías; el enfoque principal es el manejo de la Raspberry Pi Camera Module. El tutorial “Captura y transmisión rápida” consta de dos scripts: el primer script es un servidor que se implementó en la computadora portátil, el cual se mantiene a la espera de una solicitud de conexión desde la RPZW; y el segundo script es un cliente que se implementó en la RPZW y envía un flujo continuo de imágenes al servidor. Estos scripts usan un protocolo simple de comunicación: primero, se

envía la longitud de la imagen como un entero de 32 bits, luego se envía los bytes de datos de la imagen.

Con los scripts anteriormente mencionados se logró cumplir con un requerimiento del proyecto el cual es transmitir lo que observa el robot humanoide hacia el nodo central para posteriormente ser procesadas, otro requerimiento es poder enviar datos desde el nodo central hacia el robot humanoide para realizar los movimientos requeridos, pero se tuvo un inconveniente al tratar de usar los scripts anteriores debido al protocolo que estos tienen implementados.

La solución al inconveniente encontrado fue la creación de otro cliente/servidor, el cual solo permite el flujo de datos enteros de 8 bits. Para ejecutar los dos sockets al mismo tiempo se hizo una programación con hilos, los cuales permiten separar tareas complejas en tareas más simples y concretas. Un hilo se lo puede definir como cada secuencia de control dentro de un proceso que ejecuta sus instrucciones de forma independiente, de esta manera si un proceso falla no bloquea toda la aplicación.

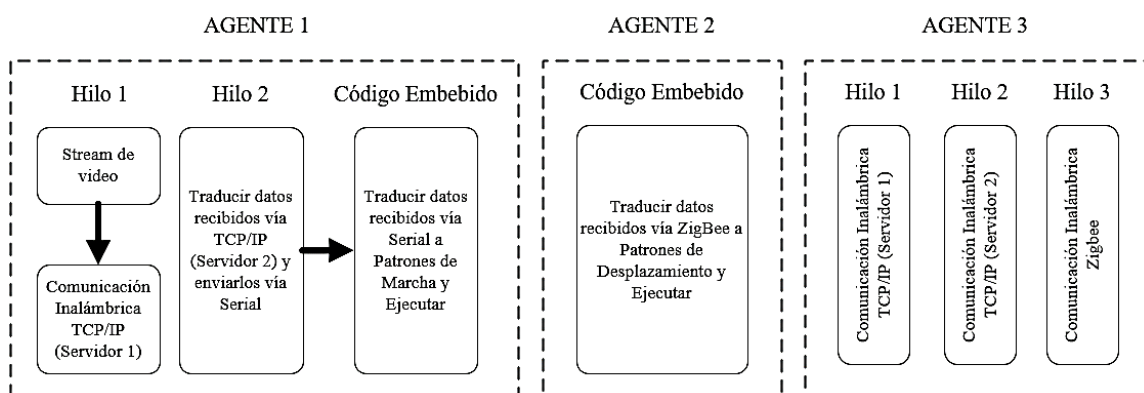
### ***Algoritmos del Sistema Multi-Agente***

El Sistema Multi-Agente al encontrarse implementado con diferentes tecnologías, las mismas que permiten la implementación de la arquitectura de coordinación centralizada; se tomó en consideración el evitar cuellos de botella en los diversos procesos que el algoritmo lleva a cabo y que puedan conllevar a la desconexión del MAS. Para este fin, el algoritmo se subdivide en varios componentes que se ejecutan en forma paralela gracias a la implementación de programación por hilos. El robot humanoide y el robot móvil se encuentran en continuo movimiento durante la ejecución del MAS, sus acciones se actualizan en función de la

información que reciban. El robot humanoide se encuentra implementado con programación por hilos y por código embebido, mientras que el robot móvil solo implementa código embebido, estos actúan de forma paralela a los hilos del tercer agente. En la Figura 56 se presenta la implementación de cada proceso simultaneo para establecer la comunicación en el MAS.

**Figura 56**

*Procesos ejecutados en forma paralela comunicación del MAS*



*Nota:* La programación en hilos del MAS permite simplificar el proceso en procesos pequeños.

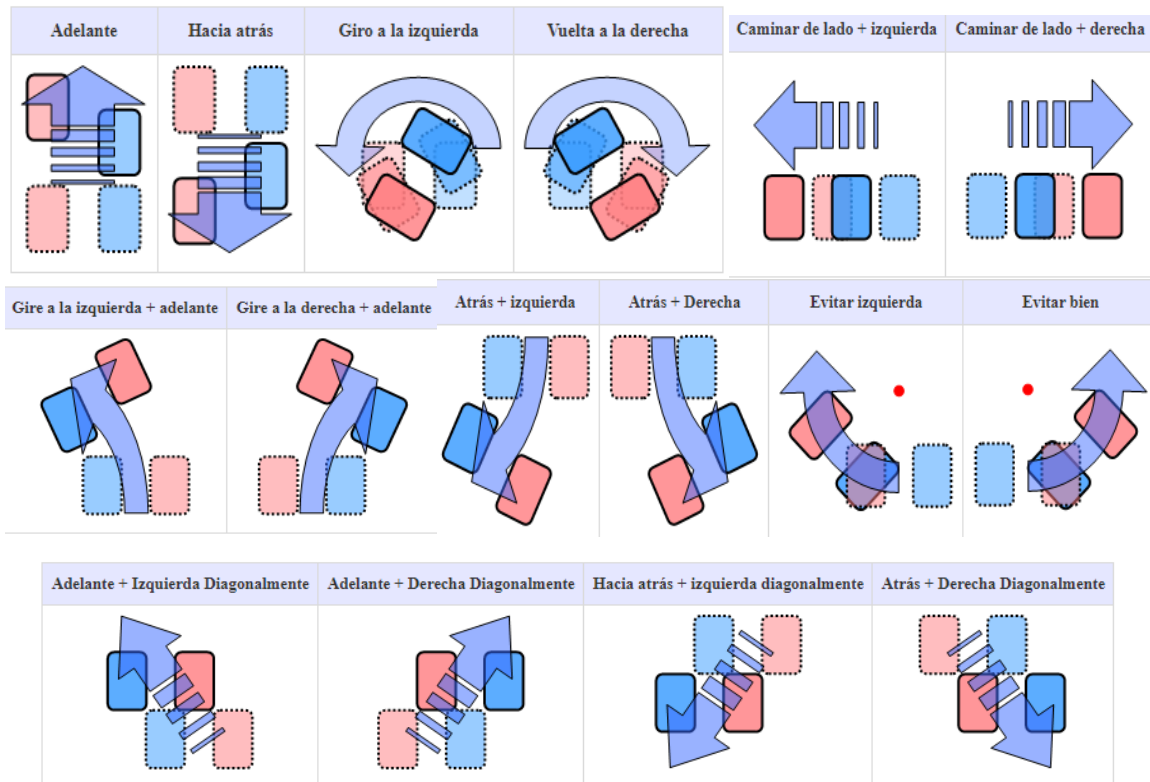
A continuación, se detalla la implementación de cada uno de estos procesos. Primero se va explicar el algoritmo del código embebido del agente 1 correspondiente al robot humanoide. Se tomó los programas ejemplos que ofrece el software R+ Task 3.0, este software permite usar al robot físico para la edición de los Motion Units, estableciendo una conexión en línea con el robot y permitiendo guardar las posiciones en la línea de tiempo, o a su vez con el robot en línea ir modificando manualmente los valores de los actuadores e ir verificando su funcionamiento con el robot físicamente.



El programa ejemplo de movimientos es para un humanoide tipo-A, este cuenta con 16 patrones diferentes de caminata como se puede observar en la Figura 57. Con el fin de que el robot humanoide camine mientras lleva un objeto, se modificó la posición de los brazos del robot, además Vargas & Castillo (2015) mencionan que el centro de masa del robot varía al combinar manipulación y locomoción, por lo cual se debe hacer una compensación cambiando la inclinación del robot a nivel de su articulación dorsal. A parte de los patrones de caminata se realizó las posturas que permitirán al robot humanoide recoger y depositar objetos con las pinzas.

**Figura 57**

*Patrones de caminata del robot humanoide*

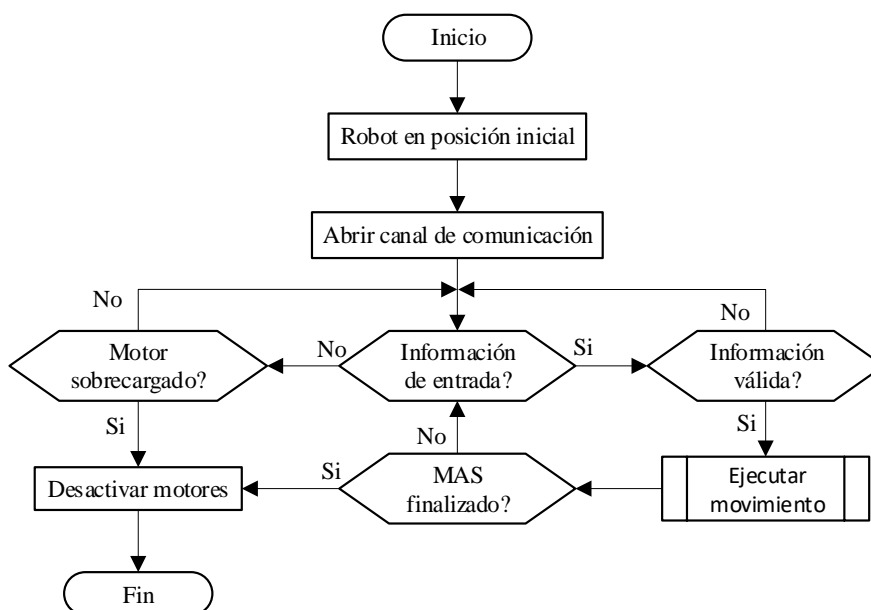


*Nota:* Recuperado de (ROBOTIS, s.f.)

De manera similar para la programación de tareas el programa ejemplo es para un robot humanoide tipo-A. La programación de tareas se encuentra basado en el diagrama de flujo de la Figura 58.

**Figura 58**

*Diagrama de flujo del código embebido del agente 1*

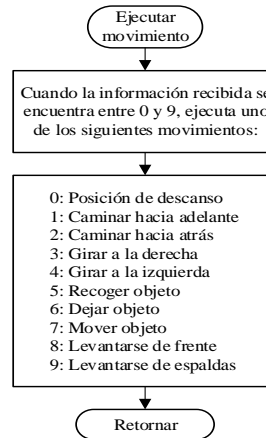


*Nota:* Diseño del autor.

La Figura 59, detalla el proceso de validación de información en el código embebido del agente 1, así como los movimientos que puede ejecutar.

**Figura 59**

*Información válida para ejecutar posibles movimientos agente 1*

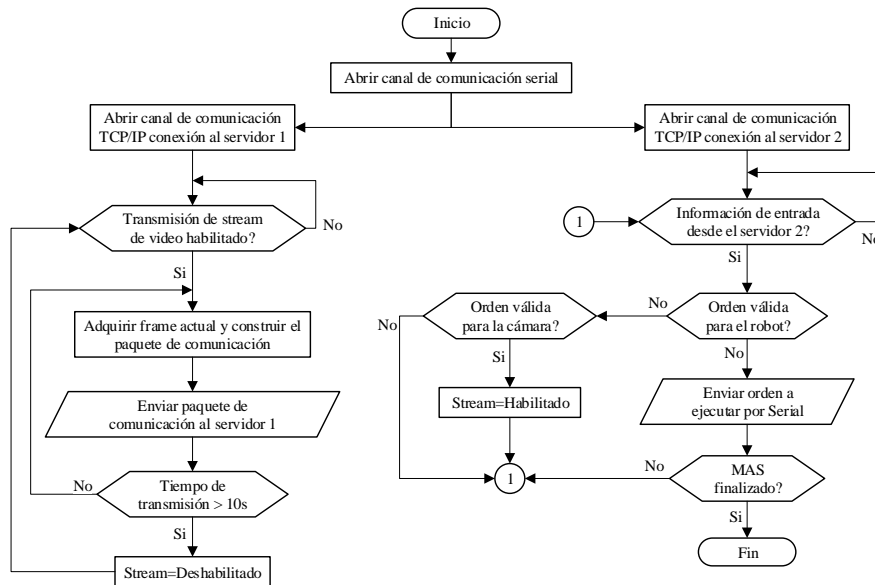


*Nota:* Diseño del autor.

La programación por hilos de la RPZW permite conectar al robot humanoide con el MAS y presenta la estructura que se muestra en la Figura 60.

**Figura 60**

Diagrama de flujo programación por hilos del agente 1

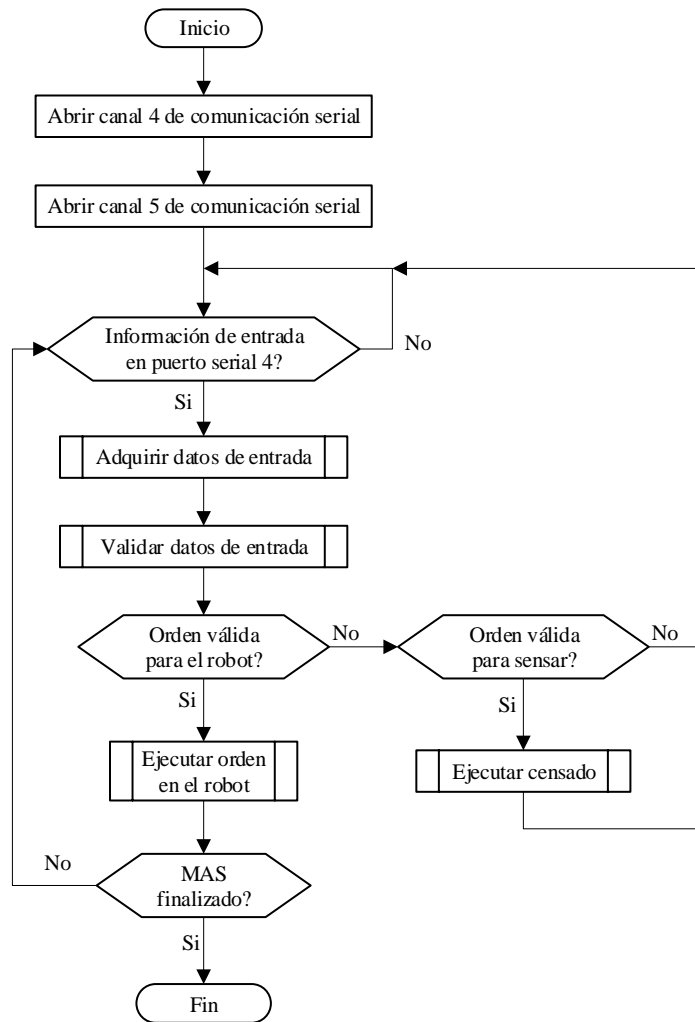


*Nota:* Diseño del autor.

La Figura 61 muestra la estructura de la parte del código embebido que permite conectar el robot móvil al MAS y ejecutar las diferentes órdenes provenientes del nodo central, para el cumplimiento del objetivo general del proyecto.

**Figura 61**

*Diagrama de flujo del código embebido del agente 2*



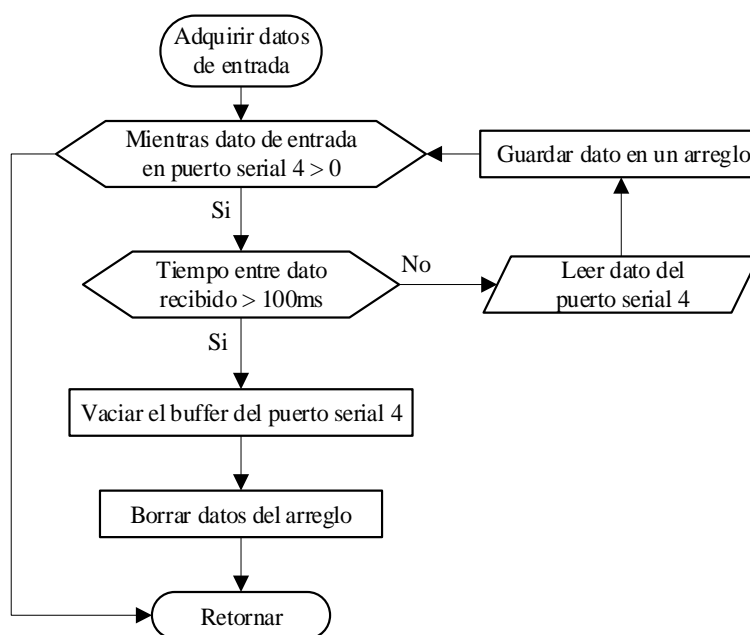
*Nota:* Diseño del autor.

El proceso que permite almacenar la información de entrada proveniente del nodo central, de una manera adecuada se muestra en la Figura 62, además esta estructura no permite

un desbordamiento (fallo) en el búfer de la memoria de la tarjeta de desarrollo. Como en todo algoritmo se debe realizar la validación de los datos recibidos, para detectar posibles fallas en la trama de datos, la estructura de la Figura 63 permite detectar posibles fallas en los datos recibidos.

**Figura 62**

*Estructura para adquisición de datos del código embebido*

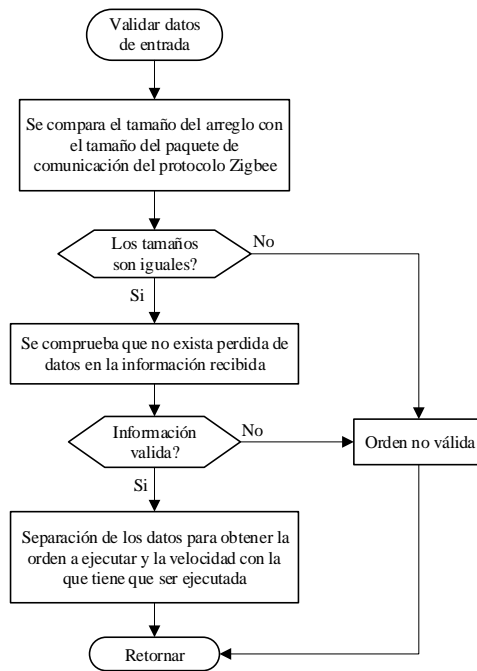


*Nota:* Diseño del autor.

El proceso de ejecutar orden en el robot, hace referencia a las órdenes que permiten al robot móvil desplazarse de un lugar a otro, así como las ordenes que permite sujetar y soltar los objetos de interés. Este proceso se muestra en la Figura 64, donde también se detalla los movimientos que puede realizar el robot. El paso final para ejecutar la orden requerida en los actuadores del robot móvil se relaciona con la implementación de librería descrita en secciones anteriores.

**Figura 63**

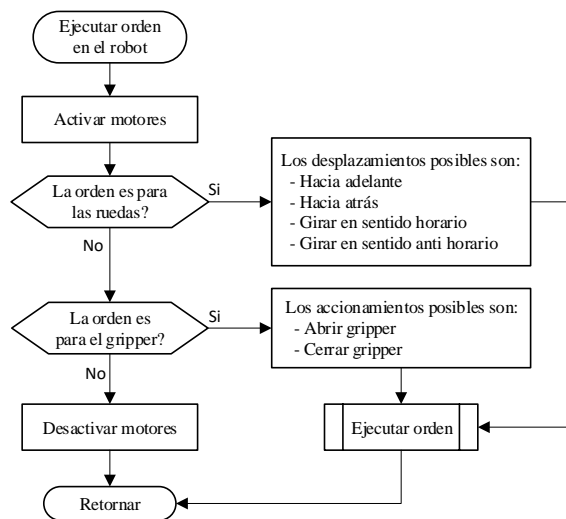
*Estructura para la validación de datos del código embebido*



*Nota:* Diseño del autor.

**Figura 64**

*Estructura para ejecutar ordenes en el robot móvil.*



*Nota:* Diseño del autor.

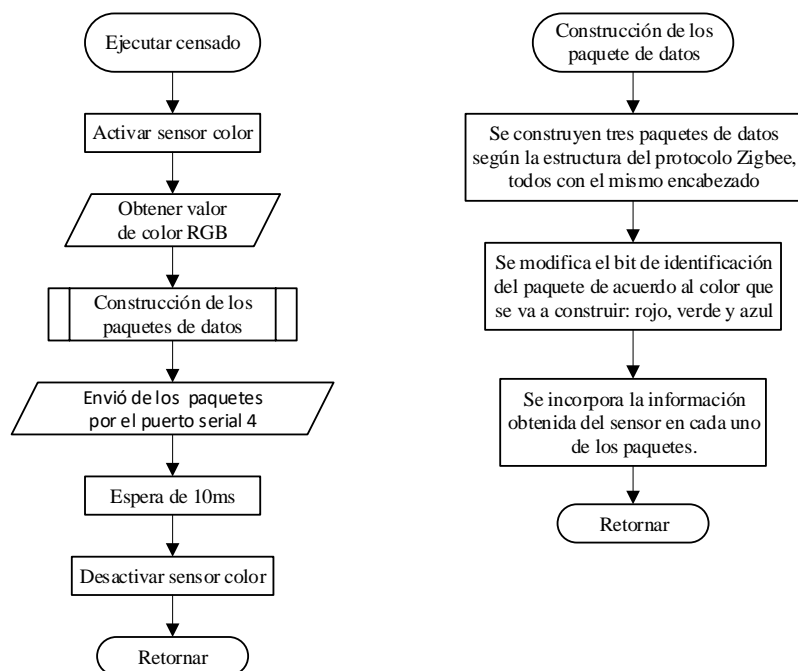
Las funciones básicas de la librería que se usaron para tener la comunicación entre los actuadores DYNAMIXEL AX-12A y la tarjeta de desarrollo Teensy se explican a continuación:

- Para establecer el puerto serial por el cual la Teensy y los actuadores se van a comunicar se usa la función *Dynamixel.setSerial(&Serial5)*.
- La función que abre la comunicación entre ambos dispositivos es *Dynamixel.begin(BAUD\_RATE,DATA\_CONTROL)*, aquí se especifica la velocidad de transmisión y el pin digital para el control de la comunicación half-duplex.
- Para permitir la rotación continua en los actuadores del sistema de locomoción se utiliza la siguiente función: *Dynamixel.setEndlessFD(ID's,ON)*.
- La ejecución de las ordenes en el robot viene dado por dos funciones, la primera función es *Dynamixel.turnFD(ID's,Desplazamiento,Velocidad)* esta permite activar los actuadores del sistema de locomoción al mismo tiempo para realizar los desplazamientos a cierta velocidad, mientras que la función *Dynamixel.moveTD(ID's,Posiciones)* permite activar los actuadores del gripper al mismo tiempo para sujetar o soltar.

Por ultimo en el código embebido del agente 2, el proceso de ejecutar el censado se lo utilizar para validar el color del objeto de interés detectado y tiene la siguiente estructura (Figura 65).

Figura 65

*Estructura para ejecutar el censado de validación de color con el robot móvil*



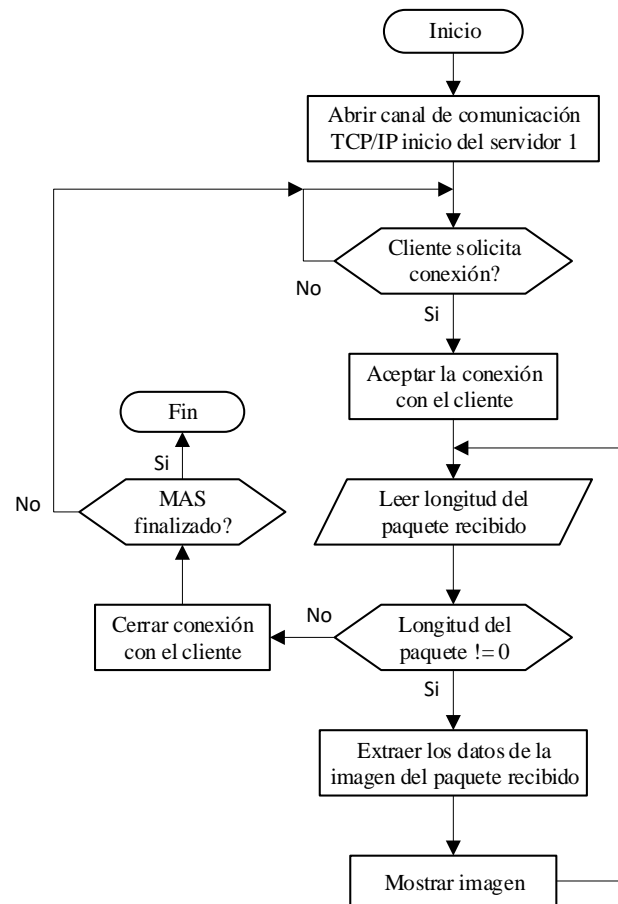
*Nota:* Diseño del autor.

Finalmente, la programación por hilos del agente 3 consta de tres estructuras que se ejecutan de forma paralela, este agente es el nodo central de la arquitectura de coordinación centralizada y el mismo se encarga que flujo de información con los demás agentes. El primer hilo de programación permite la recepción del stream de video del robot humanoide, con la estructura de la Figura 66.



Figura 66

Estructura del primer hilo de programación del agente 3

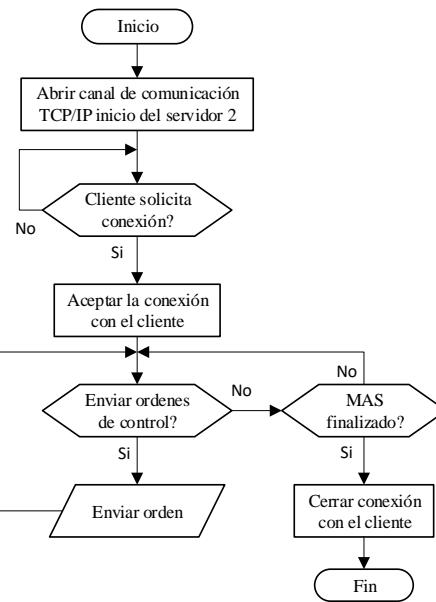


*Nota:* Diseño del autor.

El segundo hilo de programación permite la trasmisión de ordenes al robot humanoide, basado en la estructura de la Figura 67. El tercer hilo de programación permite la recepción y trasmisión de información con el robot móvil, esta estructura se muestra en la Figura 68.

Figura 67

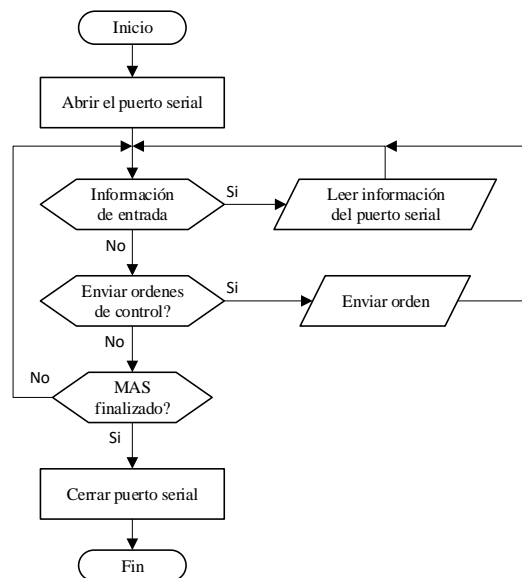
Estructura del segundo hilo de programación del agente 3



Nota: Diseño del autor.

Figura 68

Estructura del tercer hilo de programación del agente 3



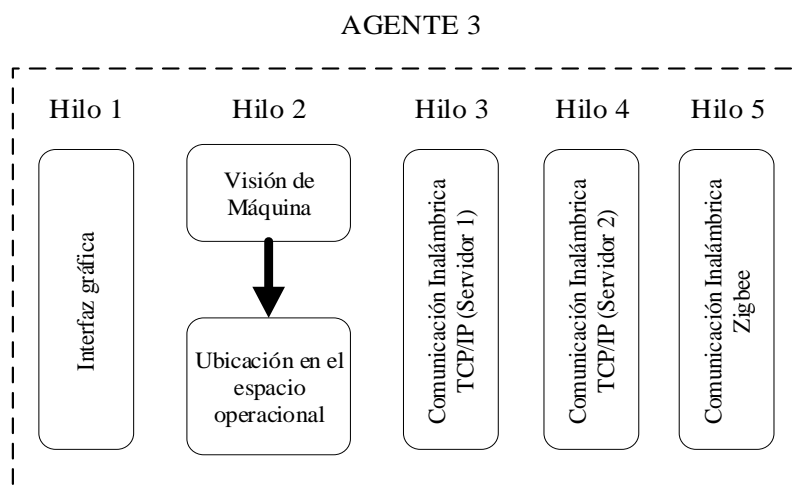
Nota: Diseño del autor.

