



UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO

**FACULTAD DE INGENIERÍA MECÁNICA Y
ELÉCTRICA**



TESIS DE INGENIERÍA

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO MECÁNICO ELECTRICISTA

**Diseño de un robot móvil para realizar tareas de
vigilancia nocturna en la facultad de ingeniería
mecánica de la U.N.P.R.G**

PRESENTADO POR:

Gustavo Adolfo Gonzales López

ASESOR:

Dr. Aníbal Jesús Salazar Mendoza

LAMBAYEQUE, PERÚ.

ENERO DEL 2020



Universidad Nacional Pedro Ruiz Gallo
Facultad de Ingeniería Mecánica y Eléctrica



Tesis de Ingeniería

Para optar el título profesional de:

Ingeniero Mecánico Electricista

**“DISEÑO DE UN ROBOT MÓVIL PARA REALIZAR
TAREAS DE VIGILANCIA NOCTURNA EN LA FACULTAD
DE INGENIERÍA MECÁNICA DE LA U.N.P.R.G”**

PRESENTADO POR:

GUSTAVO ADOLFO GONZALES LÓPEZ

PRESIDENTE : MSC. JUAN TUMIALAN HINOSTROZA

SECRETARIO : ING. CARLOS YUPANQUI RODRÍGUEZ

VOCAL : ING. OSCAR MÉNDEZ CRUZ

ASESOR : DR. ANÍBAL JESÚS SALAZAR MENDOZA

LAMBAYEQUE, PERÚ.

ENERO DEL 2020.



UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO

**FACULTAD DE INGENIERÍA MECÁNICA Y
ELÉCTRICA**



TESIS DE INGENIERÍA

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO MECÁNICO ELECTRICISTA

| CONTENIDO |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>CAPÍTULO I: PROBLEMA DE LA INVESTIGACIÓN.</p> <p>CAPITULO II: MARCO TEÓRICO.</p> <p>CAPITULO III: MARCO METODOLÓGICO.</p> <p>CAPITULO IV: PROPUESTA DE INVESTIGACIÓN.</p> <p>CAPÍTULO V: CONCLUSIONES Y RECOMENDACIONES.</p> |

| AUTOR |
|--------------------------------------|
| <p>Gustavo Adolfo Gonzales López</p> |



MSC. JUAN TUMIALAN HINOSTROZA
PRESIDENTE.



ING. CARLOS YUPANQUI RODRÍGUEZ
SECRETARIO.



ING. OSCAR MÉNDEZ CRUZ
MIEMBRO.



DR. ANÍBAL JESÚS SALAZAR MENDOZA
ASESOR.

LAMBAYEQUE, PERÚ.

ENERO DEL 2020.

INDICE.

| | |
|-------------------------------------------------------------------------------------|----|
| AGRADECIMIENTO..... | 1 |
| RESUMEN | 2 |
| INDICE TABLAS | 6 |
| INDICE FIGURAS | 7 |
| CAPÍTULO I: PROBLEMA DE LA INVESTIGACIÓN. | 9 |
| 1.1. Realidad Problemática. | 9 |
| 1.2. Formulación del Problema. | 13 |
| 1.3. Delimitación de la Investigación. | 13 |
| 1.3.1. Delimitación Espacial..... | 13 |
| 1.3.2. Delimitación Temporal. | 13 |
| 1.3.3. Delimitación Temática. | 14 |
| 1.4. Justificación e Importancia de la Tesis..... | 14 |
| 1.4.1. Justificación Técnica..... | 14 |
| 1.4.2. Justificación Económica | 15 |
| 1.4.3. Justificación Social | 16 |
| 1.5. Objetivos. | 16 |
| 1.5.1. Objetivo General..... | 16 |
| 1.5.2. Objetivos Específicos | 16 |
| CAPÍTULO II: MARCO TEÓRICO..... | 17 |
| 2.1. Antecedentes de Estudios..... | 17 |
| 1.5. Marco teórico. | 26 |
| 1.5.1. Robots móviles. | 26 |
| 1.5.1.1. Definición. | 26 |
| 1.5.1.2. Clasificación de los robots móviles según el mecanismo de locomoción. | 27 |

| | |
|----------------------------------------------------------|----|
| 1.5.1.2.1. Robots móviles con patas..... | 27 |
| 1.5.1.2.2. Robots móviles con ruedas. | 29 |
| 1.5.1.3. Arquitectura de un robot móvil. | 32 |
| 1.5.1.3.1. Sistema mecánico. | 32 |
| 1.5.1.3.2. Sistema electrónico. | 33 |
| 1.5.1.3.3. Percepción..... | 38 |
| 1.5.1.3.4. Comunicaciones. | 40 |
| 1.5.1.3.5. Electrónica de potencia. | 43 |
| 1.5.1.3.6. Sistema de control e inteligencia. | 43 |
| 1.5.1.3.7. Actuadores eléctricos. | 45 |
| 1.5.2. Conceptos utilizados..... | 48 |
| 1.5.2.1. Servidor Web HTML..... | 48 |
| 1.5.2.2. SSH (Secure Shell) | 49 |
| 1.5.2.3. PuTTY..... | 50 |
| 1.5.2.4. GitHub..... | 50 |
| 1.5.2.5. Python..... | 50 |
| CAPÍTULO III: MARCO METODOLÓGICO..... | 51 |
| 3.1. Tipo de investigación..... | 51 |
| 3.2. Procedimiento por seguir en la investigación | 51 |
| CAPÍTULO IV: PROPUESTA DE INVESTIGACIÓN | 52 |
| 4.1. DEFINICIÓN DE PARÁMETROS GENERALES DEL ROBOT MÓVIL. | 52 |
| 4.1.1. Consideraciones iniciales. | 52 |
| 4.1.1.1. Sistema de detección de presencia. | 52 |
| 4.1.1.2. Sistema de visión nocturna. | 53 |
| 4.1.2. Análisis de requerimientos de diseño. | 54 |
| 4.1.2.1. Mecánica y cinemática..... | 54 |

| | | |
|------------|----------------------------------------------------------------|----|
| 4.1.2.2. | Ambiente de trabajo. | 55 |
| 4.1.2.3. | Presupuesto | 55 |
| 4.1.2.4. | Percepción (sensores y cámara)..... | 55 |
| 4.1.2.5. | Hardware..... | 56 |
| 4.1.2.6. | Software | 56 |
| 4.1.3. | Concepto propuesto. | 59 |
| 4.1.4. | Diseño funcional del sistema de control. | 60 |
| 4.2. | SELECCIÓN HARDWARE Y DISEÑO DEL SISTEMA DE CONTROL. | 61 |
| 4.2.1. | SELECCIÓN DE SENSORES, ACTUADORES Y ELEMENTOS DE CONTROL. | 61 |
| 4.2.1.1. | Visión. | 61 |
| 4.2.1.1.1. | Parámetros generales de selección. | 61 |
| 4.2.1.1.2. | Sistema de visión seleccionado..... | 62 |
| 4.2.1.1.3. | Sistema de iluminación seleccionado..... | 62 |
| 4.2.1.2. | Sistema de control y procesamiento. | 63 |
| 4.2.1.2.1. | Parámetros generales de selección. | 63 |
| 4.2.1.2.2. | Sistema de control y procesamiento seleccionado..... | 64 |
| 4.2.1.3. | Sensores | 65 |
| 4.2.1.4. | Actuadores eléctricos..... | 66 |
| 4.2.1.5. | Sistema de potencia..... | 68 |
| 4.2.1.6. | Drivers..... | 69 |
| 4.2.1.7. | Batería..... | 70 |
| 4.2.1.8. | Comunicaciones..... | 71 |
| 4.2.2. | DISEÑO DEL HARDWARE. | 73 |
| 4.2.3. | DISEÑO DEL SISTEMA DE CONTROL..... | 74 |
| 4.2.3.1. | Elección de sistema operativo..... | 74 |
| 4.2.3.1.1. | Sistemas operativos disponibles en el mercado..... | 75 |

| | | |
|--------------------------------------------------|-----------------------------------------------------------------|----|
| 4.2.3.2. | Arquitectura del software de control..... | 77 |
| 4.2.3.2.1. | Comunicación con el usuario..... | 77 |
| 4.2.3.2.2. | Interfaz de control de locomoción con ruedas Mecanum. ... | 82 |
| 4.2.3.2.3. | Sistema de detección de movimiento. | 84 |
| 4.3. | DISEÑO MECÁNICO DEL ROBOT..... | 84 |
| CAPÍTULO V: CONCLUSIONES Y RECOMENDACIONES | | 86 |
| 5.1. | Conclusiones..... | 86 |
| 5.1.1. | Especificaciones técnicas y geométricas del sistema mecánico. . | 86 |
| 5.1.1.1. | Chasis. | 86 |
| 5.1.1.1.1. | Tapa lateral..... | 87 |
| 5.1.1.1.2. | Tapa frontal y trasera. | 88 |
| 5.1.1.1.3. | Tapa superior. | 89 |
| 5.1.1.1.4. | Base del chasis. | 90 |
| 5.1.1.2. | Soporte para el miniordenador Raspberry 3B. | 91 |
| 5.1.1.3. | Soporte cámara..... | 92 |
| 5.1.1.4. | Ruedas Mecanum. | 93 |
| 5.1.1.5. | Acople para motor y rueda..... | 94 |
| 5.1.1.6. | Pernos de sujeción..... | 95 |
| 5.1.2. | Especificaciones técnicas y geométricas del sistema electrónico | 96 |
| 5.1.2.1. | Cámara No IR v2 Raspberry Pi Sony..... | 96 |
| 5.1.2.2. | Luz LED infrarroja. | 96 |
| 5.1.2.3. | Raspberry Pi 3 B. | 96 |
| 5.1.2.4. | Microcontrolador Arduino Mega 2560 R3..... | 97 |
| 5.1.2.5. | Motor de DC Mini Gear /Core 6v - 750 rpm..... | 97 |
| 5.1.2.6. | Driver Puente H L298N. | 97 |
| 5.1.2.7. | Batería SAKO..... | 98 |
| 5.1.2.8. | Adaptador inalámbrico USB Tp-Link. | 98 |

| | | |
|------------------|-------------------------------------------------------------------|-----|
| 5.1.2.9. | Cable plano flexible CSI para cámara Raspberry Pi. | 98 |
| 5.1.3. | Especificaciones técnicas y geométricas del sistema de control. . | 99 |
| 5.1.3.1. | Creación de un servidor de transmisión de video en vivo. | 99 |
| 5.1.3.2. | Control de la locomoción..... | 100 |
| 5.1.3.2.1. | Programa Arduino. | 100 |
| 5.1.3.2.2. | GUI con Python. | 104 |
| 5.1.3.3. | Sistema de detección de intrusos. | 105 |
| 5.2. | Recomendaciones..... | 108 |
| REFERENCIAS..... | | 109 |
| ANEXOS | | 111 |

AGRADECIMIENTO

En primer lugar, agradezco a mi familia por su apoyo incondicional a lo largo de todos estos años de mi vida como universitario, sin su apoyo incondicional no hubiera culminado este importante paso en mi vida profesional. Mi familia me dio la mano en los momentos más difíciles de mi vida, su apoyo moral y económico fue la fuerza que me impulso en este camino que estuvo lleno de retos.

En segundo lugar, muy agradecido con mis formadores, fueron personas que me otorgaron una gran sabiduría durante mi estancia universitaria, me incentivaron a aprender y lograron que me apasione por mi carrera.

RESUMEN

Por mucho tiempo los robots industriales han impulsado la industria manufacturera a escalas jamás vistas antes, reemplazando la labor del ser humano en muchas tareas de manufactura. La demanda de robots no solo se limita a la industria manufacturera, por ello sus alcances se están expandiendo cada día más, siendo la agricultura, la medicina, la seguridad, la minería, los campos de aplicación más directos.

La seguridad es una labor de mucha importancia en la sociedad y demanda de un gran esfuerzo físico y mental por parte de las personas involucradas. En este contexto, algunos países están empezando a complementar las tareas de seguridad con robots móviles teleoperados o robots móviles autónomos. Los agentes de seguridad están siendo capacitados para poder trabajar de manera efectiva con estas tecnologías. Además, se verifica una reducción de la mano de obra humana, aumento de la eficiencia de trabajo y una reducción de los costos involucrados.

En comparación con los robots móviles los agentes de seguridad que trabajan en horarios nocturnos necesitan a menudo mantener una calidad de sueño reparador, si desean mantener una eficiencia de trabajo óptima, y también su salud física y mental equilibrada. Caso contrario a largo plazo pueden aparecer problemas como fatiga crónica, trastornos del sueño, trastornos gastrointestinales, perturbación del apetito, alteraciones del sistema cardiovascular, mayor incidencia de trastornos depresivos y ansiosos, entre otros.

En este contexto se plantea el proyecto “DISEÑO DE UN ROBOT MÓVIL PARA REALIZAR TAREAS DE VIGILANCIA NOCTURNA EN LA FACULTAD DE INGENIERÍA MECÁNICA DE LA U.N.P.R.G”, que tiene como objetivo diseñar un robot móvil que sea capaz de realizar tareas de vigilancia nocturna como rondas de vigilancia, inspección de ambientes y detección de intrusos. El presente diseño consta de 4 partes, primer lugar definimos los parámetros generales del robot, en segundo lugar, diseñamos el hardware y seleccionamos los sensores, actuadores y elementos de control, en tercer lugar, diseñamos el sistema de control e inteligencia, estableciendo las interfaces de control y los algoritmos para los diferentes modos de operación del robot móvil. Finalmente, el diseño mecánico del robot que consta del diseño preliminar, diseño básico y diseño detallado.

Palabras clave:

Robot Móvil, Vigilancia Nocturna, Detección De Intrusos.

ABSTRACT

For a long time, industrial robots have boosted the manufacturing industry at scales never seen before, replacing the work of the human being in many manufacturing tasks. The demand for robots is not only limited to the manufacturing industry, which is why its scope is expanding more and more, being agriculture, medicine, safety, mining, the most direct fields of application.

Security is a work of great importance in society and demands a great physical and mental effort from the people involved. In this context, some countries are beginning to complement security tasks with tele-operated mobile robots or autonomous mobile robots. Security agents are being trained to work effectively with these technologies. In addition, it verifies a reduction in human labor, an increase in work efficiency and a reduction in the costs involved.

Compared to mobile robots, security agents operating at night hours often need to maintain a good quality of sleep if they wish to maintain optimal work efficiency, and also their balanced physical and mental health. Otherwise, in the long term, problems such as chronic fatigue, sleep disorders, gastrointestinal disorders, appetite disturbance, cardiovascular system disorders, higher incidence of depressive and anxious disorders, among others, may occur.

In this context, the project "DESIGN OF A MOBILE ROBOT TO PERFORM NIGHT SURVEILLANCE TASKS IN THE MECHANICAL ENGINEERING FACULTY OF THE UNPRG" is proposed, which aims to design a mobile robot that is capable of performing night surveillance tasks such as rounds of surveillance, inspection of environments and detection of intruders. The present

constant design of 4 parts, first defined of the general parameters of the robot, secondly, we design the hardware and select the sensors, actuators, and control elements, thirdly, we design the control and intelligence system, establishing the Control interfaces and algorithms for the different modes of operation of the mobile robot. Finally, the mechanical design of the robot consisting of the preliminary design, basic design, and detailed design.

Keywords:

Mobile Robot, Night Surveillance, Intruder Detection.

INDICE TABLAS

| | |
|--------------------------------------------------------------------------------|-----|
| Tabla 1. Especificaciones de diseño de TRAXBOT. | 18 |
| Tabla 2. Ventajas y desventajas de los robots móviles con patas..... | 29 |
| Tabla 3. Resumen/Especificaciones técnicas. | 57 |
| Tabla 4. Pines de conexión / sensor PIR. | 65 |
| Tabla 5. Dirección del motor DC dependiendo de la excitación. | 69 |
| Tabla 6. Pines de conexión controlador puente H..... | 70 |
| Tabla 7. Consumo de las cargas presentes en el sistema. | 70 |
| Tabla 8. Características técnicas y geométricas del chasis. | 87 |
| Tabla 9. Características técnicas y geométricas del soporte de la cámara. | 93 |
| Tabla 10. Características técnicas y geométricas de las ruedas Mecanum. | 94 |
| Tabla 11. Especificaciones técnicas y geométricas de los pernos de sujeción. | 95 |
| Tabla 12. Costos involucrados en la construcción. | 106 |

INDICE FIGURAS

| | |
|-------------------------------------------------------------------------|----|
| Ilustración 1. Integración de ROS en la plataforma TRAXBOX. | 19 |
| Ilustración 2. Esquema de control del robot móvil. | 21 |
| Ilustración 3. Robot móvil WADORO. | 25 |
| Ilustración 4. Robot cuadrúpedo AIBO. | 28 |
| Ilustración 5. Tipos de ruedas. | 31 |
| Ilustración 6. Arduino UNO. | 37 |
| Ilustración 7. Componentes de un sensor infrarrojo. | 39 |
| Ilustración 8. Componentes básicos de una cámara digital. | 40 |
| Ilustración 9. Transmisión de datos mediante TCP/IP. | 40 |
| Ilustración 10. Modelo OSI. | 42 |
| Ilustración 11. Sistema de control de lazo abierto (actuador). | 44 |
| Ilustración 12. Sistema de control de lazo cerrado (actuador). | 44 |
| Ilustración 13. Velocidad y corriente frente al torque. | 46 |
| Ilustración 14. Vista frontal del robot móvil. | 59 |
| Ilustración 15. Vista de planta del robot móvil. | 59 |
| Ilustración 16. Vista lateral del robot móvil. | 60 |
| Ilustración 17. Esquema a bloques del diseño funcional. | 61 |
| Ilustración 18. Cámara No IR v2 Raspberry Pi Sony. | 62 |
| Ilustración 19. Leds IR. | 63 |
| Ilustración 20. Raspberry Pi 3 modelo B. | 64 |
| Ilustración 21. Arduino Mega 2560 R3. | 64 |
| Ilustración 22. Módulos PIR HC-SR501. | 65 |
| Ilustración 23. Pines de conexión. | 65 |
| Ilustración 24. Mini Gear DC motor /Core 6v - 750 rpm. | 67 |
| Ilustración 25. Circuito puente H. | 68 |
| Ilustración 26. Driver Puente H L298N. | 69 |
| Ilustración 27. Adaptador Inalámbrico USB 150 m Tp – Link. | 71 |
| Ilustración 28. Cable plano flexible CSI para cámara Raspberry Pi. | 72 |
| Ilustración 29. Rueda de aluminio Mecanum. | 72 |
| Ilustración 30. Arquitectura del sistema de control. | 74 |
| Ilustración 31. Acceso remoto a Raspberry Pi mediante VNC y PuTTY. | 78 |
| Ilustración 32. Utilización de la función generadora. | 80 |

| | |
|------------------------------------------------------------------------------------------|-----|
| Ilustración 33. Respuestas multiparte para actualizar datos en tiempo real..... | 81 |
| Ilustración 34. Generación y transmisión por partes de datos de gran volumen. | 81 |
| Ilustración 35. Interfaz física de los motores de DC. | 82 |
| Ilustración 36. Interfaz física de los sensores PIR..... | 83 |
| Ilustración 37. Vista isométrica del robot móvil de vigilancia. | 85 |
| Ilustración 38. Vistas estándar del chasis. | 86 |
| Ilustración 39. Vista 3D de la tapa lateral del chasis. | 87 |
| Ilustración 40. Vistas estándar de la tapa lateral en milímetros..... | 87 |
| Ilustración 41. Vista 3D de la tapa frontal y trasera del chasis. | 88 |
| Ilustración 42. Vistas estándar de la tapa frontal y trasera en milímetros. | 88 |
| Ilustración 43. Vista 3D de la tapa superior del chasis. | 89 |
| Ilustración 44. Vistas estándar de la tapa superior en milímetros. | 89 |
| Ilustración 45. Vista 3D de la base del chasis. | 90 |
| Ilustración 46. Vistas estándar de la base del chasis en milímetros..... | 90 |
| Ilustración 47. Vista 3D del soporte para el miniordenador Raspberry pi 3B. .. | 91 |
| Ilustración 48. Vistas estándar del soporte para Raspberry pi 3B en milímetros. | 91 |
| Ilustración 49. Vista 3D del soporte de la cámara de visión nocturna. | 92 |
| Ilustración 50. Vistas estándar del soporte de la cámara en milímetros..... | 92 |
| Ilustración 51. Vista 3D de la rueda Mecanum. | 93 |
| Ilustración 52. Vistas estándar de la rueda Mecanum..... | 93 |
| Ilustración 53. Vista 3D del acople. | 94 |
| Ilustración 54. Vistas estándar del acople en milímetros..... | 95 |
| Ilustración 55. Giro hacia arriba con ruedas Mecanum. | 101 |
| Ilustración 56. Giro hacia abajo con ruedas Mecanum..... | 101 |
| Ilustración 57. Giro a la derecha con ruedas Mecanum. | 102 |
| Ilustración 58. Giro a la izquierda con ruedas Mecanum..... | 102 |
| Ilustración 59. Giro horario con ruedas Mecanum..... | 103 |
| Ilustración 60. Giro antihorario con ruedas Mecanum. | 103 |
| Ilustración 61. Interfaz de usuario para el control de la locomoción. | 105 |

CAPÍTULO I: PROBLEMA DE LA INVESTIGACIÓN.