## Capítulo IV Desarrollo del algoritmo clasificatorio

En el presente capítulo se va a explicar el desarrollo del algoritmo clasificatorio, para el desarrollo de este algoritmo se tomó como referencia el algoritmo de organización desarrollado por Vargas & Castillo (2015) y se le adecuó al proyecto. En la sección anterior se explicó cómo se encuentra conformado el Sistema Multi-Agente, a los procesos del MAS detallados anteriormente se le han agregado dos procesos más en paralelo al agente 3 como se detalla en la Figura 69. Los procesos agregados son los que permiten al MAS realizar la distribución de tareas de clasificación, tareas emergentes, además de que el sistema toma en cuenta la cercanía de cada agente a los objetos de interés.

**Figura 69**

*Procesos del algoritmo de organización*



*Nota:* Diseño del autor.

### Interfaz gráfica

El propósito para diseñar una interfaz gráfica en el proyecto es para crear una interacción persona-maquina mediante el trabajo cooperativo de supervisión y toma de

decisiones. El diseño de las interfaces se basó en las pautas de la guía ergonómica de diseño de interfaz de supervisión GEDIS, por lo tanto, se definió: arquitectura, navegación, distribución, color, texto, equipos, valores, ingreso de datos.

### ***Arquitectura y Navegación***

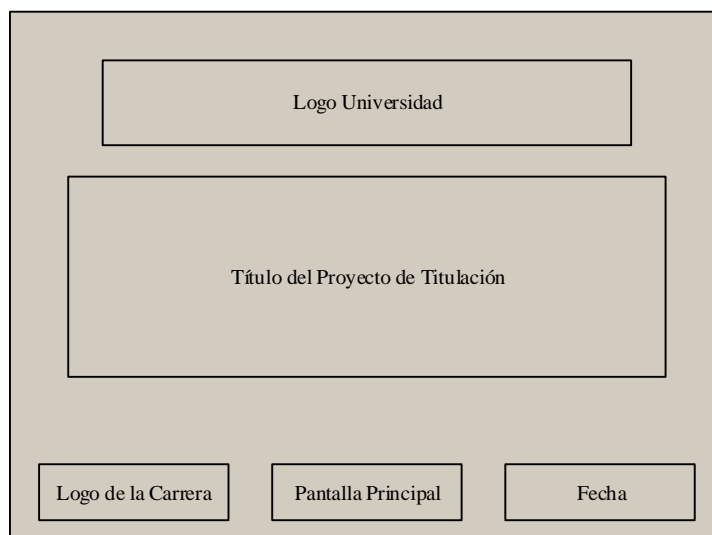
La arquitectura se la ha definido en base a una navegación muy simple, debido a que el proyecto no requiere de muchas pantallas para su funcionamiento. La navegación solo se dará desde la Pantalla de Título hacia la Pantalla Principal y en la Pantalla Principal al terminar de ejecutar el algoritmo clasificatorio permitirá cerrar la interfaz.

### ***Distribución***

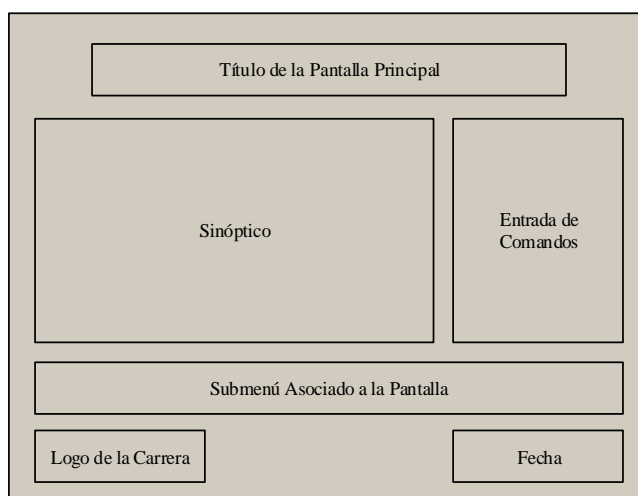
La distribución de las pantallas propuestas para la interfaz del proyecto se detalla a continuación.

### **Figura 70**

#### *Distribución de la pantalla Título*



*Nota:* Diseño del autor.

**Figura 71***Distribución de la pantalla principal**Nota:* Diseño del autor.**Color y Texto**

El color es uno de los elementos más importantes en el diseño de interfaces, su uso adecuado es determinante para la generación de una excelente interfaz bajo la metodología GEDIS. La información del proceso es presentada comúnmente en texto, la guía GEDIS regula el uso de este elemento para informar eficazmente al operador respecto al estado del proceso. Algunas características importantes sobre el texto para su correcta utilización son: de preferencia se debe usar fuentes Sans Serif, no usar más de tres tamaños de la misma fuente, combinar mayúsculas y minúsculas.

**Comandos e Ingreso de Datos**

La guía GEDIS establece la intervención del operador al suministrar datos al sistema de modo que este se comporte de acuerdo a sus objetivos. Las operaciones que normalmente efectúa el usuario son: ejecutar comandos, seleccionar opciones e ingresar datos. Las principales

características de los comandos son su visibilidad y su fácil operación. Se puede clasificar los tipos de comando como:

- Comandos de arranque y paro
- Confirmación de alarmas
- Selección excluyente de una opción
- Selección múltiple no excluyente
- Selección simple

A continuación, se muestran las diversas posibilidades de entradas de comandos que permite el IDLE de Python.

**Figura 72**

*Tipos de entrada de comandos*



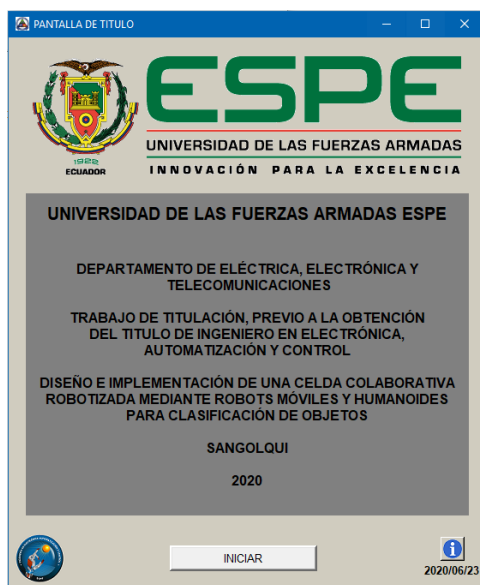
*Nota:* El tipo de entrada de comando seleccionado es el RadioButton

### ***Pantallas de la interfaz del proyecto***

Las pantallas de la interfaz diseñadas son dos: la primera es la Pantalla de Título, en la que se muestra el título del presente trabajo de titulación y el autor, tal como se puede observar en la Figura 73; la segunda es la Pantalla Principal, la cual consiste en el monitoreo y control para la clasificación de objetos tal como se puede observar en la Figura 74.

Figura 73

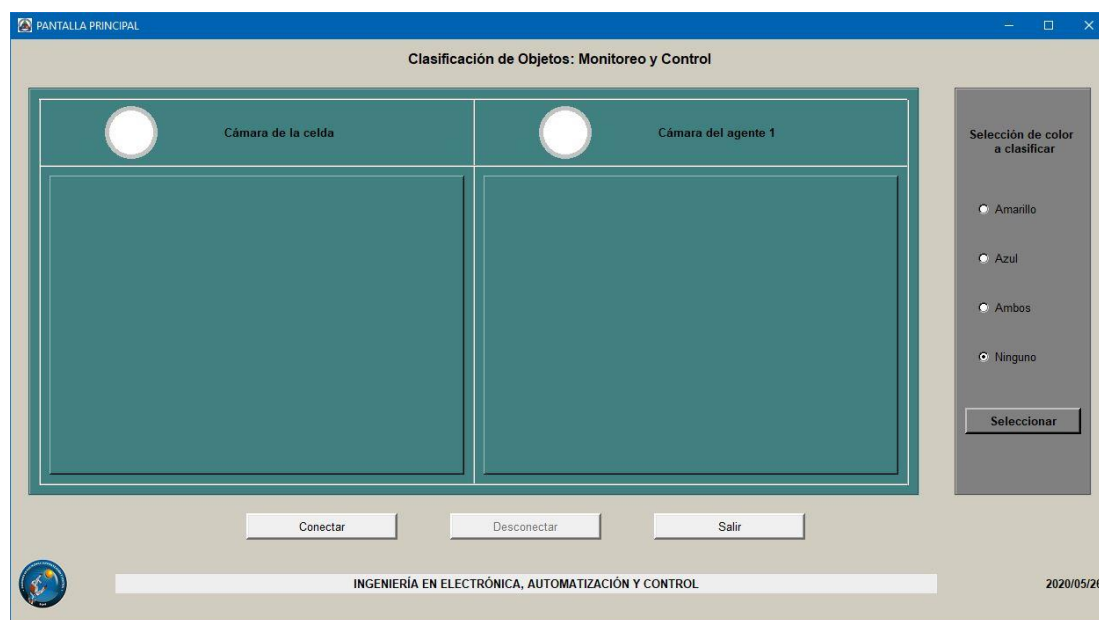
*Pantalla de título del proyecto*



*Nota:* Diseño del autor.

Figura 74

*Pantalla principal del proyecto*



*Nota:* Diseño del autor.

## Visión de máquina

La percepción visual es la encargada de extraer de la imagen todos los elementos de interés (objetos de color y los dos robots) del video obtenido de la cámara superior de la celda colaborativa. Se implementó la librería de OpenCV para Python en la versión 4.1.2, que es una librería libre de visión artificial, de fácil manejo, tiene documentación detallada, explicada y actualizada. Esta librería es utilizada en varias aplicaciones algunas de ellas son: detección de movimiento, reconocimiento de objetos, reconstrucción 3D a partir de imágenes 2D.

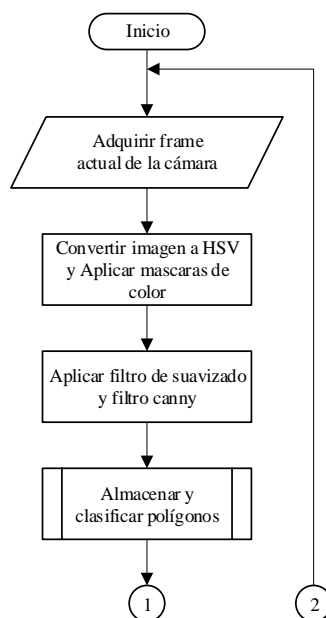
El algoritmo de referencia toma en cuenta que el diseño del robot humanoide Bioloid tienen un diseño homogéneo, razón por la cual es necesario dotarlos con algún tipo de marcador, que servirán para determinar la posición y orientación de los robots. También los objetos de interés por recolectar tienen una forma geométrica conocida, por lo cual el siguiente segmento de código se enfoca en la extracción de figuras circulares y triangulares, lograda mediante la detección de contornos. El diagrama de flujo de este componente se muestra en la Figura 75.

La cámara superior de la celda captura 30 cuadros por segundos con la resolución de 640 por 480 píxeles. El algoritmo de referencia contrasta dicho valor con el tamaño del espacio de trabajo que en el proyecto es de 120x90cm, por lo que cada píxel tiene un tamaño equivalente a un cuadrado de 1.875mm de lado. Luego el algoritmo crea una copia del cuadro actual, que se convierten a HSV. A la imagen HSV se le aplica cinco filtros de color que permitirán detectar contornos en cada uno de los colores, permitiendo así clasificar a los dos robots y los objetos de interés.



Figura 75

Diagrama de flujo para componente de visión



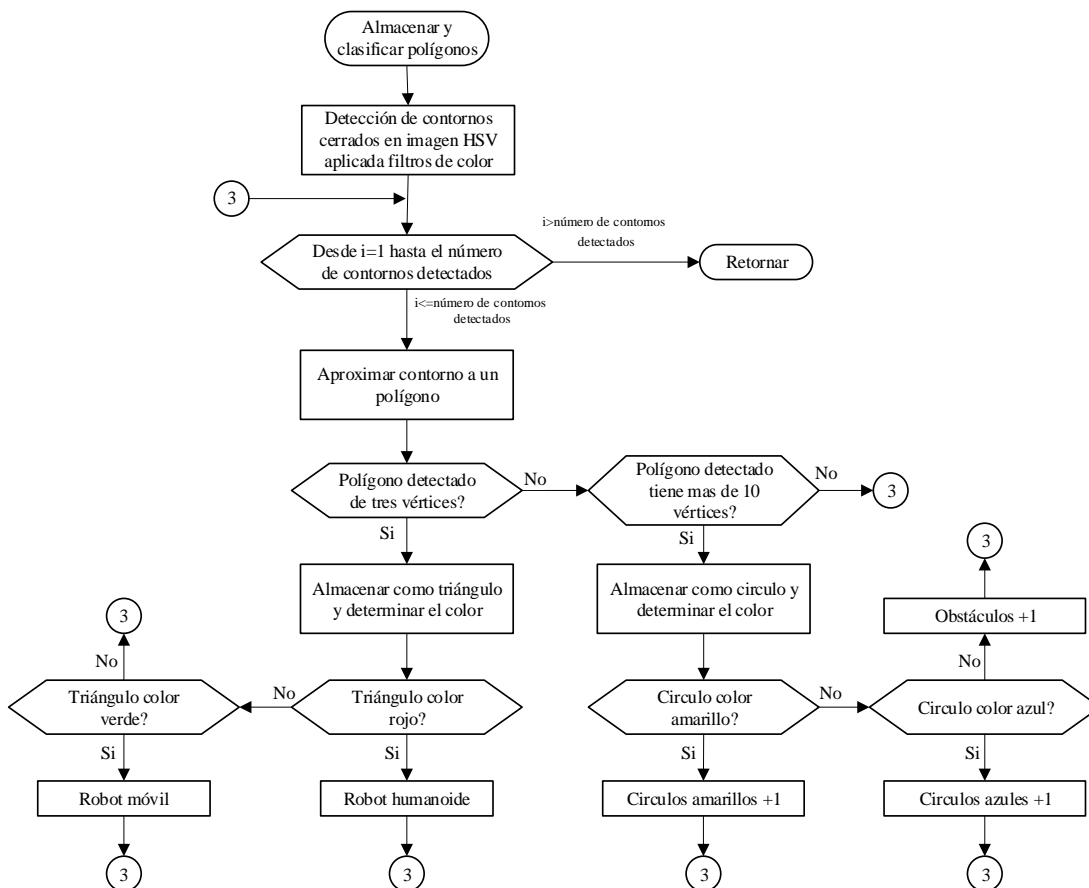
Nota: Diseño del autor.

El proceso que permite almacenar la información de los polígonos clasificados dependiendo del número de vértices y del color, presenta la estructura de la Figura 76.

La función de OpenCV-Python que permite la conversión de una imagen de un espacio de color a otro es `cv2.cvtColor()`, los argumentos que admite esta función es la imagen cuyo espacio de color se quiere convertir y el código de conversión del espacio de color. Para convertir una imagen a HSV se usa el código `cv2.COLOR_BGR2HSV`. Para aplicar el filtro de color se tiene que crear matrices tipo Numpy con los rangos mínimos y máximos de umbral. La función `cv2.inRange()` permite segmentar una región o color particular en una imagen, la función admite tres argumentos: el primero la imagen de entrada, el segundo argumento es el límite inferior del umbral y el tercer argumento es el límite superior del umbral.

Figura 76

Estructura para almacenar y clasificar polígonos de acuerdo a su forma y color



Nota: Diseño del autor.

El suavizado de una imagen se logra convolucionando la imagen con un núcleo de filtro de pasa bajo, que es útil para eliminar el ruido. Para el suavizado de la imagen se va a utilizar el filtrado gaussiano que se realiza con la función *cv2.GaussianBlur()*, los argumentos que admite esta función es la imagen a la cual se le va a aplicar el filtro, especificar el ancho y la altura del núcleo que debe ser positivo e impar y la desviación estándar en las direcciones X e Y.

La finalidad de aplicar el filtro Canny es para encontrar bordes en una imagen usando el algoritmo de Canny Edge Detection. En OpenCV la función de este algoritmo es `cv2.Canny()`, el primer argumento es la imagen de entrada, el segundo y tercer argumento es el valor mínimo y máximo de umbral de histéresis.

Después de la etapa de filtrado se procede a detectar los contornos existentes con la función `cv2.findContours()`, esta función admite tres argumentos: el primero es la imagen de entrada que en esta etapa es una imagen binaria, el segundo es el modo de recuperación de contorno y el tercero es el método de aproximación de contorno. Los modos de recuperación y los métodos de aproximación de contorno se detallan en la Tabla 19.

**Tabla 19**

*Modos de recuperación y métodos de aproximación de contorno de la función `cv.findContours()`*

<b>Modo</b>	<b>Descripción</b>
<code>cv2.RETR_EXTERNAL</code>	Recupero solo los contornos exteriores extremos.
<code>cv2.RETR_LIST</code>	Recupera todos los contornos sin establecer ninguna relación jerárquica.
<code>cv2.RETR_CCOMP</code>	Recupera todos los contornos y los organiza en una jerarquía de dos niveles.
<code>cv2.RETR_TREE</code>	Recupera todos los contornos y reconstruye una jerarquía completa de contornos anidados.
<b>Método</b>	<b>Descripción</b>
<code>cv2.CHAIN_APPROX_NONE</code>	Almacena absolutamente todos los puntos de contorno.
<code>cv2.CHAIN_APPROX_SIMPLE</code>	Comprime segmentos horizontales, verticales y diagonales y deja solo los puntos de los vértices.
<code>cv2.CHAIN_APPROX_TC89_L1</code> , <code>cv2.CHAIN_APPROX_TC89_KCOS</code>	Aplica uno de los sabores del algoritmo de aproximación de cadena Teh-Chin.

*Nota:* Dependiendo del modo y el método seleccionado se obtiene diferentes contornos, la selección correcta del mismo dará buenos resultados en el procesamiento de la imagen.

La función anterior devuelve una lista de todos los contornos encontrados en la imagen, donde cada contorno individual es una matriz que contiene las coordenadas  $(x, y)$  de cada uno de los vértices del objeto. Los contornos presentan diferentes características como el área, perímetro, centroide, cuadrado delimitador, etc.

La función *cv2.approxPolyDP()* aproxima una forma de contorno (curva o polígono) a otra forma con menos número de vértices dependiendo de la precisión que se especifique. Esta función implementa el algoritmo de Douglas-Peucker, cuyo propósito es encontrar una curva similar con menos puntos, para lo cual se basa en la distancia máxima entre la curva original y la curva simplificada. Esta función acepta tres argumentos: el primero es la matriz del contorno de entrada, el segundo es un parámetro denominado épsilon que especifica la precisión de aproximación y el tercer argumento indica a la función si la curva aproximada está cerrada o no.

La función anterior devuelve una nueva matriz con las coordenadas  $(x, y)$  de los vértices del contorno aproximado, dependiendo del número de vértices de cada contorno aproximado se los puede clasificar como polígonos regulares. Con la ayuda de estos polígonos se hace el reconocimiento de los robots y de los objetos de interés a clasificar dentro del espacio de trabajo, en la Figura 76 se puede observar cómo se hizo la clasificación de los polígonos y a que corresponde cada uno.

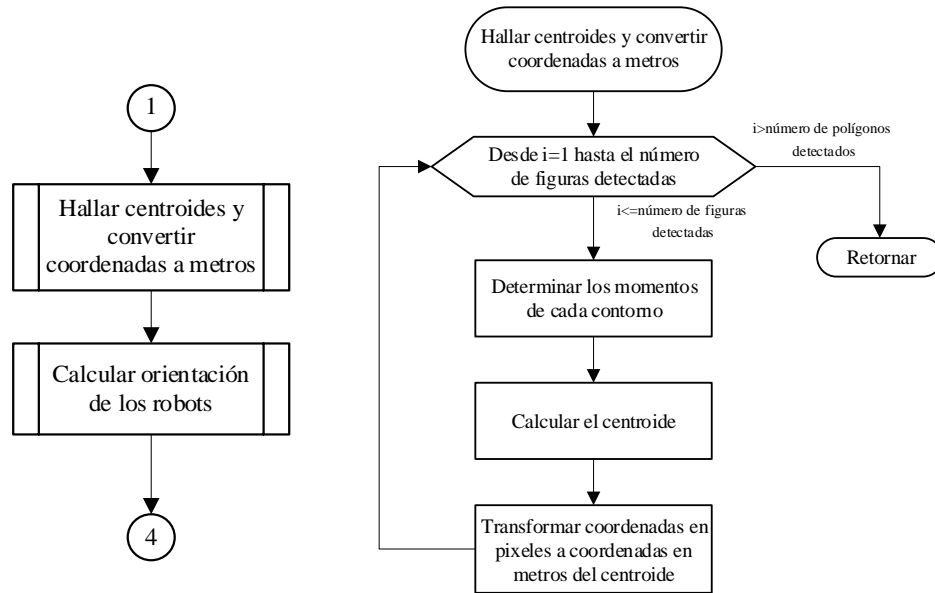
### **Ubicación en el espacio de trabajo**

En cuanto al componente de ubicación el algoritmo de referencia direcciona a que se debe obtener la posición y orientación de los robots y las posiciones de los objetos de interés, además de calcular el ángulo entre la orientación actual de cada robot y la orientación deseada.

El diagrama de flujo del proceso de ubicación dentro del espacio de trabajo se muestra en la Figura 77.

**Figura 77**

*Diagrama de flujo para componente de ubicación*



*Nota:* Diseño del autor.

Según Robologs (2019) los momentos de un contorno son una medida que indica la dispersión de una nube de puntos, y matemáticamente se define como:

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y) \quad (4)$$

donde  $x, y$  son las coordenadas de un pixel y la función  $I(x, y)$  indica su intensidad. Para calcular los momentos se debe trabajar sobre una máscara binaria, por lo tanto la función  $I(x, y)$  solo puede tomar el valor de 0 o 1. Para calcular el centroide del contorno hay tres momentos que interesan:  $M_{00}$ ,  $M_{01}$  y  $M_{10}$ . Calculando el momento  $M_{00}$ , la formula anterior se transforma en:

$$M_{00} = \sum_x \sum_y x^0 y^0 I(x, y) = \sum_x \sum_y I(x, y) \quad (5)$$

El momento  $M_{00}$  es equivalente al número de píxeles cuyo valor es uno, en otras palabras, este momento es el área en píxeles de la región blanca. Para  $M_{10}$ , la fórmula original se transforma en:

$$M_{10} = \sum_x \sum_y x^1 y^0 I(x, y) = \sum_x \sum_y x I(x, y) \quad (6)$$

Este momento es igual a la suma de las coordenadas  $x$  de los píxeles cuya intensidad sea uno. De manera análoga el momento  $M_{01}$ , es igual a la suma de las coordenadas en  $y$  de los píxeles cuya intensidad sea uno. Por lo tanto, si se divide  $M_{10}$  por  $M_{00}$  se obtiene la componente en  $x$  del centroide, mientras que si se divide  $M_{01}$  por  $M_{00}$  se obtiene la componente en  $y$  del centroide.

$$\frac{M_{10}}{M_{00}} = \frac{\sum_x \sum_y x I(x, y)}{\sum_x \sum_y I(x, y)} = \frac{x_1 + x_2 + \dots + x_n}{n} = \bar{x} \quad (7)$$

$$\frac{M_{01}}{M_{00}} = \frac{\sum_x \sum_y y I(x, y)}{\sum_x \sum_y I(x, y)} = \frac{y_1 + y_2 + \dots + y_n}{n} = \bar{y}$$

En Python existe una función que permite calcular los momentos de un contorno. La función `cv2.moments()` recibe como argumento el contorno de entrada, luego se toma los momentos  $M_{00}$ ,  $M_{10}$  y  $M_{01}$  de cada contorno y se realiza las divisiones antes mencionadas.

Se toma en consideración para la transformación de coordenadas que el origen (0,0) en la imagen se encuentra en la esquina superior izquierda, lo cual es contrario al origen del plano cartesiano que se sitúa en la esquina inferior izquierda; otra consideración tomada en cuenta

son las relaciones que existen entre las dimensiones de la imagen y del espacio de trabajo. Para realizar las transformaciones de coordenadas a centímetros se utilizan las siguientes ecuaciones:

$$X_{cm} = X_{pixeles} * 0.1875 \quad [cm] \quad (8)$$

$$Y_{cm} = 90 - (Y_{pixeles} * 0.1875) \quad [cm] \quad (9)$$

donde  $X_{pixeles}$  y  $Y_{pixeles}$  son las coordenadas en pixeles provenientes de la cámara. Al realizar la transformación de coordenadas se genera un sistema de referencia el cual se denominará sistema A, dentro de este sistema se localizará los robots y los objetos de interés.

Tabango (2014) detalla que se debe realizar la corrección de las coordenadas de ubicación de los objetos que se encuentran dentro del área de trabajo, esto se debe realizar debido a la diferencia de alturas que existe entre los robots, los objetos a recolectar y el campo visual de trabajo de la cámara superior de la celda.

Como se puede apreciar en la Figura 78 la cámara ve la proyección del objeto en la ubicación  $P_1(x_1, y_1)$  (valor conocido) pero la verdadera ubicación es  $P_2(x_2, y_2)$ , es por esto que se debe corregir la ubicación vista por la cámara para que sea la verdadera ubicación del objeto. Para realizar la corrección de la ubicación del objeto es necesario determinar el valor de la distancias  $d_1, d_3$  que se detallan a continuación:

$$d_1 = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} \quad (10)$$

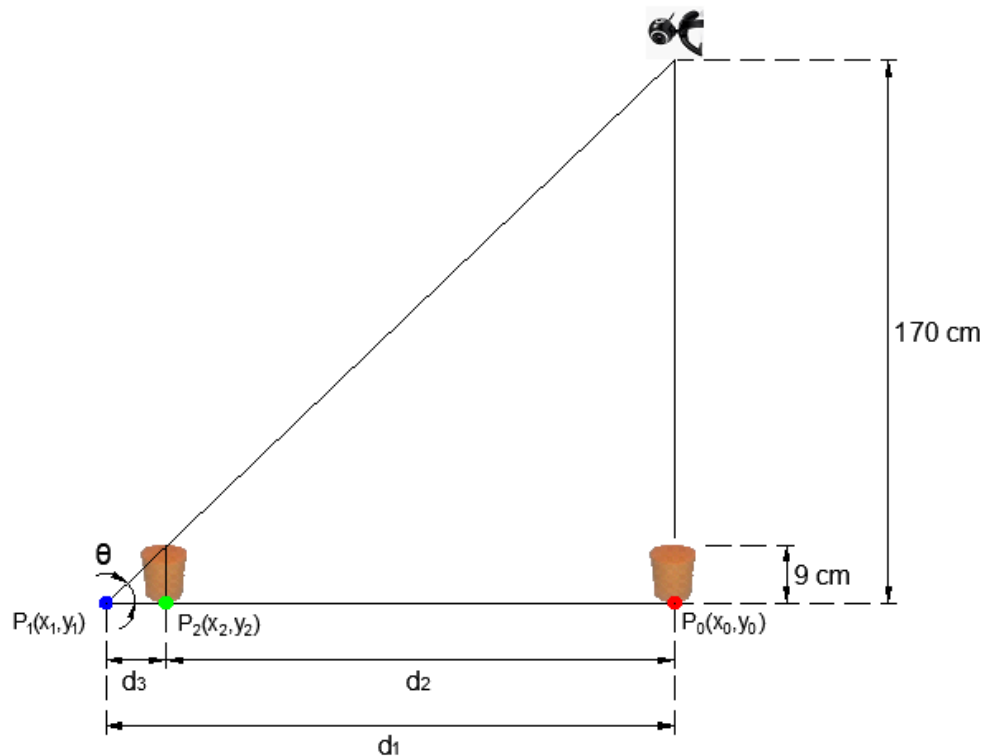
$$d_3 = d_1 - d_2 \quad (11)$$

donde  $P_0(x_0, y_0)$  es la ubicación debajo de la cámara (valor conocido),  $d_1$  es la distancia entre la ubicación proyectada del objeto visto por la cámara y la ubicación debajo de la cámara,  $d_2$  es la

distancia entre la ubicación real del objeto y la ubicación debajo de la cámara y  $d_3$  es la diferencia entre la ubicación proyectada y la ubicación real del objeto.

**Figura 78**

*Diferentes ubicaciones del objeto a recolectar*



*Nota:* Diseño del autor.

La corrección de ubicación se lo realiza tanto en el eje  $x$  como en el eje  $y$ , cuando se lo hace en el eje  $x$  las coordenadas del eje  $y$  se las pone en cero, y cuando se lo hace en el eje  $y$  las coordenadas del eje  $x$  se las pones en cero.

Como se ve en la Figura 78 se forman dos triángulos rectángulos que son semejantes por tener el ángulo agudo igual  $\theta$ , en donde las alturas de estos triángulos son conocidos la una es la altura a la que se encuentra la cámara desde el piso de la celda y la otra es la altura del



objeto a recolectar. Utilizando la propiedad de la semejanza de triángulos donde sus lados homólogos son proporcionales se tiene:

$$\frac{d_1}{d_3} = \frac{170}{9} \quad (12)$$

despejando  $d_3$  de la ecuación (12) se tiene:

$$d_3 = \frac{9}{170} d_1 \quad (13)$$

reemplazando la ecuación (13) en la ecuación (11) y despejando  $d_2$ :

$$\frac{9}{170} d_1 = d_1 - d_2 \quad (14)$$

$$d_2 = \frac{161}{170} d_1 \approx 0.947d_1 \quad (15)$$

Con la ecuación (15) se tiene el resultado aproximado de la ubicación real del robot con respecto a la ubicación proyectada del objeto que ve la cámara, con esto el algoritmo hace la corrección de las coordenadas del objeto a recolectar. De manera análoga se realiza el mismo procedimiento tanto para el robot humanoide como para el robot móvil, en donde el valor que cambia es la altura de cada uno obteniendo los siguientes resultados de aproximación:

Donde la altura del robot humanoide es de 33 cm.

$$d_3 = \frac{33}{170} d_1 \quad (16)$$

reemplazando la ecuación (16) en la ecuación (11) y despejando  $d_2$ :

$$\frac{33}{170} d_1 = d_1 - d_2 \quad (17)$$

$$d_2 = \frac{137}{170} d_1 \approx 0.806d_1 \quad (18)$$

mientras que la altura del robot móvil es de 10 cm.

$$d_3 = \frac{10}{170} d_1 \quad (19)$$

reemplazando la ecuación (19) en la ecuación (11) y despejando  $d_2$ :

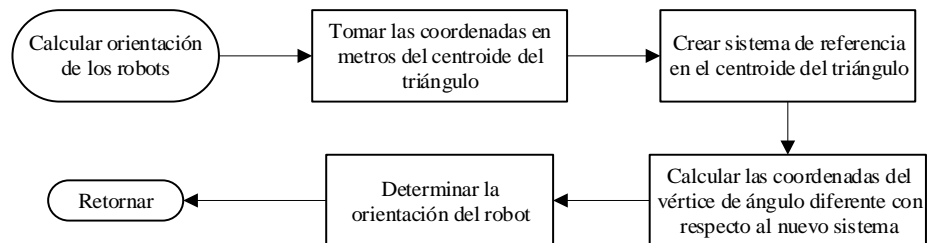
$$\frac{1}{17} d_1 = d_1 - d_2 \quad (20)$$

$$d_2 = \frac{16}{17} d_1 \approx 0.941d_1 \quad (21)$$

El algoritmo de referencia usa marcadores para los robots en forma de triángulos isósceles, donde el vértice de ángulo diferente se encuentra apuntando hacia el frente del robot. Con la utilización de estos marcadores el algoritmo calcula la orientación de los robots, utilizando en cada uno de los robots la estructura que se muestra en la Figura 79.

**Figura 79**

*Estructura para determinar la orientación de los robots*



*Nota:* Diseño del autor.

Como se explicó en la sección anterior, al encontrar el contorno aproximado se obtiene las coordenadas (x, y) de los vértices y luego de la clasificación se toma los polígonos triangulares correspondientes a los robots para calcular la orientación de cada uno. Primero con

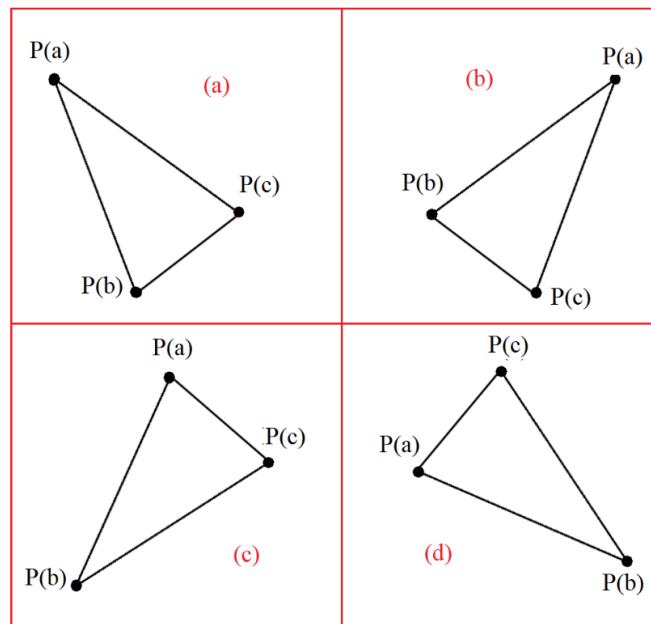
las coordenadas de los vértices en pixeles se los transforma a coordenadas en metros usando las ecuaciones antes mencionadas, estas coordenadas se encuentra en referencia al sistema A. Luego se calcula la longitud de cada lado del triángulo, para lo cual se usa la ecuación de la distancia entre dos puntos de la geometría analítica.

$$d_{P_1P_2} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad [cm] \quad (22)$$

Una vez obtenido las tres distancias se detecta cual es el lado de diferente longitud y el vértice de ángulo diferente del triángulo isósceles. Al realizar las pruebas del algoritmo hasta este parte, se determinó que las coordenadas de los vértices en Python se guardan en distinto orden, esto depende de la rotación en la que este el triángulo como se puede observar en la Figura 80.

**Figura 80**

*Detección de coordenadas del triángulo en diferentes rotaciones*

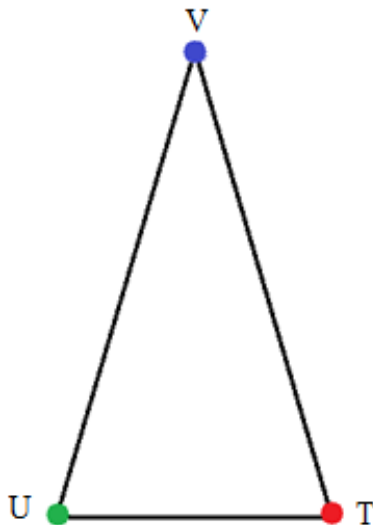


*Nota:* Diseño del autor.

Para que el algoritmo pueda calcular la orientación del robot, se toma en consideración que las coordenadas deberían respetar el orden que se detalla en la Figura 81.

**Figura 81**

*Orden considerado de las coordenadas*



*Nota:* Diseño del autor.

Tomando en cuenta la consideración planteada en la Figura 81, se determinó que si el lado de menor longitud del triángulo se tiene como en la Figura 80-(a) y (b), el orden se determina como:  $T = P(c)$ ,  $U = P(b)$  y  $V = P(a)$ ; mientras que si el lado de menor longitud se tiene como en la Figura 80-(c) y (d), el orden es:  $T = P(a)$ ,  $U = P(c)$  y  $V = P(b)$ . **Figura 80**

En el centroide C del triángulo se coloca un sistema de referencia que se denominará sistema B. Se procede a encontrar la matriz de transformación homogénea del sistema A con respecto al sistema B.

$$T_A^B = \begin{bmatrix} R_A^B & t_A^B \\ 0 & 1 \end{bmatrix}_{3 \times 3} \quad (23)$$

donde  $R_A^B$  es la matriz de rotación del sistema A con respecto al sistema B, y  $t_A^B$  es el vector de posición del origen del sistema A con respecto al sistema B. Se procede a encontrar  $R_A^B$  y  $t_A^B$ :

$$R_A^B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}_{2 \times 2} ; \quad t_A^B = \begin{bmatrix} -c_x \\ -c_y \end{bmatrix}_{2 \times 1} \quad (24)$$

donde  $R_A^B$  es la matriz identidad ya que no existe ninguna rotación en los ejes y  $C(c_x, c_y)$  son las coordenadas del centroide, quedando así la matriz de transformación homogénea:

$$T_A^B = \begin{bmatrix} 1 & 0 & -c_x \\ 0 & 1 & -c_y \\ 0 & 0 & 1 \end{bmatrix}_{3 \times 3} \quad (25)$$

Las coordenadas del vértice de menor ángulo del triángulo V se encuentran con respecto al sistema A y lo que se requiere es encontrar las coordenadas con respecto al sistema B utilizando la matriz de transformación homogénea.

$$\begin{bmatrix} V^B \\ 1 \end{bmatrix} = \begin{bmatrix} R_A^B & t_A^B \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V^A \\ 1 \end{bmatrix} \quad (26)$$

reemplazando los datos antes determinados en la ecuación anterior se obtiene:

$$\begin{bmatrix} V^B \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -c_x \\ 0 & 1 & -c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ 1 \end{bmatrix} \quad (27)$$

Finalmente, para encontrar la orientación del robot primero se calcula el ángulo  $\alpha$  que forma el punto  $V^B$  con el eje x positivo del sistema B, luego con dicho valor se determina la orientación verdadera  $\beta$ . Para calcular el ángulo  $\alpha$  se utiliza la función  $atan2()$ , esta función recibe dos argumentos: el primero la componente  $V_y$  y el segundo la componente  $V_x$ ; el valor que devuelve la función es un valor numérico entre  $-\pi$  y  $\pi$ .

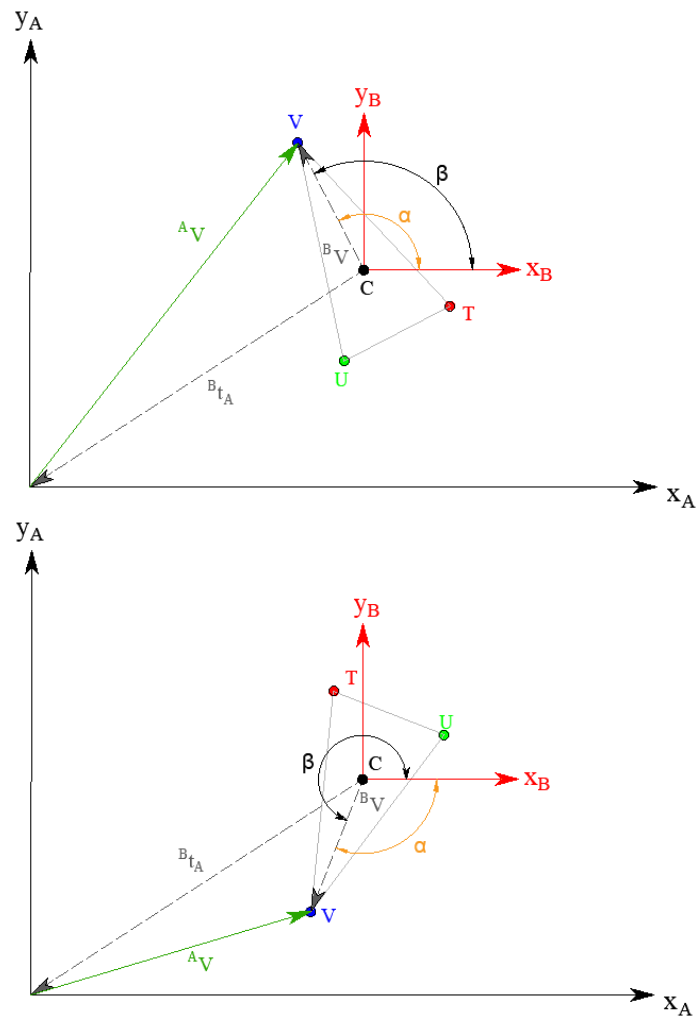
Dependiendo del valor de  $\alpha$  se calcula el valor del ángulo  $\beta$  en base al siguiente sistema de ecuaciones:

$$\begin{aligned} \beta &= \alpha & ; & \text{ si } 0 \leq \alpha \leq \pi \\ \beta &= 2\pi + \alpha & ; & \text{ si } -\pi < \alpha < 0 \end{aligned} \quad (28)$$

La Figura 82 muestra un ejemplo grafico de la estructura presentada en la Figura 79.

**Figura 82**

*Ejemplos de la representación de posición y orientación*



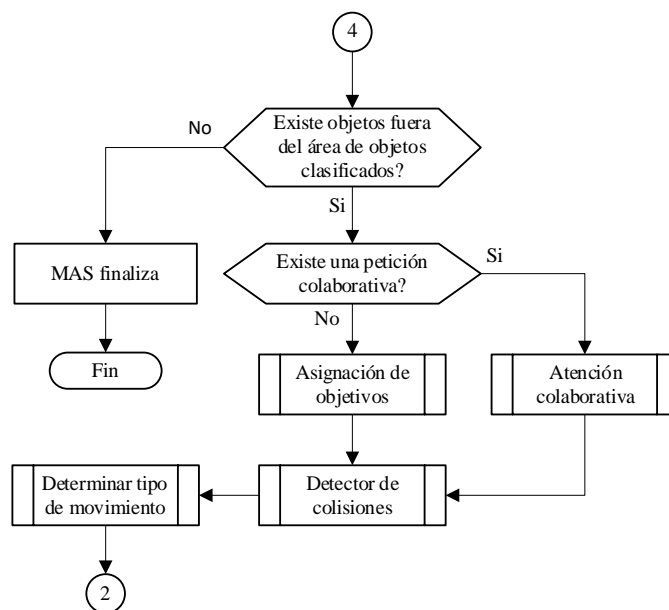
*Nota:* Ejemplo demostrativo de lo que realizará el algoritmo diseñado.

## Trabajo colaborativo

El componente de trabajo colaborativo se enfoca en la asignación de objetivos a cada uno de los robots, la ejecución de movimientos que los robots deben realizar para alcanzar los objetivos asignados, y así como la verificación y cumplimiento de las peticiones de ayuda colaborativa, bajo el diagrama de flujo de la Figura 83.

**Figura 83**

*Diagrama de flujo del componente de trabajo colaborativo*



*Nota:* Diseño del autor.

El algoritmo primero comprueba que existan objetos fuera del área de objetos clasificados, en caso de no existir objetos fuera de esta área esto significa que se han clasificado todos los objetos de interés seleccionados por el usuario o que no existe ningún objeto que se pueda clasificar y esto conlleva a la finalización del algoritmo. Para el caso de que si existan

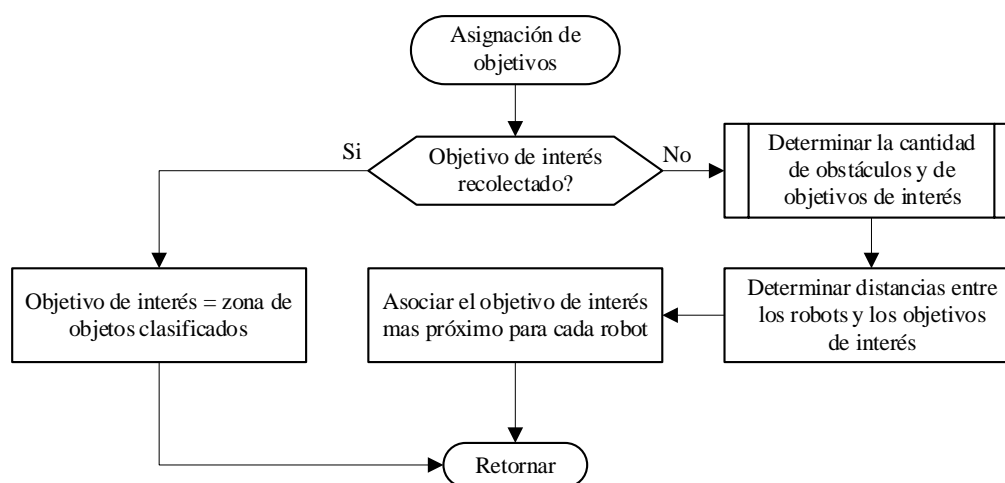
objetos fuera de dicha área el algoritmo procede a evaluar la existencia de alguna petición de ayuda colaborativa, las peticiones de ayuda pueden provenir de cualquiera de los dos robots.

Para no tener inconvenientes en la toma de decisiones dentro del algoritmo se introdujo nivel de rango entre los robots, en donde el robot con rango 1 tiene más peso en las tomas de decisiones dentro del algoritmo. En el presente proyecto se asigna con rango 1 al robot móvil y con rango 2 al robot humanoide.

Luego el algoritmo procede con la asignación de los objetivos de interés a cada uno de los robots de acuerdo a la petición ingresada por el usuario, para lo cual utiliza la estructura que se muestra en la Figura 84.

**Figura 84**

*Estructura para la asignación de objetivos*



*Nota:* Diseño del autor.

Antes de realizar la asignación de objetivos de interés a cada robot se valida si el objetivo de interés ya ha sido recogido previamente. Una vez recolectado el objeto de interés el

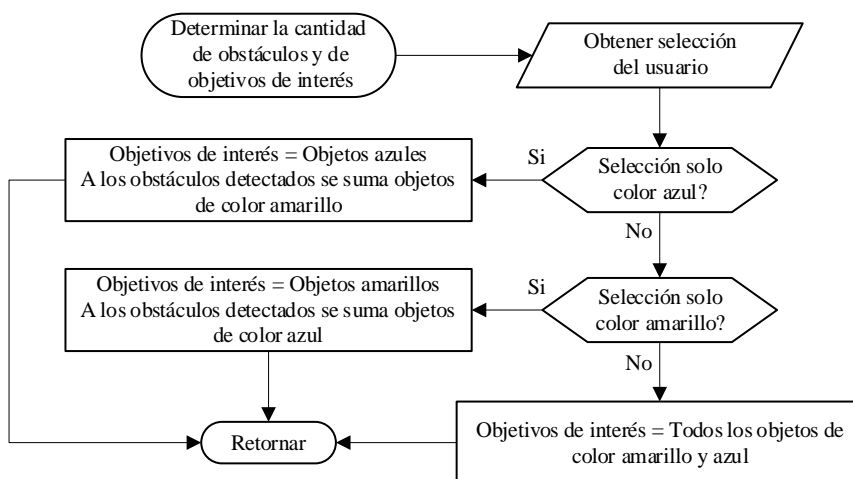


siguiente paso es trasladarlo al área de objetos clasificados, para lo cual el nuevo objetivo de interés del robot es dicha área.

Algo importante en este punto es determinar la cantidad de objetos de interés, así como la cantidad de obstáculos detectados en el espacio de trabajo, para esto el algoritmo se encuentra basado en la estructura de la Figura 85. En esta estructura se obtiene la selección ingresada por el usuario a través de la interfaz gráfica diseñada, los valores posibles que puede seleccionar el usuario son tres: objetos de color azul, objetos de color amarillo y objetos de color azul y amarillo.

**Figura 85**

*Estructura para determinar cantidad de obstáculos y de objetos de interés*



*Nota:* Diseño del autor.

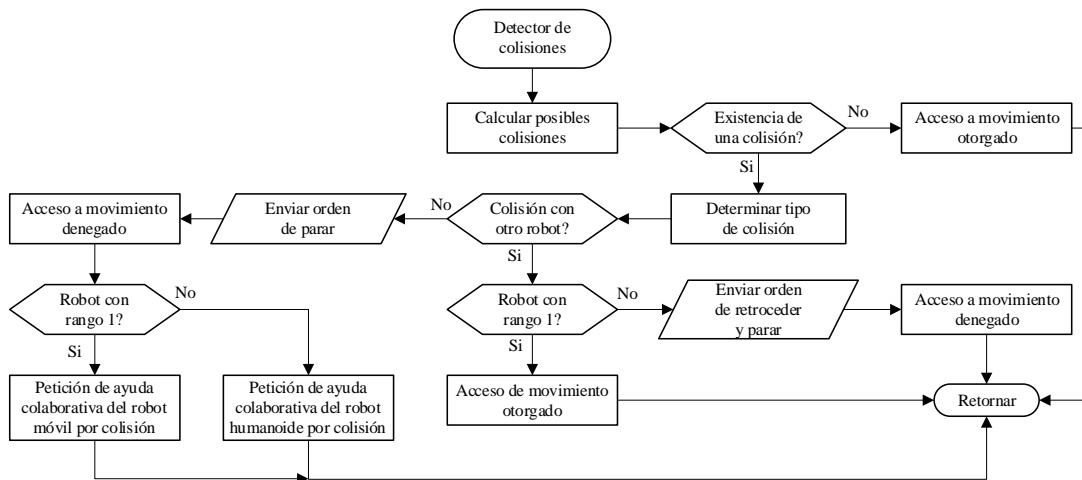
Una vez determinado la cantidad de objetivos de interés se procede a asociar el objetivo de interés más próximo a cada robot. Primero se calcula la distancia que existe entre cada robot con cada uno de los objetivos de interés, luego para asociar se lo hace en relación a la distancia mínima entre las antes calculadas.

Luego de la asignación de objetivos de interés el algoritmo usa la estructura de la Figura 86, para detectar y evitar las posibles colisiones que se puedan dar durante los desplazamientos de los robots con el entorno. Para determinar si existe una colisión con algún objeto, se crea virtualmente una circunferencia que envuelva a cada uno de los robots, y si algún objeto llega a estar dentro de este espacio significa que hay una colisión.

Matemáticamente la detección de colisiones se lo realiza determinando si existen puntos de intersección entre dos circunferencias, para lo cual es necesario resolver el sistema de ecuaciones de segundo orden formado por sus ecuaciones.

**Figura 86**

*Estructura que permite detectar y evitar colisiones*



*Nota:* Diseño del autor.

Partiendo de la ecuación de la circunferencia que se muestra a continuación:

$$(x - h)^2 + (y - k)^2 = r^2 \quad (29)$$

donde  $h$  y  $k$  son las coordenadas del centro de la circunferencia en el eje  $x$  e  $y$  respectivamente, mientras que  $r$  es el radio de la circunferencia. Con la ecuación anterior se forma el siguiente sistema de ecuaciones:

$$\begin{cases} (x - h_1)^2 + (y - k_1)^2 = r_1^2 \\ (x - h_2)^2 + (y - k_2)^2 = r_2^2 \end{cases} \quad (30)$$

donde  $h_1, k_1, r_1$  son los valores conocidos de la circunferencia 1, mientras que  $h_2, k_2, r_2$  son los valores conocidos de la circunferencia 2. Expandiendo la ecuación anterior se obtiene:

$$\begin{cases} x^2 + y^2 - 2xh_1 - 2yk_1 + h_1^2 + k_1^2 - r_1^2 = 0 \\ x^2 + y^2 - 2xh_2 - 2yk_2 + h_2^2 + k_2^2 - r_2^2 = 0 \end{cases} \quad (31)$$

sustituyendo  $h_1^2 + k_1^2 - r_1^2 = C_1$  y  $h_2^2 + k_2^2 - r_2^2 = C_2$  en la ecuación anterior se tiene:

$$\begin{cases} x^2 + y^2 - 2xh_1 - 2yk_1 + C_1 = 0 \\ x^2 + y^2 - 2xh_2 - 2yk_2 + C_2 = 0 \end{cases} \quad (32)$$

resolviendo la ecuación anterior usando el método de suma y resta se obtiene:

$$\begin{array}{r} x^2 + y^2 - 2xh_1 - 2yk_1 + C_1 = 0 \\ -x^2 - y^2 + 2xh_2 + 2yk_2 - C_2 = 0 \\ \hline 2(h_2 - h_1)x + 2(k_2 - k_1)y + C_1 - C_2 = 0 \end{array} \quad (33)$$

despejando  $x$  de la ecuación anterior y sustituyendo  $C_1 - C_2 = C_3$  se tiene:

$$x = -\frac{C_3}{2(h_2 - h_1)} - \frac{(k_2 - k_1)}{(h_2 - h_1)}y \quad (34)$$

sustituyendo  $-\frac{C_3}{2(h_2 - h_1)} = C_4$  y  $\frac{(k_2 - k_1)}{(h_2 - h_1)} = C_5$  se tiene:

$$x = C_4 - C_5y \quad (35)$$

reemplazando la ecuación (35) en la primera ecuación del sistema de ecuaciones (32), resolviendo y agrupando se tiene:

$$(C_5^2 + 1)y^2 + (2C_5h_1 - 2C_4C_5 - 2k_1)y + (C_4^2 - 2C_4h_1 + C_1) \quad (36)$$

luego se encuentra las raíces del polinomio usando la formula general:

$$y = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad (37)$$

donde  $a = C_5^2 + 1$ ,  $b = 2C_5h_1 - 2C_4C_5 - 2k_1$  y  $c = C_4^2 - 2C_4h_1 + C_1$ . Encontrando las raíces  $y_1, y_2$  estas se reemplazan en la ecuación (35) y se encuentra los valores  $x_1, x_2$ , donde el par  $(x_1, y_1)$  y  $(x_2, y_2)$  son las coordenadas de los puntos de intersección.

La condición para que exista intersección entre dos circunferencias es que  $(b^2 - 4ac)$  sea mayor o igual a cero, mientras que si  $(b^2 - 4ac)$  es menor que cero esto quiere decir que no existe intersección entre dos circunferencias. Utilizando este procedimiento entre cada uno de los robots con los objetos detectados se determina la existencia de posibles colisiones, para así posteriormente realizar la toma de decisiones.

Una vez que se haya detectado o evitado alguna colisión, el siguiente proceso es determinar el tipo de movimiento que tiene que realizar cada uno de los robots para llegar a los diferentes lugares dentro del área de trabajo, para lo cual el algoritmo usa la estructura de la Figura 87. En esta estructura la primera validación es si el robot tiene acceso a moverse, lo cual viene dado por el proceso anterior, en caso de tenerlo, se calcula la orientación del objetivo con respecto al sistema de referencia de cada uno de los robots utilizando la ecuación (38).

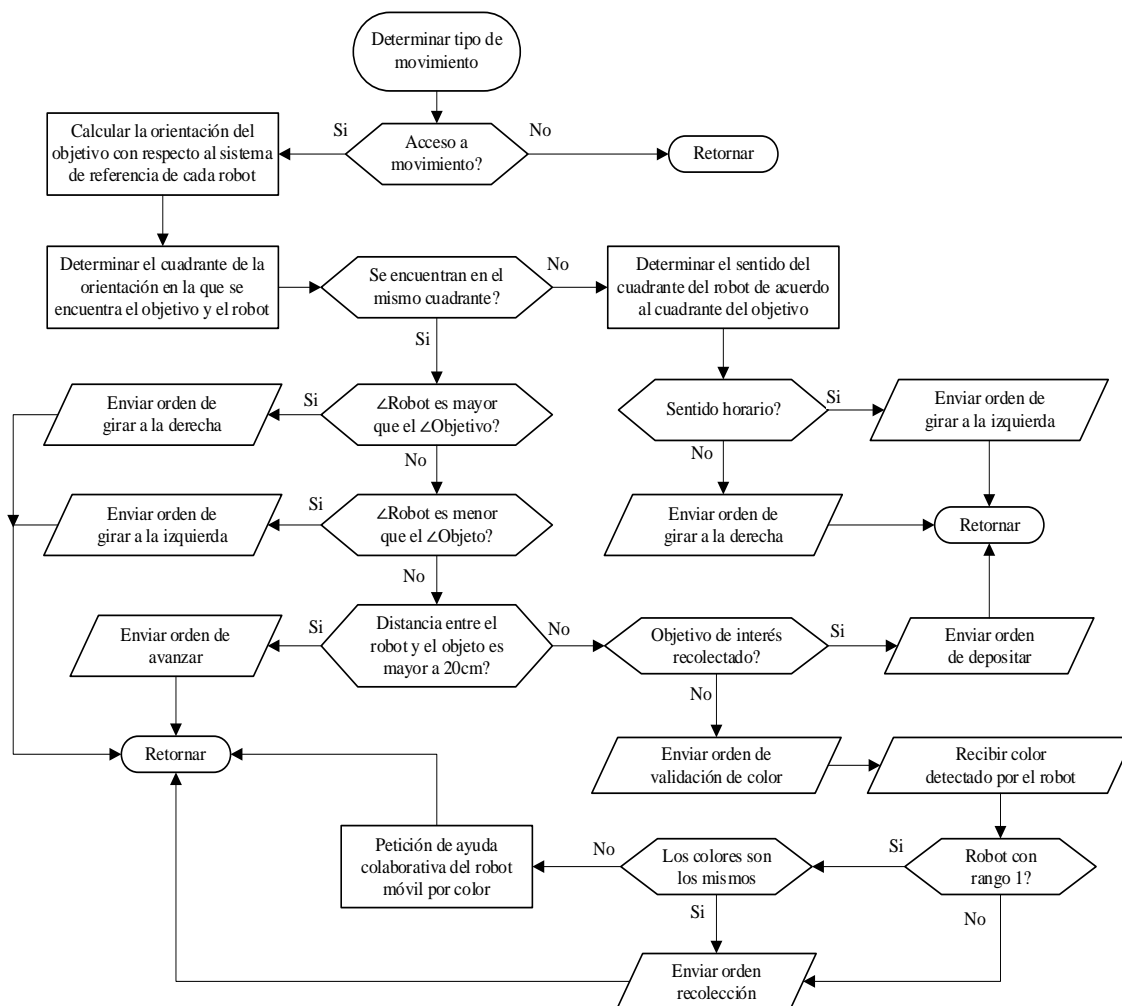
$$\begin{bmatrix} O^B \\ 1 \end{bmatrix} = \begin{bmatrix} R_A^B & t_A^B \\ 0 & 1 \end{bmatrix} \begin{bmatrix} O^A \\ 1 \end{bmatrix} \quad (38)$$

donde  $R_A^B$  tiene los mismo valores antes mencionados,  $t_A^B$  es el vector de traslación correspondiente a las coordenadas del centroide de cada uno de los robots como se detalló anteriormente y  $O^A$  son las coordenadas conocidas del objetivo de interés con respecto al sistema A.