1.1. Realidad Problemática.

En los últimos años los alcances aplicativos de la robótica se han expandido a muchas áreas que hasta hace poco no se pensaba, conformando áreas como la robótica de servicio. La robótica de servicio está alcanzando áreas como agricultura, minería, construcción, energía, espacio, seguridad y defensa, medicina, etc. En todas estas áreas y en muchas más, hay una enorme cantidad de tareas que se pueden robotizar (Aracil et al., 2008). Esto requiere del diseño de robots con características más específicas a diferencia de la robótica industrial donde mayormente se realizan tareas repetitivas.

En la literatura podemos verificar que se están adoptando alternativas robóticas (robots móviles o aéreos) para realizar tareas de vigilancia en áreas de diferente extensión (Acevedo et al., 2012; Bora et al., 2013; Lee et al., 2011; López et al., 2013; Meghana et al., 2017; Thakur et al., 2013). Esta alternativa a permitido reducir el personal humano y a su vez mantener vigilado los ambientes sin límite de tiempo, pero fue necesario dotar a los agentes de nuevas capacidades para interactuar con los robots de forma efectiva.

La creciente adquisición de las tecnologías robóticas en el rubro de la vigilancia en los países desarrollados se debe a los siguientes factores:

- **Costo.** Contratar agentes de seguridad puede resultar costoso, y aparte de sus salarios, la instalación de cámaras que deben cubrir son un costo extra. Se ha encontrado que una mejor alternativa es contratar solo unos pocos guardias y alquilar un robot de seguridad.

- **Tardanza.** Los guardias tienden a llegar tarde, y esto puede ser una molestia para el personal que necesita cubrirlo. Los robots de seguridad nunca llegan tarde a sus trabajos y, de hecho, pueden trabajar 24/7.
- **Pereza.** Los guardias tienden a aflojar en sus trabajos y hacen su trabajo a medias. Los robots móviles no dejan de hacer su trabajo hasta que los apagas.
- **Soborno.** Los guardias de seguridad son propensos a recibir sobornos. Los robots no detendrán totalmente el soborno, ya que todavía hay guardias de seguridad humana involucrados para hacer el trabajo estratégico, pero podrían registrar y minimizar el acto de soborno.
- **Vigilancia.** Incluso si están mirando, todavía hay posibilidades de que pierdan algunas acciones sospechosas. Los robots de seguridad pueden llegar a tener hasta 4 cámaras que capturan los alrededores en 360 grados, lo que seguramente supera la instalación de múltiples cámaras de vigilancia en un área.
- **Riesgos.** Podría haber casos en que la vida de los oficiales de seguridad podría estar en juego y eso debería evitarse. Los robots pueden evitar que los humanos arriesguen sus vidas siempre que haya una situación peligrosa.

Las tareas de vigilancia se realizan durante todo el día, pero durante los turnos nocturnos existen mayores efectos negativos sobre la persona. Se han identificado que el trabajo por turnos nocturnos resulta en problemas de salud si no se lleva los cuidados necesarios, resultando en cambios a nivel biológico, familiar y social, siendo el sueño inadecuado la causa de una amplia gama de trastornos de salud.

Durante el trabajo nocturno, los agentes están sometidos a largas jornadas de trabajo sin poder dormir, resultando a largo plazo afectada su salud física y mental. Además, el sueño interrumpido aumenta el riesgo de mortalidad por cualquier causa en hombres y a largo plazo la falta de sueño causa incapacidad laboral y jubilación temprana (Avila, 2016). Algunos de los problemas asociados en personas que trabajan durante largos periodos de tiempo en estas condiciones sin los cuidados necesarios son (Avila, 2016):

- La aparición de un estado de fatiga crónica.
- La aparición de trastornos del sueño.
- La aparición de trastornos gastrointestinales y perturbación del apetito.
- Alteraciones del sistema cardiovascular.
- Mayor incidencia de trastornos depresivos y ansiosos.
- Perturbaciones en la vida familiar y social.

Por ello un sueño reparador es fundamental para el correcto desempeño en todas las esferas de la vida del trabajador, ya que, si no se toma en cuenta, el desempeño en su trabajo también se verá afectado. Los riesgos cognitivos asociados a la privación de sueño son (Kusztor et al., 2019)

- **Disminución del rendimiento.** Genera una disminución del nivel de alerta y sobre esfuerzo para realizar actividades normales.
- **Aumento del número de errores.** La privación de sueño hace que cualquier actividad esté sujeta a numerosos errores.
- **Incapacidad para conducir.** Errores en el manejo de máquinas que pueden resultar muy perjudiciales para el trabajador.

- **Merma la capacidad de atención y concentración.** Pérdida de la capacidad para trabajo sobre la marcha, procesando información nueva al mismo tiempo.
- **Estado de ánimo inadecuado.** Puede provocar estallidos emocionales como irritabilidad, impaciencia, comportamiento inmaduro, alteración de la relación con otras personas, etc.
- **Aumento de la temeridad.** Los estudios de imágenes cerebrales muestran que la privación de sueño provoca una activación de las regiones cerebrales responsables de la toma de decisiones arriesgadas, mientras que las regiones que regulan el control racional y lógica muestran una menor activación.
- **Incapacidad para reaccionar y rectificar.** La falta de sueño afecta el pensamiento flexible, la contención de pensamiento o acción, la actualización estratégica a partir de nueva información.

El sueño es vital en el rendimiento del agente de seguridad, no solo porque su salud física y mental resulta afectada, sino porque también su trabajo no será totalmente efectivo.

En conclusión, las tareas de patrullar, detectar actividades irregulares, registrar incidentes y verificar el estado del ambiente pueden ser reemplazadas por los robots móviles y en comparación con los agentes de seguridad, estos tienen capacidades sensoriales superiores, no se fatigan y además no pueden resultar dañados frente a situaciones adversas.

1.2. Formulación del Problema.

¿Cuáles serán las especificaciones técnicas y geométricas del robot móvil de vigilancia nocturna?

1.3. Delimitación de la Investigación.

Se presentan las siguientes delimitaciones de la investigación:

1.3.1. Delimitación Espacial.

El robot móvil de vigilancia nocturna se propone para su uso en los alrededores de la Facultad de Ingeniería Mecánica y Eléctrica como muestra inicial, con sede en la Universidad Nacional Pedro Ruíz Gallo, en el distrito de Lambayeque, ubicado en la Provincia de Lambayeque del departamento de Lambayeque, en el norte del Perú. Ubicado en: UTM ZONA 17M: 621069.68 m E, 9258353.52 m S.

- Altitud: 78 m.s.n.m.
- Norte: Con el distrito de Mochumí, Provincia de Lambayeque.
- Este: Con el distrito de Ferreñafe, Provincia de Ferreñafe.
- Sur: Con el distrito de Chiclayo, Provincia de Chiclayo.
- Oeste: Con el distrito de San José. Provincia de Lambayeque.

1.3.2. Delimitación Temporal.

El tiempo de desarrollo del proyecto será entre los meses de junio a diciembre del año 2019.

1.3.3. Delimitación Temática.

El desarrollo de esta tesis está enfocado en el diseño de un robot móvil para realizar tareas de vigilancia nocturna en las instalaciones de la Facultad de ingeniería mecánica y eléctrica (específicamente en los laboratorios con los equipos de mayor valor adquisitivo). El proyecto de tesis cubrirá la selección de los componentes del robot móvil, desarrollo del sistema de control, diseño mecánico y establecimiento de los costos para su futura construcción.

1.4. Justificación e Importancia de la Tesis.

1.4.1. Justificación Técnica.

La vigilancia nocturna es un trabajo que involucra un enorme sacrificio físico y mental por parte del personal de seguridad y muchas veces puede resultar afectada su calidad de vida si no se mantienen los cuidados necesarios. Por ello, los trabajos donde se involucra demasiado la integridad física del personal deberán ser reemplazados por una nueva fuerza de trabajo, que tenga una eficiencia de trabajo igual o mayor. Muchos países que se encuentran en la vanguardia tecnológica nos muestran el camino, y proponen la utilización de la robótica como solución al problema.

Los robots móviles son una tecnología bastante desarrollada en la actualidad y muchos entusiastas, investigadores y profesionales del área ponen sus proyectos a disposición de toda la comunidad, favoreciendo la investigación y la construcción de prototipos de manera autodidacta.

Las tareas que realiza un agente de seguridad pueden ser reemplazadas por robots móviles, y a largo plazo se espera que el personal humano involucrado

en las tareas de vigilancia se vaya reduciendo poco a poco. En este contexto los agentes de seguridad deberán adquirir nuevas capacidades técnicas para poder trabajar de manera complementaria con los robots de vigilancia.

En conclusión, el diseño de un robot móvil para vigilancia nocturna nos dará las herramientas para generar un prototipo que aportará un nuevo enfoque para ejercer la vigilancia nocturna, que es más eficiente y reduce los costos de operación.

1.4.2. Justificación Económica

Según la Superintendencia Nacional de Control de Servicios de Seguridad, Armas, Municiones y Explosivos de Uso Civil habría unas 900 empresas a nivel nacional brindando servicios de traslado, custodia y vigilancia bajo sistemas cerrados y con personal de confianza, además el personal disponible para este negocio está alrededor de 150 mil agentes de seguridad (Gestión, 2015). Estos números han tenido un crecimiento exponencial en el tiempo, y posiblemente sigan en aumento en los próximos años.

La seguridad privada es un sector que se ha desarrollado mucho en los últimos años, debido a las deficiencias notorias en la seguridad pública. Por ello las diversas empresas y entidades se han visto en la necesidad de recurrir a la seguridad privada para acaparar sus necesidades de vigilancia.

En conclusión, nuestro mercado objetivo seguirá requiriendo una demanda de agentes de seguridad a largo plazo.

1.4.3. Justificación Social

Las tecnologías de vanguardia solo están disponibles para instituciones con gran poder adquisitivo. En el rubro de la seguridad no es excepción, y muchas instituciones tienen que invertir los pocos recursos que tienen en sistemas de seguridad sofisticados que son una obligación por la situación de inseguridad que afronta nuestro país. La investigación y desarrollo de tecnologías en nuestro país reducen los costos de construcción y su posterior adquisición dramáticamente, al contrario de adquirirlos del extranjero. Además, el presente proyecto de tesis servirá como base para investigaciones posteriores en el campo de los robots de seguridad.

1.5. Objetivos.

1.5.1. Objetivo General

- Diseñar el robot móvil para realizar tareas de vigilancia nocturna en la facultad de ingeniería mecánica y eléctrica.

1.5.2. Objetivos Específicos

- Determinar las especificaciones técnicas y geométricas del sistema mecánico del robot móvil de vigilancia nocturna.
- Determinar las especificaciones técnicas y geométricas del sistema electrónico del robot móvil de vigilancia nocturna.
- Determinar las especificaciones técnicas y geométricas del sistema de control del robot móvil vigilancia nocturna.

CAPÍTULO II: MARCO TEÓRICO.

2.1. Antecedentes de Estudios.

En la investigación “Desarrollo de un robot móvil compacto integrado en el middleware ROS” (Araújo et al., 2014), desarrollada en el Instituto de Sistemas e Robótica (ISR), Universidad de Coímbra, Portugal (Araújo, Portugal, Couceiro, Sales & Rocha, 2014); se describe el desarrollo y montaje del robot personalizado apodado “TRAXBOT”, construido alrededor de una placa educativa Arduino, el desarrollo de un controlador de ROS para el TRAXBOT, lo que permite el control de la plataforma mediante ROS, la abstracción de hardware y reutilización de software, y finalmente la evaluación de la plataforma y el controlador utilizando múltiples plataformas TRAXBOT. El robot fue diseñado atendiendo las características de la Tabla 1.

El robot utiliza un chasis tipo oruga y de aluminio, que le otorga una mayor flexibilidad para maniobrar por terrenos accidentados; motores de CC con reductor y codificador óptico de rueda de cuadratura (624 pulsos), para 2kg de peso aproximado y 0.95 m/s de velocidad; sensores ultrasónicos con un rango máximo de 6.45 metros; una placa Arduino Uno con un microcontrolador de 8 bits ATmega328p de Atmel, que presenta una limitación en términos de memoria RAM; un módulo de comunicación inalámbrica Xbee que tiene una potencia de 1mW y alcanza entre 30 y 100 metros, para el uso en interiores y al aire libre, respectivamente; y dos paquetes de baterías de 12V y 2300 mAh de Ni-MH, para tres horas de autonomía.

Tabla 1. Especificaciones de diseño de TRAXBOT.

| | |
|---------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| ROBUSTEZ | Todo el hardware es de aluminio o acero inoxidable. |
| COSTE | El coste de la plataforma es de alrededor de 470 euros. |
| OPERABILIDAD | Tiene la capacidad de maniobrar en muchos tipos diferentes de terreno y topografías de superficie. |
| AUTONOMÍA ENERGÉTICA | Puede funcionar continuamente alrededor de 2-3 horas. |
| SISTEMA DE SENSORES | Equipado con sensores ultrasónicos de rango para permitir la interacción con el medio ambiente. |
| DIMENSIONES | Es adecuado para experimentos tanto en interiores como en exteriores. |
| FLEXIBILIDAD | Se le pueden incorporar muchas extensiones nuevas y componentes. |
| DISEÑO HÍBRIDO | Es capaz de trabajar con y sin un pequeño netbook en la parte superior de la plataforma de acuerdo con los requerimientos computacionales del usuario. |
| COMUNICACIÓN | Soporta comunicación inalámbrica ZigBee o 802.11b/g cuando se utiliza el netbook. |
| ROS COMPATIBLE | Capacidad para utilizar herramientas de ROS. |

Fuente: Recuperado de (Araújo et al., 2014)

Todos los procesos del robot fueron operados de manera remota desde un Netbook que tiene instalado ROS en su sistema operativo. Se usó una experimentación virtual y real. Los 3 sensores ultrasónicos fueron puestos a prueba con el mapeo y evasión de obstáculos en un entorno real y virtual, y también se programaron rutas autónomas simuladas en un escenario real y uno virtual.

Los autores destacan la relevancia de ROS para la reusabilidad del código, indicando la necesidad de mantener el ritmo de la investigación robótica y no tener que escribir código ya existente. ROS proporciona las bibliotecas y herramientas para ayudar a los desarrolladores de software a crear aplicaciones robóticas. Es muy beneficiosa ya que reduce en gran medida el tiempo de desarrollo y permite un mayor enfoque en las tareas científicas.

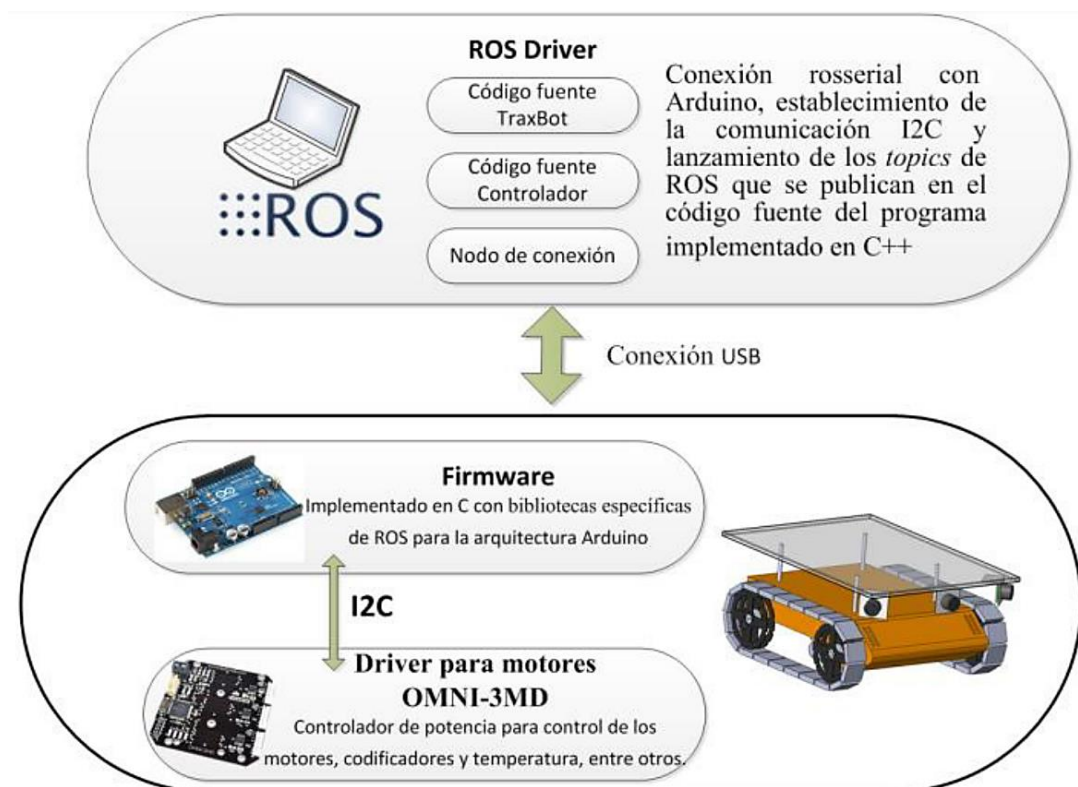


Ilustración 1. Integración de ROS en la plataforma TRAXBOX.

Recuperado de (Araújo et al., 2014)

En la tesis “Diseño y construcción de un robot móvil tele operado para la asistencia en operaciones de alto riesgo del cuerpo de bomberos” (Calle Rojas & Ullauri Guerrero, 2012), se describe el diseño de un robot móvil con tele operación para realizar labores de apoyo al cuerpo de bomberos de cuenca. El robot fue diseñado para exploración de incendios en industrias donde debe manejar

materiales peligrosos y su operación es netamente manual (tele operada) mediante una interfaz de usuario.

En el capítulo 1 se hace una breve descripción sobre la robótica móvil (tipos de robots móviles y prototipos existentes), además se describen los componentes que conforman la estructura de un robot móvil de manera general.