Luego de determinar la orientación del objetivo de interés se procede a determinar en qué sentido tiene que girar cada uno de los robots para igualar las orientaciones de ambos, una vez que se encuentre en la misma orientación los robots empiezan a avanzar hasta llegar a la posición en la que se encuentra el objetivo de interés, verifica el color detectado por el agente 3 y en caso de ser iguales se procede con la recolección y el traslado hacia el área de objetos clasificados.

### **Figura 87**

*Estructura para determinar el tipo de movimiento que tiene que ejecutar cada robot*



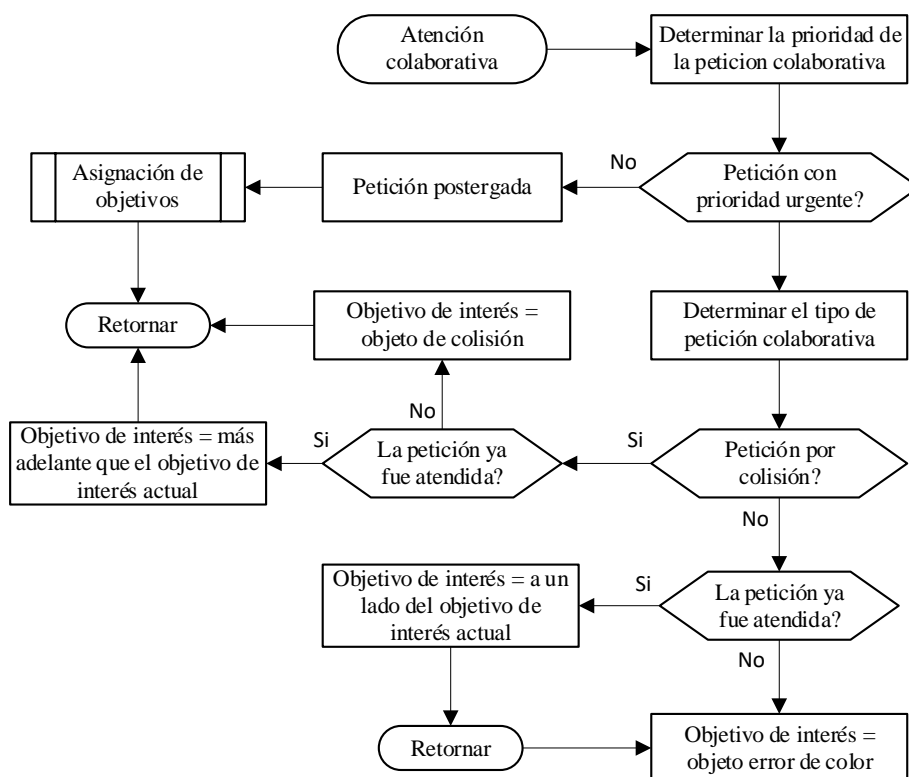
*Nota:* Diseño del autor.

Finalmente, la parte del algoritmo que permite realizar el trabajo colaborativo entre ambos robots se encuentra basado en la estructura de la Figura 88. La atención a peticiones de ayuda colaborativa se da de acuerdo a la prioridad de la petición, donde una prioridad urgente se atiende de manera inmediata, mientras que una prioridad normal se atiende cuando el robot haya finalizado su tarea actual.

También se analiza el tipo de petición ya sea por una posible colisión con varios obstáculos que puede provenir tanto del robot móvil como del robot humanoide, y el otro tipo de petición se da por error en la validación de color del robot móvil.

**Figura 88**

*Estructura que permite la realización del trabajo colaborativo*



*Nota:* Diseño del autor.

### Comunicación inalámbrica

Este componente se encarga de establecer la comunicación bilateral con todos los robots, otorgándoles los comandos de movimiento necesario para aproximarlos a los objetos por recolectar, esto ya se lo detalló en la capítulo anterior.

La comunicación usando el ZIG2Serial y el USB a Serial, se logró con la siguiente configuración: puerto COM2, velocidad de transmisión de 57600 y un tiempo de espera de 1 segundo. El computador y la RPZW se conectaron a una red de celular 4G, donde se asignaron direcciones IP estáticas y utilizando la misma IP, pero diferentes puertos se crean los servicios para la comunicación bilateral entre ellos, tal como se puede observar en la Tabla 20.

**Tabla 20**

*Direcciones IP y puertos utilizados*

<b>Dispositivo</b>	<b>Dirección - Puerto</b>
Computador - Servidor 1	192.168.43.149:9000
Computador - Servidor 2	192.168.43.149:7000
RPZW	192.168.43.243

*Nota:* Es necesario conocer las direcciones IP asignadas a los dispositivos, para poder alzar la red de comunicación del MAS.



## Capítulo V Pruebas y resultados

### Pruebas

Para la realización de las pruebas y el análisis de los resultados obtenidos de la celda colaborativa robotizada, se utilizó la metodología de diseño de experimentos (DoE) que es una técnica estadística que se basa en la creación de una serie de experimentos de forma que con el mínimo número de pruebas y repeticiones se consiga extraer información útil.

#### ***Prueba 1: Análisis de la iluminación dentro de la celda***

La primera prueba consiste en analizar la cantidad de objetos que algoritmo pudo detectar con las diferentes intensidades de iluminación que posee la celda internamente. Para realizar esta prueba se colocan a los robots y los objetos a clasificar de manera aleatoria dentro del área de trabajo, y manteniendo esas posiciones se hace la variación entre los tres tipos de intensidades de iluminación que tienen las lámparas: baja, media y alta. Posterior se aplica cada uno de los filtros de color que el algoritmo utiliza para detectar los objetos.

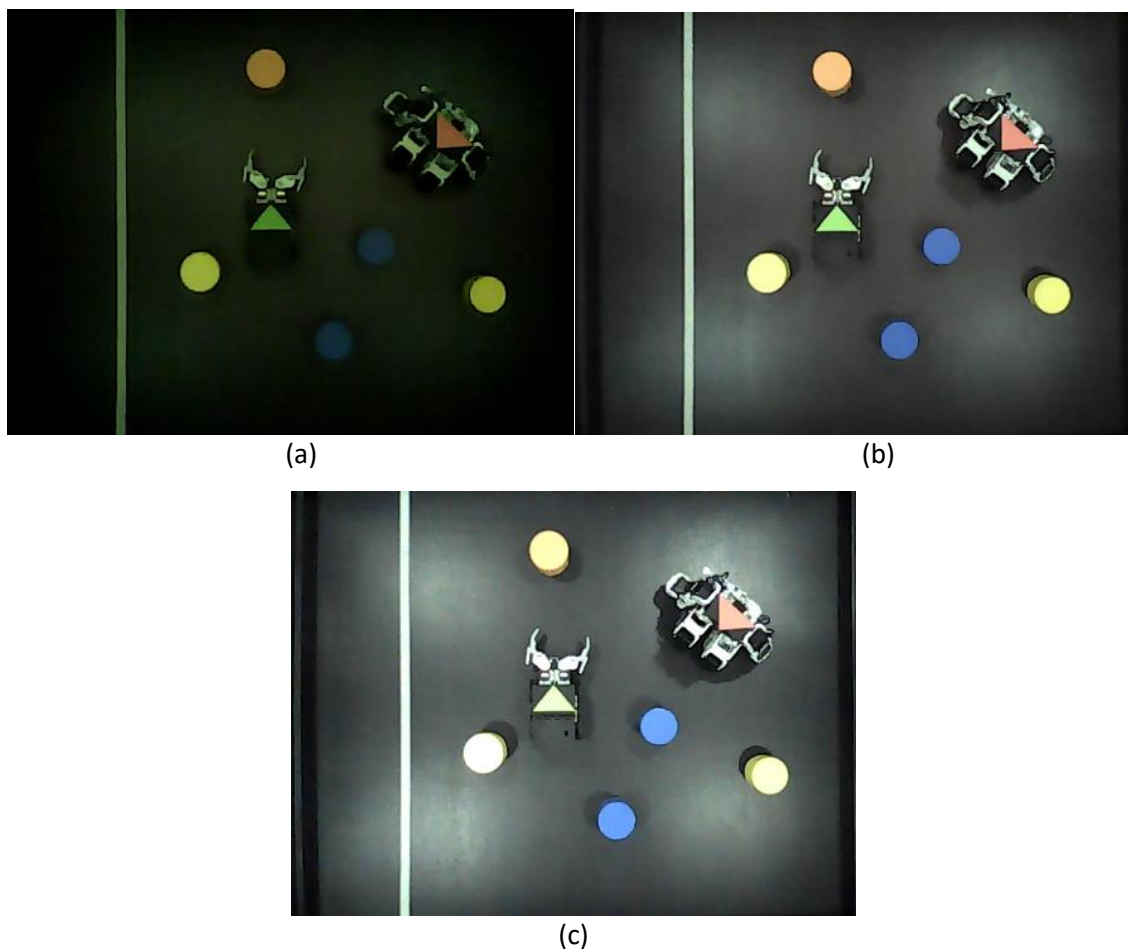
En este caso se tiene los siguientes factores con sus respectivos niveles:

- Mascara de color: Amarillo, Azul, Naranja, Rojo, Verde
- Intensidad de iluminación: Baja, Media, Alta

En la Figura 89-(a) se puede observar la prueba realizada para una intensidad de iluminación baja, en la Figura 89-(b) se observa la prueba realizada para una intensidad de iluminación media y finalmente en la Figura 89-(c) se observa la prueba para una intensidad de iluminación alta.

**Figura 89**

*Prueba 1 con una intensidad de iluminación baja (a), intensidad de iluminación media (b) y con una intensidad de iluminación alta (c).*



*Nota:* En las figuras se puede observar que, las diferentes intensidades de iluminación con la que cuenta la celda afectan directamente a los colores de los objetos dentro de la celda.

### ***Prueba 2: Error entre el área calculada y el área real***

Esta prueba consiste en calcular el error entre el área calculada por el algoritmo y el área real de los marcadores de los robots. Para la ejecución de esta prueba se colocó a los dos robots dentro del área de trabajo uno a la vez, y se los roto en los cuatro cuadrantes.

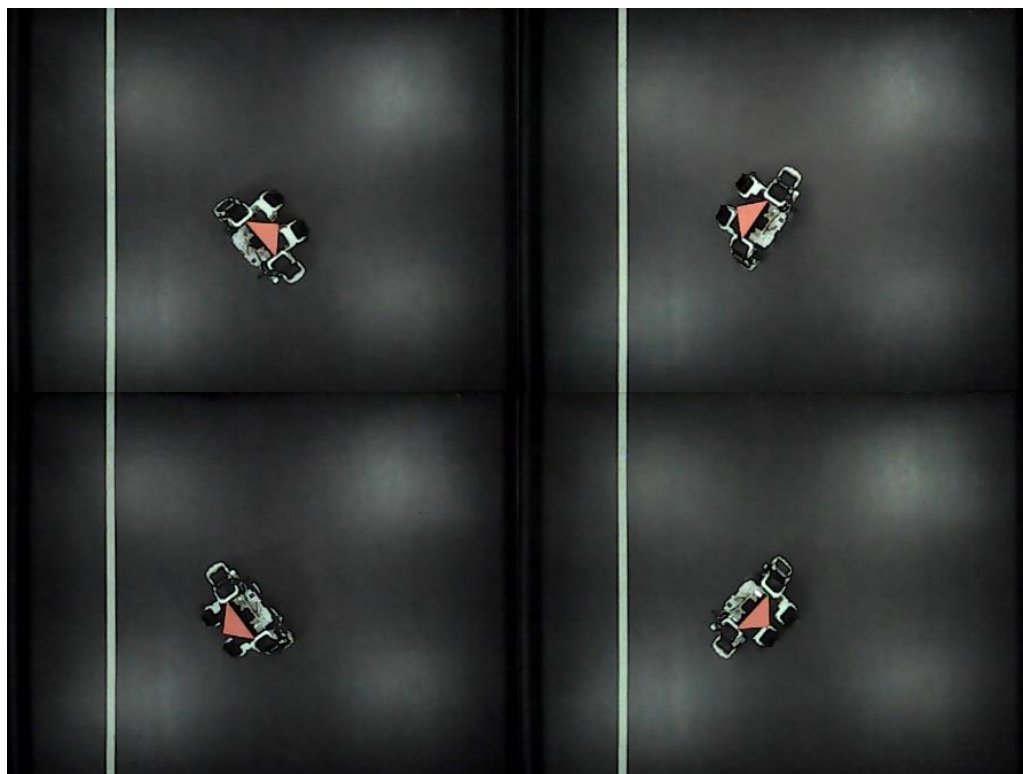
En este caso se tienen los siguientes factores con sus respectivos niveles:

- Marcador del robot: Humanoide, Móvil.
- Cuadrantes: I, II, III, IV

También se realiza un análisis cualitativo de la orientación del robot que el algoritmo calcula con respecto a los marcadores de los robots. En la Figura 90 se puede observar las pruebas realizadas con el marcador del robot humanoide en cada uno de los cuadrantes y de igual manera en la Figura 91 se observa las pruebas realizadas con el marcador del robot móvil.

### Figura 90

*Prueba 2 con el marcador robot humanoide rotado en los cuatro cuadrantes*



*Nota:* El objetivo principal de rotar al robot en los cuatro cuadrantes es para saber si el algoritmo calcula la misma área en diferentes posiciones.

**Figura 91**

Prueba 2 con el marcador robot móvil rotado en los cuatro cuadrantes



*Nota:* El objetivo principal de rotar al robot en los cuatro cuadrantes es para saber si el algoritmo calcula la misma área en diferentes posiciones.

***Prueba 3: Error entre la distancia calculada y la distancia real***

Para esta prueba se calcula el error que existe entre la distancia calculada desde el centro del marcador del robot hacia el centro del objeto de interés detectado los cuales se encuentra localizados dentro del área de trabajo. Para realizar esta prueba se colocó a los robots uno a la vez de forma aleatoria dentro del espacio de trabajo, y se ubicó en forma aleatoria los objetos.

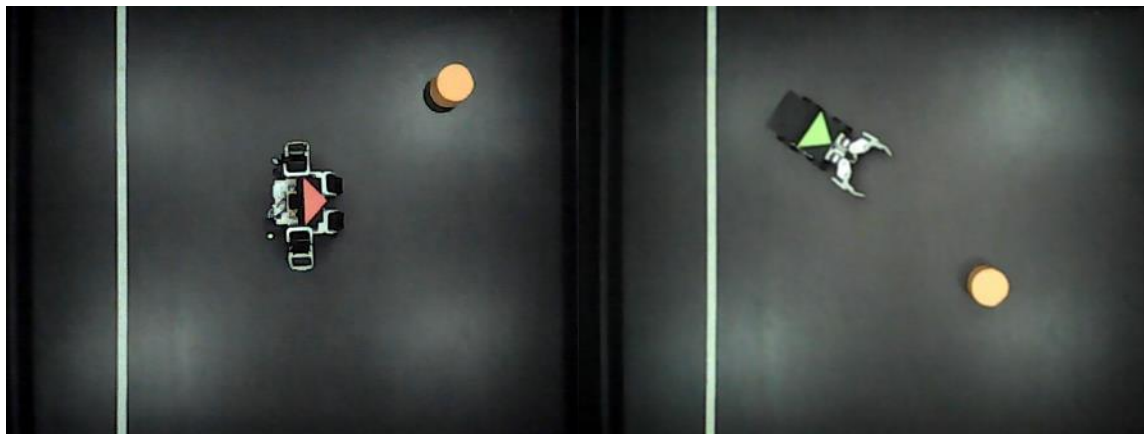
En este caso se tienen los siguientes factores con sus respectivos niveles:

- Marcador del robot: Humanoide, Móvil.
- Cantidad de objetos: 1, 2, 3, 4 y 5.

Además, en esta prueba se hace un análisis cualitativo sobre la asignación correcta de objetivos al robot, en donde el algoritmo hace la asignación en relación al objeto más cercano al robot, para lo cual solo se consideró los casos donde la cantidad de objetos es mayor a 2. Desde la Figura 92 hasta la Figura 96 se puede observar las pruebas realizadas tanto con el robot humanoide como con el robot móvil.

### Figura 92

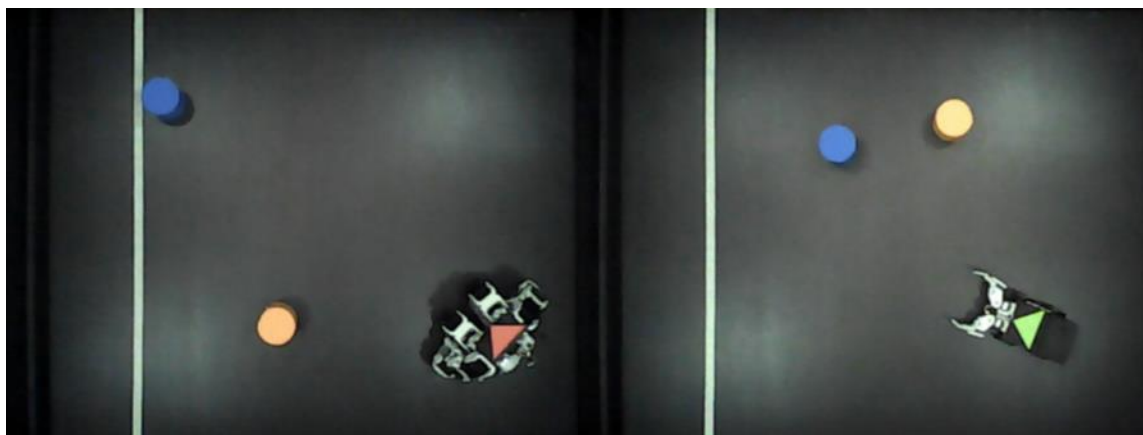
*Prueba 3 con los robots y un objeto*



*Nota:* Se toma el mismo objeto para calcular la distancia hasta los robots.

**Figura 93**

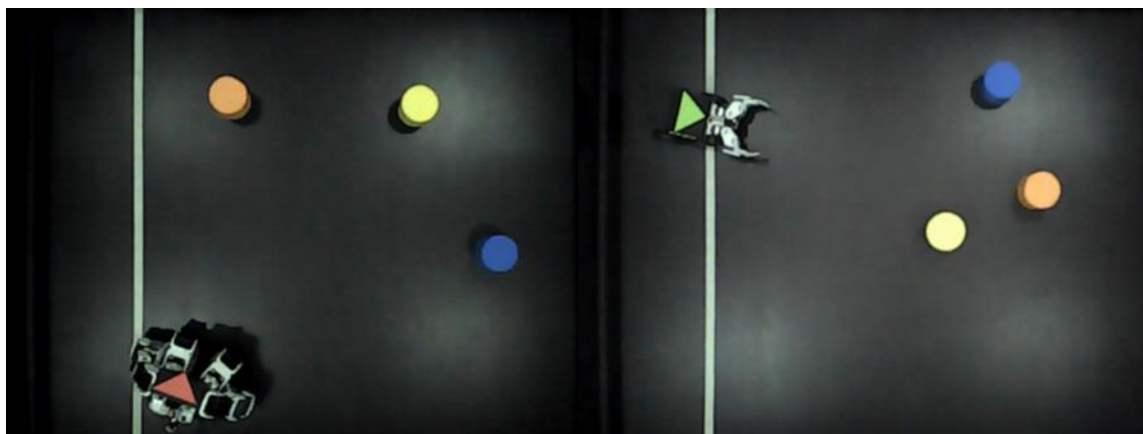
*Prueba 3 con los robots y dos objetos*



*Nota:* Se toma los mismos objetos para calcular la distancia hasta los robots.

**Figura 94**

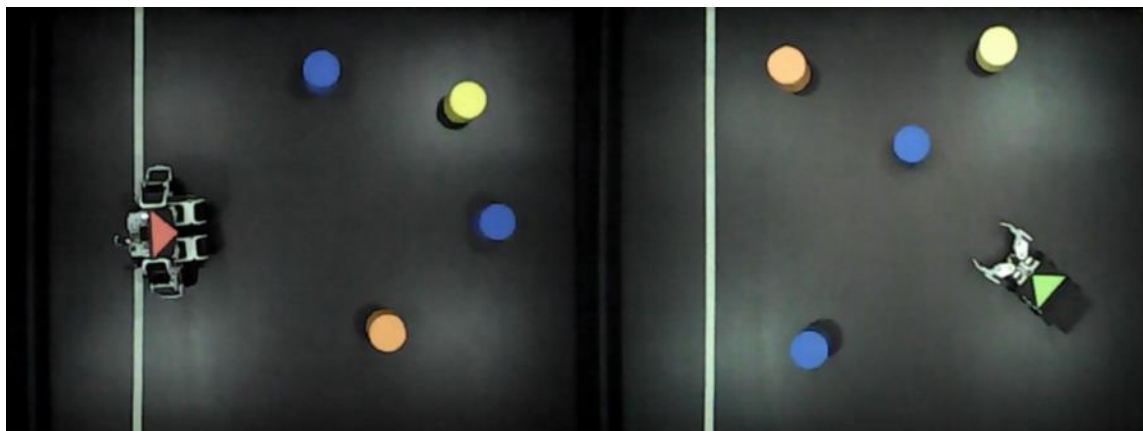
*Prueba 3 con los robots y tres objetos*



*Nota:* Se toma los mismos objetos para calcular la distancia hasta los robots.

**Figura 95**

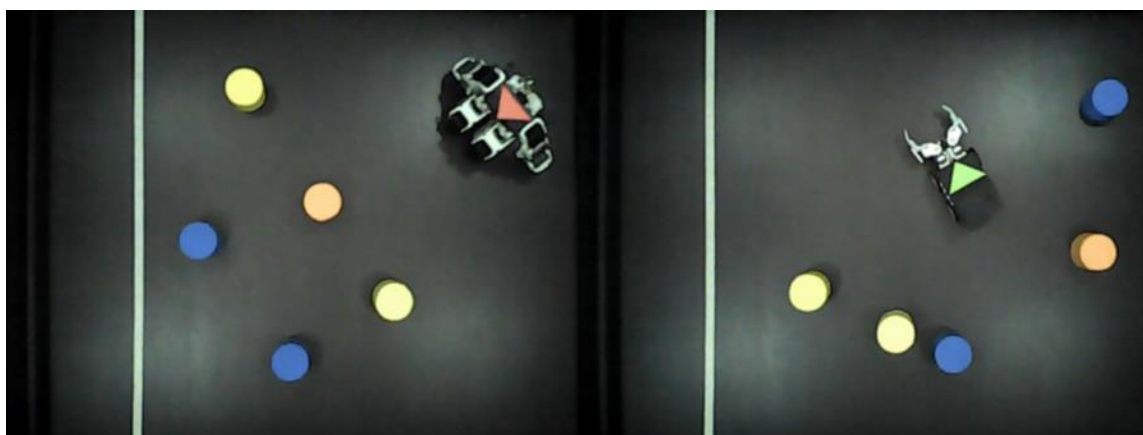
*Prueba 3 con los robots y cuatro objetos*



*Nota:* Se toma los mismos objetos para calcular la distancia hasta los robots.

**Figura 96**

*Prueba 3 con los robots y cinco objetos*



*Nota:* Se toma los mismos objetos para calcular la distancia hasta los robots.

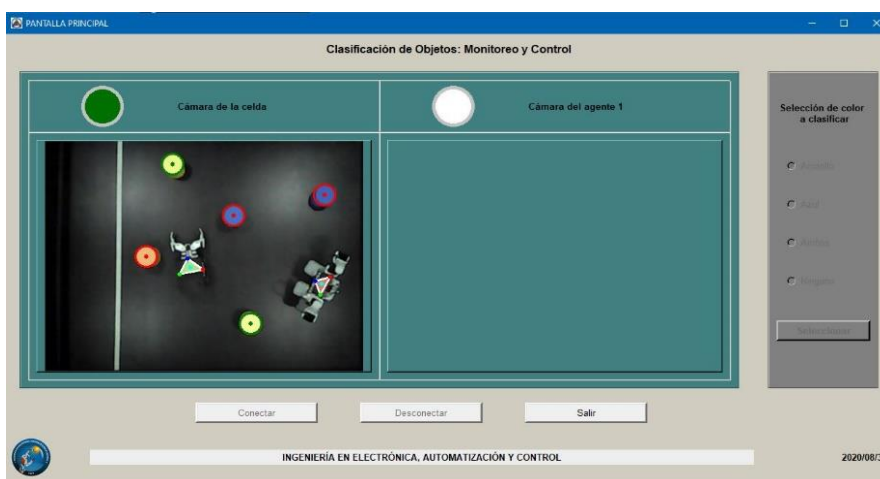
#### ***Prueba 4: Análisis del trabajo colaborativo***

La última prueba consiste en analizar el trabajo colaborativo que se realiza entre los robots por el surgimiento de peticiones de ayuda colaborativa, además se analiza los tiempos que demoran los robots en completar la tarea asignada. La primera parte de esta prueba se

plantea tres escenarios de acuerdo a los objetos a clasificar que puede ser: solo objetos de color amarillo, solo objetos de color azul y objetos de color amarillo y azul, los cuales son elegidos a través de la interfaz de usuario.

**Figura 97**

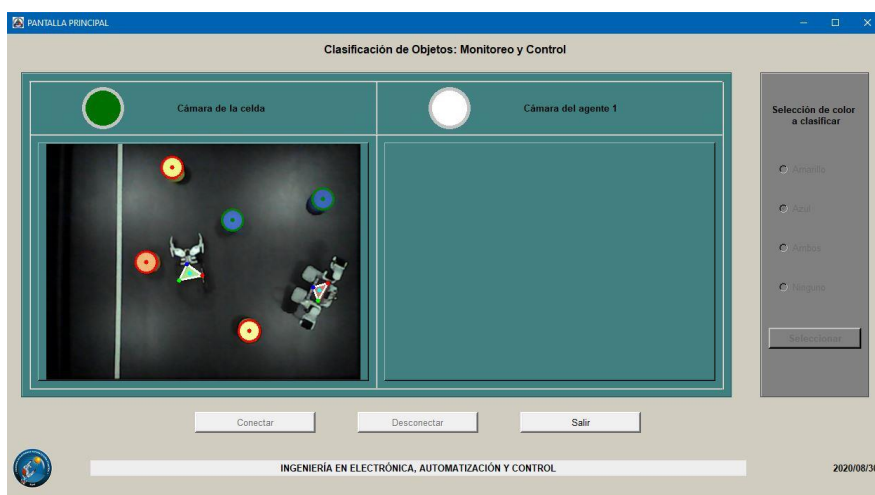
*Prueba 4 selección de objetos amarillos.*



*Nota:* La selección de los objetos a clasificar viene por parte del usuario en la interfaz gráfica.

**Figura 98**

*Prueba 4 selección de objetos azules*

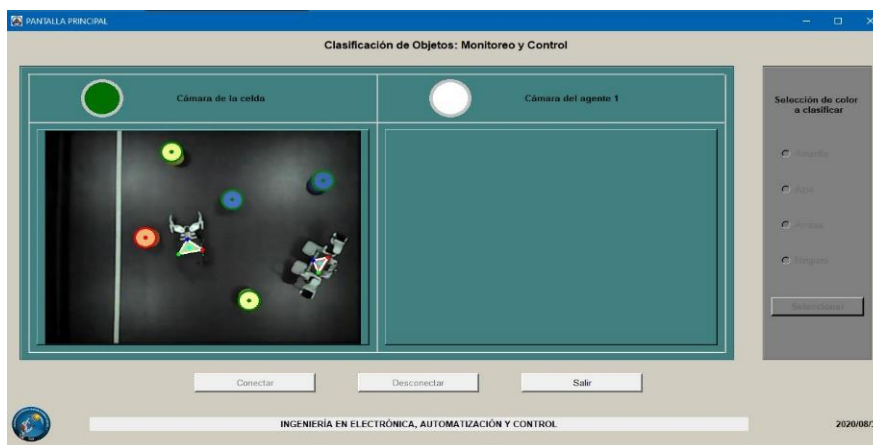


*Nota:* La selección de los objetos a clasificar viene por parte del usuario en la interfaz gráfica.



**Figura 99**

*Prueba 4 selección de objetos amarillos y azules*



*Nota:* La selección de los objetos a clasificar viene por parte del usuario en la interfaz gráfica.

El procedimiento que se realiza en esta prueba es el siguiente:

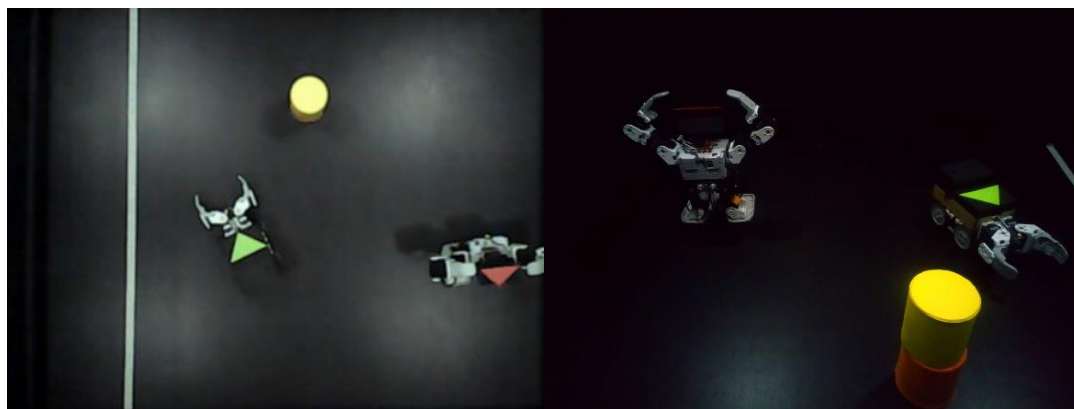
- En la interfaz se selecciona el color que se desee clasificar.
- El algoritmo busca, detecta, muestra y asigna el objeto de interés más próximo al robot.
- Una vez asignado el objeto de interés el robot procede a desplazarse a su ubicación.
- Para el caso del robot móvil cuando este se encuentre a una distancia de 15cm del objeto de interés este se detiene, se hace la validación del color del objeto, en caso afirmativo el robot agarra el objeto de interés y se desplaza a la zona de objetos clasificados, al llegar a la zona de objetos clasificados el robot suelta el objeto y en caso de haber otro objeto a clasificar procede a desplazarse hacia la nueva ubicación y se repite el proceso hasta que no queden más objetos.
- Para el caso del robot humanoide cuando este se encuentre a una distancia de 10cm del objeto de interés este se detiene, se activa la cámara y se detecta la posición del objeto

a recolectar, de acuerdo a la posición en la que se encuentre el objeto el robot procede a recogerlo y empieza a desplazarse hacia la zona de objetos clasificados, al llegar a la zona de objetos clasificados el robot deja el objeto en el suelo y de igual manera si hay más objetos por clasificar este procede a desplazarse a su ubicación hasta que ya no haya más objetos por clasificar.

La segunda parte de esta prueba se recrea los escenarios en los que se requiere del trabajo colaborativo entre los dos robots: el de objetos superpuestos como en la Figura 100 y el de obstáculos que le impiden llegar al objeto de interés como en la Figura 101.

#### **Figura 100**

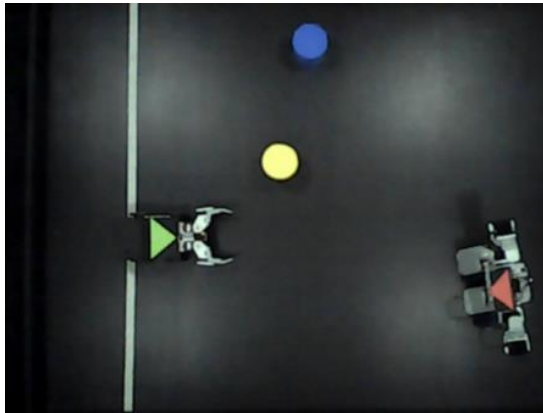
*Prueba 4 de petición de ayuda colaborativa por objetos superpuestos*



*Nota:* La figura de la izquierda vista en planta y la figura de la derecha vista lateral.

**Figura 101**

*Prueba 4 de petición de ayuda colaborativa por colisión del robot móvil con algún objeto*



*Nota:* Figura de la izquierda muestra un obstáculo frente al robot humanoide y la figura de la derecha muestra un obstáculo frente al robot móvil.

En esta prueba cuando se da el caso de petición de ayuda por objetos superpuestos el procedimiento que se realiza es el siguiente:

- Se selecciona el color del objeto a clasificar.
- El robot móvil procede a desplazarse hacia la ubicación del objeto asignado.
- Al llegar realiza la validación del color del objeto a clasificar y aquí los colores no son los mismos.
- Al no ser los mismos colores el robot móvil hace una petición de ayuda al robot humanoide y se aleja de la ubicación del objeto.
- El robot humanoide recibe la petición de ayuda y empieza a desplazarse hacia la ubicación de la petición.
- Al llegar a la ubicación el robot humanoide realiza la validación del color del objeto, lo recoge, empieza a desplazarse para atrás, lo deposita en el suelo y se aleja.

- Cuando el robot deja al objeto en el suelo el robot móvil empieza a desplazarse hacia la nueva ubicación.
- Se repite la validación del color y al ser el color correcto procede a llevarlo a la zona de objetos clasificados.

En esta prueba cuando se da el caso de petición de ayuda por colisión el procedimiento que se realiza es el siguiente:

- Se selecciona el color del objeto a clasificar.
- El robot móvil empieza a desplazarse a su ubicación, pero en su trayecto hacia el objeto de interés, se detecta que este tiene una posible colisión con un obstáculo y el robot móvil envía una petición de ayuda al robot humanoide.
- Al momento de detectar la colisión el robot móvil se detiene de manera inmediata, mientras que el robot humanoide analiza la petición y procede a desplazarse hacia el obstáculo para retirarlo.
- Cuando el robot humanoide llegue a la ubicación del obstáculo, lo recoge y empieza a desplazarse para atrás para retirarlo.
- Al momento de que el robot humanoide este retirando el obstáculo el robot móvil ya no detecta la colisión con el objetivo por lo que reanuda su desplazamiento hacia el objeto de interés.

## Análisis y discusión de resultados

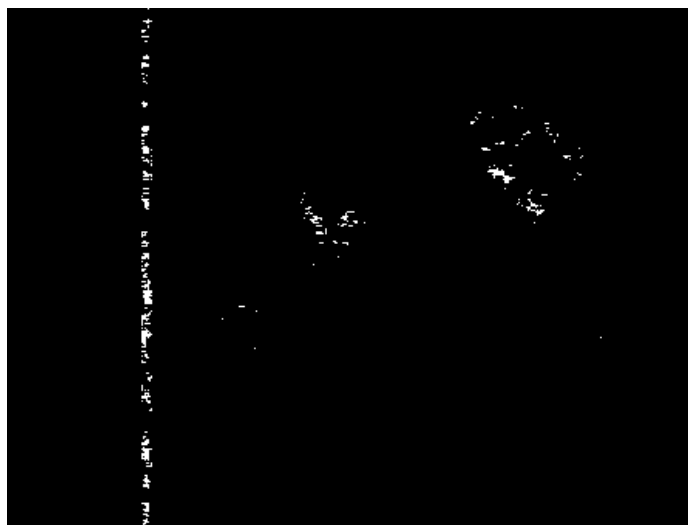
### **Prueba 1**

A continuación, se muestran los resultados obtenidos para cada una de las máscaras de color utilizadas en el desarrollo del trabajo de titulación.

**Mascara de color amarilla.** En la Figura 102 se puede observar el resultado obtenido filtrando el color amarillo del frame de la cámara superior de la celda con una intensidad de iluminación baja, mientras que en la Figura 103 y la Figura 104 se observa el resultado obtenido para una intensidad de iluminación media y alta respectivamente.

### **Figura 102**

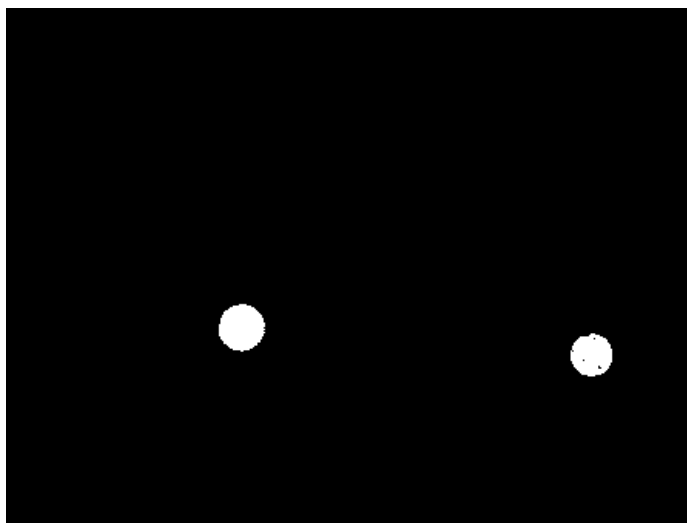
*Resultado obtenido para máscara color amarillo con intensidad de iluminación baja.*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación no se logra diferenciar las tonalidades de colores amarillos, detectando otras formas.

**Figura 103**

*Resultado obtenido para máscara color amarillo con intensidad de iluminación media*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación se logra diferenciar las tonalidades de colores amarillos y detecta los objetos de color amarillo.

**Figura 104**

*Resultado obtenido para máscara color amarillo con intensidad de iluminación alta*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación no se logra diferenciar las tonalidades de colores amarillos, detecta un objeto de color amarillo y al objeto de color naranja lo considera como amarillo.

En la Tabla 21 se realiza una prueba diagnóstica entre la cantidad de objetos detectados con el filtro de color amarillo y las diferentes intensidades de iluminación.

**Tabla 21**

*Cantidad de objetos detectados con la máscara color amarillo*

Intensidad de iluminación	Resultado de la prueba	Diagnostico verdadero	
		Amarillo	No amarillo
Baja	Detectado	0	0
	No detectado	2	5
Media	Detectado	2	0
	No detectado	0	5
Alta	Detectado	1	1
	No detectado	1	4

*Nota:* La información de la tabla muestra un resumen de los resultados obtenidos en este experimento.

Analizando los datos de la Tabla 21 se puede determinar la eficiencia de la prueba para cada una de las intensidades de iluminación, utilizando la siguiente ecuación:

$$Precisión = \frac{VP + VN}{VP + FP + FN + VN} \quad (39)$$

donde VP son los detectados amarillos, VN son los no detectados no amarillos, FP son los detectados no amarillos y FN son los no detectados amarillos.

Para una intensidad de iluminación baja se tiene el siguiente valor de precisión:

$$Precisión = \frac{0 + 5}{0 + 0 + 2 + 5} = \frac{5}{7} \quad (40)$$

$$Precisión = 0.71 \quad (41)$$

Para una intensidad de iluminación media se tiene el siguiente valor de precisión:

$$Precisión = \frac{2 + 5}{2 + 0 + 0 + 5} = \frac{7}{7} \quad (42)$$

$$Precisión = 1 \quad (43)$$

Para una intensidad de iluminación alta se tiene el siguiente valor de precisión:

$$Precisión = \frac{1 + 4}{1 + 1 + 1 + 4} = \frac{5}{7} \quad (44)$$

$$Precisión = 0.71 \quad (45)$$

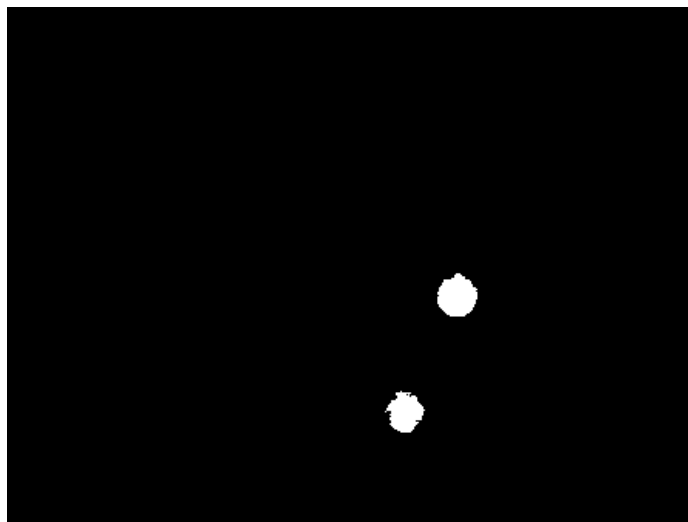
Con los datos calculados de la eficiencia para cada una de las intensidades se puede determinar que, para la máscara de color amarillo el mejor resultado se obtiene cuando se usa la intensidad de iluminación media.

**Mascara de color azul.** En la Figura 105 se puede observar el resultado obtenido filtrando el color azul del frame de la cámara superior de la celda con una intensidad de iluminación baja, mientras que en la Figura 106 y la Figura 107 se observa el resultado obtenido para una intensidad de iluminación media y alta respectivamente.

### Figura 105

*Resultado obtenido para máscara color azul con intensidad de iluminación baja*

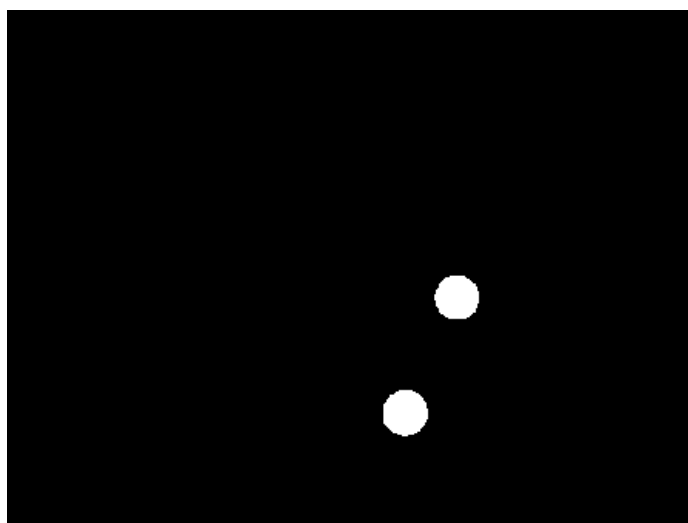




*Nota:* En la figura se puede observar que con esta intensidad de iluminación se logra diferenciar las tonalidades de colores azul, detectando los objetos de color azul pero no en su totalidad.

### **Figura 106**

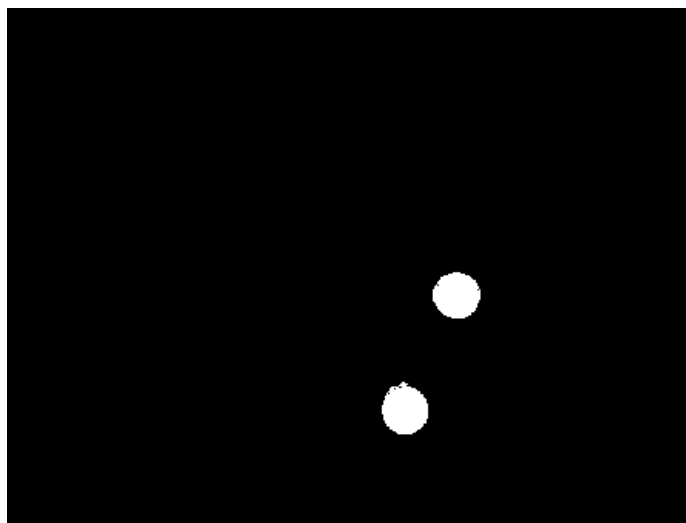
*Resultado obtenido para máscara color azul con intensidad de iluminación media*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación se logra diferenciar las tonalidades de colores azul, detectando los objetos de color azul ya en su totalidad.

**Figura 107**

*Resultado obtenido para máscara color azul con intensidad de iluminación alta*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación se logra diferenciar las tonalidades de colores azul, detectando los objetos de color azul y un poco más.

En la **¡Error! No se encuentra el origen de la referencia.** se realiza una prueba diagnóstica entre la cantidad de objetos detectados con el filtro de color azul y las diferentes intensidades de iluminación.

**Tabla 22**

*Cantidad de objetos detectados con la máscara color azul*

Intensidad de iluminación	Resultado de la prueba	Diagnostico verdadero	
		Azul	No azul
Baja	Detectado	2	0
	No detectado	0	5
Media	Detectado	2	0
	No detectado	0	5
Alta	Detectado	2	0
	No detectado	0	5

*Nota:* La información de la tabla muestra un resumen de los resultados obtenidos en este experimento.

Analizando los datos de la Tabla 22 se puede determinar la eficiencia de la prueba para cada una de las intensidades de iluminación, utilizando la siguiente ecuación:

$$Precisión = \frac{VP + VN}{VP + FP + FN + VN} \quad (46)$$

donde VP son los detectados azules, VN son los no detectados no azules, FP son los detectados no azules y FN son los no detectados azules.

Dado que para los tres tipos de intensidad de iluminación se obtuvo los mismos valores el valor calculado de precisión va a ser el mismo para los tres.

$$Precisión = \frac{2 + 5}{2 + 0 + 0 + 5} = \frac{7}{7} \quad (47)$$

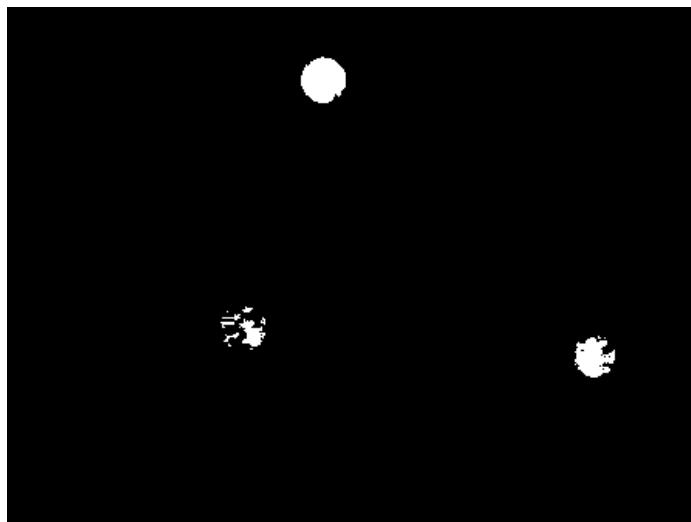
$$Precisión = 1 \quad (48)$$

Con los datos calculados de la eficiencia para cada una de las intensidades se puede determinar que, para la máscara de color azul con cualquiera de las intensidades de iluminación se obtiene el mejor resultado.

**Máscara de color naranja.** En la Figura 108 se puede observar el resultado obtenido filtrando el color naranja del frame de la cámara superior de la celda con una intensidad de iluminación baja, mientras que en la Figura 109 y la Figura 110 se observa el resultado obtenido para una intensidad de iluminación media y alta respectivamente.

#### **Figura 108**

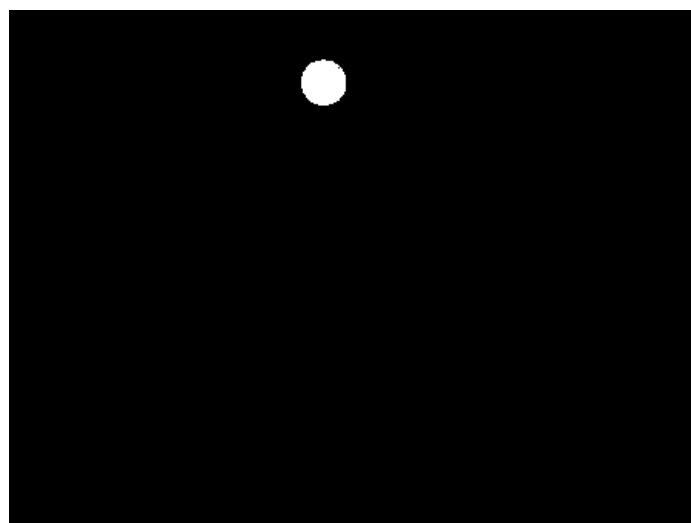
*Resultado obtenido para máscara color naranja con intensidad de iluminación baja*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación no se logra diferenciar las tonalidades de colores naranja, detectando el objeto de color naranja y otros dos objetos que los considera naranja.

### **Figura 109**

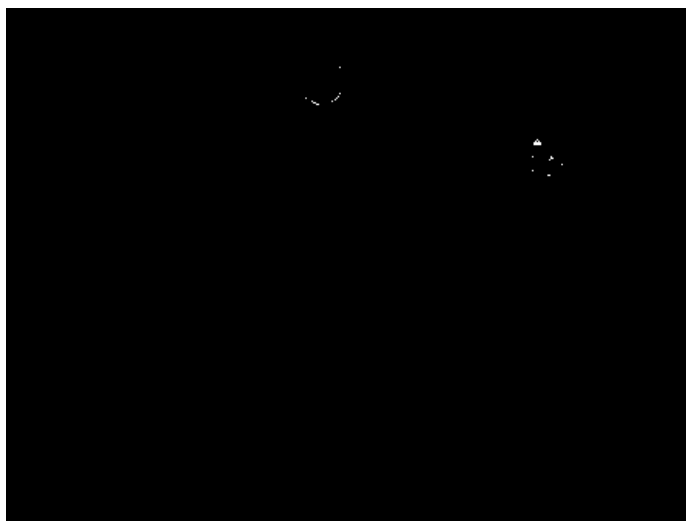
*Resultado obtenido para máscara color naranja con intensidad de iluminación media*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación se logra diferenciar las tonalidades de colores naranja, detectando el objeto de color naranja.

**Figura 110**

*Resultado obtenido para máscara color naranja con intensidad de iluminación alta*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación no se logra diferenciar las tonalidades de colores naranja y no detectando el objeto de color naranja.

En la Tabla 23 se realiza una prueba diagnóstica entre la cantidad de objetos detectados con el filtro de color naranja y las diferentes intensidades de iluminación.

**Tabla 23**

*Cantidad de objetos detectados con la máscara color naranja*

<b>Intensidad de iluminación</b>	<b>Resultado de la prueba</b>	<b>Diagnostico verdadero</b>	
		Naranja	No naranja
Baja	Detectado	1	2
	No detectado	0	4
Media	Detectado	1	0
	No detectado	0	6
Alta	Detectado	0	0
	No detectado	1	6

*Nota:* La información de la tabla muestra un resumen de los resultados obtenidos en este experimento.

Analizando los datos de la Tabla 23 **Tabla 22** se puede determinar la eficiencia de la prueba para cada una de las intensidades de iluminación, utilizando la siguiente ecuación:

$$Precisión = \frac{VP + VN}{VP + FP + FN + VN} \quad (49)$$

donde VP son los detectados naranjas, VN son los no detectados no naranjas, FP son los detectados no naranjas y FN son los no detectados naranjas.

Para una intensidad de iluminación baja se tiene el siguiente valor de precisión:

$$Precisión = \frac{1 + 4}{1 + 2 + 0 + 4} = \frac{5}{7} \quad (50)$$

$$Precisión = 0.71 \quad (51)$$

Para una intensidad de iluminación media se tiene el siguiente valor de precisión:

$$Precisión = \frac{1 + 6}{1 + 0 + 0 + 6} = \frac{7}{7} \quad (52)$$

$$Precisión = 1 \quad (53)$$

Para una intensidad de iluminación alta se tiene el siguiente valor de precisión:

$$Precisión = \frac{0 + 6}{0 + 0 + 1 + 6} = \frac{6}{7} \quad (54)$$

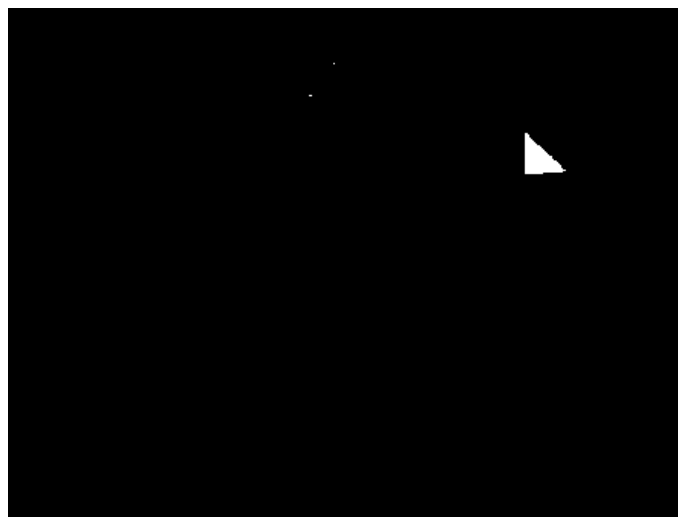
$$Precisión = 0.86 \quad (55)$$

Con los datos calculados de la eficiencia para cada una de las intensidades se puede determinar que, para la máscara de color naranja el mejor resultado se obtiene cuando se usa la intensidad de iluminación media.

**Mascara de color rojo.** En la Figura 111 se puede observar el resultado obtenido filtrando el color rojo del frame de la cámara superior de la celda con una intensidad de iluminación baja, mientras que en la Figura 112 y la Figura 113 se observa el resultado obtenido para una intensidad media y alta respectivamente.

**Figura 111**

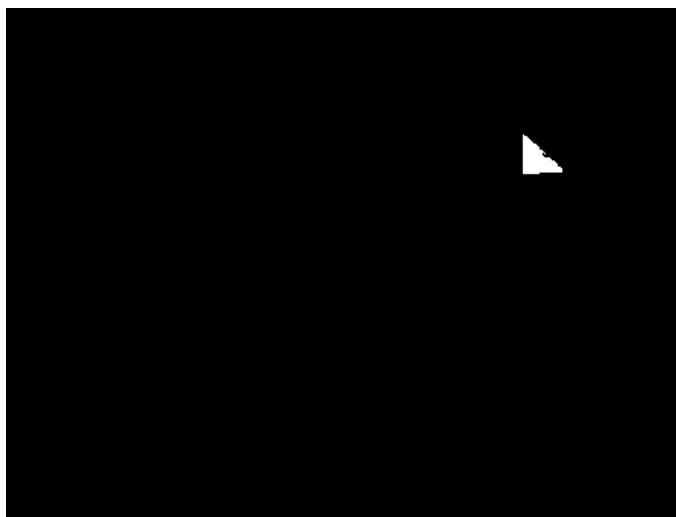
*Resultado obtenido para máscara color rojo con intensidad de iluminación baja*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación se logra diferenciar las tonalidades de colores rojo, detectando la marca de color rojo y por ahí otros puntos rojos.

**Figura 112**

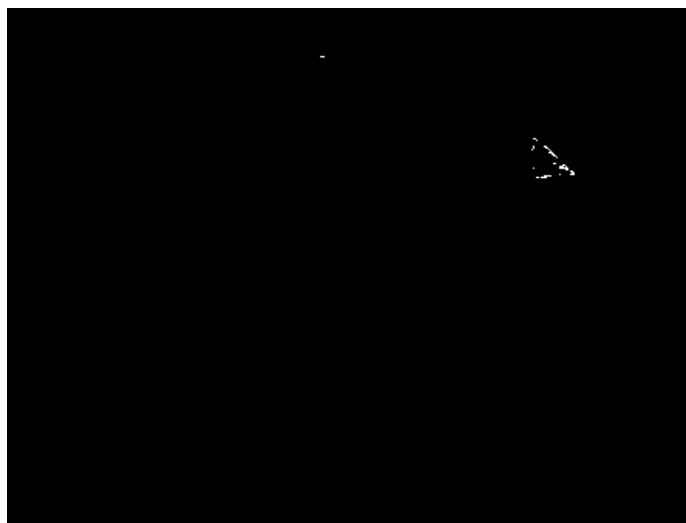
*Resultado obtenido para máscara color rojo con intensidad de iluminación media*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación se logra diferenciar las tonalidades de colores rojo, detectando solo la marca de color rojo.

**Figura 113**

*Resultado obtenido para máscara color rojo con intensidad de iluminación alta*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación no se logra diferenciar las tonalidades de colores rojo y no se detecta bien la marca de color rojo.



En la Tabla 24 se realiza una prueba diagnóstica entre la cantidad de objetos detectados con el filtro de color rojo y las diferentes intensidades de iluminación.

**Tabla 24**

*Cantidad de objetos detectados con la máscara color rojo*

Intensidad de iluminación	Resultado de la prueba	Diagnostico verdadero	
		Rojo	No rojo
Baja	Detectado	1	0
	No detectado	0	6
Media	Detectado	1	0
	No detectado	0	6
Alta	Detectado	0	0
	No detectado	1	6

*Nota:* La información de la tabla muestra un resumen de los resultados obtenidos en este experimento.

Analizando los datos de la Tabla 24 **Tabla 22** se puede determinar la eficiencia de la prueba para cada una de las intensidades de iluminación, utilizando la siguiente ecuación:

$$Precisión = \frac{VP + VN}{VP + FP + FN + VN} \quad (56)$$

donde VP son los detectados rojos, VN son los no detectados no rojos, FP son los detectados no rojos y FN son los no detectados rojos.