En el capítulo 2 se describe el procedimiento de diseño, que inicialmente fue elegir las tecnologías para el sistema electrónico, en segundo lugar, se procedió a hacer el diseño mecánico, y en tercer lugar el diseño del sistema de control e inteligencia para finalmente integrarlo al hardware del robot.

En el capítulo 3 se describe el proceso de construcción del robot, iniciando la con la construcción mecánica, posteriormente se explicó la construcción del sistema electrónico y de control. Finalmente se desarrollaron los pasos para ensamblar la electrónica en la estructura mecánica.

En el capítulo 4 se desarrollaron pruebas de funcionamiento, divididas en pruebas de laboratorio y pruebas de campo. Las primeras consistían en la depuración del software de control, configuración y puesta a punto de los dispositivos, y confirmación de correcta movilidad y accionamiento de periféricos. Las pruebas de campo consisten en operar en robot por diferentes terrenos al mismo tiempo que tomamos medidas de las diferentes variables de interés.

Las especificaciones de diseño tomadas en cuenta inicialmente fueron, de 75 cm a 120 cm de largo, de 50 cm a 75 cm de ancho, de 20 cm a 40 cm de alto, de 30 kg a 50 kg de peso, de 5 Km/h a 15 Km/h de velocidad, 45° de pendiente máxima a superar, una distancia de tele operación de 400 metros, una

autonomía de energía de 1 hora, soportar una temperatura máxima de 400 °C y finalmente una inversión máxima de 4500 dólares.

El autor resalta las ventajas de utilizar sistemas operativos libres como PUPPY LINUX que gracias a una gran comunidad de desarrolladores que lo soportan, ofrece facilidad para su instalación, alta velocidad en la ejecución de procesos e incluso la posibilidad de administrar y desarrollar remotamente.

Menciona también que el uso del lenguaje de programación C++ para el desarrollo de este proyecto ha resultado desafiante, sin embargo, se demostró que se pueden utilizar herramientas de licencia libre para desarrollar sistemas complejos de control y comunicación. Además, que la mejor opción para el flujo de vídeo es Open CV ya que permite tener un Software propio dedicado, al cual se le puede ir mejorando la eficiencia.

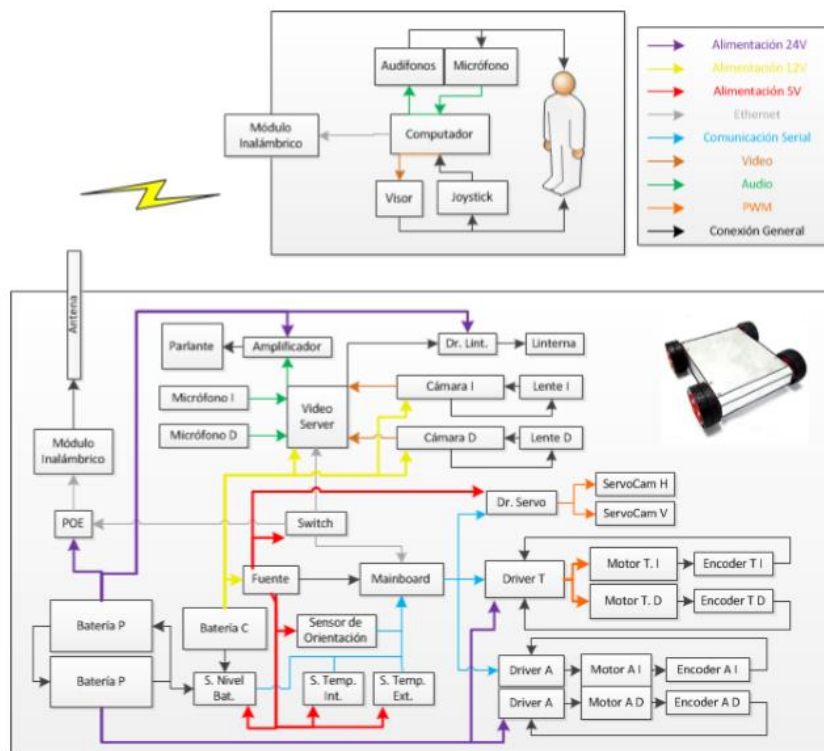


Ilustración 2. Esquema de control del robot móvil.

Recuperado de (Calle Rojas & Ullauri Guerrero, 2012)

En la investigación "Wadoro: An autonomous mobile robot for surveillance," publicado en la IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (Mittal & Rai, 2017), se presenta a WADORO (WATCH DOG ROBOT), un robot móvil autónomo para vigilancia doméstica que tiene la capacidad de detectar humanos casi en tiempo real durante todo el día usando sensores pasivos de infrarrojos y cámara de visión nocturna.

El ciclo de trabajo del robot se divide en fases de detección humana, seguimiento, rastreo, reconocimiento, y generación de alertas con fase simultánea de autoprotección.

Plantea la siguiente problemática: "Las cámaras de circuito cerrado de televisión por sí solas no han ayudado mucho a disminuir las tasas de criminalidad, ya que solo ayudan en las investigaciones posteriores a la delincuencia, a menos que sean monitoreadas continuamente por un operador humano", y plantea para compensar la necesidad de operador, algoritmos de visión por computadora que detectan movimiento utilizando la sustracción de fondo incluso en entornos complejos que se implementan en tiempo real usando tales cámaras conectadas a una computadora. Además, resalta la fusión de la cámara y múltiples sensores PIR para hacer un mejor seguimiento del objetivo humano en el seguimiento en tiempo real.

WADORO está equipado con Raspberry Pi 2 (Modelo B +) interconectado en serie con Arduino Uno. El Pi exhibe inestabilidad durante el control del servo usando la modulación de ancho de pulso (PWM); ya que el sistema operativo Raspbian (OS) instalado no es un sistema operativo en tiempo real y los tiempos

inferiores a 10 milisegundos son inexactos. Por lo tanto, este control se implementa utilizando un Arduino Uno.

El sistema de locomoción cuenta con cuatro motores reductores de corriente continua (CC) de 300 rpm; cada uno con un par de torsión de 25 kg.cm y un peso de alrededor de 400 g. Se fija una rueda de 10 cm de diámetro al eje de cada motor. Cada rueda mide 4 cm de ancho.

Para controlar la velocidad; dirección y conmutación de motores; el robot utiliza el módulo controlador de motor dual L298N. El sistema de locomoción le permite la movilidad terrestre de WADORO con una velocidad máxima de alrededor de 1,5 m/s.

La actividad del robot se puede monitorear en una PC a través de una conexión de escritorio remota, siempre que la PC y Raspberry Pi estén conectadas a la misma red Wifi.

El sistema de visión consta de una cámara Raspberry Pi No IR, que es un sensor de imagen de 5 MP sin filtro IR que le permite capturar en la oscuridad con iluminación infrarroja. El módulo de la cámara está instalado en el centro de un iluminador IR de forma circular que tiene 48 LED IR. La cámara está montada en un conjunto de giro e inclinación que es un accesorio de dos servomotores con engranajes de tal manera que permite que la cámara se desplace de 0 ° a 180 ° y se incline de 0 ° a 70 °. El sistema de autoprotección y evasión de obstáculos cuenta con sensores ultrasónicos HCSR05. Se instala otro sensor ultrasónico debajo de la base para detectar la elevación desde el suelo y otro en la parte frontal para detectar obstáculos a 160 cm.

El sistema de detección humana consiste en sensores PIR dispuestos en un triángulo. Cada ciclo de trabajo ejecutado por WADORO durante la vigilancia puede verse como una combinación de tres modos; a saber, modo inactivo; modo de patrulla y modo de seguimiento.

A. En modo inactivo

- Sigue monitoreando visualmente la presencia de cualquier persona o rostro y también detecta cualquier movimiento humano usando la disposición del sensor PIR.
- Si la presencia se detecta a través de la cámara; comienza a funcionar en modo de seguimiento.
- Si detecta algún movimiento humano a través de sensores PIR; gira hacia la dirección estimada y entra en el modo de patrulla.
- La duración máxima del modo inactivo es de 60 segundos, luego de lo cual el robot ingresa automáticamente al modo de patrulla.

B. En modo patrulla

- Continúa paseando con una velocidad de alrededor de 0.5 m/s mientras captura simultáneamente cuadros para detectar cualquier humano o rostro.
- Mientras tanto; Se protege de posibles colisiones utilizando un sensor ultrasónico. Al detectar cualquier obstáculo dentro de un rango de 100 cm por delante; inclina su cámara a 45 ° para capturar un cuadro para detectar la presencia de rostro humano; ausencia de lo cual exige que el robot navegue hasta el espacio libre de obstáculos y continúe su paseo.
- Después de 30 segundos de patrullaje; el robot ingresa nuevamente al modo inactivo para continuar con el siguiente ciclo de trabajo.

Mientras está inactivo o patrullando; si el robot encuentra a la persona visualmente; entra en el modo de seguimiento; donde sigue a la persona para obtener su toma facial de tamaño adecuado. Después de capturar la imagen facial; lo compara con las caras conocidas en la base de datos para su reconocimiento. Si el resultado del reconocimiento indica la cara como familiar; el robot se vuelve y entra en el modo de patrulla. En otro caso de rostro reconocido como intruso; el robot genera inmediatamente la alerta de llamada telefónica.

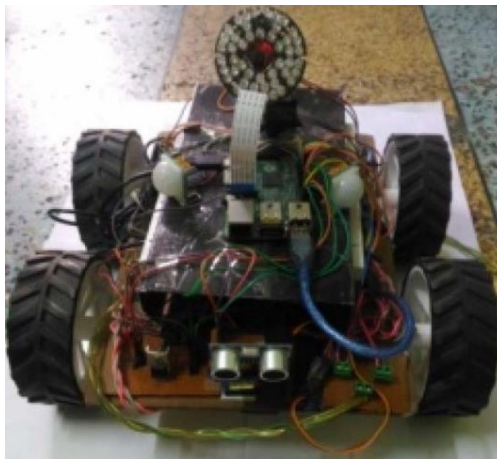


Ilustración 3. Robot móvil WADORO.

Recuperado de (Mittal & Rai, 2017)

1.5. Marco teórico.

1.5.1. Robots móviles.

1.5.1.1. Definición.

La robótica está cambiando mucho la industria manufacturera de nuestro tiempo, incrementando la velocidad, eficiencia, y calidad de la producción. La aparición de los robots industriales ha marcado un hito en el desarrollo de la industria manufacturera, y en la actualidad son una tecnología altamente consolidada en el mercado. Por ello, en la última década se está tratando que los robots extiendan sus funcionalidades para abrirse campo en otras áreas distintas de las industriales (Garcia et al., 2007).

La definición de robot de servicio surgió a partir de esta necesidad. La Federación Internacional de Robótica, organismo que coordina las actividades en esta área tecnológica de los países con mayor nivel de desarrollo define un robot de servicio como: “Un robot que opera de manera automática o semiautomática para realizar servicios útiles al bienestar de los humanos o a su equipamiento, excluyendo las operaciones de fabricación”.

Se pueden mencionar como los principales sectores de aplicación a la agricultura, la construcción, la minería, la energía, el espacio, la seguridad y defensa, y la sanidad.

1.5.1.2. Clasificación de los robots móviles según el mecanismo de locomoción.

1.5.1.2.1. Robots móviles con patas.

La facilidad que tiene la locomoción con patas para moverse por terrenos de cualquier superficie los hace superiores a los robots con ruedas, pero la desventaja de estos sistemas radica en la complejidad de su control. Muchas veces se necesitan múltiples grados de libertad para operarlos con precisión, sobre todo si se quiere replicar la biomecánica animal o humana.

En estos mecanismos es importante analizar la estabilidad debido a que no todas las patas están sobre la superficie, al contrario de la locomoción por ruedas. Además, la resistencia al movimiento se genera por el contacto de cada pata con la superficie.

A. Robots de una extremidad.

Los robots de una extremidad tienen la ventaja de poder superar con mayor facilidad los terrenos abruptos, ya que solo existe un punto de contacto con el suelo. Además, el control solo se limita a tratar de maniobrar una sola pierna, lo que es una ventaja porque la complejidad se reduce en comparación con los robots móviles que tienen más de una extremidad. Estos robots comúnmente conocidos como saltadores pueden cruzar grandes obstáculos con tan solo una zancada, pero el desafío al diseñar estos sistemas es lograr el equilibrio en su posición de reposo y durante la marcha.

B. Robots bípedos.

En el caso de los robots bípedos, se ha demostrado que corren, saltan, suben y bajan escaleras, e incluso hacen trucos aéreos como saltos mortales.

En la actualidad pueden llegar a tener hasta 4 grados de libertad en cada pierna, lo que aumenta su maniobrabilidad, pero en consecuencia aumenta la masa, el número de actuadores, la energía consumida, y la complejidad del control.

C. Robots cuadrúpedos y hexápodos

En los robots cuadrúpedos (Ilustración 4), conservar la estabilidad durante su movimiento sigue siendo un desafío, ya que el centro de gravedad debe desplazarse activamente en cada paso. Por otra parte, los robots hexápodos son altamente estables durante la marcha, y en consecuencia la complejidad del control se reduce. En la Tabla 2 se mencionan las ventajas y desventajas más notables de este sistema de locomoción (Siegwart et al., 2011),



Ilustración 4. Robot cuadrúpedo AIBO.

Recuperado de (Siegwart et al., 2011)

Tabla 2. Ventajas y desventajas de los robots móviles con patas.

| VENTAJAS | DESVENTAJAS |
|--------------------------------------------------------|------------------------------------------------------|
| Adaptabilidad y maniobrabilidad en terrenos difíciles. | Elevada potencia. |
| Capacidad para manipular objetos con gran facilidad. | Complejidad mecánica (múltiples grados de libertad). |

Fuente: Elaboración Propia.

1.5.1.2.2. Robots móviles con ruedas.

Las ruedas son elementos de locomoción simples, pero con una eficiencia considerablemente alta cuando trabaja en terrenos planos, esto hace que sea el sistema de locomoción por excelencia en la actualidad.

En el caso de utilizar ruedas en terrenos irregulares, se hace necesario la utilización de sistemas de suspensión para mantener todas las ruedas sobre el terreno y así lograr la estabilidad del robot. En lugar de preocuparse por el equilibrio, la investigación del robot con ruedas tiende a centrarse en los problemas de tracción y estabilidad, maniobrabilidad y control. Las características por evaluar para la selección de las ruedas son las siguientes:

A. Estabilidad.

- Número y geometría de puntos de contacto.
- Centro de gravedad.
- Estabilidad dinámica y estática.
- Inclinação del terreno.

B. Características de contacto.

- Tamaño y forma del punto de contacto.
- Ángulo de contacto.
- Fricción.

C. Tipo de ambiente.

- Estructura.
- Medio (agua, aire, suelo blando o duro).

A partir de estas características, se seleccionará la arquitectura de rueda óptima. Los modelos de ruedas mayormente utilizados en robots móviles se muestran en la Ilustración 5. En los modelos “a” y “b”, la rueda estándar y la rueda giratoria, “a” solo se mueve en una sola dirección y el giro en su propio eje está restringido, por otro lado, en “b” el giro en su propio eje es libre. El modelo “c”, rueda sueca, tiene rodillos adheridos con cierto ángulo de inclinación (90°), esto le permite maniobrabilidad en otras direcciones, no solo adelante o hacia atrás como en la rueda estándar. El modelo “d”, la rueda esférica es totalmente omnidireccional, lo que quiere decir que no hay dirección en la cual no se pueda mover. Al momento de seleccionar las ruedas para nuestro robot móvil debemos analizar con cuidado la disposición y geometría de las ruedas. Estas dos características son importantes porque afectan la maniobrabilidad, capacidad de control y estabilidad del sistema.

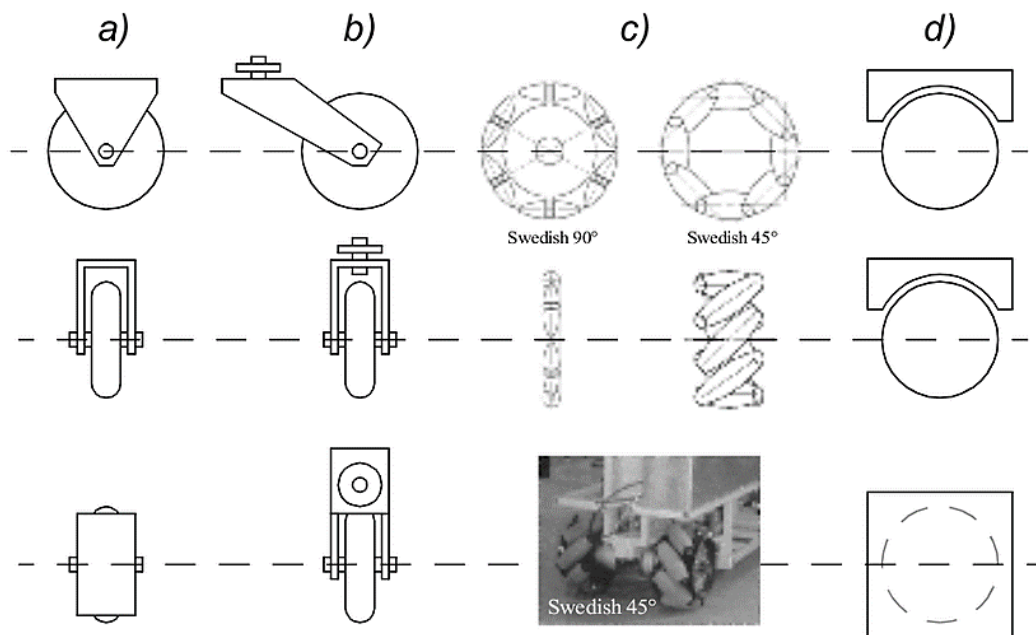


Ilustración 5. Tipos de ruedas.

Fuente: Recuperado de (Siegwart et al., 2011)

A. Estabilidad.

La estabilidad depende principalmente del número de ruedas y su distribución sobre el robot. El número mínimo de ruedas para alcanzar la estabilidad es dos. Además, el centro de gravedad debe permanecer en posiciones de equilibrio durante su reposo y movimiento.

B. Maniobrabilidad.

La maniobrabilidad es la facultad que tiene un robot para desplazarse con facilidad en diferentes direcciones. Los robots con ruedas omnidireccionales tienen una excelente maniobrabilidad.

C. Capacidad de control.

Existe una correlación inversa entre la capacidad de control y la maniobrabilidad. Por ejemplo, los diseños omnidireccionales requieren un procesamiento significativo para convertir las velocidades de rotación y traslación deseadas en comandos de rueda individuales. Además, tales diseños omnidireccionales a menudo tienen mayores grados de libertad al volante. Por ejemplo, la rueda sueca tiene un conjunto de rodillos libres a lo largo del perímetro de la rueda. Estos grados de libertad provocan una acumulación de deslizamiento, tienden a reducir la precisión de los cálculos muertos y aumentan la complejidad del diseño.

Hay que tener en cuenta que cada aplicación de robot móvil impone restricciones únicas al problema de diseño del robot, y la tarea del diseñador es elegir la configuración de unidad más apropiada posible entre este espacio de compromisos.

1.5.1.3. Arquitectura de un robot móvil.

1.5.1.3.1. Sistema mecánico.

El sistema mecánico es la estructura física del robot y cumple la función de proveer protección a los sistemas internos, y además de otorgar los mecanismos necesarios para generar el movimiento mecánico. El sistema mecánico está conformado por el sistema de tracción y el sistema de soporte.

El sistema de tracción se acciona por medio de los actuadores (hidráulicos, neumáticos o eléctricos), y contiene a las ruedas y los elementos mecánicos necesarios para transmitir la tracción si es necesario. Los actuadores

pueden ser neumáticos, hidráulicos o eléctricos según su fuente de energía y para su selección deben determinarse la mejor alternativa de potencia y eficiencia de trabajo.

Si el terreno es accidentado, el sistema de amortiguamiento es necesario para mantener todas las ruedas sobre el suelo, y mantener la estabilidad durante la locomoción.

La forma externa del robot depende del sistema de soporte, que está conformado por el chasis y la carrocería. El diseño del sistema de soporte depende del tipo de cargas (ligeras o pesadas) a las que estará sometido el robot, dependiendo del trabajo a realizar.

A menos que se use un robot móvil en partidos de lucha de sumo o vientos muy fuertes, tiene sentido mantener el peso bajo. Por ello se seleccionan los materiales con menor densidad y se controla al mínimo la cantidad de material utilizada, sin descuidar la resistencia de este. Una gran diversidad de metales, plásticos, composites, y maderas, son los materiales más comúnmente utilizados en la estructura mecánica de robots.

1.5.1.3.2. Sistema electrónico.

El sistema electrónico está conformado por un juego de sensores, que permiten al robot poder percibir su entorno; los actuadores eléctricos, que permiten su locomoción; el sistema embebido que contiene al microcontrolador o microprocesador que se encargará del control de las operaciones del robot; y finalmente las comunicaciones, que tienen la finalidad de conectar el sistema y transmitir los datos entre ellos de manera.

A. Sistema embebido.

En un sistema embebido la mayoría de los componentes se encuentran incluidos en la placa base y muchas veces los dispositivos resultantes no tienen el aspecto de lo que se suele asociar a una computadora. Algunos ejemplos de sistemas embebidos podrían ser dispositivos como un sistema de control de acceso, la electrónica que controla una máquina expendedora o el sistema de control de una fotocopiadora entre otras múltiples aplicaciones.

Los sistemas embebidos se pueden programar directamente en el lenguaje ensamblador del microcontrolador, o también en lenguajes de alto nivel como C y C++, para tareas de precisión, o lenguajes orientados a objetos como Python o Java, cuando el tiempo de repuesta no es un factor crítico. Si el sistema embebido tiene incorporado un microprocesador en vez de un microcontrolador, el dispositivo es capaz de soportar la instalación de un sistema operativo, desde el cual se puede controlar sus funcionalidades. Entre los sistemas embebidos más utilizados del mercado están ARDUINO, MBED, RASPBERRY PI, BEAGLEBONE, etc. De manera general un sistema embebido se compone de los siguientes componentes electrónicos:

- **Microprocesador, microcontrolador, DSP, etc.** Incorpora a la CPU o unidad que aporta capacidad de cómputo al sistema, pudiendo incluir memoria interna o externa, un micro con arquitectura específica según requisitos.
- **Comunicaciones.** Lo normal es que el sistema pueda comunicarse mediante interfaces estándar de cable o inalámbricas. Así un SI

normalmente incorporará puertos de comunicaciones del tipo RS-232, RS-485, SPI, I²C, CAN, USB, IP, WI-FI, GSM, GPRS, DSRC, etc.

- **Subsistema de presentación.** Suele ser una pantalla gráfica, táctil, LCD, alfanumérico, etc.
- **Salidas de señal PWM.** Para control de la velocidad en motores de corriente continua.
- **El módulo de E/S analógicas y digital.** Suele emplearse para digitalizar señales analógicas procedentes de sensores, activar diodos ledes, reconocer el estado abierto cerrado de un conmutador o pulsador, etc.
- **El módulo de reloj.** Es el encargado de generar las diferentes señales de reloj a partir de un único oscilador principal. El tipo de oscilador es importante por varios aspectos: por la frecuencia necesaria, por la estabilidad necesaria y por el consumo de corriente requerido. El oscilador con mejores características en cuanto a estabilidad y coste son los basados en resonador de cristal de cuarzo, mientras que los que requieren menor consumo son los RC.
- **El módulo de energía.** Se encarga de generar las diferentes tensiones y corrientes necesarias para alimentar los diferentes circuitos del SE. Usualmente se trabaja con un rango de posibles tensiones de entrada que mediante conversores ca/cd o cd/cd se obtienen las diferentes tensiones necesarias para alimentar los diversos componentes activos del circuito.

B. Arduino.

Arduino es una compañía de desarrollo de hardware y software libre muy utilizados para proyectos de electrónica y educación. Las placas Arduino están disponibles en forma placas de microcontrolador a un costo muy accesible, para

acercar y facilitar la programación y la electrónica a más personas. Arduino se ha vuelto tan popular que en la actualidad existe una amplia comunidad que respalda y desarrolla nuevo contenido sobre sus proyectos en internet, por lo que su aprendizaje es muy fácil.

El control de actuadores y sensores resulta muy fácil con la ayuda de una placa Arduino debido a la facilidad de su lenguaje de programación y a las librerías disponibles. Por supuesto este hardware tuvo como finalidad inicial facilitar la educación, así que sus prestaciones para proyectos de mayor complejidad deben ser analizadas con cuidado.

El hardware está constituido por un microcontrolador Atmel AVR (8 bits) conectado sobre una placa de circuito impreso a la que se le pueden conectar placas de expansión (Shields) a través de la disposición de los puertos de entrada y salida presentes en la placa. Las Shields complementan la funcionalidad del modelo de placa empleada, agregando circuitería, sensores y módulos de comunicación externos a la placa original.

El software de Arduino consiste en dos elementos: un entorno de desarrollo (IDE) (basado en el entorno de “PROCESSING” y en la estructura del lenguaje de programación “WIRING”), y en el cargador de arranque que es ejecutado de forma automática dentro del microcontrolador en cuanto este se enciende.

Las placas Arduino (Ilustración 6) se programan mediante un computador, usando comunicación serial. Algunos de los modelos más utilizados por los usuarios son Arduino Galileo, Arduino Uno, Arduino Leonardo, Arduino Due, Arduino Mega 2560, Arduino Nano.



Ilustración 6. Arduino UNO.

Recuperado de <https://store.arduino.cc/usa/arduino-uno-rev3>.

C. Raspberry Pi.

Raspberry Pi es un ordenador de placa reducida de bajo costo, que tuvo como finalidad inicial la educación de informática en escuelas. Su uso se ha extendido a otras aplicaciones como la robótica. La Raspberry Pi fue posible en parte por los avances en chips de computadoras móviles que han sucedido en los últimos años. En su corazón se encuentra un chip Broadcom BCM2835 que contiene una unidad de procesamiento central (CPU) ARM y una unidad de procesamiento de gráficos (GPU) Video Core 4. La CPU y la GPU comparten la memoria entre ellas. La GPU es lo suficientemente potente como para poder manejar la reproducción de video con calidad de Blu-ray.

Aunque inicialmente el miniordenador Raspberry Pi se desarrolló para ser utilizado como un ordenador personal, en la actualidad contiene una variabilidad de periféricos que le permite interactuar con videocámaras, sensores, actuadores, leds, interruptores, etc.

Al igual que Arduino, Raspberry Pi cuenta con una amplia comunidad de desarrolladores que la respalda, desarrollando proyectos en todo tipo de áreas, siendo algunas de ellas, Internet de las cosas, robótica, inteligencia artificial, etc.

1.5.1.3.3. Percepción.

Está conformado por los sensores que permitirán realizar el monitoreo de los parámetros del ambiente interno y externo al robot. Mientras más precisas sean las mediciones mejor se podrá recrear el ambiente donde se desenvuelve el robot. Si es operado manualmente, el operador tomará mejores decisiones y por el contrario si el robot es autónomo, este actuará de la mejor manera frente a los estímulos que se presenten. Entre los sensores más utilizados en robótica móvil se encuentran los siguientes:

A. Sensores infrarrojos

Particularmente, el sensor infrarrojo es un dispositivo optoelectrónico capaz de medir la radiación electromagnética infrarroja de los cuerpos en su campo de visión. Todos los cuerpos emiten una cierta cantidad de radiación, esta resulta invisible para nuestros ojos, pero no para estos aparatos electrónicos, ya que se encuentran en el rango del espectro justo por debajo de la luz visible.

- **Sensores pasivos.** Están formados únicamente por el fototransistor con el cometido de medir las radiaciones provenientes de los objetos.
- **Sensores activos.** Se basan en la combinación de un emisor y un receptor próximos entre ellos, normalmente forman parte de un mismo circuito integrado. El emisor es un diodo LED infrarrojo (IRED) y el componente receptor el fototransistor.

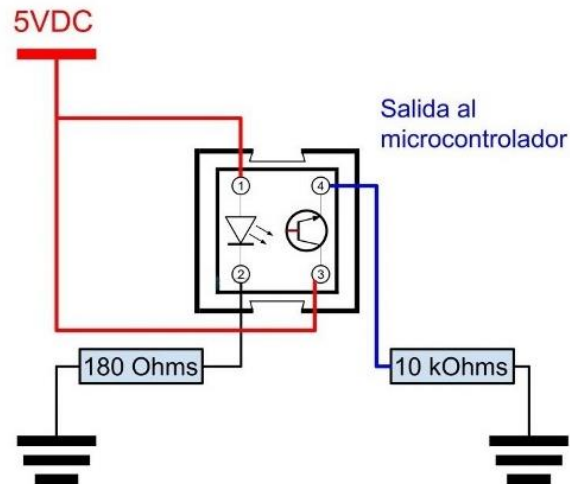


Ilustración 7. Componentes de un sensor infrarrojo.

Recuperado https://es.wikipedia.org/wiki/Sensor_infrarrojo.

B. Visión / cámaras.

Las cámaras se utilizan para capturar datos de imágenes de un objeto o ubicación examinados para una serie de aplicaciones de imágenes, como visión artificial o automatización de fábrica. Los componentes electrónicos de formación de imagen y los componentes ópticos de formación de imagen, juegan un papel importante en el rendimiento de un sistema de formación de imágenes. La integración adecuada de todos los componentes, incluyendo la cámara, la tarjeta de captura, el software y los cables, da como resultado un rendimiento óptimo del sistema (Ilustración 8).

El corazón de cualquier cámara es el sensor; los sensores modernos son dispositivos electrónicos de estado sólido que contienen hasta millones de sitios foto detectores discretos llamados píxeles. Hoy en día, casi todos los sensores en visión de máquina caen en una de dos categorías: Dispositivo de Carga Acoplada (CCD) y Semiconductor Complementario de Óxido Metálico (CMOS).



Ilustración 8. Componentes básicos de una cámara digital.

Fuente. Elaboración Propia.

1.5.1.3.4. Comunicaciones.

A. Protocolo TCP/IP.

TCP / IP, el Protocolo de Control de Transmisión / Protocolo de Internet, se puede definir como un conjunto de reglas o protocolos, utilizados para intercambiar datos entre dispositivos de hardware conectados a Internet, incluidos (entre otros) los equipos cliente y servidores. TCP / IP se subdivide en "capas" (Ilustración 9).

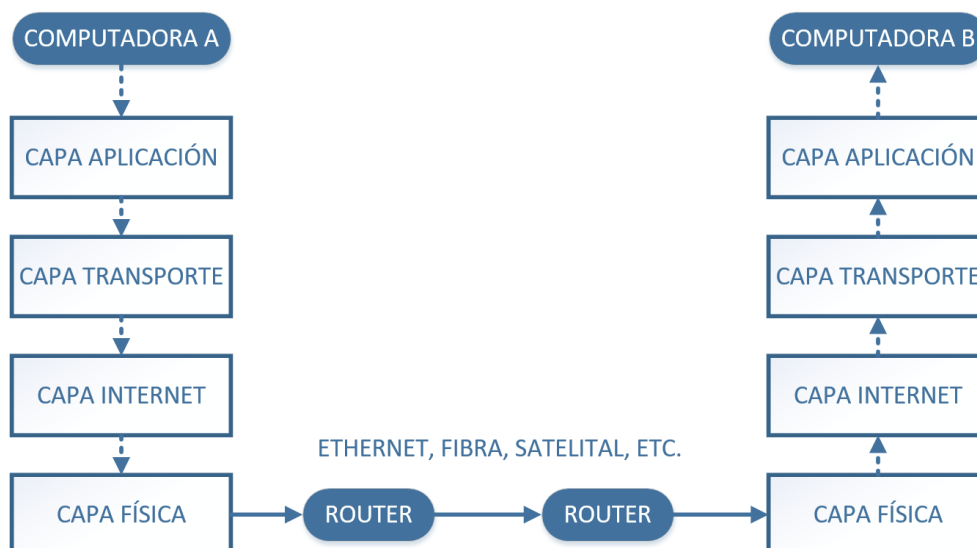


Ilustración 9. Transmisión de datos mediante TCP/IP.

Fuente. Elaboración Propia.

B. Proceso de transporte.

En la capa aplicación se realiza el procesamiento inicial de la información. Los datos completos (páginas web, correos electrónicos, etc.) se gestionan en las capas superiores. Posteriormente pasa la capa de transporte para la fragmentación en paquetes. Los segmentos se pasan a la capa de Internet/red. Aquí es donde están encapsulados en un llamado datagrama. El datagrama agrega información sobre las direcciones IP de origen y destino para los datos. Los datagramas se pasan a la capa de enlace OSI (capa 2 de OSI), donde se encapsulan en un marco que contiene información sobre la dirección MAC del dispositivo emisor, antes de pasar a la capa física para la transmisión a través de la red.

La capa física es donde los paquetes se mueven físicamente, usando cables de ethernet, satélites, fibras, sistemas inalámbricos, dependiendo de lo que esté disponible entre la computadora A y B. Si las computadoras están en la misma habitación, pueden conectarse a través de un par de ethernet cables con un enrutador en el medio, o tal vez de forma inalámbrica, si la computadora está conectada de forma inalámbrica al mismo enrutador.

C. Modelo OSI.

El modelo OSI (Ilustración 10) proporciona un marco importante para comprender mejor la estructura de los paquetes TCP/IP y cómo se intercambian los datos entre dispositivos conectados dentro de una red y a través de redes a través de dispositivos de red físicos, es decir, conmutadores y enrutadores, que se analizarán con más detalle en la siguiente sección de este capítulo.

D. VNC (Virtual Network Computing).

VNC (Computación Virtual en Red) es un software libre basado en una estructura cliente / servidor, donde un ordenador cliente puede acceder y observar lo que sucede en el ordenador servidor. Esta plataforma permite acceder a un ordenador desde cualquier parte del mundo, ya que funciona a través de internet. El software VNC es simple y sencillo de usar, y muchos sistemas operativos lo incluyen. Esto permite trabajar de forma remota, y con ordenadores que cuenten con el mismo software. VNC es utilizado mayormente por administradores de sistemas, ya que desde un solo ordenador se pueden gestionar muchos ordenadores, evitando estar presente en el lugar. Otra aplicación común es la educación, donde todos los equipos del aula pueden mostrar una copia de lo que sucede en el ordenador del profesor a medida que este desarrolla la clase.

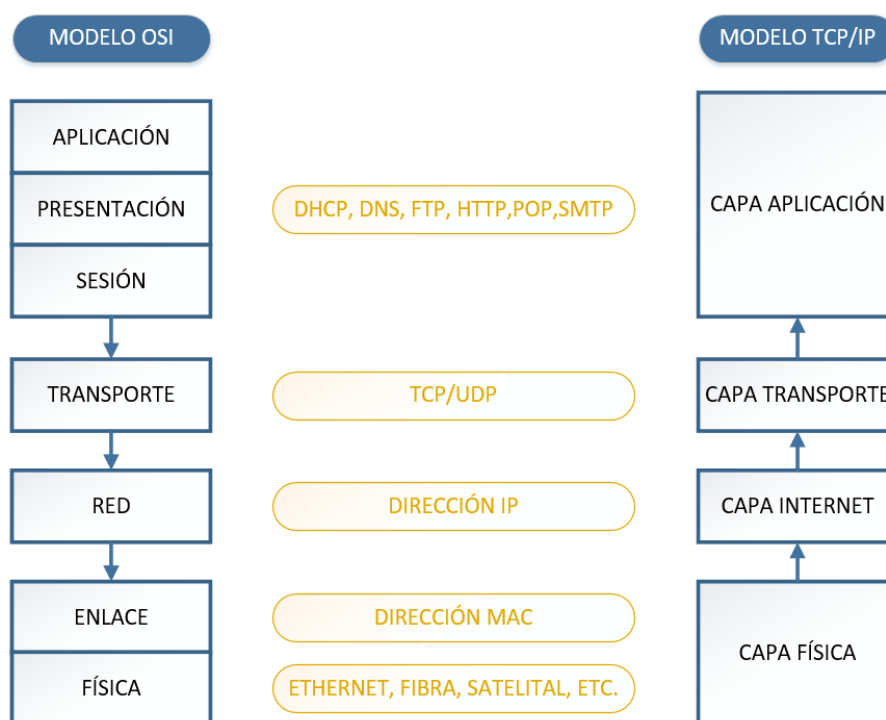


Ilustración 10. Modelo OSI.

Fuente. Elaboración Propia.

1.5.1.3.5. Electrónica de potencia.

Son circuitos que sirven de interfaz entre los elementos de control y los actuadores. Se encargan de proveer controladamente la potencia requerida por los actuadores. Entre ellos tenemos:

- Fuentes de alimentación.
- Drivers de motores.
- Contactores.
- Relés.

1.5.1.3.6. Sistema de control e inteligencia.

Es el sistema encargado de dotar al robot de la capacidad de tomar decisiones, ya sea siguiendo las órdenes que le suministre un operario en tiempo real, o para tener un comportamiento autónomo. Esto se logra mediante la implementación de algoritmos dentro de un microcontrolador o microprocesador. Los algoritmos tienen que estar desarrollados en un lenguaje de programación, que pueden ser de dos tipos, lenguajes de alto nivel y lenguajes de bajo nivel. Desarrollar algoritmos en lenguajes de programación de alto nivel es menos tedioso y entendible que hacerlo en lenguajes de bajo nivel como el ensamblador. Los lenguajes de alto nivel más comunes son, Java Script, C++, Python, Matlab, etc. Los sistemas de control pueden ser de dos tipos:

A. Lazo abierto.

En los primeros el sistema de control responde a la señal de entrada y cambia la señal de salida en consecuencia. En los sistemas de control de bucle abierto (Ilustración 11), la información tiende a fluir solo de una manera. Por

ejemplo, el sistema de control dentro del control remoto nunca descubre si el televisor se enciende o no. Además, el botón de encendido del control remoto nunca indica si el haz de infrarrojos se envió o no.

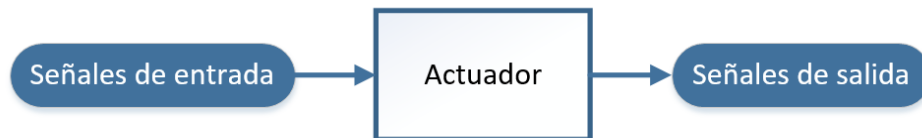


Ilustración 11. Sistema de control de lazo abierto (actuador).

Fuente. Elaboración Propia.

B. Lazo cerrado.

Los sistemas de control de circuito cerrado (Ilustración 12) también se conocen como sistemas de control de retroalimentación, porque la información fluye hacia atrás en algún punto dentro del sistema de control. En general, esta información inversa fluye desde la salida del sistema de control hacia atrás, hacia la entrada. La información que fluye hacia atrás permite que el sistema de control realice correcciones en su salida.