Para una intensidad de iluminación baja se tiene el siguiente valor de precisión:

$$Precisión = \frac{1 + 6}{1 + 0 + 0 + 6} = \frac{7}{7} \quad (57)$$

$$Precisión = 1 \quad (58)$$

Para una intensidad de iluminación media se tiene el siguiente valor de precisión:

$$\text{Precisión} = \frac{1 + 6}{1 + 0 + 0 + 6} = \frac{7}{7} \quad (59)$$

$$\text{Precisión} = 1 \quad (60)$$

Para una intensidad de iluminación alta se tiene el siguiente valor de precisión:

$$\text{Precisión} = \frac{0 + 6}{0 + 0 + 1 + 6} = \frac{6}{7} \quad (61)$$

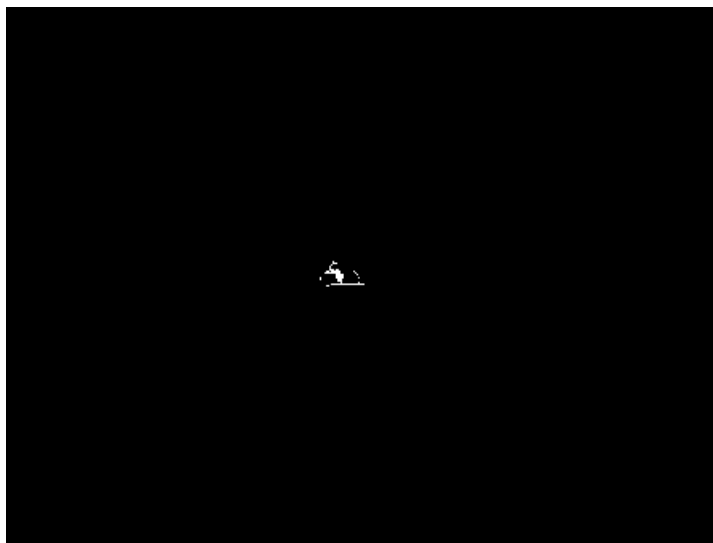
$$\text{Precisión} = 0.86 \quad (62)$$

Con los datos calculados de la eficiencia para cada una de las intensidades se puede determinar que, para la máscara de color rojo el mejor resultado se obtiene usando una intensidad de iluminación baja y media.

**Mascara de color verde.** En la Figura 114 se puede observar el resultado obtenido filtrando el color rojo del frame de la cámara superior de la celda con una intensidad de iluminación baja, mientras que en la Figura 115 y la Figura 116 se observa el resultado obtenido para una intensidad de iluminación media y alta respectivamente.

#### **Figura 114**

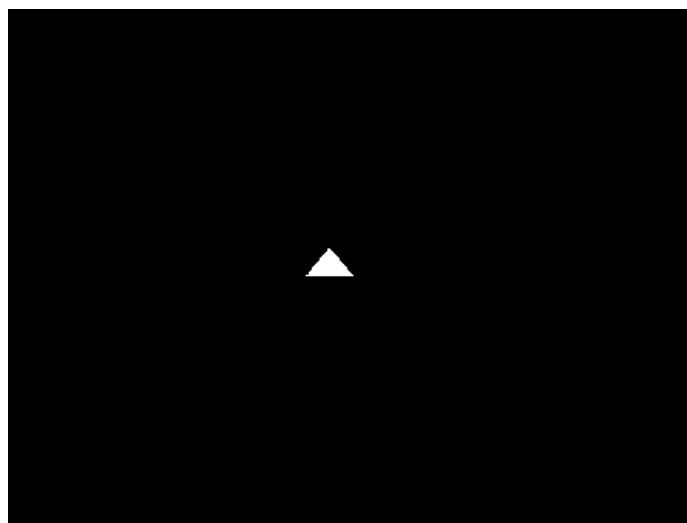
*Resultado obtenido para máscara color verde con intensidad de iluminación baja*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación no se logra diferenciar las tonalidades de colores verde, por lo que no detecta bien la marca de color verde.

**Figura 115**

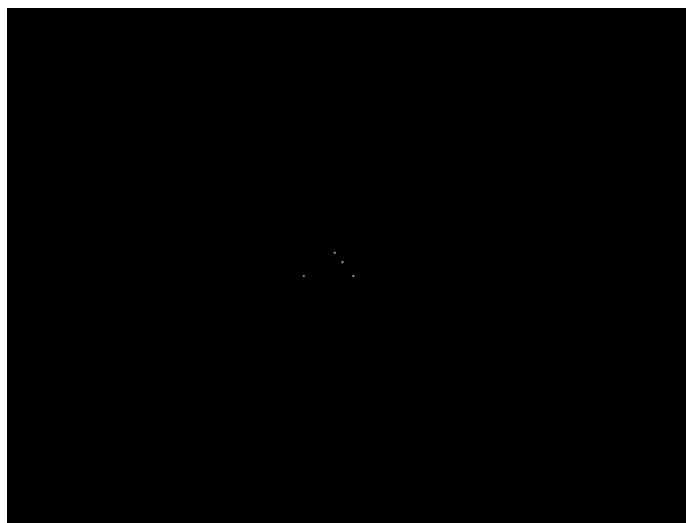
*Resultado obtenido para máscara color verde con intensidad de iluminación media*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación se logra diferenciar las tonalidades de colores verde, detectando la marca de color verde.

**Figura 116**

*Resultado obtenido para máscara color verde con intensidad de iluminación alta*



*Nota:* En la figura se puede observar que con esta intensidad de iluminación no se logra diferenciar las tonalidades de colores verde y no detectando la marca de color verde.

En la Tabla 25 se realiza una prueba diagnóstica entre la cantidad de objetos detectados con el filtro de color verde y las diferentes intensidades de iluminación.

**Tabla 25**

*Cantidad de objetos detectados con la máscara color verde*

<b>Intensidad de iluminación</b>	<b>Resultado de la prueba</b>	<b>Diagnostico verdadero</b>	
		Verde	No verde
Baja	Detectado	0	0
	No detectado	1	6
Media	Detectado	1	0
	No detectado	0	6
Alta	Detectado	0	0
	No detectado	1	6

*Nota:* La información de la tabla muestra un resumen de los resultados obtenidos en este experimento.

Analizando los datos de la Tabla 25 **Tabla 22** se puede determinar la eficiencia de la prueba para cada una de las intensidades de iluminación, utilizando la siguiente ecuación:

$$Precisión = \frac{VP + VN}{VP + FP + FN + VN} \quad (63)$$

donde VP son los detectados verdes, VN son los no detectados no verdes, FP son los detectados no verdes y FN son los no detectados verdes.

Para una intensidad de iluminación baja se tiene el siguiente valor de precisión:

$$Precisión = \frac{0 + 6}{0 + 0 + 1 + 6} = \frac{7}{7} \quad (64)$$

$$Precisión = 0.86 \quad (65)$$

Para una intensidad de iluminación media se tiene el siguiente valor de precisión:

$$Precisión = \frac{1 + 6}{1 + 0 + 0 + 6} = \frac{7}{7} \quad (66)$$

$$Precisión = 1 \quad (67)$$

Para una intensidad de iluminación alta se tiene el siguiente valor de precisión:

$$Precisión = \frac{0 + 6}{0 + 0 + 1 + 6} = \frac{6}{7} \quad (68)$$

$$Precisión = 0.86 \quad (69)$$

Con los datos calculados de la eficiencia para cada una de las intensidades se puede determinar que, para la máscara de color verde el mejor resultado se obtiene para una intensidad de iluminación media.

Con los resultados obtenidos en cada una de las máscaras de color con las diferentes intensidades de iluminación, se concluye que los mejores resultados para detectar los colores se dan con una intensidad de iluminación media en el interior de la celda, por tal motivo para las siguientes pruebas se utiliza esta intensidad esperando obtener buenos resultados.

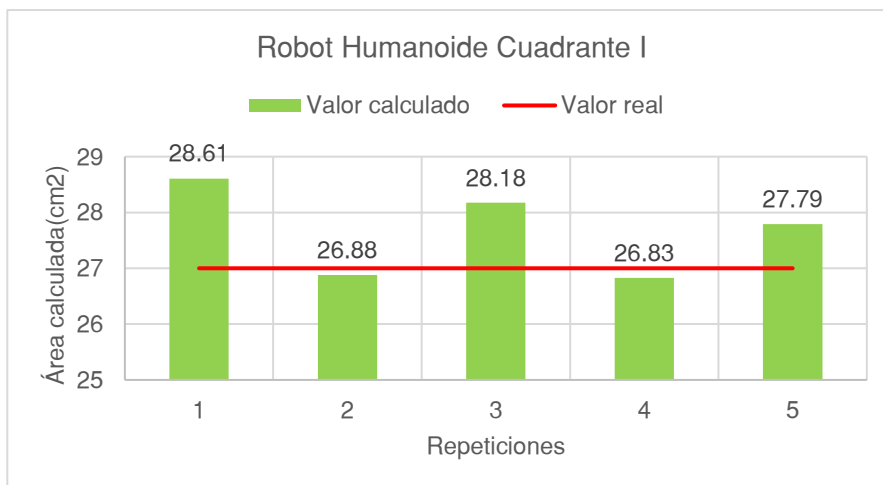
### ***Prueba 2***

Para cada experimento se realizaron cinco repeticiones, en cada repetición se tomaron 30 datos tanto del área de la marca como de la orientación del robot que el algoritmo calcula, de los cuales en cada repetición se saca la media de los datos. A continuación, se muestran los resultados obtenidos para cada uno de los robots.

**Robot humanoide.** La marca del robot humanoide es un triángulo isósceles con las siguientes dimensiones: 8.5cm de base y 3.2cm de altura, con estas dimensiones el área real de la marca es de 27cm<sup>2</sup>. En la Figura 117 se puede observar los resultados obtenidos del área calculada para el primer cuadrante, en la Figura 118 para los del segundo cuadrante, en la Figura 119 para los del tercer cuadrante y en la Figura 120 para el cuarto cuadrante.

**Figura 117**

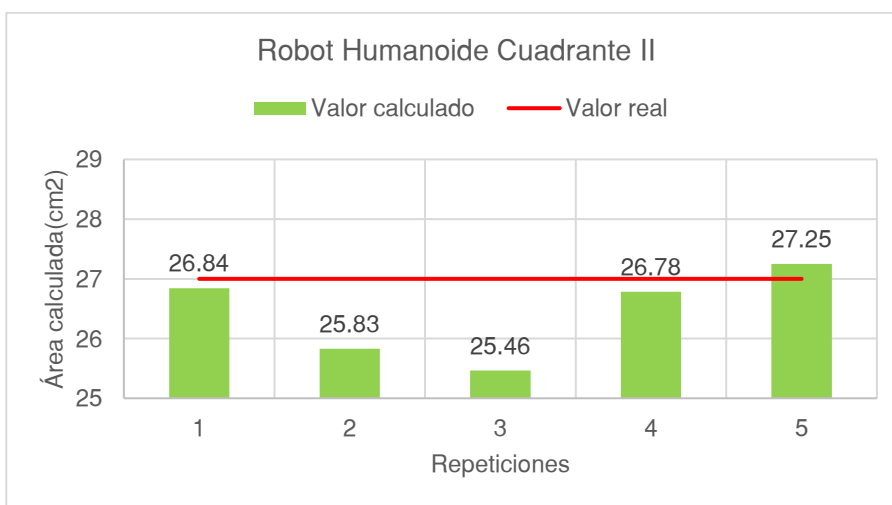
*Resultados obtenidos para el robot humanoide en el cuadrante I*



*Nota:* La grafica muestra que los datos calculados no se encuentran muy dispersos.

**Figura 118**

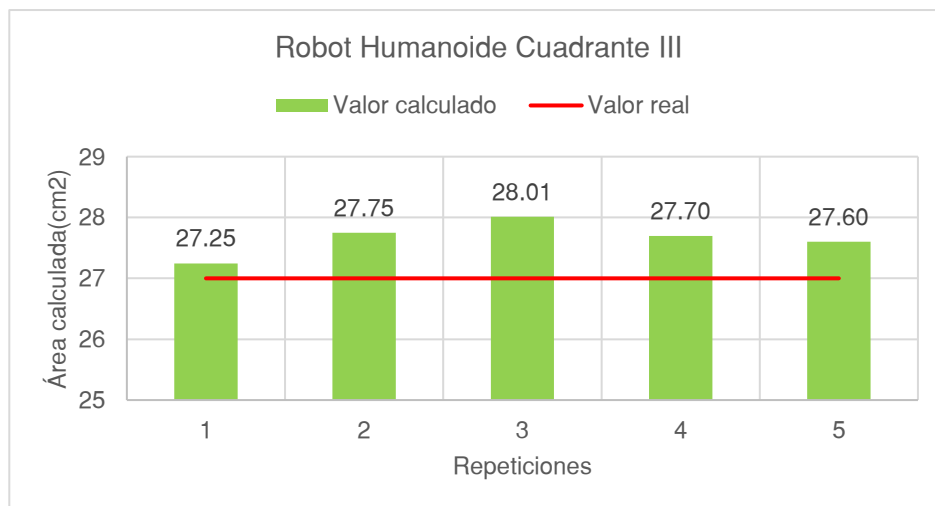
*Resultados obtenidos para el robot humanoide en el cuadrante II*



*Nota:* La grafica muestra que los datos calculados no se encuentran muy dispersos.

**Figura 119**

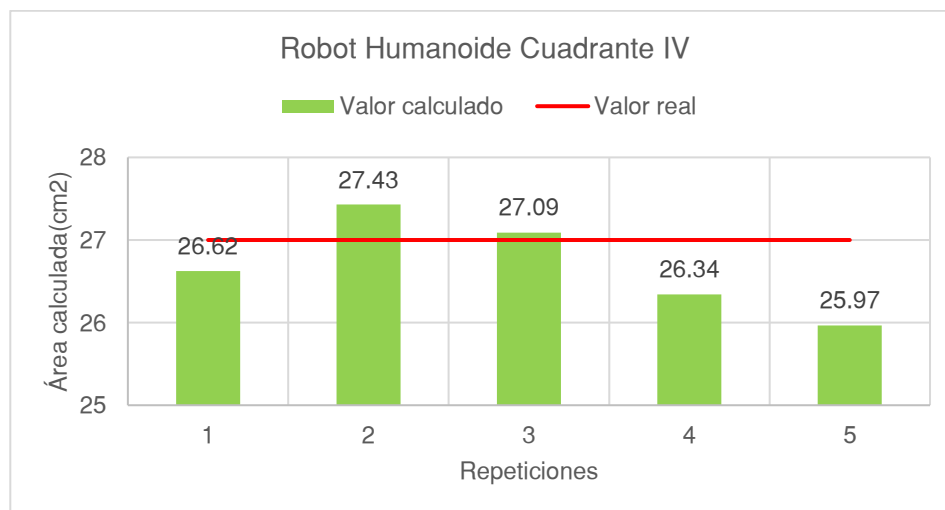
*Resultados obtenidos para el robot humanoide en el cuadrante III*



*Nota:* La grafica muestra que los datos calculados no se encuentran muy dispersos.

**Figura 120**

*Resultados obtenidos para el robot humanoide en el cuadrante IV*



*Nota:* La grafica muestra que los datos calculados no se encuentran muy dispersos.

En la Tabla 26 se muestra el error calculado para cada una de las repeticiones en los diferentes cuadrantes para la marca del robot humanoide.



Tabla 26

*Error porcentual del cálculo del área de la marca del robot humanoide*

Repetición	Área de la marca calculada			
	Cuadrante I	Cuadrante II	Cuadrante III	Cuadrante IV
1	5.95%	0.58%	0.91%	1.39%
2	0.44%	4.33%	2.77%	1.59%
3	4.35%	5.69%	3.74%	0.34%
4	0.65%	0.81%	2.58%	2.44%
5	2.94%	0.93%	2.23%	3.83%

*Nota:* Los datos del error porcentual permiten conocer que tan efectivo está siendo el algoritmo en los cálculos de la marca del robot humanoide.

En la Tabla 27 se muestran los resultados obtenidos de la orientación calculada del robot humanoide en cada uno de los cuadrantes, en cada cuadrante se calculó la media y la desviación estándar de los datos obtenidos en cada una de las repeticiones.

Tabla 27

*Resultados obtenidos para la orientación del robot humanoide*

Repetición	Valor del ángulo calculado							
	Cuadrante I		Cuadrante II		Cuadrante III		Cuadrante IV	
	No.	$\bar{X}$	$\sigma$	$\bar{X}$	$\sigma$	$\bar{X}$	$\sigma$	$\bar{X}$
1	20.58	1.54	101.67	1.37	201.76	2.76	288.83	3.57
2	30.96	0.00	114.18	4.84	225.07	4.12	301.23	1.02
3	45.00	0.00	136.37	4.67	240.61	3.72	318.87	1.30
4	73.30	0.00	150.88	5.39	253.16	3.44	335.85	1.13
5	77.97	1.86	171.69	4.74	266.67	3.86	351.17	3.26

*Nota:* Los datos de la tabla muestran la orientación del robot en cada una de las pruebas, esto es necesario conocer ya que esto depende la decisión de los movimientos a ejecutar.

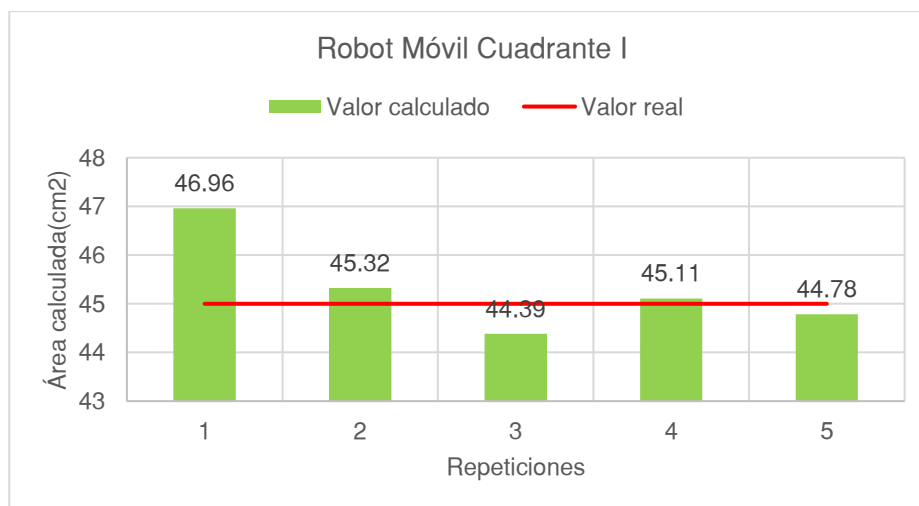
Como se puede observar en Tabla 26, los porcentajes de error son bajos cuando el robot humanoide se encuentra orientado en el cuadrante III y IV, para el cuadrante I y II se tiene un solo valor superior al 5% en cada uno, lo cual no es representativo. En la Tabla 27 se puede

observar que el ángulo calculado en cada una de las repeticiones de cada cuadrante corresponde al cuadrante en el que se encuentra orientado el robot humanoide, además que los valores obtenidos no están muy dispersos lo que permitirá tomar decisiones de movimientos estables.

**Robot móvil.** La marca del robot móvil es un triángulo isósceles con las siguientes dimensiones: 9cm de base y 5cm de altura, con estas dimensiones el área real de la marca es de  $45\text{cm}^2$ . En la Figura 121 se puede observar los resultados obtenidos del área calculada para el primer cuadrante, en la Figura 122 para los del segundo cuadrante, en la Figura 123 para los del tercer cuadrante y en la Figura 124 para el cuarto cuadrante.

**Figura 121**

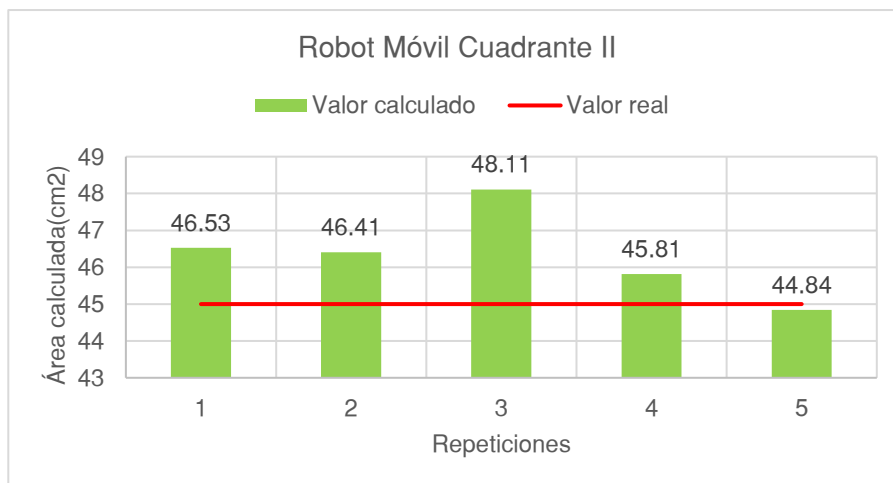
*Resultados obtenidos para el robot móvil en el cuadrante I*



*Nota:* La grafica muestra que los datos calculados no se encuentran muy dispersos.

**Figura 122**

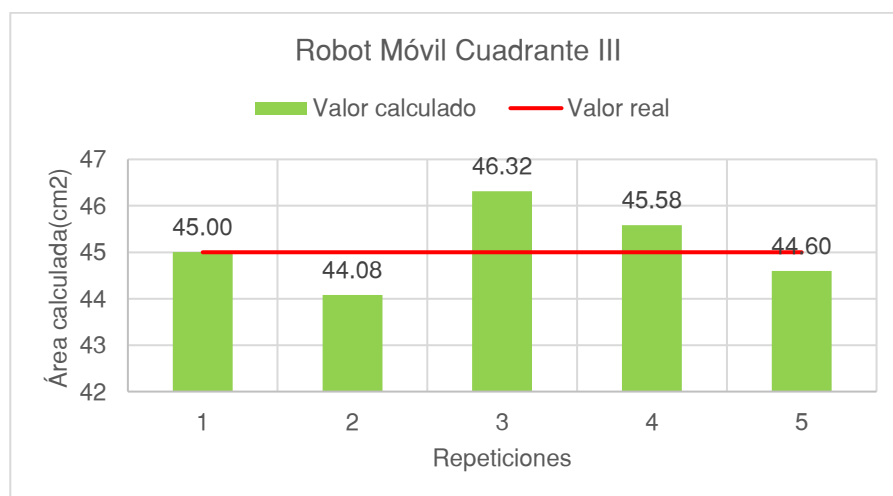
*Resultados obtenidos para el robot móvil en el cuadrante II*



*Nota:* La grafica muestra que los datos calculados no se encuentran muy dispersos.

**Figura 123**

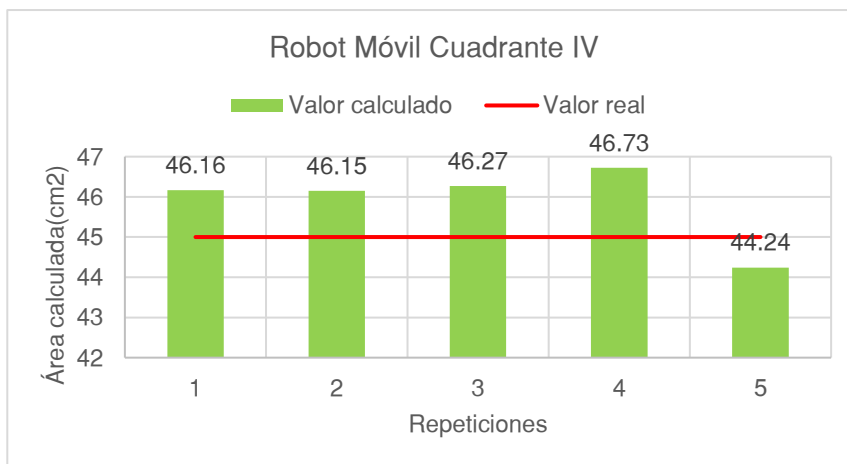
*Resultados obtenidos para el robot móvil en el cuadrante III*



*Nota:* La grafica muestra que los datos calculados no se encuentran muy dispersos.

**Figura 124**

*Resultados obtenidos para el robot móvil en el cuadrante IV*



*Nota:* La grafica muestra que los datos calculados no se encuentran muy dispersos.

En la Tabla 28 se muestra el error calculado para cada una de las repeticiones en los diferentes cuadrantes para la marca del robot móvil.

**Tabla 28**

*Error porcentual del cálculo del área de la marca del robot móvil*

Repetición	Área de la marca			
	Cuadrante I	Cuadrante II	Cuadrante III	Cuadrante IV
1	4.35%	3.39%	0.01%	2.58%
2	0.72%	3.13%	2.04%	2.55%
3	1.36%	6.91%	2.92%	2.83%
4	0.65%	1.81%	1.29%	3.83%
5	0.48%	0.35%	0.89%	1.69%

*Nota:* Los datos del error porcentual permiten conocer que tan efectivo está siendo el algoritmo en los cálculos de la marca del robot móvil.

En la Tabla 29 se muestran los resultados obtenidos de la orientación calculada del robot humanoide en cada uno de los cuadrantes, en cada cuadrante se calculó la media y la desviación estándar de los datos obtenidos en cada una de las repeticiones.

**Tabla 29**

*Resultados obtenidos para la orientación del robot móvil*

Repetición No.	Valor del ángulo calculado							
	Cuadrante I		Cuadrante II		Cuadrante III		Cuadrante IV	
	$\bar{X}$	$\sigma$	$\bar{X}$	$\sigma$	$\bar{X}$	$\sigma$	$\bar{X}$	$\sigma$
1	21.95	0.55	115.99	4.15	195.69	2.76	294.78	0.00
2	41.48	1.34	127.46	2.84	209.40	1.91	311.44	1.18
3	53.97	0.00	146.26	3.17	217.60	1.80	329.58	1.85
4	66.59	1.61	155.14	3.43	243.58	1.73	340.35	0.00
5	80.07	1.66	165.02	2.57	257.28	2.49	347.25	0.40

*Nota:* Los datos de la tabla muestran la orientación del robot en cada una de las pruebas, esto es necesario conocer ya que esto depende la decisión de los movimientos a ejecutar.

En la Tabla 28, los porcentajes de error son bajos cuando el robot móvil se encuentra orientado en el cuadrante I, III y IV, mientras que en el cuadrante II se tiene un solo valor superior al 5%, lo cual no es representativo. En la Tabla 29 se puede observar que el ángulo calculado en cada una de las repeticiones de cada cuadrante corresponde al cuadrante en el que se encuentra orientado el robot móvil, además que los valores obtenidos no están muy dispersos y permite tomar decisiones de movimientos eficientes.

Con los resultados obtenidos en esta prueba se puede concluir mencionando que tanto el área de la marca triangular de color rojo del robot humanoide como el área de la marca triangular de color verde del robot móvil, se encuentran calculadas por el algoritmo implementado dentro lo admitido en otras palabras son menor a un error del 5%. En cuanto a la

dispersión de los valores obtenidos del ángulo calculado el robot móvil presenta menor dispersión que el robot humanoide y esto se debe a que la marca del robot móvil se encuentra de forma paralela al suelo, mientras que la marca del robot humanoide esta de forma inclinada debido al soporte de la cámara del mismo.

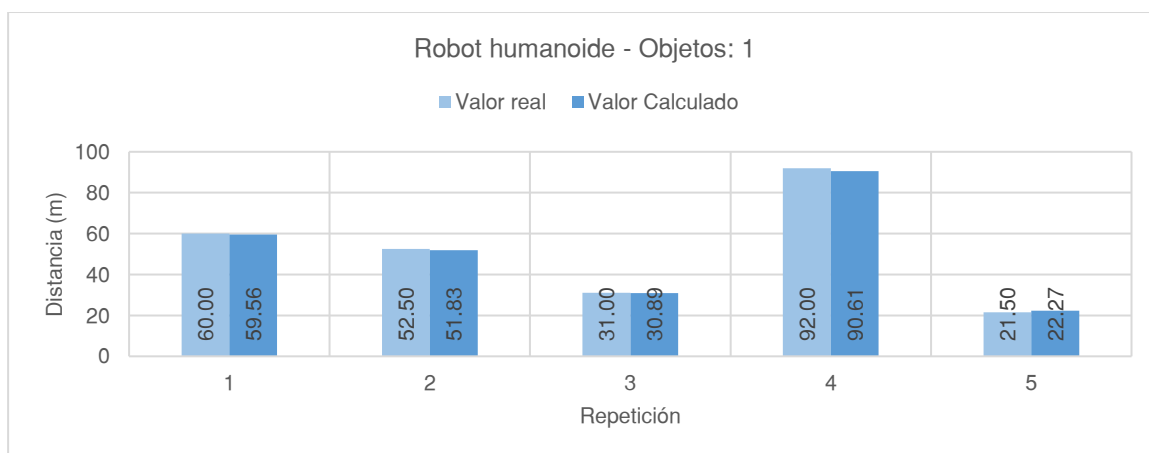
### Prueba 3

En cada experimento se realizaron cinco repeticiones y en cada repetición se tomaron 30 datos de la distancia que el algoritmo calcula. A continuación, se muestran los resultados obtenidos para cada uno de los robots.

**Robot humanoide.** Se colocó el robot humanoide y los objetos de manera aleatoria dentro del espacio de trabajo, en la Figura 125 se muestran los resultados obtenidos de la distancia calculada desde el robot hacia un objeto, mientras que: en la Figura 126 hacia dos objetos, en la Figura 127 hacia tres objetos, en la Figura 128 hacia cuatro objetos y en la Figura 129 hacia cinco objetos.