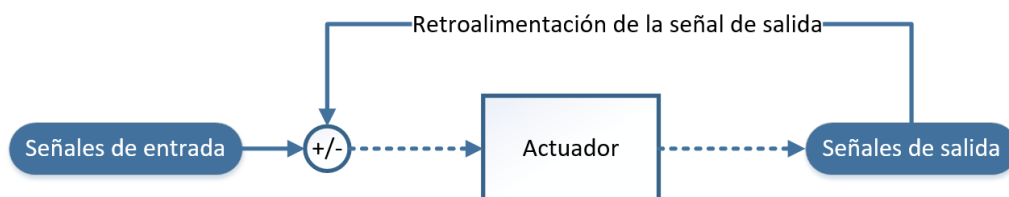


Ilustración 12. Sistema de control de lazo cerrado (actuador).

Fuente. Elaboración Propia.

1.5.1.3.7. Actuadores eléctricos.

A. Tipos de actuadores eléctricos.

- **Servo modificado:** Los servos son muy utilizados en robótica. Tienen un control electrónico que les permite mantener una posición y ejercer una fuerza considerable. Es necesario desmontarlos y modificarlos para que puedan rotar continuamente. Tienen la desventaja de no ser muy rápidos.
- **Motores de corriente continua.** Los motores de DC proporcionan una velocidad proporcional al voltaje aplicado, además pueden alcanzar velocidades de giro altas por lo que es necesario usar cajas reductoras para adaptar la velocidad a un rango utilizable.
- **Motores paso a paso.** Los motores paso a paso tienen una gran precisión y repetibilidad en cuanto al posicionamiento, por ello son más caros que los motores de corriente continua.

B. Funcionamiento de un motor de DC.

Aunque el motor de DC no tiene un comportamiento lineal en la realidad, se puede simplificar con mucha precisión a ello. Como vemos en la siguiente gráfica la velocidad de giro del motor depende linealmente del torque (Ilustración 13). Si el motor no tiene que ejercer fuerza alguna, será libre de girar a su velocidad máxima, y su consumo de corriente es mínimo. Cada vez que la carga aumente sobre él, su velocidad de giro empezará a reducirse, y su consumo de corriente irá en aumento. Si la carga es muy elevada el motor dejará de girar, este estado se llama “STALL” (corriente máxima).

El voltaje del motor aproximadamente es proporcional a la corriente y el torque, por ejemplo, si usamos el doble de voltaje el motor girará al doble de velocidad se necesitará el doble de corriente y ejercerá el doble de fuerza. Pero siempre considerando los rangos de voltaje de funcionamiento.

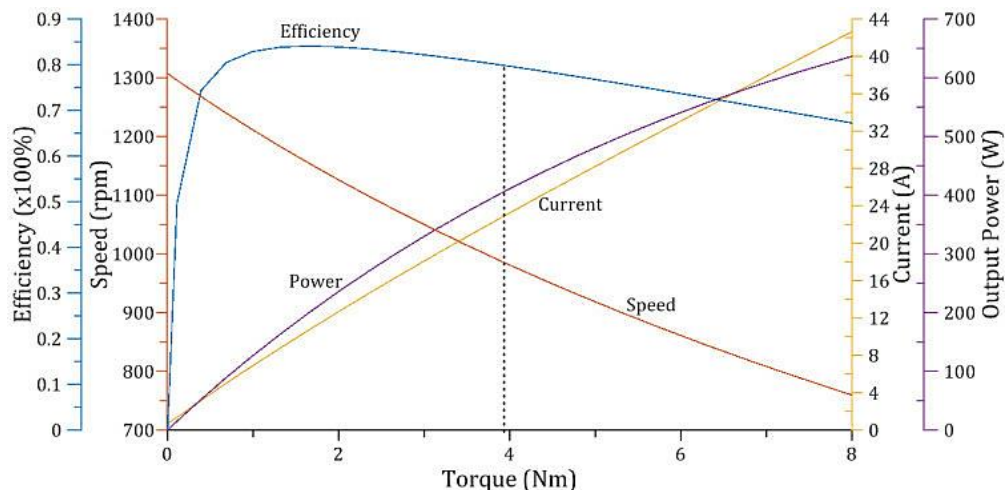


Ilustración 13. Velocidad y corriente frente al torque.

Recuperado de (Karnavas et al., 2018)

- **Parámetros del motor.** Torque, es la fuerza que ejerce el motor. Velocidad, es la rapidez con la que gira el motor. Corriente, es la intensidad que necesita el motor para funcionar.
- **Especificaciones del motor.** Voltaje al que se suministran las especificaciones, velocidad de giro libre, corriente de giro libre, torque de Stall, corriente de Stall, punto de funcionamiento del motor.

C. Relaciones importantes en las variables de un motor eléctrico.

- **Relación entre el torque y la velocidad de giro.**

$$N(T) = N_1 \left(\frac{T_s - T}{T_s} \right)$$

Donde: N: Velocidad de giro, N_l : Velocidad de giro libre, T: Torque aplicado, T_s : Torque de Stall.

- **La relación entre el torque y la corriente:**

$$I(T) = I_l + (I_s - I_l) \frac{T}{T_s}$$

Donde: T: Torque aplicado, T_s : Torque de Stall, I: Corriente, I_l : Corriente de giro libre, I_s : Corriente de Stall.

- **Relación de los parámetros con el voltaje.**

$$T(V) = T_0 \cdot \frac{V}{V_0}$$

$$I(V) = I_0 \cdot \frac{V}{V_0}$$

$$N(V) = N_0 \cdot \frac{V}{V_0}$$

- **Torque del motor a partir de la potencia de salida:**

$$P = T \cdot W$$

$$P = POT_e - P_m - P_e - P_o$$

Donde: P: Potencia del motor (Watts), POT_e : Potencia eléctrica, P_m : Pérdidas mecánicas, P_e : Pérdidas eléctricas, P_o : Otras pérdidas, T: Torque del motor (N. m), W: Velocidad angular del motor (rad/s).

- **Tracción un vehículo móvil o fuerza de trabajo.** Es equivalente a la fuerza de rozamiento del robot con el suelo.

$$F_R = u \cdot P$$

Donde: P: Peso, u: Coeficiente de rozamiento.

- **Torque requerido.**

$$T = F_R \cdot r$$

Donde: F_R : Tracción o fuerza de trabajo, r : Radio de la rueda.

- **Velocidad en función del radio de la rueda.**

$$V = 2\pi \cdot R \cdot N_l \left(1 - \frac{F \cdot R}{T_s}\right)$$

Donde: F : Tracción o fuerza de trabajo, N_l : Velocidad de giro libre,

T_s : Torque de Stall, R : Radio de la rueda.

1.5.2. Conceptos utilizados.

1.5.2.1. Servidor Web HTML.

Quienes tengan conocimientos sobre lo que es un servidor no deben confundirlo con un servidor web, porque son dos cosas distintas, aunque sí es cierto que uno forma parte del otro, ya que de hecho el servidor web es uno de los componentes de un servidor. El server (o servidor) es el equipo en el cual se alojan los sitios o aplicaciones web, mientras que el servidor web es un software que forma parte del servidor. Los servidores dedicados en particular son equipos utilizados para el alojamiento de sitios y aplicaciones web, así como para servidor de bases de datos y otros tipos de contenido. Cuentan con los componentes de hardware tradicionales de una computadora, como un CPU, memoria RAM, unidades de almacenamiento, etc. El servidor web es el software que se encarga de despachar el contenido de un sitio web al usuario.

El proceso de despacho comienza en nuestro navegador web. Al escribir la dirección de un sitio web y presionar “ENTER” comienza la siguiente secuencia: el sistema hace una búsqueda para encontrar en cuál servidor está alojado el sitio en cuestión. Cuando el server es encontrado, el navegador le pide el contenido del sitio web, y acto seguido el web server procesa este pedido y envía dicho contenido al navegador, lo cual da como resultado la visualización del sitio en nuestra pantalla. Para que el servidor web pueda cumplir con su rol, tendrá primero que nada que recibir una petición de parte de un navegador, y a continuación buscar en los archivos del servidor la información que le está siendo solicitada. Una vez que los archivos del sitio son localizados, el servidor web procede a interpretar las líneas de código y a enviar el resultado de regreso al navegador, el cual lo mostrará para nosotros.

1.5.2.2. SSH (Secure Shell)

SSH es un protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor y que permite a los usuarios conectarse a un host remotamente. SSH encripta la sesión de conexión, haciendo imposible que alguien pueda obtener contraseñas no encriptadas.

SSH está diseñado para reemplazar los métodos más viejos y menos seguros para registrarse remotamente en otro sistema a través de la Shell de comando. El uso de métodos seguros para registrarse remotamente a otros sistemas reduce los riesgos de seguridad tanto para el sistema cliente como para el sistema remoto (recuperado de <https://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-es-4/ch-ssh.html>).

1.5.2.3. PuTTY

Es un cliente SSH, disponible para Windows y mediante el cual podemos comunicarnos a servidores remotos. PuTTY nos permitirá ejecutar comandos y hacer modificaciones a alguna parte del servidor remoto. Más información en <https://www.putty.org/>. Algunas de las ventajas de PuTTY son las siguientes:

- Es gratuito y código abierto.
- Disponible para varias plataformas.
- Es una aplicación portable.
- Interfaz sencilla y manejable.
- Muy completo y ofrece una gran flexibilidad con multitud de opciones.
- Está en constante desarrollo.

1.5.2.4. GitHub

GitHub es una plataforma de desarrollo colaborativo que facilita el alojamiento y revisión de código, administración proyectos y creación software junto con 40 millones de desarrolladores. Para más información acceder a <https://github.com/>.

1.5.2.5. Python

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma. Para más información ingresar a <https://www.python.org/>.

CAPÍTULO III: MARCO METODOLÓGICO

3.1. Tipo de investigación

Descriptiva.

3.2. Procedimiento por seguir en la investigación

El diseño de un robot móvil implica el uso iterativo de las metodologías de diseño mecánico, electrónico y de control.

- **Definición de los parámetros generales del robot.** Los parámetros generales o especificaciones técnicas del robot móvil son definidas por el diseñador y dependen de las características de trabajo que tendrá el robot.
- **Diseño del hardware y selección de sensores, actuadores y elementos de control.** A partir de los parámetros del robot se seleccionan las tecnologías más adecuadas para suplir esas demandas.
- **Diseño del sistema de control e inteligencia.** En este paso se establecen las interfaces de control para los sensores y motores de DC, los algoritmos para los diferentes modos de operación del robot móvil, y la interfaz para la comunicación remota.
- **Diseño mecánico del robot.** Podemos distinguir 3 etapas necesarias: diseño preliminar, diseño básico y diseño detallado.

CAPÍTULO IV: PROPUESTA DE INVESTIGACIÓN

4.1. DEFINICIÓN DE PARÁMETROS GENERALES DEL ROBOT MÓVIL.

4.1.1. Consideraciones iniciales.

Los agentes de seguridad realizan las siguientes tareas durante los turnos nocturnos. La primera es la detección de actividades irregulares y verificación del estado del ambiente. Durante la tarea el agente debe detectar actividades sospechosas y dar aviso de inmediato a la central de monitoreo si es necesario. Se utilizan radioteléfonos para la comunicación remota entre el agente y la central de monitoreo, donde otros agentes están disponibles. Es una tarea que se complementa con rondas de seguridad continuamente durante la noche con la finalidad de monitorear las condiciones en las que se encuentra la propiedad.

La segunda tarea llevada a cabo por los agentes es el registro de incidentes. Durante esta tarea el agente debe hacer una descripción de las actividades sospechosas detectadas durante el turno de trabajo. Esta tarea tiene que ser lo más detallado posible y debe ser incluida la hora del suceso, además se debe expresar de forma escrita u oral. Según las tareas mencionadas se deben considerar los siguientes sistemas de percepción:

4.1.1.1. Sistema de detección de presencia.

Los sensores PIR detectarán la variación del espectro infrarrojo en el ambiente, permitiendo detectar la presencia de una persona con mucha precisión. Aún durante la noche los sensores serán de mucha ayuda para los agentes de seguridad ya que aún con la incorporación de una cámara, existen puntos que están fuera de su alcance de visión. El sistema cuenta con 4

sensores PIR dispuestos sobre las 4 caras del cuerpo del robot móvil. Los sensores PIR tiene un rango de detección de 110° y un alcance de detección de 7 metros como máximo, por lo que no existirán puntos ciegos. Los sensores PIR al momento de detectar una presencia, generan una señal de salida por uno de sus pines de conexión, en este momento se dará una señal de alerta al operador.

4.1.1.2. Sistema de visión nocturna.

Una de las limitaciones humanas durante las tareas de vigilancia es la visión nocturna. Por ello una cámara con visión nocturna montada sobre el robot móvil dotará al operador con la capacidad de reconocer objetos con mucha precisión. Un operario capacitado debe ser capaz de tele operar el robot por medio de un computador personal y un ratón. Se desarrollará una interfaz de usuario que constará de una serie de comandos que permitirán realizar la locomoción del robot móvil de manera sencilla.

Todos los sistemas deberán trabajar en paralelo, por ejemplo, si el sistema de detección de presencia detecta algún intruso y el operador recibe la alerta, la cámara de visión nocturna debe estar activa en todo momento, para así poder darle seguimiento al intruso. La cámara de visión nocturna transmitirá video en tiempo real y en todo momento, por lo que el operador podrá percatarse de cada detalle al mismo tiempo que controla en movimiento del robot mediante la interfaz de usuario.

4.1.2. Análisis de requerimientos de diseño.

4.1.2.1. Mecánica y cinemática.

Como condición inicial se plantea la construcción de un prototipo de pequeñas dimensiones, ya que el costo de los materiales empleados aumenta con el tamaño del robot. La construcción de un prototipo de pequeñas dimensiones nos da la facilidad de probar los algoritmos de control sin incurrir a gastos elevados, además reduce su peso y lo hace fácilmente transportable.

Se busca una arquitectura compacta y robusta con la finalidad de proteger a los componentes internos de los agentes externos que puedan afectar su integridad (polvo, lluvia, gases, personas, etc.). Los componentes externos como la cámara de visión nocturna también deben estar protegidos, por ello llevarán su respectiva protección. Se seleccionó aluminio por sus buenas características de resistencia, densidad y anticorrosión. También por su fácil adquisición en el mercado y maquinabilidad, además de su buena resistencia a la corrosión atmosférica.

Un sistema de locomoción por ruedas otorga mayor facilidad de control y tiene excelente rendimiento si trabaja sobre superficies planas. El robot móvil se desplazará en ambientes interiores como centros comerciales, universidades, colegios, hospitales, etc., que en la mayor parte de los casos estarán libres de superficies abruptas. Por esta razón, no será necesario la incorporación de un sistema de amortiguamiento. Se elige un sistema de locomoción con 4 ruedas, debido a las ventajas de estabilidad durante su movimiento y reposo. Además, la incorporación de ruedas omnidireccionales nos otorgará una maniobrabilidad en todas las direcciones.

Los motores seleccionados deben ser capaces de girar continuamente 360° y poseer cambio de sentido de giro. Por ello los motores de DC o los motores PAP serán los indicados. Ambos necesitarán de una fuente de corriente eléctrica de DC que en nuestro caso serán baterías. La batería debe contar con la posibilidad de recargarse y poseer una duración de 1 hora con todos los sistemas del robot activados. La velocidad de un agente durante las rondas de vigilancia oscila entre 0.8 – 2 m/s. Por ello la velocidad de diseño seleccionada estará entre este intervalo.

4.1.2.2. Ambiente de trabajo.

El material seleccionado debe sostener las siguientes condiciones.

- Humedad relativa. Oscila entre 70/75 %.
- Temperatura máxima. 33 °C. Los valores máximos de temperatura se miden en los meses de enero hasta abril.
- Temperatura mínima. 15 °C. La temperatura llega a bajar hasta 25 °C los meses de junio a setiembre, y de Setiembre a diciembre se reduce hasta 15 °C.

4.1.2.3. Presupuesto

El presupuesto inicial no debe superar los 2000 soles.

4.1.2.4. Percepción (sensores y cámara)

El robot móvil debe tener una cámara con visión nocturna (mínimo 720p y 60fps). La visión nocturna se logra con una cámara sin filtro IR. El filtro IR está dentro de las cámaras de uso común, y tiene la finalidad de eliminar el efecto de la radiación infrarroja sobre la imagen de salida. Una fuente de luz IR proyectada

por una fuente externa nunca será detectada por una persona, ya que la luz IR no está en el espectro visible para el ojo humano. El robot necesitará una luz infrarroja artificial que iluminará su recorrido y mejorará la calidad de la imagen transmitida. La detección de intrusos se llevará a cabo mediante la incorporación de 4 sensores PIR, que tienen un costo muy cómodo y una precisión aceptable para nuestra aplicación.

4.1.2.5. Hardware

El hardware utilizado debe permitir la flexibilidad necesaria para incorporar nuevos componentes y extensiones en el futuro. Además, debe permitir la transmisión remota de datos a un ordenador personal de manera inalámbrica y a una distancia máxima de 100 metros al aire libre. Se debe priorizar el uso de hardware de bajo costo destinado solo a la elaboración de prototipos, como Raspberry Pi y Arduino.

4.1.2.6. Software

El software debe ser de fuente abierta, lo que significa que debe estar libre de derechos de autor y no debe involucrar costos adicionales. Además, debe ser soportado por una amplia comunidad de desarrolladores y ser libre de modificar.

Tabla 3. Resumen/Especificaciones técnicas.

| | | |
|-------------------|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Geometría | Ancho promedio | 30cm |
| | Largo promedio | 30cm |
| | Alto promedio | 30cm |
| | Arquitectura externa | Diseño compacto. |
| Cinemática | Velocidad promedio | 0.5 - 2 m/s. |
| | Tipos de movimientos | Omnidireccional. -Avance/Retroceso. -Derecha/Izquierda. -Giro Horario/Antihorario sobre su propio eje. |
| Locomoción | Superficies | Planas. |
| Fuerzas | Peso máximo | 1.5 kg |
| | Cargas extras | 0.5 kg. |
| Energía | Cargas eléctricas | Motores paso a paso. Tarjetas electrónicas. -Raspberry Pi 3B. -Arduino Mega. Sensores PIR. Cámara de visión nocturna. Comunicación Wifi. |
| | Tiempo de operación | Una duración de 1 hora manteniendo todos los sistemas activos. |
| | Tipo | Baterías eléctricas. |
| | Eficiencia | 90 – 95 %. |

Fuente. Elaboración Propia.

Tabla 4. Resumen/Especificaciones técnicas (continuación).

| | | |
|---------------------------------|-----------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Señales | Entradas | Sensores PIR. Cámara de visión nocturna. |
| | Salidas | Motores paso a paso. |
| | Distancia de control. | 100 metros. |
| | Equipo de control | Arduino Mega. Raspberry Pi. |
| Ergonomía | Relación hombre maquina | La interfaz de usuario es a través de una aplicación desarrollada en lenguaje de programación Python. |
| | Claridad del diseño | Fácilmente entendible por una persona con conocimientos básicos en computación. |
| | Tipo de operación | El uso de una PC y un ratón permite la ejecución de los comandos de la interfaz de control. Así el operador tiene la facilidad de controlar la locomoción del robot móvil de manera remota. |
| Transporte | Fácilmente transportable. Bajo peso y tamaño. | |
| Material | Aluminio. | |
| Condiciones de Operación | Humedad relativa. Oscila entre 70/75 %. Temperatura máxima. 33 °C. Temperatura mínima. 15 °C. | |

Fuente. Elaboración Propia.

4.1.3. Concepto propuesto.

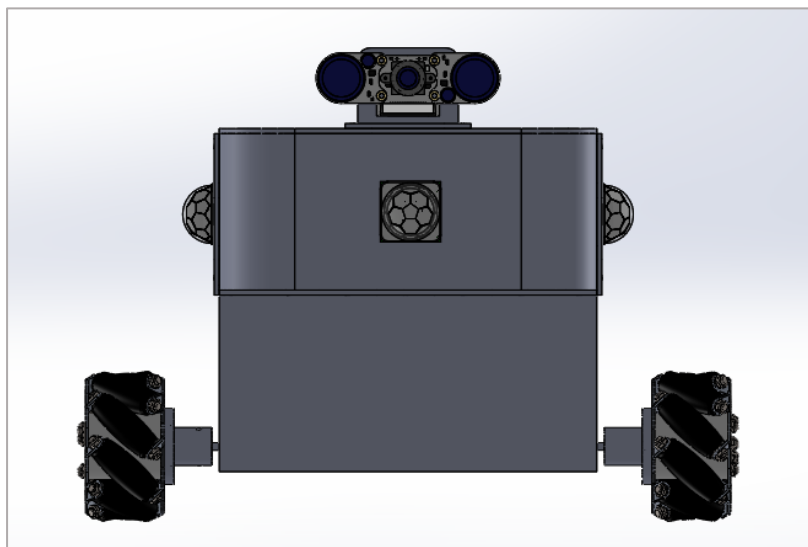


Ilustración 14. Vista frontal del robot móvil.

Fuente: Elaboración propia.

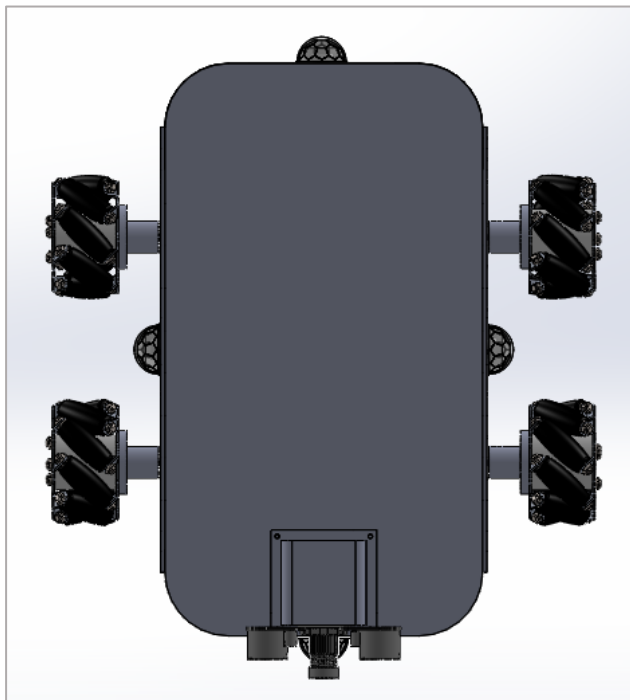


Ilustración 15. Vista de planta del robot móvil.

Fuente: Elaboración propia.

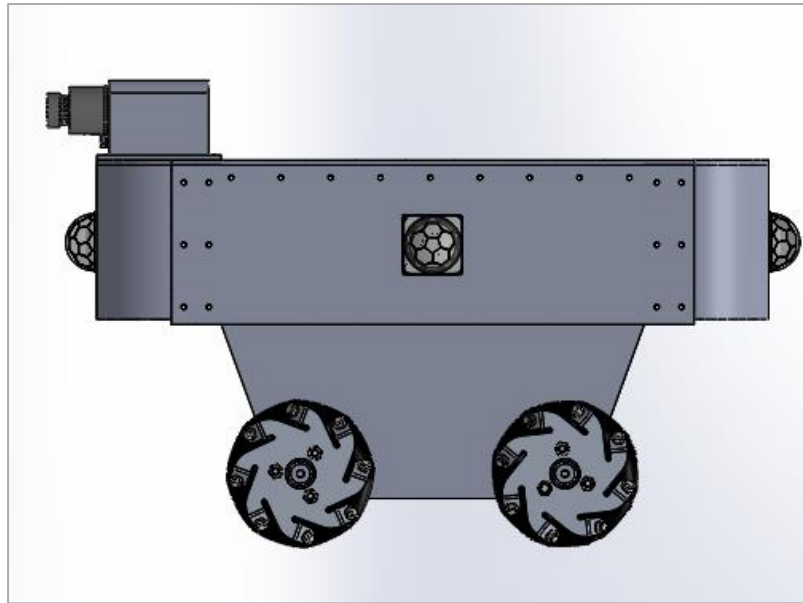


Ilustración 16. Vista lateral del robot móvil.

Fuente: Elaboración propia.

4.1.4. Diseño funcional del sistema de control.

Se define el diseño conceptual del sistema de control como se muestra en la Ilustración 17. El sistema necesita una unidad central de procesamiento, a la cual se conectarán los demás bloques y se encargará del procesamiento y el control de las señales entrada y las señales de video. El sistema de potencia se encarga de manejar a los actuadores. El sistema de sensores consta de sensores internos y externos, los internos miden las variables de los actuadores y los externos las variables del ambiente.

El sistema de comunicaciones enviará toda la información al usuario de manera remota (equipo de visualización y comando), además el usuario a su vez podrá enviar información de comando hacia el robot móvil mediante una interfaz instalada en el computador personal. La fuente de energía mantiene activo a todo el sistema.

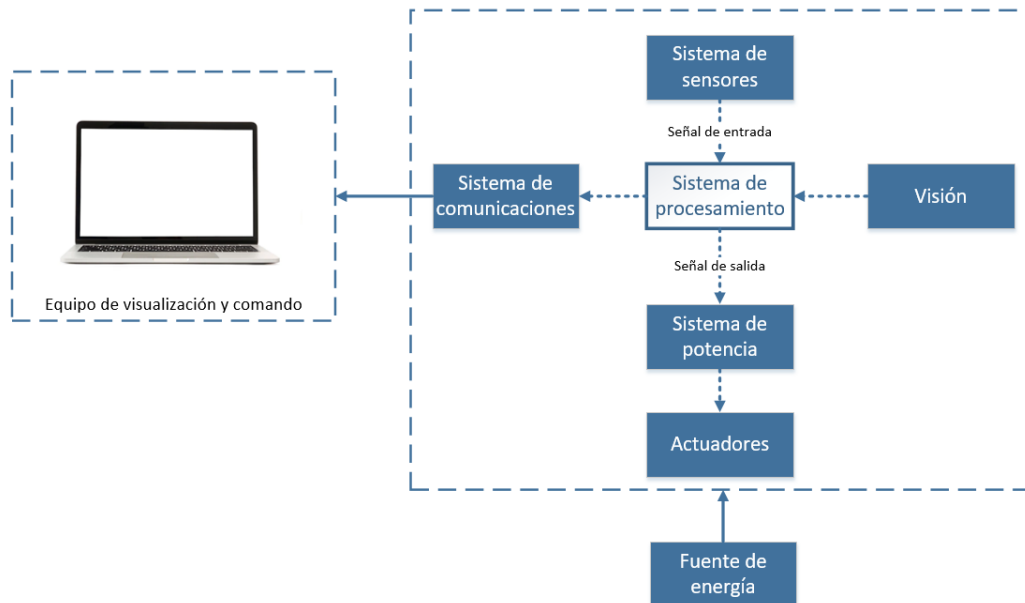


Ilustración 17. Esquema a bloques del diseño funcional.

Fuente: Elaboración Propia.

4.2. SELECCIÓN HARDWARE Y DISEÑO DEL SISTEMA DE CONTROL.

4.2.1. SELECCIÓN DE SENSORES, ACTUADORES Y ELEMENTOS DE CONTROL.

4.2.1.1. Visión.

4.2.1.1.1. Parámetros generales de selección.

A. Volumen requerido. Que sea lo suficientemente pequeña para moverse con libertad dentro del robot.

B. Desarrollo de la tecnología. Debe ser compatible con los equipos de comunicación modernos.

C. Iluminación. Capaz de funcionar con visión nocturna.

D. Flexibilidad de características de visión. Que se pueda configurar sus características como enfoque, ángulo de visión, etc. para acomodarse a la persona.

E. Transmisión de la señal. Que se pueda enviar una señal a través de una red inalámbrica de datos.

4.2.1.1.2. Sistema de visión seleccionado.

Se selecciona una Cámara No IR v2 Raspberry Pi Sony (Ilustración 18).

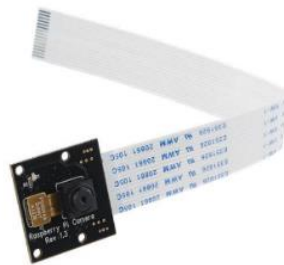


Ilustración 18. Cámara No IR v2 Raspberry Pi Sony.

Fuente: <https://raspberrypi.org.uk/products/raspberry-pi-camera-board-v2-no-ir>

4.2.1.1.3. Sistema de iluminación seleccionado.

Se selecciona una Luz LED infrarroja (Ilustración 19). El sensor óptico utilizado por la cámara es altamente sensible al espectro infrarrojo. Por ello será necesario iluminar el ambiente con una fuente de luz infrarroja para mejorar el reconocimiento de objetos durante la noche.

Las fuentes de luz convencionales también emiten cierto porcentaje de radiación IR, pero en casos donde los niveles de luz sean muy bajos, será necesario apoyarse en los leds. La cámara No IR v2 Raspberry Pi Sony soporta la instalación de estos leds de manera sencilla.



Ilustración 19. Leds IR.

Fuente: <https://www.amazon.es>

4.2.1.2. Sistema de control y procesamiento.

4.2.1.2.1. Parámetros generales de selección.

- A. Velocidad de procesamiento.** Que este dentro de las velocidades de los sistemas embebidos actuales para poder tener evolución de software.
- B. Recursos.** Debe poseer variedad de interfaces para comunicación con otros periféricos.
- C. Método de programación.** Capacidad de implementar programas pesados.
- D. Procesamiento en tiempo real.** Debe tener soporte para software en tiempo real.
- E. Especialización.** Debe ser utilizado en aplicaciones robóticas.
- F. Flexibilidad.** Posibilidad de ser sujeto a futuras mejoras en cuanto a su capacidad de procesamiento e integración de nuevos componentes.
- G. Consumo de energía.** Debe consumir la mínima energía posible de acuerdo a sus características de desempeño.
- H. Ambiente de funcionamiento.** Debe tener soporte para su funcionamiento en ambientes hostiles e industriales.

4.2.1.2.2. Sistema de control y procesamiento seleccionado.

- **Raspberry Pi 3 B.** Ilustración 20.



Ilustración 20. Raspberry Pi 3 modelo B.

Fuente: <https://www.raspberrypi.com/products/raspberry-pi-3-model-b/>

- **Arduino Mega 2560 R3.** Ilustración 21.



Ilustración 21. Arduino Mega 2560 R3.

Fuente: <https://store.arduino.cc/products/arduino-mega-2560-rev3>

4.2.1.3. Sensores

Sensor PIR HC-SR501. Ilustración 22



Ilustración 22. Módulos PIR HC-SR501.

Fuente: <https://www.elektor.com/hc-sr501-pir-motion-sensor-module>

Tabla 5. Pines de conexión / sensor PIR.

| PINES | FUNCIÓN |
|---------|--------------------------------------------------------------------|
| +5v/GND | Estos son los pines de suministro del sensor. |
| Salida | Al detectar un cambio en el espectro infrarrojo produce una señal. |

Fuente: Elaboración Propia.

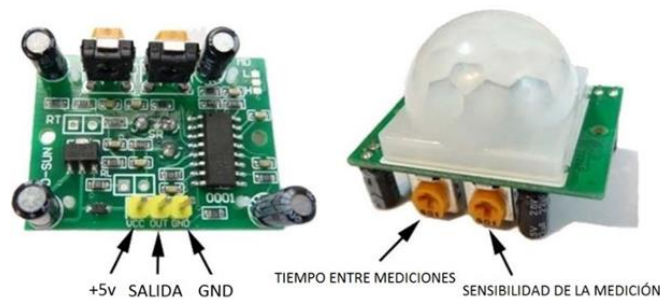


Ilustración 23. Pines de conexión.

Fuente: <https://www.theengineeringprojects.com>

4.2.1.4. Actuadores eléctricos

Se realiza el cálculo para determinar la potencia que consumirá el robot. Además, de considerar una velocidad de locomoción promedio de 1 m/s, su masa será como máximo 2 kg, y un radio de rueda mínimo de 58mm.

A. Inicialmente calculamos las RPM aproximadas del motor.

$$\text{RPM} = \frac{60V}{3.14D}$$

Donde: V: Velocidad del robot (m/s), D: Diámetro de las ruedas (m). Por lo tanto, las RPM son,

$$\text{RPM} = \frac{60(1)}{3.14(0.058)} = 329.45 \text{ RPM} = 330 \text{ RPM}$$

B. Calcularemos el torque del motor.

- Número de ruedas: 4 ruedas.
- Número de motores: 4 motores.
- El coeficiente de fricción se asume: 0.6 y el radio de la rueda: 58 mm
- Peso total del robot = Peso aproximado del robot + carga útil = 1.5 kg + 0.5 kg = 2 kg.
- El peso que actúa sobre las cuatro ruedas se puede escribir como $4 * (N1) = W$, es decir, N1 es el peso que actúa en cada rueda del motor.
- Suponga que el robot está parado. Se requiere el par máximo cuando el robot comienza a moverse. Entonces debe superar la fricción.

$$\mu N r - T = 0$$

Donde: μ : es el coeficiente de fricción, N: es el peso promedio que actúa sobre cada rueda (N), r: es el radio de la rueda (m), T: es el torque (N. m).

- Reemplazando:

$$N1 = W/4 = 2/4 = 0.5 \text{ kg} = 5 \text{ N}$$

Suponiendo que el peso del robot se distribuya equitativamente en las tres ruedas. Por lo tanto, obtenemos:

$$0.6 (5) 0.029 - T = 0$$

$$T = 0.087 \text{ N} - \text{m o } 0.87 \text{ Kg} - \text{cm}$$

C. La potencia del motor en HP.

- Motor RPM: 330 RPM.
- Motor torque: 0.87 Kg.cm.
- Diámetro de la rueda: 58mm.

D. Elección del motor de DC. Mini Gear DC motor /Core 6v - 750 rpm (Ilustración 24).

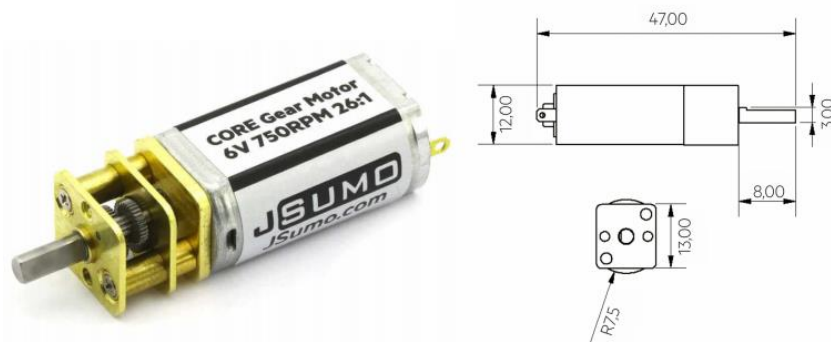


Ilustración 24. Mini Gear DC motor /Core 6v - 750 rpm.

Fuente: <https://www.jsumo.com/core-dc-motor-6v-750rpm>

4.2.1.5. Sistema de potencia.

Un controlador de motor o controlador de motor es un circuito que puede controlar la velocidad del motor. Controlar motores significa que podemos controlar el voltaje a través del motor y también podemos controlar la dirección y velocidad del motor. Los motores pueden girar en sentido horario o antihorario, si cambiamos la polaridad del terminal del motor.

Los circuitos de puente H (Ilustración 25) se usan comúnmente en controladores de motor. El puente H es un circuito electrónico que puede aplicar voltaje en cualquier dirección. Tiene altas propiedades de manejo de corriente y puede cambiar la dirección del flujo de corriente.

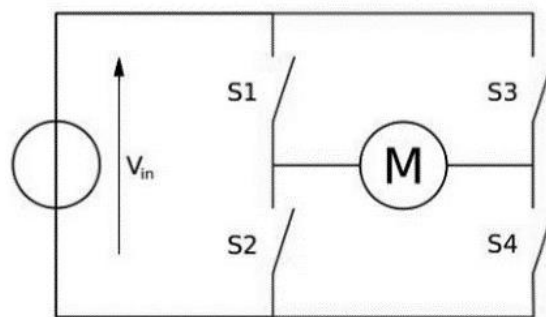


Ilustración 25. Circuito puente H.

Fuente: https://es.wikipedia.org/wiki/Puente_H_%28electr%C3%B3nica%29

La dirección del motor, dependiendo de los cuatro interruptores, se da de la siguiente manera:

Tabla 6. Dirección del motor DC dependiendo de la excitación.

| S1 | S2 | S3 | S4 | RESULTADOS |
|----|----|----|----|--------------------------------|
| 1 | 0 | 0 | 1 | Motor se mueve a la derecha. |
| 0 | 1 | 1 | 0 | Motor se mueve a la izquierda. |
| 0 | 0 | 0 | 0 | Motor gira libre. |
| 0 | 1 | 0 | 1 | Freno del motor. |
| 1 | 0 | 1 | 0 | Freno del motor. |
| 1 | 1 | 0 | 0 | Motor se dispara. |
| 0 | 0 | 1 | 1 | Motor se dispara. |
| 1 | 1 | 1 | 1 | Motor se dispara. |

Fuente: Elaboración Propia.

4.2.1.6. Drivers.

Se selecciona un Driver Puente H L298N (Ilustración 26).

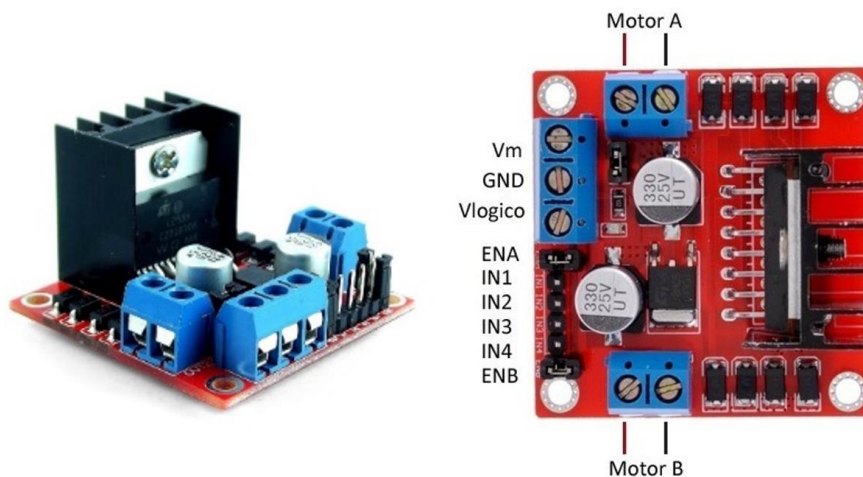


Ilustración 26. Driver Puente H L298N.

Fuente: <https://avelectronics.cc/producto/modulo-l298n/>

Tabla 7. Pines de conexión controlador puente H.

| PINES. | FUNCIÓN. |
|--------|--------------------------------------------------------------------------|
| Vm | Voltaje de alimentación. |
| GND | Tierra. |
| ENA | Control de velocidad del Motor A. Debe estar conectado a un pin PWM. |
| IN1 | Control de dirección del Motor A. Debe estar conectado a un pin digital. |
| IN2 | Control de dirección del Motor A. Debe estar conectado a un pin digital. |
| IN3 | Control de dirección del Motor B. Debe estar conectado a un pin digital. |
| IN4 | Control de dirección del Motor B. Debe estar conectado a un pin digital. |
| ENB | Control de la velocidad del Motor B. Debe estar conectado a un pin PWM. |

Fuente: Elaboración Propia.