### Figura 125

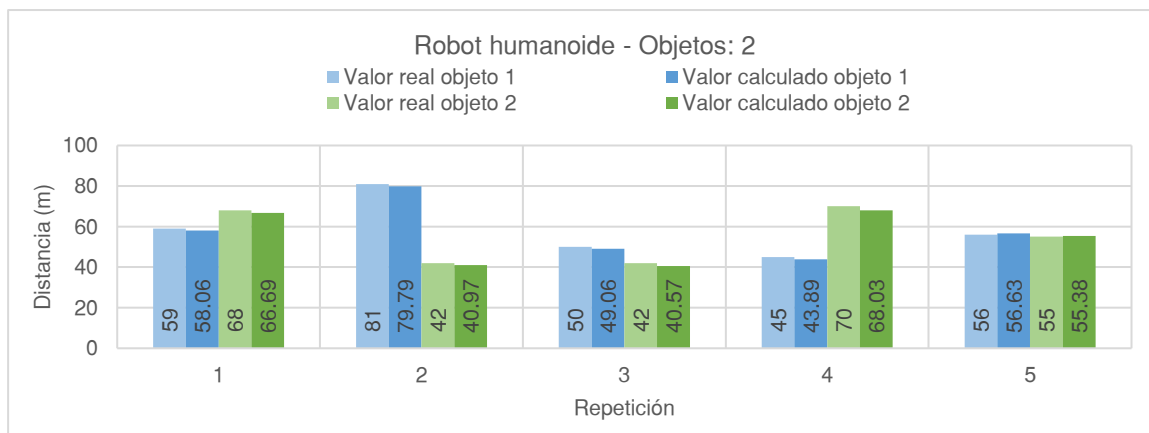
*Resultados obtenidos para el robot humanoide con un objeto*



*Nota:* Es necesario conocer la distancia real y calculada a la que se encuentra el objeto del robot.

**Figura 126**

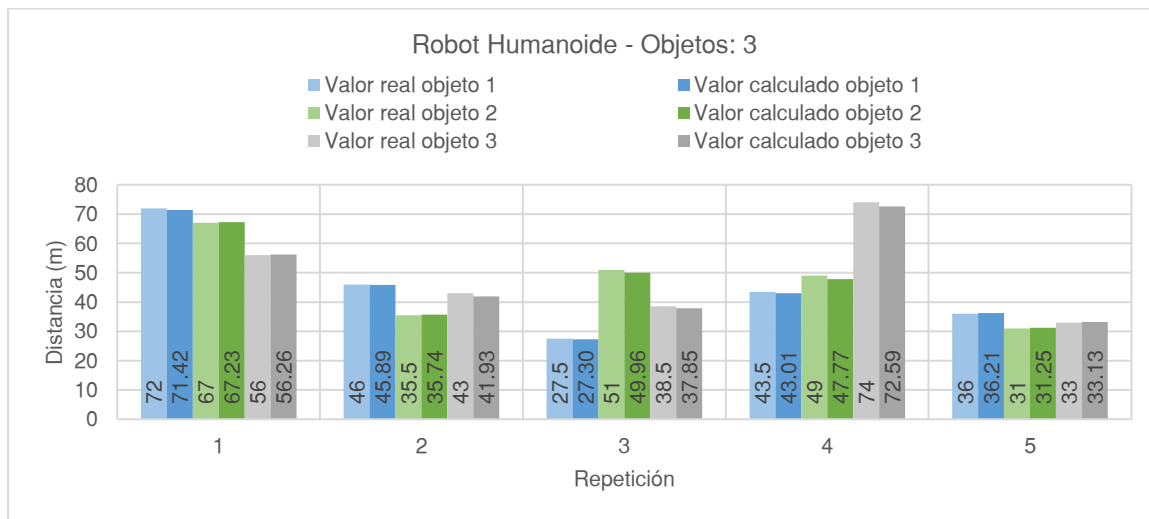
*Resultados obtenidos para el robot humanoide con dos objetos*



*Nota:* Es necesario conocer las distancias reales y calculadas a la que se encuentra cada objeto con respecto al robot.

**Figura 127**

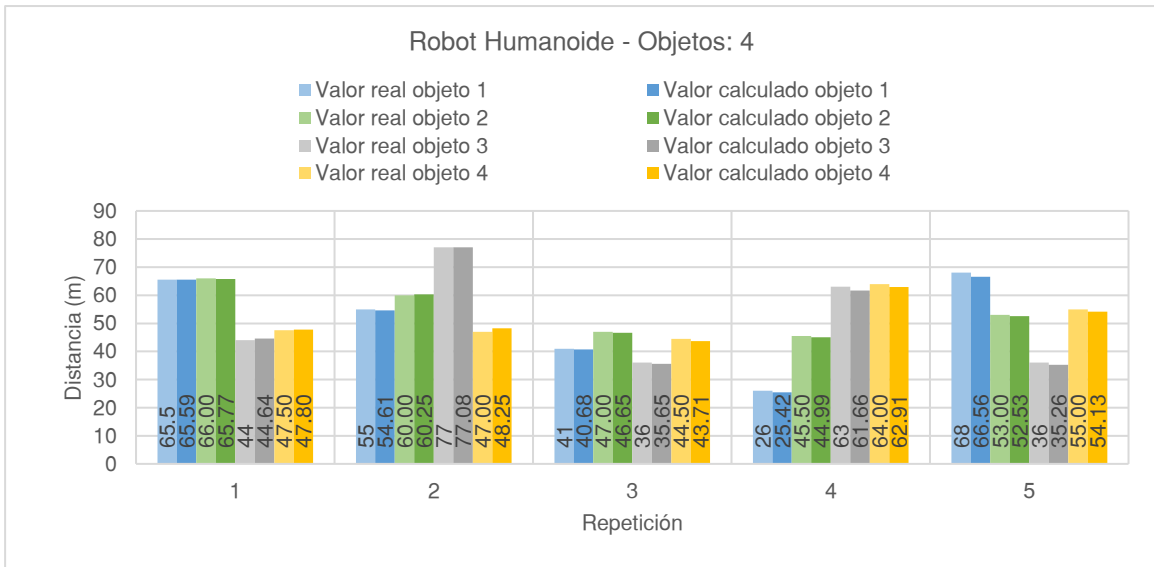
*Resultados obtenidos para el robot humanoide con tres objetos*



*Nota:* Es necesario conocer las distancias reales y calculadas a la que se encuentra cada objeto con respecto al robot.

**Figura 128**

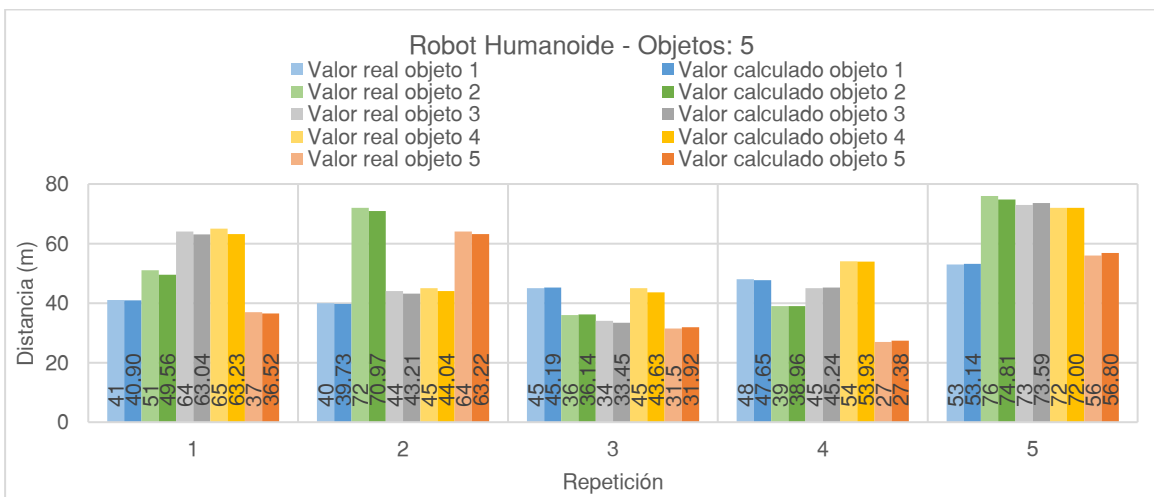
*Resultados obtenidos para el robot humanoide con cuatro objetos*



*Nota:* Es necesario conocer las distancias reales y calculadas a la que se encuentra cada objeto con respecto al robot.

**Figura 129**

*Resultados obtenidos para el robot humanoide con cinco objetos*

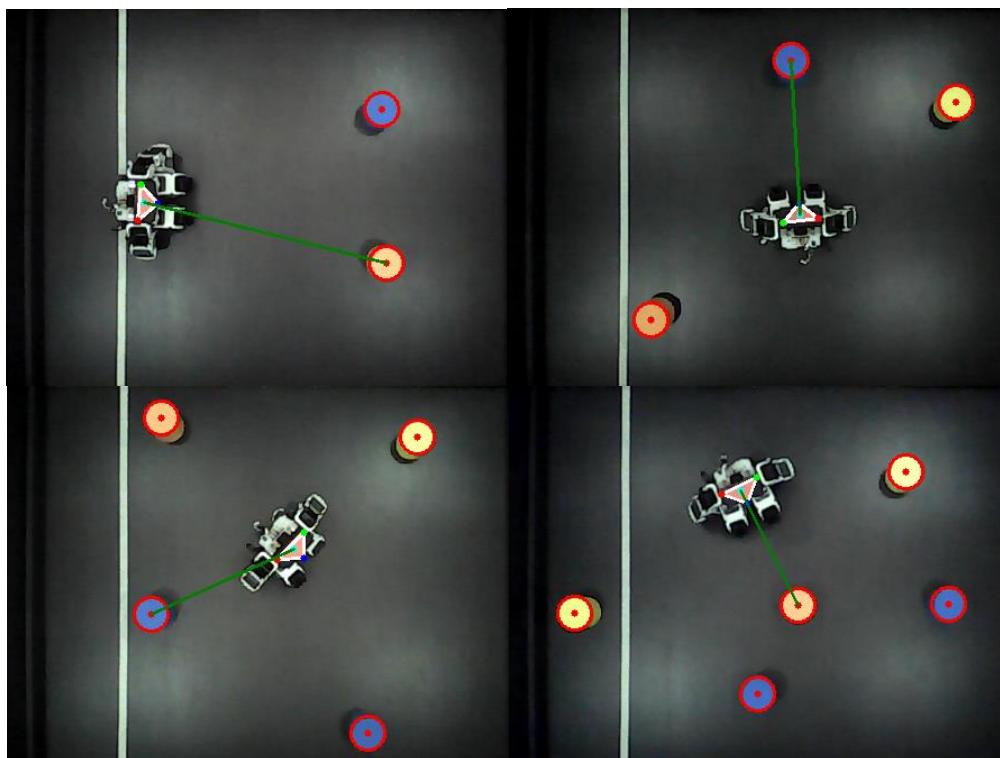


*Nota:* Es necesario conocer las distancias reales y calculadas a la que se encuentra cada objeto con respecto al robot.



**Figura 130**

*Resultados obtenidos de la asignación del objeto más cercano al robot humanoide*



*Nota:* En cada experimento se determina que objeto se encuentra a menor distancia del robot, se lo asigna como objeto de interés y este procederá a trasladarse hacia su ubicación.

En la Figura 130 se observa unos ejemplos de la asignación del objetivo más cercano, y para que se pueda observar la asignación se dibujó una línea desde el centro de la marca del robot hacia el centro del objetivo asignado. La Tabla 30 muestra el objetivo asignado en cada repetición hecha de esta prueba.

Tabla 30

*Asignación de objetivos en el robot humanoide*

Repetición	Asignación de objetivo al robot humanoide			
	Dos objetos	Tres objetos	Cuatro objetos	Cinco objetos
1	Objeto 1	Objeto 3	Objeto 3	Objeto 5
2	Objeto 2	Objeto 2	Objeto 4	Objeto 1
3	Objeto 2	Objeto 1	Objeto 3	Objeto 5
4	Objeto 1	Objeto 1	Objeto 1	Objeto 5
5	Objeto 2	Objeto 2	Objeto 3	Objeto 1

*Nota:* En la asignación de objetivo se pudo determinar experimentalmente que el algoritmo no distingue entre objetos del mismo color.

El cálculo de las distancias se lo realizó a cada objeto por lo que cada uno tiene una distancia real y una calculada por ende cada una tiene un error, por lo que en la Tabla 31 se muestra el error porcentual promedio de cada prueba realizada.

Tabla 31

*Error porcentual de la distancia calculada con el robot humanoide*

Repetición	Distancia				
	Un objeto	Dos objetos	Tres objetos	Cuatro objetos	Cinco objetos
1	0.74%	1.76%	0.54%	0.64%	1.72%
2	1.27%	1.97%	1.14%	0.97%	1.45%
3	0.34%	2.64%	1.48%	1.08%	1.36%
4	1.51%	2.64%	1.85%	1.80%	0.58%
5	3.56%	0.91%	0.59%	1.66%	0.81%

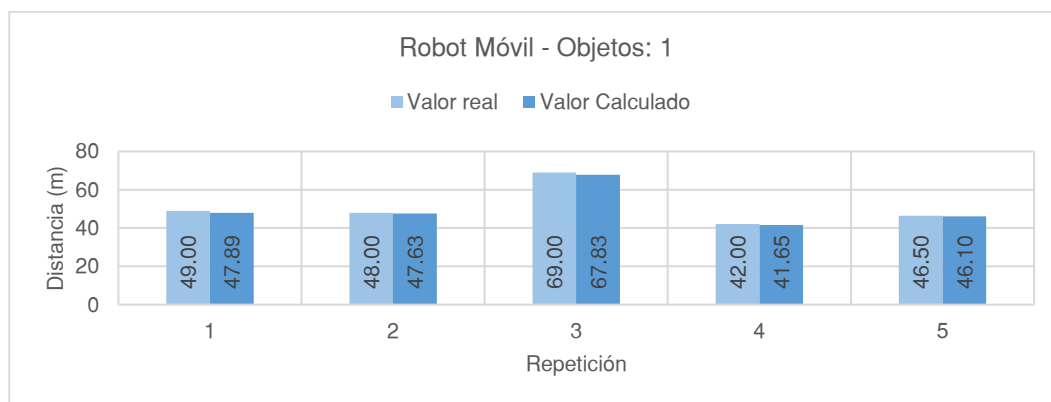
*Nota:* Con los datos obtenidos en la tabla se puede observar que todos los errores se encuentran por debajo del 5%.

**Robot móvil.** De manera similar se colocó el robot móvil y los objetos de manera aleatoria dentro del espacio de trabajo, en la Figura 131 se muestran los resultados obtenidos de la distancia calculada desde el robot hacia un objeto, mientras que: en la Figura 132 hacia dos

objetos, en la Figura 133 hacia tres objetos, en la Figura 134 hacia cuatro objetos y en la Figura 135 hacia cinco objetos.

**Figura 131**

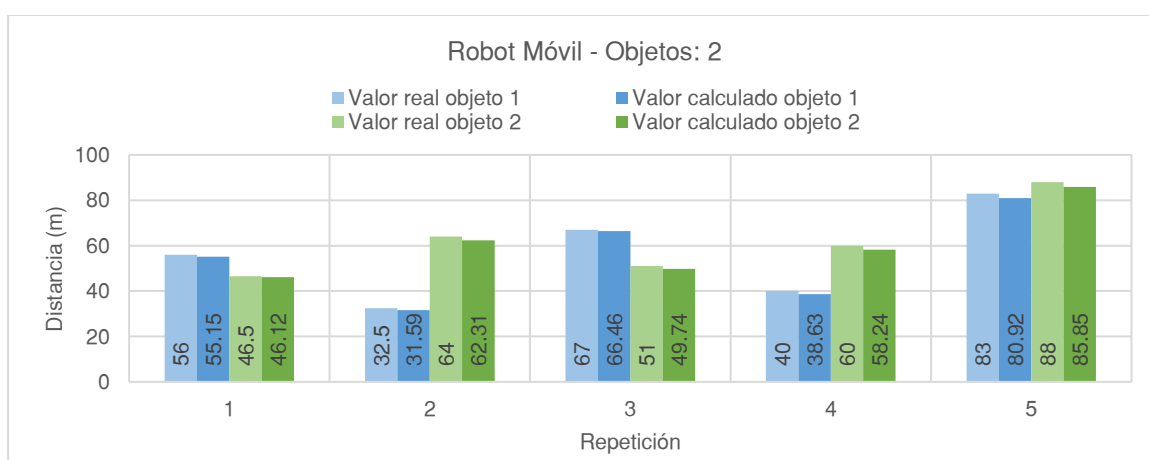
*Resultados obtenidos para el robot móvil con un objeto*



*Nota:* Es necesario conocer la distancia real y calculada a la que se encuentra el objeto con respecto al robot.

**Figura 132**

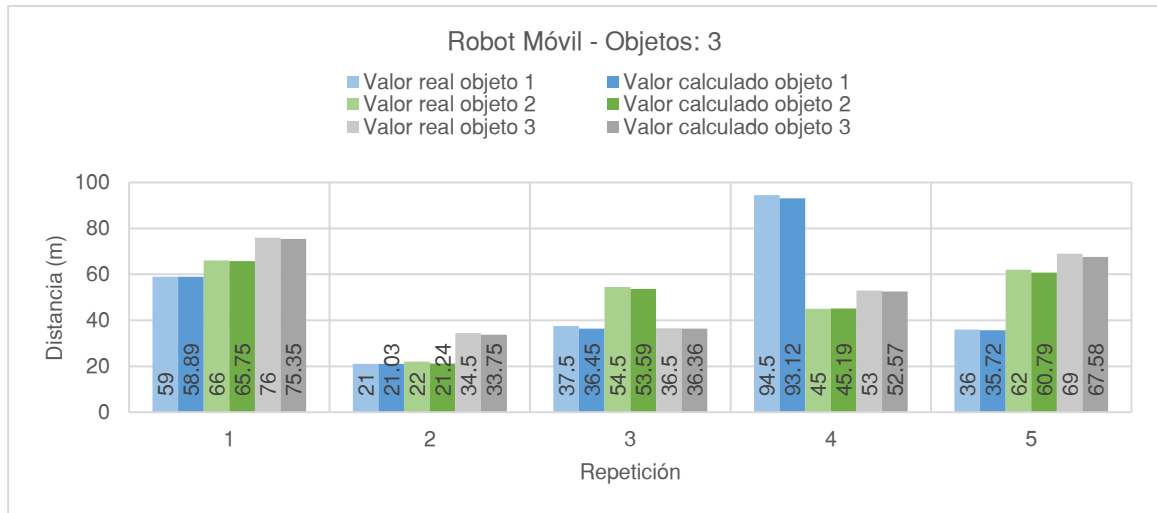
*Resultados obtenidos para el robot móvil con dos objetos*



*Nota:* Es necesario conocer las distancias reales y calculadas a la que se encuentra cada objeto con respecto al robot.

**Figura 133**

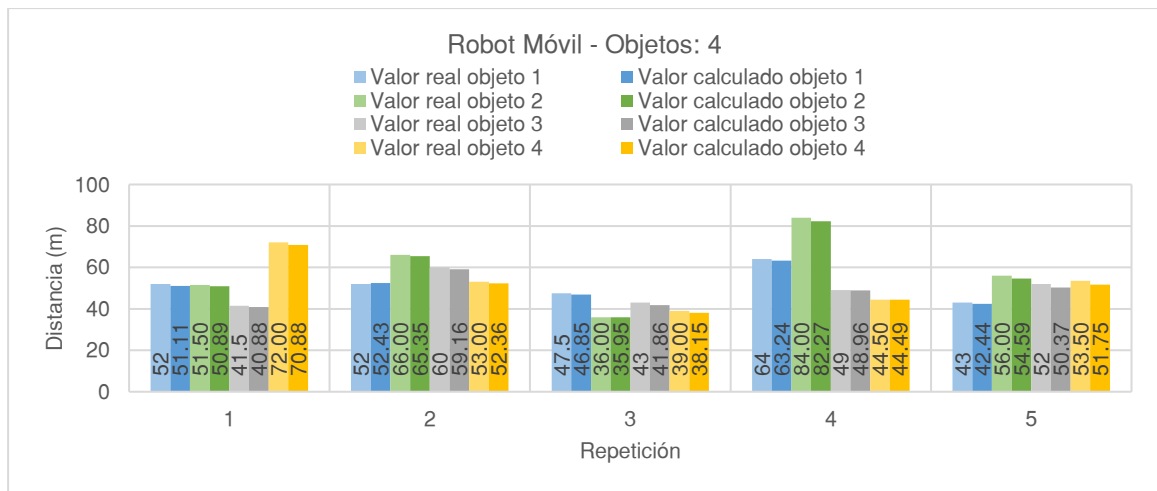
*Resultados obtenidos para el robot móvil con tres objetos*



*Nota:* Es necesario conocer las distancias reales y calculadas a la que se encuentra cada objeto con respecto al robot.

**Figura 134**

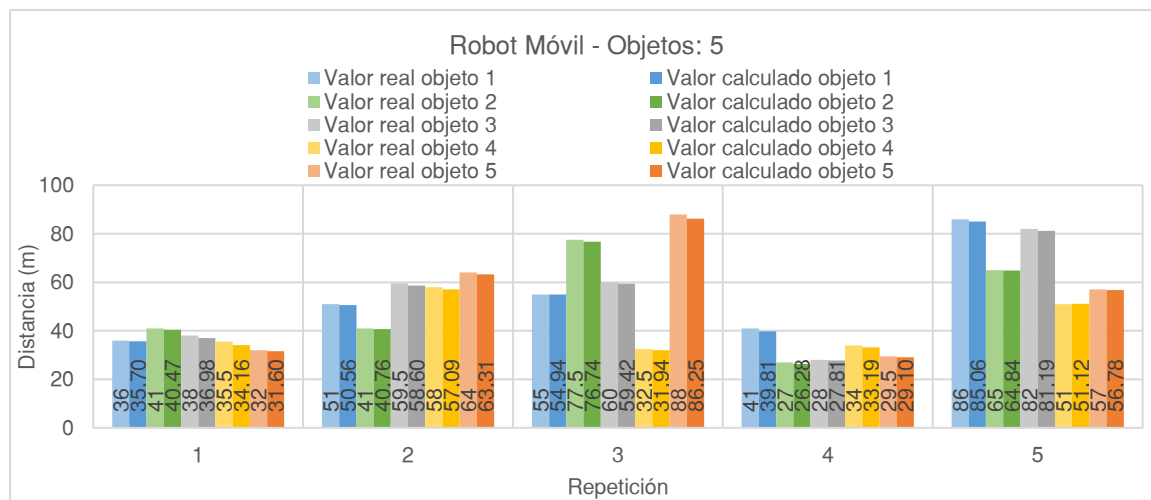
*Resultados obtenidos para el robot móvil con cuatro objetos*



*Nota:* Es necesario conocer las distancias reales y calculadas a la que se encuentra cada objeto con respecto al robot.

Figura 135

Resultados obtenidos para el robot móvil con cinco objetos

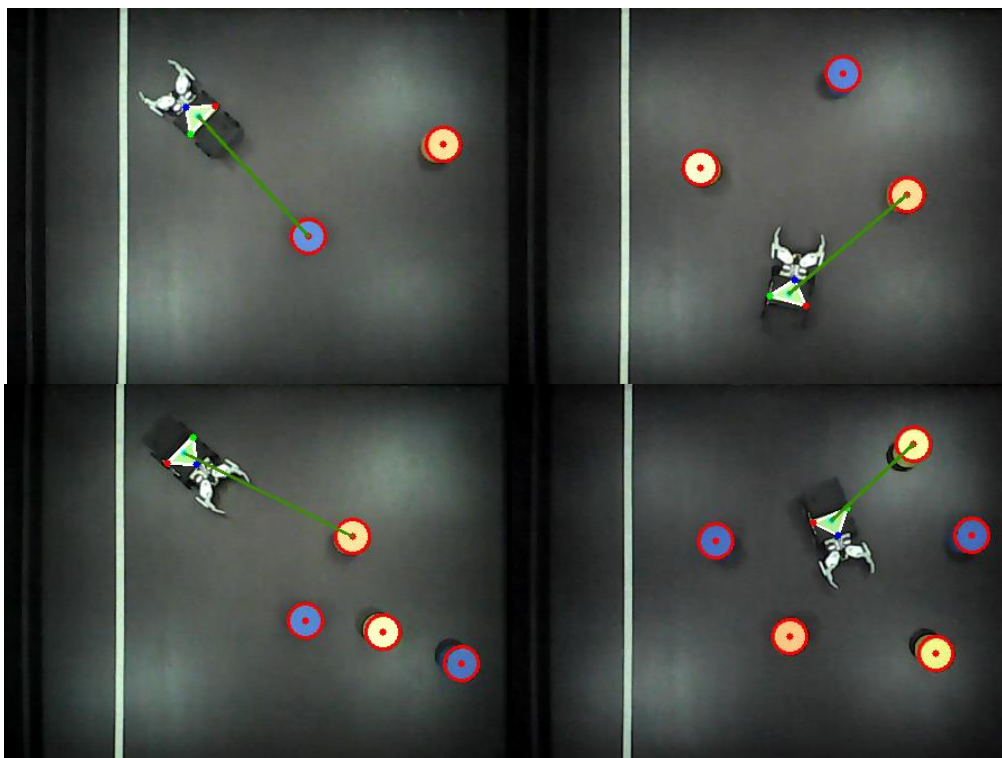


*Nota:* Es necesario conocer las distancias reales y calculadas a la que se encuentra cada objeto con respecto al robot.

En la Figura 136 se observa unos ejemplos de la asignación del objetivo más cercano, y para que se pueda observar la asignación se dibujó una línea desde el centro de la marca del robot hacia el centro del objetivo asignado. La Tabla 32 muestra el objetivo asignado en cada repetición hecha de esta prueba.

Figura 136

*Resultados obtenidos de la asignación del objeto más cercano al robot móvil*



*Nota:* En cada experimento se determina que objeto se encuentra a menor distancia del robot, se lo asigna como objeto de interés y este procederá a trasladarse hacia su ubicación.

Tabla 32

*Asignación de objetivos en el robot móvil*

Repetición	Asignación de objetivo al robot móvil			
	Dos objetos	Tres objetos	Cuatro objetos	Cinco objetos
1	Objeto 2	Objeto 1	Objeto 3	Objeto 5
2	Objeto 1	Objeto 1	Objeto 1	Objeto 2
3	Objeto 2	Objeto 3	Objeto 2	Objeto 4
4	Objeto 1	Objeto 2	Objeto 4	Objeto 2
5	Objeto 1	Objeto 1	Objeto 1	Objeto 4

*Nota:* En la asignación de objetivo se pudo determinar experimentalmente que el algoritmo no distingue entre objetos del mismo color.

El cálculo de las distancias se lo realizó a cada objeto por lo que cada uno tiene una distancia real y una calculada por ende cada una tiene un error, en la Tabla 33 se muestra el error porcentual promedio de cada prueba realizada.

**Tabla 33**

*Error porcentual de la distancia calculada con el robot móvil*

Repetición	Distancia				
	Un objeto	Dos objetos	Tres objetos	Cuatro objetos	Cinco objetos
1	2.26%	1.16%	0.47%	1.25%	1.97%
2	0.77%	2.73%	1.92%	1.47%	1.13%
3	1.69%	1.64%	1.62%	1.49%	1.15%
4	0.82%	3.18%	0.90%	1.26%	1.99%
5	0.85%	2.48%	1.60%	2.14%	0.59%

*Nota:* Con los datos obtenidos en la tabla se puede observar que todos los errores se encuentran por debajo del 5%.

Con los resultados obtenidos tanto con el robot humanoide como con el robot móvil, se obtuvo errores en todas las pruebas realizadas menores al 5% lo que indica que el algoritmo implementado funciona de una manera correcta y permite trabajar con valores cercanos a la realidad, esto permitirá que la toma de decisiones al momento de ejecutar ordenes de movimientos en los robots sean más estables.

#### **Prueba 4**

Durante la ejecución de la última prueba se presentaron inconvenientes con algunos actuadores AX-12A pertenecientes al robot humanoide, los cuales impedían que el robot pueda concretar la ejecución de sus movimientos. Como se detalló anteriormente solo se contaba con 20 actuadores para el robot humanoide y 6 para el robot móvil, por lo que no hay actuadores de repuestos que permitieran solventar el inconveniente surgido. La adquisición de nuevos

actuadores no se encuentra dentro de la planificación del proyecto y por tal motivo se modificó el diseño del robot humanoide a uno de 10 GDL conformado por los actuadores que no presentaron ningún problema en su funcionamiento.