4.2.1.7. Batería.

Tabla 8. Consumo de las cargas presentes en el sistema.

| COMPONENTE ELECTRÓNICO. | NÚMERO DE ELEMENTOS. | CORRIENTE TOTAL (mA). |
|-------------------------|----------------------|-----------------------|
| Raspberry Pi 3B. | 01 | 2500. |
| Arduino Mega 2560. | 01 | 90. |
| Motores DC. | 04 | 5200. |
| Sensores PIR. | 05 | Despreciable. |

Fuente: Elaboración Propia.

La corriente total consumida por el sistema es 7790 mA, y el sistema debe tener una autonomía de 1 horas. La capacidad de la batería debe ser:

$$Q = I (H)$$

Donde: Q: Capacidad eléctrica (mAh), I: Corriente eléctrica (mA), H: Tiempo de autonomía (h). Entonces:

$$Q = 7792(1) = 7790 \text{ mAh.}$$

4.2.1.8. Comunicaciones.

- **Conexión Wifi.**

Adaptador inalámbrico USB Tp-Link (Ilustración 27)



Ilustración 27. Adaptador Inalámbrico USB 150 m Tp – Link.

Fuente: <https://www.tp-link.com/pe/home-networking/adapter/tl-wn722n/>

- **Cable plano flexible CSI para cámara Raspberry Pi.** Ilustración 28.



Ilustración 28. Cable plano flexible CSI para cámara Raspberry Pi.

Fuente: <https://www.adafruit.com/product/2144>

- **Ruedas Mecanum.** Ilustración 29.



Ilustración 29. Rueda de aluminio Mecanum.

Fuente: <https://jp.robotshop.com>

4.2.2. DISEÑO DEL HARDWARE.

El sistema de control consiste en un miniordenador Raspberry pi que se puede comunicar con un ordenador personal de cualquier característica, desde ahí el operador del robot puede enviar comandos y recibir datos de estado, todo a través de una comunicación TCP/IP. La PC y la Raspberry Pi 3B formarán una red WLAN, para ello es necesario un router o enrutador Wifi, que será el dispositivo encargado de dar la conectividad inalámbrica a los dispositivos. Esta funcionalidad permite acceder a la operación de la Raspberry Pi 3B de manera remota, pudiendo acceder a su sistema operativo “Raspbian” desde nuestro ordenador (Ilustración 30).

La Raspberry Pi 3B se comunica con la placa Arduino MEGA mediante el protocolo USB. Por otro lado, la Raspberry Pi 3B tiene conectados los 4 sensores PIR en sus pines GPIOs, por donde ingresan las señales que se generan cuando se detecta una presencia en el ambiente.

Las señales generadas por los comandos ejecutados a través de la interfaz de usuario son enviadas por medio de Raspberry hasta Arduino, para que posteriormente generen las señales que utilizan los drivers para controlar los motores de DC. Los pines digitales de Arduino Mega son utilizados para conectar los Drivers de los motores de DC. La cámara de visión nocturna está conectada mediante el protocolo CSI (Interfaz Serie para Cámaras) a la placa Raspberry Pi 3 B.

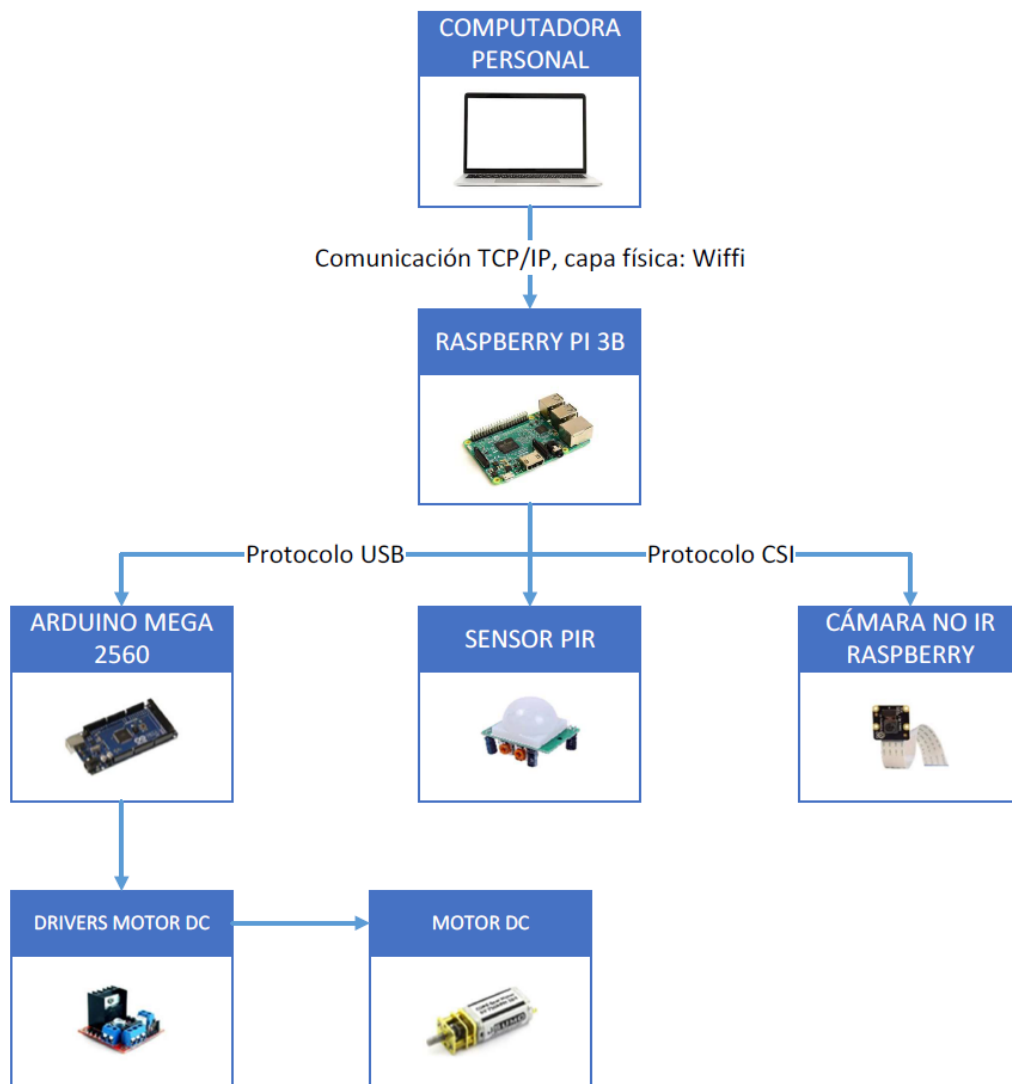


Ilustración 30. Arquitectura del sistema de control.

Fuente. Elaboración Propia.

4.2.3. DISEÑO DEL SISTEMA DE CONTROL.

4.2.3.1. Elección de sistema operativo

La elección del sistema operativo para el miniordenador Raspberry Pi 3B debe priorizar el uso de herramientas de software libre y ser respaldada por una amplia comunidad de desarrolladores.

4.2.3.1.1. Sistemas operativos disponibles en el mercado.

A. Debian.

Debian GNU/Linux es un sistema operativo libre, desarrollado por miles de voluntarios de todo el mundo, que colaboran a través de Internet.

Características:

- Debian es un sistema operativo de libre distribución (es decir sin coste alguno).
- Permite a varios usuarios acceder al mismo tiempo a través de terminales, y distribuye los recursos disponibles entre todos.
- Puede correr en la mayoría de las plataformas del mercado (procesadores de la gama Intel y AMD, Motorola, etc.).
- Debian nace como una apuesta por separar en sus versiones el software libre del software no libre, para esto debe respetar 4 libertades:
- Libertad para usarlo.
- Libertad para modificarlo.
- Libertad para copiarlo.
- Libertad para distribuir las modificaciones.
- Debian es una distribución que ha probado su estabilidad y utilidad, muchos desarrolladores la han tomado para crear otras nuevas como: Knoppix, Ubuntu, Sidux, etc.
- Debian puede ser usado por cualquier persona que sepa qué es Linux, que haya tenido un Linux instalado, que sepa configurar las cosas básicas, que no se rinda ante las primeras dificultades.

B. Raspbian.

Raspbian es el sistema operativo recomendado para Raspberry Pi (al estar optimizado para su hardware) y se basa en una distribución de GNU/Linux llamada Debian.

Características:

- Raspbian está altamente optimizado para la Raspberry Pi.
- Raspbian proporciona más que un sistema operativo puro: viene con más de 35,000 paquetes, software pre compilado incluido en un formato agradable para una fácil instalación en su Raspberry Pi.
- En esta plataforma se puede poner a disposición de todos los usuarios de Raspbian, mediante moderación y posterior lanzamiento, contenidos gratuitos o de pago, como archivos binarios, código Python, imágenes, audio o vídeo.

C. Ubuntu Mate.

Ubuntu MATE es una distribución Linux basada en Ubuntu. Está mantenida por la comunidad y usa el entorno de escritorio MATE.

Características:

- La versión 16.04 LTS tiene imágenes creadas específicamente para dispositivos Raspberry Pi 2 y Raspberry Pi 3.
- Es un sistema operativo orientado principalmente a los ordenadores personales, aunque también cuenta con un soporte especial para los servidores.
- Tiene una fácil instalación.

D. ARCH LINUX.

Es una distribución Linux para computadoras x86 y x64 orientada a usuarios avanzados. Se compone en su mayor parte de software libre y de código abierto (FOSS) y apoya la participación comunitaria.

Características:

- Para instalar y configurar el sistema se necesita un grado de conocimiento superior al básico. Además, Posee aplicaciones que permiten mantener el sistema de forma sencilla.
- A pesar de las múltiples alternativas disponibles, el modelo Raspberry Pi tiene limitaciones para soportar Ubuntu Mate. Arch Linux es un sistema operativo de uso general, y no está equipado con las herramientas para trabajar con mayor facilidad con Raspberry Pi. Así que se selecciona el sistema operativo Raspbian, debido a su fácil uso y por ser un sistema operativo recomendado y diseñado especialmente para Raspberry Pi. Raspbian provee las herramientas para la gestión del sistema de manera remota (VNC) y protocolos de compartición de archivos con otros sistemas operativos. La instalación del sistema operativo se realizará en una tarjeta SD externa, que debe tener una memoria mayor a 6 Gb por recomendación.

4.2.3.2. Arquitectura del software de control.

4.2.3.2.1. Comunicación con el usuario.

El operador del robot debe ser capaz de acceder de manera remota desde cualquier computadora personal (Ilustración 31). Para ello se utilizará VNC (Virtual Network Computing), que es un protocolo de administración remota que

le permite a los usuarios controlar y modificar sus servidores remotos a través de internet con la ayuda de un mecanismo de autenticación. La condición para utilizarlos es que las computadoras deben estar conectadas a internet. En nuestro la Raspberry Pi y el ordenador personal estarán conectados a la misma red a través de un enrutador Wifi.

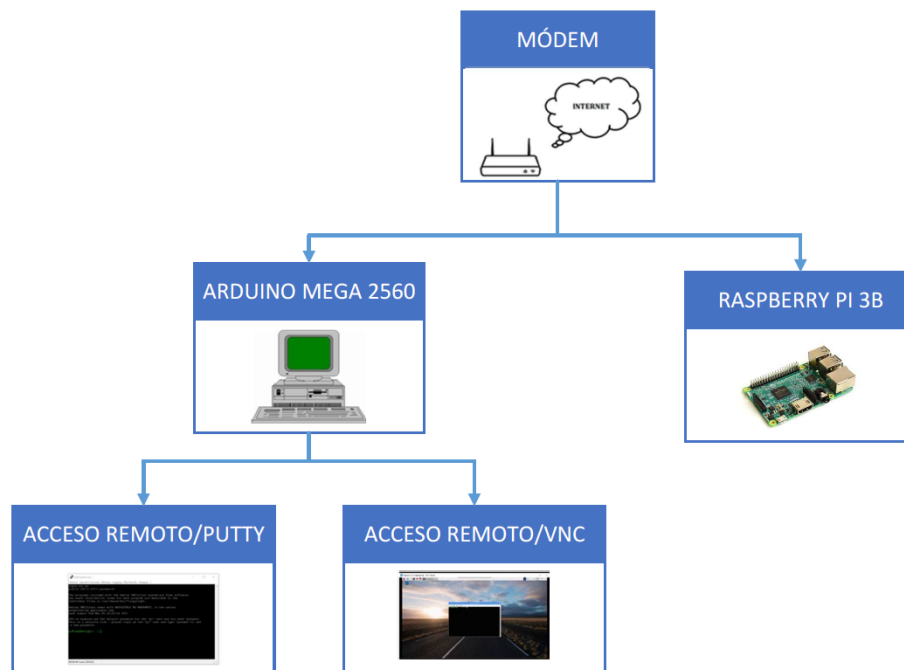


Ilustración 31. Acceso remoto a Raspberry Pi mediante VNC y PuTTY.

Fuente: Elaboración Propia.

A. Instalación de PuTTY en Windows.

Putty es un cliente SSH (protocolo de administración remota) con licencia libre y está disponible para múltiples sistemas operativos. Nos permite acceder de forma remota a cualquier computadora personal, pero en comparación con VNC (accede a la interfaz completa de Raspberry Pi), se opera a través del terminal de Windows. Lo primero que debemos hacer es instalar la aplicación PuTTY en nuestro ordenador, y posteriormente, ejecutarlo. Luego colocamos la dirección IP de nuestra Raspberry Pi donde se indica, y finalmente podemos

acceder al terminal de comandos de la Raspberry Pi para así poder operarla fácilmente desde nuestra computadora personal. Desde esta terminal instalaremos el software VNC, colocando algunas líneas de código. El software VNC debe estar instalado en ambos dispositivos por lo que también se realiza la operación de instalación en nuestra computadora personal. Después de finalizada la instalación se ejecuta VNC y colocando la dirección IP se puede acceder al entorno de Raspberry Pi de forma remota.

B. Comunicación con la Cámara No IR Raspberry Pi.

La cámara está conectada mediante una interfaz CSI al ordenador Raspberry Pi, que es una interfaz de comunicación serial para la familia de cámaras de Raspberry Pi. Raspberry Pi tiene un procesador multimedia llamado Video Core que otorga procesamiento de video a alta velocidad, pero a baja potencia para no gastar energía eléctrica.

C. FLASK.

Para acceder a la transmisión de video en tiempo real, primero necesitamos acceder a nuestra Raspberry Pi por medio del software PuTTY (acceso remoto por SSH). Después de acceder al terminal de comandos de Raspberry Pi, procederemos a instalar “FLASK”, que es un micro framework escrito en Python que permite crear aplicaciones web rápidamente y con un mínimo número de líneas de código, ello nos facilitará realizar el video streaming. Las transmisiones de FLASK, otorga la facilidad de dar respuestas grandes en pequeños fragmentos durante un largo periodo de tiempo. La transmisión es una técnica en la que el servidor proporciona una respuesta a una solicitud en fragmentos, y es muy útil al momento de transmitir datos de video o audio en tiempo real.

D. Video Streaming con FLASK.

FLASK genera una transmisión de respuestas usando funciones generadoras (Ilustración 32). Un generador es una función especial que puede ser interrumpida o reanudada. La función generadora nos permite retornar múltiples resultados en una secuencia. Estos datos posteriormente pueden ser enviados a un servidor web donde transmitidos y almacenados. Podemos probar los siguientes comandos en el IDLE de Python para verificar la operación del comando `gen ()`.

```
def gen():
    yield 1
    yield 2
    yield 3

>>> x = gen()
>>> x
<generator object gen at 0x7f06f3059c30>
>>> next(x)
1
>>> next(x)
2
>>> next(x)
3
>>> next(x)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
StopIteration
```

Ilustración 32. Utilización de la función generadora.

Fuente: Elaboración Propia.

Una funcionalidad interesante es reemplazar cada fragmento con el próximo en el servidor web. Ello permite que las transmisiones se reproduzcan en la ventana del navegador. Así cada fragmento de la transmisión puede ser una imagen, generando un video streaming en el servidor web.

Esta funcionalidad se implementa mediante las respuestas “MULTIPART”, donde el encabezado contiene los tipos de contenido multiparte (Content – Type), que en este caso es “multipart/x-mixed-replace” y es específica para

transmitir y reemplazar los datos constantemente en tiempo real (Ilustración 33), seguido de las partes separadas por un marcador de limite (--frame).

```
HTTP/1.1 200 OK
Content-Type: multipart/x-mixed-replace; boundary=frame

--frame
Content-Type: image/jpeg

<jpeg data here>
--frame
Content-Type: image/jpeg

<jpeg data here>
...
```

Ilustración 33. Respuestas multiparte para actualizar datos en tiempo real.

Fuente: Elaboración Propia.

Podemos generar una tabla con un gran número de datos, sin tener que ensamblar la tabla entera en la memoria. FLASK se encarga de invocar el generador y enviar todos los resultados parciales como fragmentos al cliente (Ilustración 34).

```
from flask import Response, render_template
from app.models import Stock

def generate_stock_table():
    yield render_template('stock_header.html')
    for stock in Stock.query.all():
        yield render_template('stock_row.html', stock=stock)
    yield render_template('stock_footer.html')

@app.route('/stock-table')
def stock_table():
    return Response(generate_stock_table())
```

Ilustración 34. Generación y transmisión por partes de datos de gran volumen.

Fuente: Elaboración propia.

4.2.3.2.2. Interfaz de control de locomoción con ruedas Mecanum.

Como podemos ver en la Ilustración 35 cada controlador se encargará del control de 2 motores de DC. De las 6 salidas del controlador conectadas a la placa Arduino Mega, 3 de ellos corresponden a un motor y las restantes al otro. De estos 3 pines dos de ellos son para el control de giro (representados por IN) y el restante para el control de la velocidad (representados por EN). En nuestro caso solo usaremos los dos pines para el control de la dirección el tercero no será necesario, ya que no deseamos realizar un control de la velocidad. Los motores de DC necesitan un voltaje de 6V para su correcto funcionamiento. La batería se encargará de alimentar a los drivers y la placa Arduino Mega.

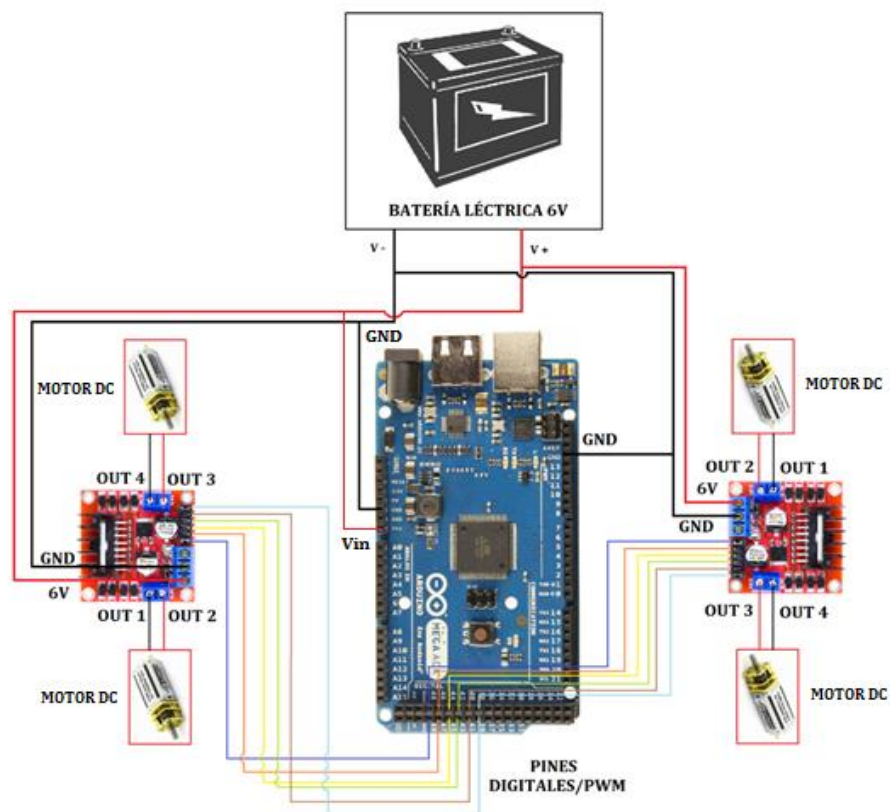


Ilustración 35. Interfaz física de los motores de DC.