Para la primera parte de la prueba se colocó primero al robot móvil con los objetos a clasificar de manera aleatoria dentro del área de trabajo, luego se colocó al robot humanoide y a los objetos a clasificar en las mismas posiciones del experimento anterior, y finalmente se colocan los objetos a clasificar en las mismas posiciones anteriores y se colocó los dos robots de manera aleatoria. Este procedimiento se lo hizo para cada caso (selección de objetos a clasificar solo de color amarillo, solo de color azul y ambos colores) y en cada una se registraron los tiempos que se demoran en llevar a cabo la clasificación, además se analiza si la tarea fue terminada correctamente o no. A continuación se muestran los resultados obtenidos en esta primera parte.

**Tabla 34**

*Resultados obtenidos en la repetición 1 de la clasificación de objetos de la prueba 4*

Clasificación de objetos						
Color	Robot	Tiempo (minutos)	Termina la clasificación	Objetos recolectados	Objetos caídos	Validación de color
Azul	Móvil	4.3	Si	2	0	2
	Humanoide	6.49	No	1	1	2
	Ambos	5.18	Si	2	0	2
Amarillo	Móvil	5.65	No	1	1	1
	Humanoide	11.45	Si	2	0	2
	Ambos	7.45	No	1	1	1
Ambos	Móvil	10.48	Si	4	0	4
	Humanoide	37.33	No	3	1	3
	Ambos	25.05	Si	4	0	4

*Nota:* Para que el robot o los robots terminen una clasificación correctamente estos no deben dejar caer ningún objeto al momento de la recolección y el traslado.



**Tabla 35**

*Resultados obtenidos en la repetición 2 de la clasificación de objetos de la prueba 4*

Color	Robot	Clasificación de objetos			Objetos caídos	Validación de color
		Tiempo (minutos)	Termina la clasificación	Objetos recolectados		
Azul	Móvil	5.5	Si	2	0	2
	Humanoide	7.33	Si	2	0	2
	Ambos	6.98	Si	2	0	2
Amarillo	Móvil	4.95	Si	2	0	2
	Humanoide	8.38	No	0	2	2
	Ambos	5.85	No	2	1	2
Ambos	Móvil	14.24	No	2	2	2
	Humanoide	32.78	Si	4	0	4
	Ambos	21.35	No	3	1	4

*Nota:* Para que el robot o los robots terminen una clasificación correctamente estos no deben dejar caer ningún objeto al momento de la recolección y el traslado.

**Tabla 36**

*Resultados obtenidos en la repetición 3 de la clasificación de objetos de la prueba 4*

Color	Robot	Clasificación de objetos			Objetos caídos	Validación de color
		Tiempo (minutos)	Termina la clasificación	Objetos recolectados		
Azul	Móvil	4.18	Si	2	0	2
	Humanoide	9.77	Si	2	0	2
	Ambos	7.32	Si	2	0	2
Amarillo	Móvil	5.35	Si	2	0	2
	Humanoide	10.12	Si	2	0	2
	Ambos	6.67	Si	2	0	2
Ambos	Móvil	13.32	Si	4	0	4
	Humanoide	35.58	Si	4	0	4
	Ambos	19.85	Si	4	0	4

*Nota:* Para que el robot o los robots terminen una clasificación correctamente estos no deben dejar caer ningún objeto al momento de la recolección y el traslado.

Como se detalló anteriormente en el documento se cuenta con dos objetos de color amarillo y dos objetos de color azul por lo que para el análisis de los datos de la Tabla 34 ,la Tabla 35 y la Tabla 36 se toman en cuentas los datos de ambos colores en cada uno de los robots.

El tiempo promedio que se demora en clasificar dos objetos el robot móvil es de 4.98 minutos, con una efectividad en el cumplimiento de la prueba del 83.33%, la cantidad de objetos de interés recolectados correctamente es del 92% debido a que en una de las pruebas uno objeto no se realizó bien la validación del color, lo que da una efectividad en el algoritmo de verificación de color en el robot móvil del 92%. El motivo por el cual no se pudo realizar la validación del color en esa prueba es que el objeto y el robot no quedaron a la distancia establecida por lo que el sensor de color detecto otro color.

En cuanto a la clasificación de cuatro objetos el robot móvil hizo un tiempo promedio de 12.68 minutos, efectividad en el cumplimiento de la prueba del 66.67%, cantidad de objetos recolectados correctamente del 83.33% y la efectividad del algoritmo de validación de color del 83.33%.

El tiempo promedio que se demora en clasificar dos objetos el robot humanoide es de 8.92 minutos, con una efectividad en el cumplimiento de la prueba del 66.64%, la cantidad de objetos de interés recolectados correctamente es del 75%, las razones por las cuales el robot humanoide no pudo recolectar bien es que al momento de la recolección del objeto de interés este no lo pudo coger bien y al momento de levantarlo o en el traslado hacia el área de objetos clasificados este se cayó; otra razón es que el robot pierde el equilibrio por los brazos y se cae.

El motivo por el cual el robot humanoide no puede coger bien el objeto de interés es porque el robot utilizado para esta prueba cuenta con 4 GDL en las extremidades inferiores, lo que impide que este pueda agacharse o inclinarse hacia adelante para conseguir un mejor agarre del objeto de interés. El modelo del robot humanoide utilizado en esta prueba es de estatura pequeña, por lo que las modificaciones hechas en los brazos para poder coger los objetos hicieron que el centro de gravedad del robot sea variado en la ejecución de sus movimientos haciendo en ocasiones que pierda el equilibrio y caiga al suelo.

En cuanto al algoritmo de validación de color en el robot humanoide tiene una efectividad del 100%. En cuanto a la clasificación de cuatro objetos el robot humanoide hizo un tiempo promedio de 35.23 minutos, efectividad en el cumplimiento de la prueba del 66.67%, cantidad de objetos recolectados correctamente del 91.67% y la efectividad del algoritmo de validación de color del 91.67%.

Los tiempos en los que tardan los robots en cumplir la tarea asignada son extensos mayores a 4 minutos, comparando los tiempos entre el robot humanoide y el robot móvil, se determina que el robot móvil cumple la tarea asignada en menor tiempo que el robot humanoide. Estos tiempos extensos se dan debido a que las velocidades de desplazamiento de los robots son bajas, las ejecuciones de los movimientos son fijas y no son del todo precisas haciendo que los robots se encuentren moviéndose constantemente tratando de alcanzar la ubicación de interés, estos movimientos son:

- En el robot humanoide la ejecución de un movimiento hacia adelante hace que este avance entre 2.5 y 3.5 cm, mientras que un giro ya sea a la izquierda o derecha está entre 10° y 15°.

- En el robot móvil la ejecución de un movimiento hacia adelante hace que avance entre 2 y 3 cm, y un giro entre 10° y 20°.

Finalmente se analiza el trabajo en conjunto entre los dos robots. Cuando se clasifican dos objetos se tiene un tiempo promedio de 6.57 minutos, efectividad en el cumplimiento de la prueba del 66.67%, cantidad de objetos recolectados correctamente es de 91.67%, y la efectividad de usar los dos algoritmos de validación de color al mismo tiempo del 91.67%.

Cuando se clasifican cuatro objetos se tiene un tiempo promedio de 22.08 minutos, efectividad en el cumplimiento de la prueba del 66.67%, cantidad de objetos recolectados correctamente es de 91.67%, y la efectividad de usar los dos algoritmos de validación de color al mismo tiempo del 100%.

Analizando los tiempos individuales con los grupales se determina que, al realizar el trabajo con ambos robots el tiempo de ejecución es menor que el tiempo que hace el robot humanoide, pero mayor que el tiempo que realiza el robot móvil, esto sucede e a que el robot humanoide se demora mucho más en alcanzar el objetivo de interés que el robot móvil, por lo que el limitante en la realización del trabajo colaborativo está dado por el robot humanoide. La efectividad de usar al mismo tiempo los algoritmos de validación de color es del 95%.

En la segunda parte de esta prueba se colocó un obstáculo entre el robot y el objeto de interés el cual impide que el robot alcance su ubicación, para lo cual se analiza el funcionamiento del algoritmo de colisiones y las peticiones de ayuda por colisión los resultados obtenidos en esta prueba se detallan a continuación.

**Tabla 37**

*Resultados obtenidos en el trabajo colaborativo de la prueba 4*

Repetición	Tiempo (minutos)	Trabajo colaborativo		
		Termina la clasificación	Colisión con objeto	Atención a la petición
1	5.02	Si	No	Si
2	3.21	No	No	Si
3	3.19	No	Si	No
4	3.35	Si	No	Si
5	5.38	Si	No	Si
6	3.72	Si	No	Si
7	4.47	Si	No	Si
8	5.27	Si	No	Si
9	6.35	Si	No	Si
10	4.48	Si	No	Si

*Nota:* En la ejecución de esta prueba los resultados obtenidos son trabajando con los dos robots.

Realizando el análisis de los datos obtenidos en la Tabla 37 se tiene que el tiempo promedio en la ejecución de la prueba del trabajo colaborativo por petición de ayuda por colisión es de 4.24 minutos. A demás el porcentaje de efectividad en la culminación del trabajo colaborativo es del 80%, donde se obtuvo dos pruebas con resultados negativos. Finalmente, la efectividad del algoritmo de detección de colisiones y del algoritmo de atención a peticiones de ayuda colaborativa son del 90%.

En la segunda repetición de la prueba la razón por la cual no se terminó de realizar la clasificación fue porque el robot humanoide después de retirar el obstáculo de la ubicación de la petición queda divagando y el robot móvil no avanza, la posible falla en esta parte es que la ubicación del obstáculo no se actualizó por lo que el robot móvil todavía detectaba la colisión y por eso no avanza y seguía enviando la petición de ayuda al robot humanoide y este al estar atendiendo una petición y estar con el obstáculo agarrado produjo una confusión haciendo que el mismo no cumpla con la tarea.

Durante la ejecución de la tercera prueba el robot móvil colisiono con el obstáculo, trasladando consigo el obstáculo hacia el objeto de interés lo cual produjo que nunca alcance al objeto de interés, con lo que se podría decir que en esta prueba el algoritmo de colisiones no funciona.

## Capítulo VI Conclusiones y Recomendaciones

### Conclusiones

La programación del agente central de control del MAS se lo realizó con una programación por hilos utilizando el lenguaje de programación Python. La programación por hilos permitió tener en paralelo la ejecución de la interfaz para el usuario, la visión artificial dentro de la celda y la comunicación inalámbrica tanto con el robot móvil como con el robot humanoide, teniendo un flujo de datos constante entre todos los componentes del MAS.

La utilización de diferentes tecnologías dentro del proyecto de titulación hizo que cada uno de los robots cuente con un programa embebido diferente en sus controladores, los cuales les permite interpretar y ejecutar las ordenes provenientes del agente central de control.

La técnica de iluminación implementada brindo en el mayor grado posible una intensidad de iluminación homogénea constante dentro de la celda colaborativa robotizada, reduciendo significativamente las sobras generadas por los robots y los objetos a recolectar. Además, permitió que los umbrales de tolerancia de color seleccionados solo detecten los pixeles requeridos y excluyan los demás pixeles.

Para la clasificación de objetos según el color la resolución de la cámara utilizada en el proyecto fue suficiente, ya que en el procesamiento de la imagen se extrajo las características requeridas de cada color.

Se implementaron técnicas fundamentales de diversas ramas como algebra lineal y geométrica analítica que permitieron determinar la ubicación de todos los componentes y las

distancias entre los robots y objetos de interés que se encuentran dentro del espacio de trabajo de la celda colaborativa robotizada.

Se desarrolló dos algoritmos funcionales que permitieron la ejecución de los movimientos de los robots, el primero permite calcular la orientación de cada uno de los robots con respecto al sistema de referencia del espacio de trabajo, y el segundo permite determinar el sentido giro en el que tiene ir el robot para alcanzar el objetivo de interés asignado.

Los desplazamientos ejecutados por ambos robots no se pueden ajustar en tiempo real durante la ejecución del programa lo que conlleva a trabajar con una tolerancia de  $\pm 10\%$ , pero aun así haciéndoles trabajar a los robots bajo esta tolerancia se pudo obtener resultados favorables en los desplazamientos que estos ejecutan para ir a otra ubicación y cumplir con la tarea asignada.

En la ejecución del trabajo colaborativo se obtuvieron mejores resultados en el robot móvil que el robot humanoide, debido a las fallas presentadas durante la ejecución de la misma. Los algoritmos implementados, así como el método utilizado para validación del color de los objetos de interés durante el desarrollo del trabajo colaborativo tuvo una eficacia del 95.83%.

Pese a todos los inconvenientes surgidos durante en la realización del proyecto de titulación se pudo diseñar e implementar la celda colaborativa con la utilización de dos robots de características diferentes, además cabe recalcar que cada uno cuenta con tecnologías distintas que se integraron para la implementación de un sistema multi-agente y permitieron realizar la tarea de clasificación de objetos de acuerdo al color.



## Recomendaciones

En proyectos en los que se utilice técnicas de visión artificial se recomienda la implementación de alguna técnica de iluminación que permita tener intensidades de iluminación uniformes y constantes que cubran todo el espacio de trabajo, además para reducir los errores producidos por factores lumínicos externos es recomendable cubrir la totalidad del espacio de trabajo con un material aislante de luz.

Se recomienda elegir bien los colores de los objetos a clasificar procurando que estos hagan contraste con el color de fondo, por lo general los colores de fondo más usados son el color negro y el blanco, debido a que estos colores en el procesamiento de imágenes son fáciles de filtrar.

Para determinar los intervalos de las máscaras en el filtrado por color en el espacio HSV se recomienda obtenerlos bajo las mismas condiciones de iluminación, para evitar falsos positivos al momento de realizar el procesamiento de la imagen y la detección de objetos.

Se recomienda la adquisición de una cámara de alta definición con el fin de mejorar el procesamiento de la imagen y conseguir extraer las características de la imagen de manera más precisas.

Debido a las fallas encontradas durante la realización del proyecto se recomienda la adquisición de los elementos faltantes en los kits Bioloid, así como de repuestos y nuevos componentes que mejoren la manipulación de los robots.

Pese a utilizar una técnica de iluminación homogénea dentro del espacio de trabajo, algunos movimientos del robot humanoide generan sombras que impiden la detección de los

objetos de interés, por lo cual se recomienda ajustar los movimientos que se ejecutan evitando en lo mayor posible la generación de sobras, además de que estos movimientos no afecten en el equilibrio del robot.

Con los resultados obtenidos en el robot humanoide se recomienda el cambio de robot humanoide por uno que tenga mejores prestaciones y permita la integración al sistema multi-agente.

En el robot móvil el cual se tiene la comunicación inalámbrica utilizando tecnología Zigbee propia del kit Bioloid, se produjeron pequeñas desconexiones que causan retardos en la ejecución del algoritmo, se recomienda cambiarlo y usar un dispositivo de comunicación inalámbrica más actual.

### **Trabajos futuros**

La finalidad del presente proyecto de titulación es establecer una base para trabajos futuros que busquen implementar mejoras en los resultados mostrados en el documento.

Para mejorar el control de los movimientos de los robots se plantea determinar el modelo cinemático y el modelo dinámico de cada robot, y en base a estos aplicar técnicas de control que permitirán ejecutar los movimientos de una manera más precisa.

Se plantea ampliar la clasificación de objetos ya no solo por la detección del color sino también por la forma, lo que conllevaría a la adquisición de un nuevo dispositivo para la adquisición de imagen y mejorar las técnicas de procesamiento de imagen.

Otra mejora que se podría plantear es la implementación de algoritmos de planificación de trayectorias, que permitirán que los robots cumplan con la tarea asignada en un menor tiempo.

La ultima mejora planteada es en el robot humanoide, la cual consiste en agregar a la RPZW la parte del control de los actuadores AX-12A que realiza el controlador CM-510, utilizando ROS.

### Referencias Bibliográficas

- Aguirre, B., & Moyano, C. (2019). *DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA MULTIROBOT DESCENTRALIZADO PARA REALIZAR TRABAJO COLABORATIVO CON APLICACIONES LOGISTICAS, FLEXIBLES Y ESCALABLES*. Universidad de las Fuerzas Armadas ESPE, Departamento de Ciencias de la Energía y Mecánica.
- Bano, N., Roppel, T., & Gokhale, I. (2010). Use of Mobility Models for Communication in Collaborative Robotics. *42nd South Eastern Symposium on System Theory*, 143-146.  
<http://doi:10.1109/SSST.2010.5442846>
- Bañó, A. (2003). *Análisis y Diseño del Control de Posición de un Robot Móvil con Tracción Diferencial*.
- Barranco, A., & Salgado, Y. (2015). Segmentación de imágenes usando HSV y Gaussiana de Segundo orden. *AMITE Conference 2015*.
- Barrientos, V., García, J., & Silva, R. (2007). *Robots Móviles: Evolución y Estado del Arte*.  
[https://www.polibits.gelbukh.com/2007\\_35/Robots%20Moviles\\_%20Evolucion%20y%20Estado%20del%20Arte.pdf](https://www.polibits.gelbukh.com/2007_35/Robots%20Moviles_%20Evolucion%20y%20Estado%20del%20Arte.pdf)
- Bauer, W., Dworschak, B., & Zaiser, H. (2015). Weiterbildung und Kompetenzentwicklung für die Industrie. *Handbuch Industrie 4.0*, 1-14. [https://doi.org/10.1007/978-3-662-45537-1\\_36-1](https://doi.org/10.1007/978-3-662-45537-1_36-1)
- Behnke, S. (2008). *Humanoid Robots - From Fiction to Reality?* [https://www.ais.uni-bonn.de/nimbro/papers/KI08\\_Behnke.pdf](https://www.ais.uni-bonn.de/nimbro/papers/KI08_Behnke.pdf)

Bonilla, G. (2019). *DISEÑO E IMPLEMENTACIÓN DE ALGORITMOS CON ROBÓTICA COOPERATIVA PARA LA CLASIFICACIÓN DE OBJETOS DE COLOR EN UN ENTORNO ESTRUCTURADO*.

Universidad de las Fuerzas Armadas ESPE, Departamento de Eléctrica, Electrónica y Telecomunicaciones. <http://repositorio.espe.edu.ec/handle/21000/15900>

Brenes, J. (2016). *INTERFAZ DE PROGRAMACIÓN DE APLICACIONES PARA LOCOMOCION DE ROBOT HUMANOIDE BIOLOID*. Universidad Politecnica de Valencia, Departamento de Ingeniería de Sistemas y Automática.

Cancelas, J., González, R., Álvarez, I., & Enguita, J. (2016). PROCESAMIENTO MORFOLÓGICO. En Varios, *Conceptos y métodos en Visión por Computador* (págs. 77-96).

<https://intranet.ceautomatica.es/sites/default/files/upload/8/files/ConceptosyMetodos enVxC.pdf>

Castillo, R., Rosário, J., & Vargas, G. (Abril de 2013). Estrategia de Coordinación y Comunicación para Sistemas Robóticos Colaborativos. *Scientia et Technica Año XVIII, Vol. 18, No 1*, 101-107.

Domínguez, A. (1996). *Procesamiento digital de imágenes*. México: Perfiles Educativos.

EL UNIVERSO. (18 de Agosto de 2018). La industria 4.0 se desarrolla en Ecuador.

<https://www.eluniverso.com/2018/08/19/video/6914051/industria-40-se-desarrolla-ecuador>

Espinosa, J. L., Fernández, M., & Vállez, N. (2016). OPERACIONES SOBRE EL HISTOGRAMA Y FILTRADO DE LA IMAGEN. En Varios, *Conceptos y métodos en Visión por Computador* (págs. 31-46).

<https://intranet.ceautomatica.es/sites/default/files/upload/8/files/ConceptosyMetodosenVxC.pdf>

Flores, A., Chávez, G., & Rodríguez, J. (2018). Educación 4.0: Transformación del aula tradicional en las escuelas de educación básica en México. *Conference Proceedings EDUNOVATIC 2018: 3rd Virtual International Conference* (págs. 144-147). REDINE.

<https://books.google.com.ec/books?hl=es&lr=&id=s3maDwAAQBAJ&oi=fnd&pg=PA144&dq=la+educacion+4.0&ots=mhS4t2NWWe&sig=g-a2sStFhMaxVYgeuRDAPi5Tf3Q#v=onepage&q&f=false>

Flores, D. M., Guzmán, F., Martínez, Y., Ibarra, E., & Alvear, E. (2019). Educación 4.0, origen para su fundamentación. *EDUNOVATIC2019, Conference Proceedings, 4th Virtual International Conference on Education, Innovation and ICT* (págs. 655-659). REDINE.

[https://www.researchgate.net/profile/Maria\\_Cristina\\_Fernandez-Laso/publication/339984851\\_Propuesta\\_de\\_innovacion\\_didactica\\_para\\_Educacion\\_Primaria\\_basada\\_en\\_la\\_tecnologia\\_de\\_Realidad\\_Aumentada\\_aplicada\\_a\\_la\\_anatomia\\_humana\\_y\\_la\\_Prehistoria/links/5e70fc529](https://www.researchgate.net/profile/Maria_Cristina_Fernandez-Laso/publication/339984851_Propuesta_de_innovacion_didactica_para_Educacion_Primaria_basada_en_la_tecnologia_de_Realidad_Aumentada_aplicada_a_la_anatomia_humana_y_la_Prehistoria/links/5e70fc529)

Fukuda, T., Dario, P., & Yang, G.-Z. (2017). Humanoid robotics-History, current state of the art, and challenges. *Science Robotics*, 2(13), eaar4043. <https://doi:10.1126/scirobotics.aar4043>

Gómez, J. A. (2011). *ROBÓTICA MÓVIL. ESTUDIO Y CARACTERIZACIÓN DEL ROBOT MÓVIL KJUNIOR DESARROLLO DE APLICACIÓN DE ROBOT LABERINTO*. Universidad Politécnica de Cartagena.

<https://repositorio.upct.es/bitstream/handle/10317/2003/pfc4048.pdf;jsessionid=7BC950D224E40DC6EE3B44A09BACCF97?sequence=1>

González, D., & Verdugo, A. (2018). *Diseño e implementación de una arquitectura IOT para robótica colaborativa*. Universidad de las Fuerzas Armadas ESPE, Carrera de Ingeniería en Mecatrónica. <http://repositorio.espe.edu.ec/handle/21000/14730>

INFAIMON. (2020). *Visión Artificial*. <https://www.infaimon.com/enciclopedia-de-la-vision/>

Joyanes, L. (2017). Ciberseguridad: la colaboración público-privada en la era de la cuarta revolución industrial (Industria 4.0 versus ciberseguridad 4.0). *Dialnet*(185), 19-64. <https://dialnet.unirioja.es/descarga/articulo/6115620.pdf>

Kragic, D., & Vincze, M. (2009). *Vision for Robotics. Foundations and Trends in Robotics*. Now Foundations and Trends. <https://doi:10.1561/23000000001>

López, I. (2009). *Sistema Multi-Robot para detección automática de "fallas" en un circuito utilizando redes bayesianas*. [https://ccc.inaoep.mx/~esucar/Clases-mgp/Proyectos/reporte\\_final.pdf](https://ccc.inaoep.mx/~esucar/Clases-mgp/Proyectos/reporte_final.pdf)

LUCA ROBOTICS. (2020). A BRIEF HISTORY OF ROBOTICS: THE ORIGIN OF THE FIRST HUMANOID ROBOT. <https://www.lucarobotics.com/blog/first-humanoid-robot>

Martín, D., García, F., & Armigol, J. (2016). EL COLOR: MODELOS Y TRANSFORMACIONES DE LOS ESPACIOS DE COLOR. En Varios, *Conceptos y métodos de Visión por Computador* (págs. 47-60). <https://intranet.ceautomatica.es/sites/default/files/upload/8/files/ConceptosyMetodosenVxC.pdf>

Mohamed, K., El Shenawy, A., & Harb, H. (2018). A Hybrid Decentralized Coordinated Approach for Multi-Robot Exploration Task. *The Computer Journal*.

<https://doi:10.1093/comjnl/bxy107>

Moreno, F., & Ramírez, E. (2017). *Algoritmos de Visión por Computador para un SBC*.

Universidad Central de Venezuela, Facultad de Ciencias: Escuela de Computación.

[https://www.researchgate.net/publication/320107559\\_Algoritmos\\_de\\_Vision\\_por\\_Computador\\_para\\_un\\_SBC](https://www.researchgate.net/publication/320107559_Algoritmos_de_Vision_por_Computador_para_un_SBC)

MOTOROLA. (s.f.). *ALLDATASHEET.COM*. Electronic Components Datasheet Search:

<https://www.electroschematics.com/wp-content/uploads/2013/07/74LS240-datasheet.pdf>

Ogata, K. (2013). *Ingeniería de Control Moderna*. *Journal of Chemical Information and Modeling*.

<https://doi.org/10.1017/CBO978107415324.004>

Paladines, L. (2015). *Modelo cinemático directo e inverso para las piernas del robot humanoide*

*Bioloid Premium*. Escuela Superior Politécnica del Litoral, Facultad de Ingeniería en Mécanica y Ciencias de la Producción.

Pelegri, J. (21 de Junio de 2019). *Blog UNIVERSAL ROBOTS*. [https://blog.universal-](https://blog.universal-robots.com/es/beneficios-robots-colaborativos)

[robots.com/es/beneficios-robots-colaborativos](https://blog.universal-robots.com/es/beneficios-robots-colaborativos)

*Picamera*. (s.f.). <https://picamera.readthedocs.io/en/release-1.13/index.html#>

Revista de Robots. (31 de Marzo de 2020). Robots Humanoides.

<https://revistaderobots.com/robots-y-robotica/robots-humanoides/>



- Robologs. (24 de Junio de 2019). *ROBOLOGS. TUTORIALES, ROBOTS Y HUMOR CIBERNÉTICO*:  
<https://robologs.net/2019/06/24/buscar-el-centro-de-un-contorno-con-opencv-python/>
- ROBOTIS. (s.f.). *ROBOTIS e-Manual*. ROBOTIS e-Manual: <http://emanual.robotis.com/>
- Robotnik Automation S.L.L. (2020). *Robotnik*. <https://www.robotnik.es/robots-moviles/>
- Sánchez, A. J., & Rricolfe, C. (2016). SISTEMA DE CAPTURA DE IMÁGINES. En Varios, *Conceptos y métodos en Visión por Computador* (págs. 11-28).  
<https://intranet.ceautomatica.es/sites/default/files/upload/8/files/ConceptosyMetodos enVxC.pdf>
- Sandakalum, T., Suraji, S., Weerasekara, I., & Chathuranga, D. (2018). Design and Characterization of a 1-axis Compact Torque Sensor for a Collaborative Robot Arm. *2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS)*.  
<https://doi:10.1109/iciafs.2018.8913373>
- SENPLADES. (2017). *Plan Nacional de Desarrollo 2017-2021 "Toda una Vida"*.
- Skrede, O. (08 de Marzo de 2017). *COLOR IMAGES, COLOR SPACES AND COLOR IMAGE PROCESSING*. University of Oslo, Departament of Informatics The Faculty of Mathematics and Natural Sciences.  
[https://www.uio.no/studier/emner/matnat/ifi/INF2310/v17/undervisningsmateriale/slides\\_inf2310\\_s17\\_week08.pdf](https://www.uio.no/studier/emner/matnat/ifi/INF2310/v17/undervisningsmateriale/slides_inf2310_s17_week08.pdf)
- Tabango, R. (2014). *Desarrollo de un sistema de robótica colectiva con procesamiento centralizado entre dos robots humanoïdes Bioloid Premium*. Universidad de las Fuerzas

Armadas ESPE, Carrera de Ingeniería Mecatronica.

<http://repositorio.espe.edu.ec/handle/21000/7793>

Trenkle, A., & Furmans, K. (2015). Der Mensch als Teil von Industrie 4.0. *Handbuch Industrie 4.0*, 1-15. [https://doi.org/10.1007/978-3-662-45537-1\\_5-1](https://doi.org/10.1007/978-3-662-45537-1_5-1)

Vargas, G. A., & Castillo, R. A. (2015). Desarrollo de algoritmo para detección y comando de robots humanoides en tareas de recolección. *Revista Tecnura*, 127-139. <https://doi:10.14483/udistrital.jour.tecnura.2015.3.a09>

Wegener, D. (2016). Industrie 4.0 - Chancen und Herausforderungen fur einen Global Player. *Handbuch Industrie 4.0*, 1-15. [https://doi.org/10.1007/978-3-662-45537-1\\_67-2](https://doi.org/10.1007/978-3-662-45537-1_67-2)

Zambrano, F. (2018). *SIMULACIÓN, INTEGRACIÓN E IMPLEMENTACIÓN DE UN ALGORITMO DE ROBOTICA COLABORATIVA PARA DOS ROBOTS MOVILES DE ARQUITECTURA HOMOGENEA*. Universidad Militar Nueva Granda, Facultad de Ingeniería. <https://repository.unimilitar.edu.co/bitstream/handle/10654/17713/ZambranoPinillaFreyGiovanni2018.pdf?sequence=2&isAllowed=y>

**Anexos**