Fuente: Elaboración Propia.

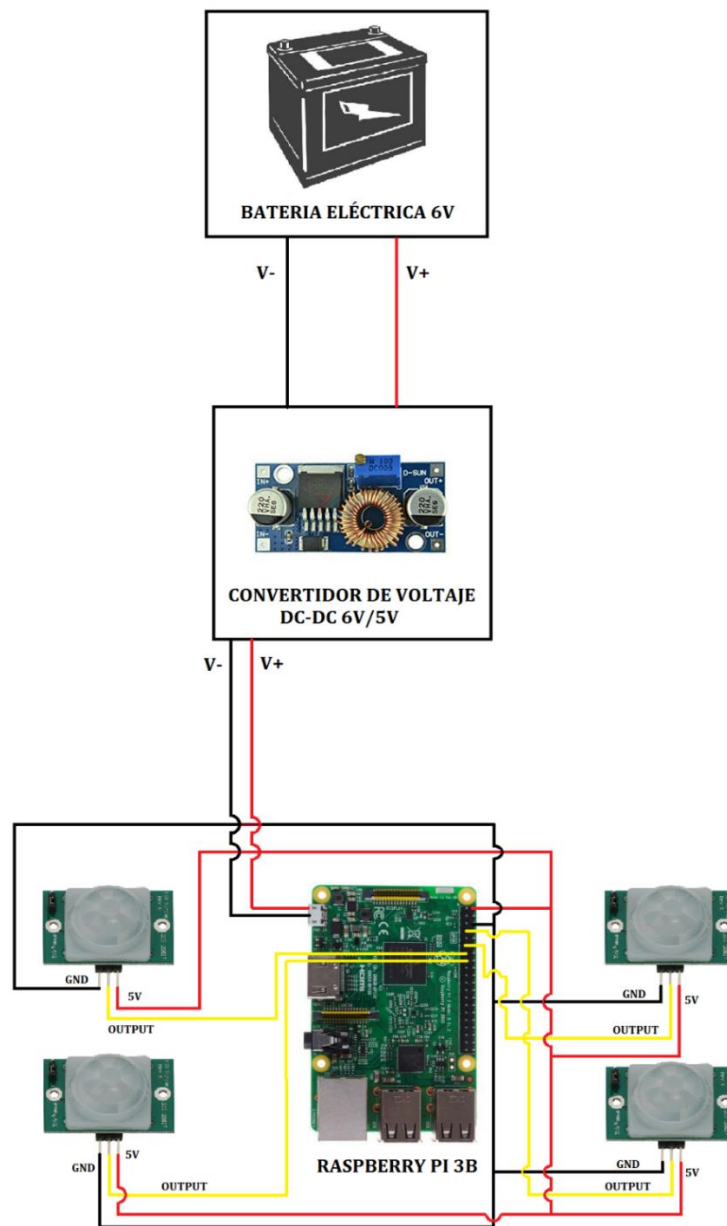


Ilustración 36. Interfaz física de los sensores PIR.

Fuente: Elaboración Propia.

Se utilizan dos programas 2 programas que están interconectados. El código utilizado se desarrolló modificando un código disponible en GITHUB para una aplicación similar. El primer programa fue desarrollado con “Python”, esta pequeña aplicación permite la creación de la GUI (interfaz de usuario), que será una pequeña ventana con algunas teclas para ejecutar los comandos de movimiento del robot móvil. Al ejecutar una tecla de la GUI mencionada, se enviará un carácter (en nuestra GUI es un número) vía comunicación serial hacia Arduino, en donde ejecutará el movimiento especificado, eso se cumple para todas las teclas de la GUI.

Como se menciona el segundo programa está desarrollado en Arduino IDLE, y contiene las operaciones para controlar los motores de DC para cada movimiento que necesitamos generar en el robot móvil. Este programa recepción los dígitos enviados desde la GUI y ejecuta las operaciones asociadas.

4.2.3.2.3. Sistema de detección de movimiento.

El sistema de detección de movimiento consta de 4 sensores PIR (sensor infrarrojo pasivo) dispuestos en cada una de las 4 caras del robot móvil. Cada sensor está conectado a los pines GPIO del miniordenador Raspberry Pi y a la alimentación (Ilustración 36).

4.3. DISEÑO MECÁNICO DEL ROBOT

La estructura mecánica tendrá el objetivo de soportar al sistema embebido, que consta de las placas electrónicas, sensores, actuadores, baterías, comunicaciones, y a la cámara de visión nocturna (Ilustración 37). El modelo 3D de la estructura se ejecutó en una plataforma de diseño asistido por computadora (CAD) llamada SolidWorks. El robot debe tener un peso de 2kg, y

el material seleccionado para cumplir esas prestaciones es el aluminio, debido a sus características de peso y resistencia.

Vista isométrica.

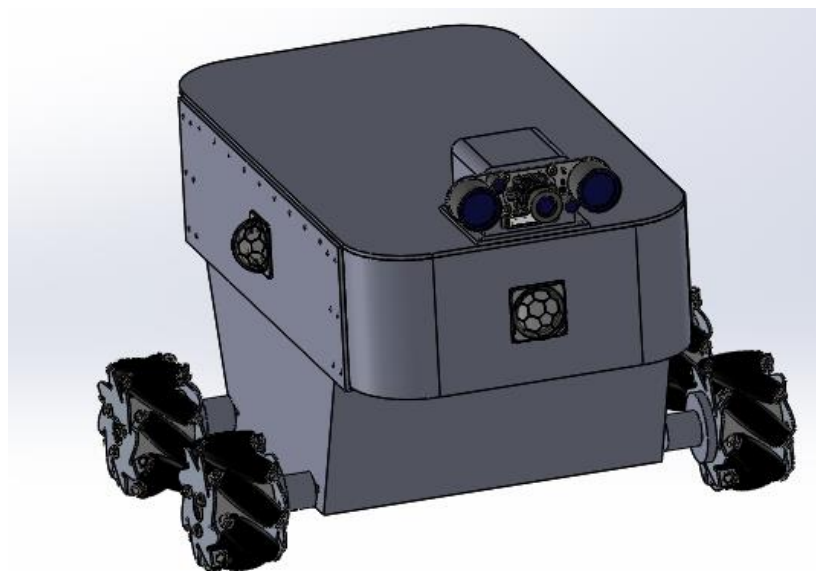


Ilustración 37. Vista isométrica del robot móvil de vigilancia.

Fuente: Elaboración Propia.

CAPÍTULO V: CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones.

5.1.1. Especificaciones técnicas y geométricas del sistema mecánico.

5.1.1.1. Chasis.

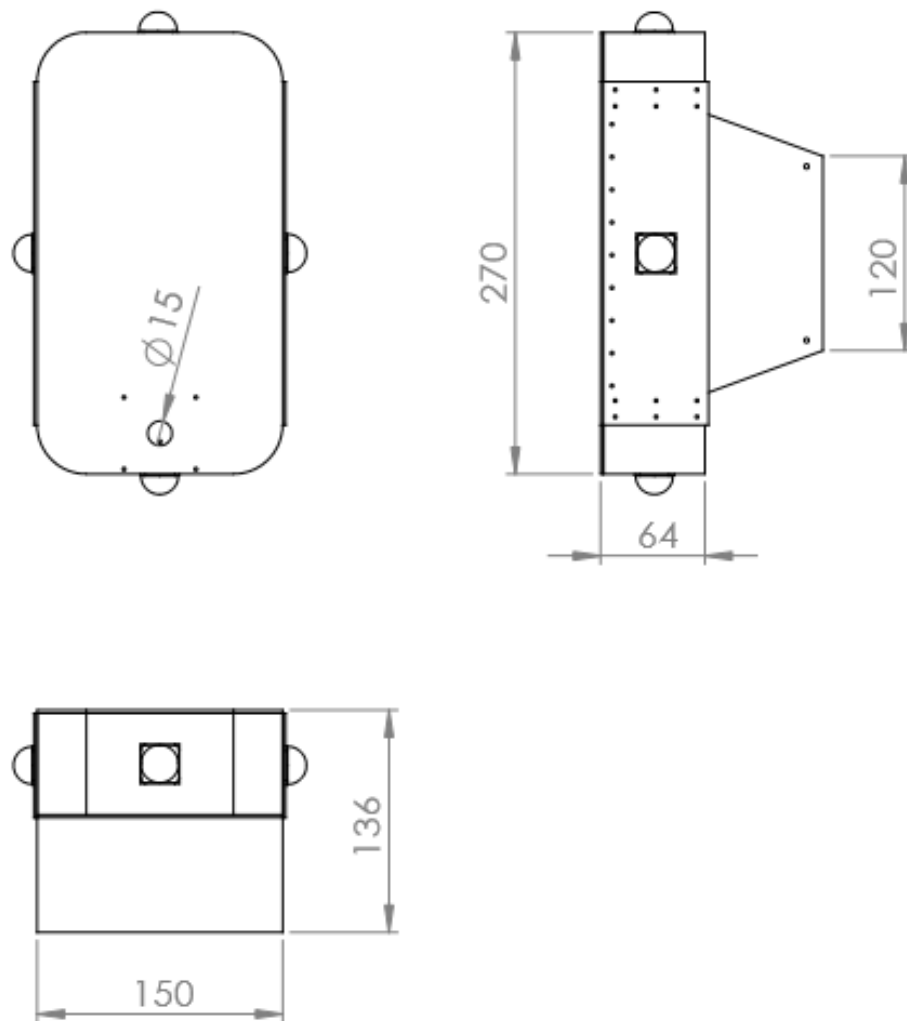


Ilustración 38. Vistas estándar del chasis.

Fuente Elaboración Propia.

Tabla 9. Características técnicas y geométricas del chasis.

| CARACTERÍSTICA | VALOR | UNIDAD |
|----------------|-----------------------------|--------|
| Altura. | 136. | mm |
| Longitud. | 270. | mm |
| Ancho. | 150. | mm |
| Material. | Plancha de aluminio 2mm. | |

Fuente: Elaboración Propia.

5.1.1.1.1. Tapa lateral.

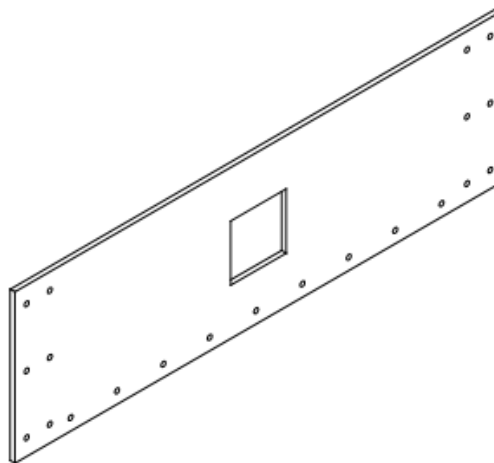


Ilustración 39. Vista 3D de la tapa lateral del chasis.

Fuente: Elaboración Propia.

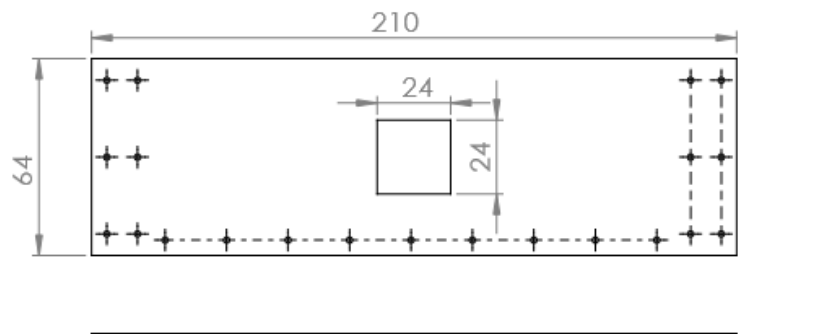


Ilustración 40. Vistas estándar de la tapa lateral en milímetros.

Fuente: Elaboración Propia.

5.1.1.1.2. Tapa frontal y trasera.

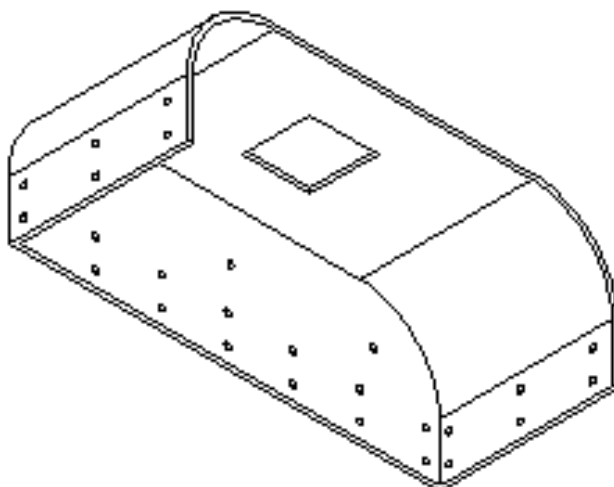


Ilustración 41. Vista 3D de la tapa frontal y trasera del chasis.

Fuente: Elaboración Propia.

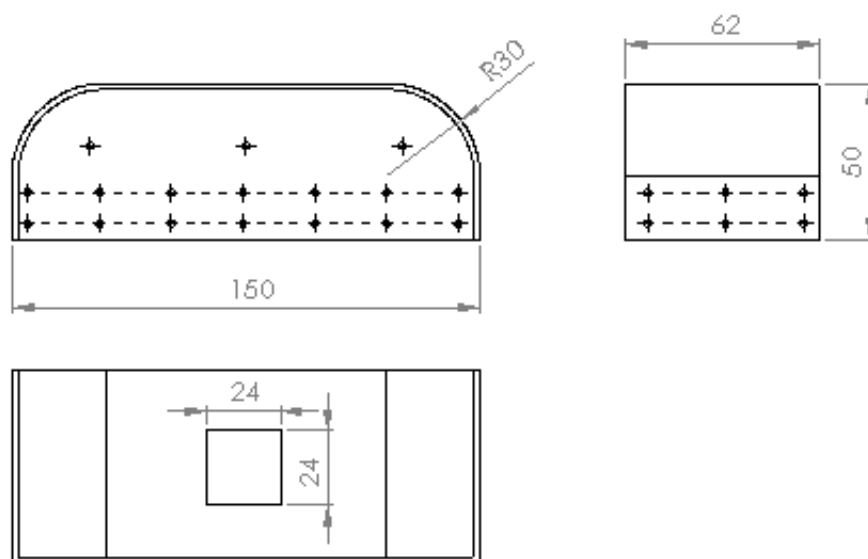


Ilustración 42. Vistas estándar de la tapa frontal y trasera en milímetros.

Fuente: Elaboración Propia.

5.1.1.1.3. Tapa superior.

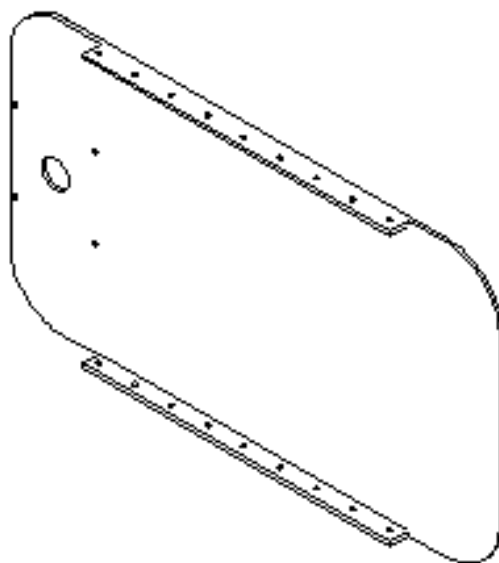


Ilustración 43. Vista 3D de la tapa superior del chasis.

Fuente: Elaboración Propia.

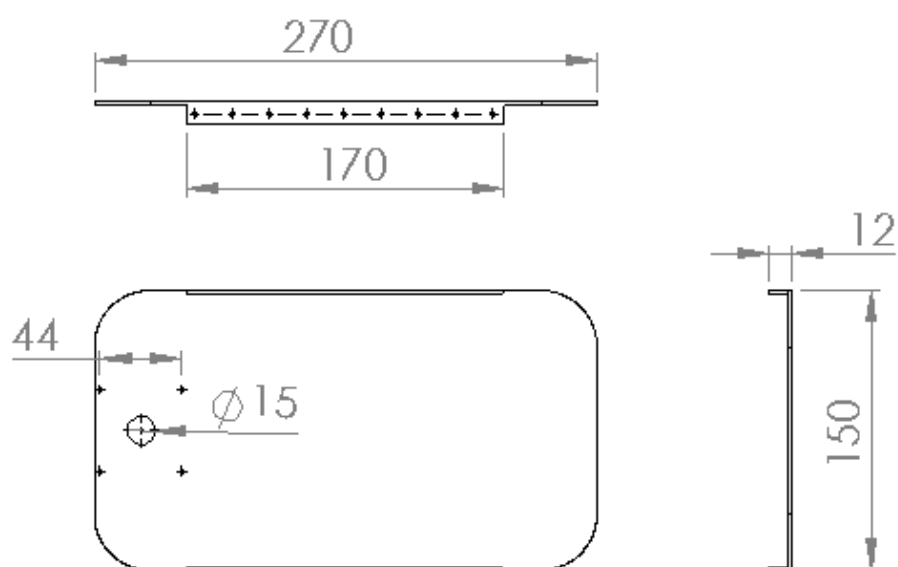


Ilustración 44. Vistas estándar de la tapa superior en milímetros.

Fuente: Elaboración Propia.

5.1.1.1.4. Base del chasis.

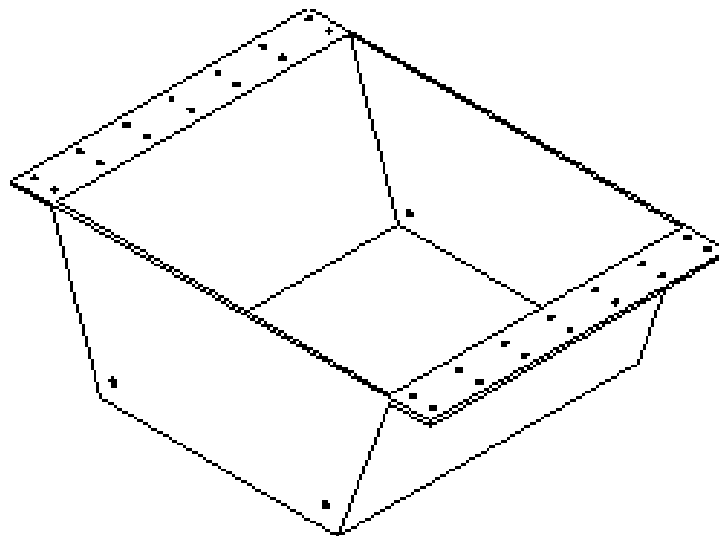


Ilustración 45. Vista 3D de la base del chasis.

Fuente: Elaboración Propia.

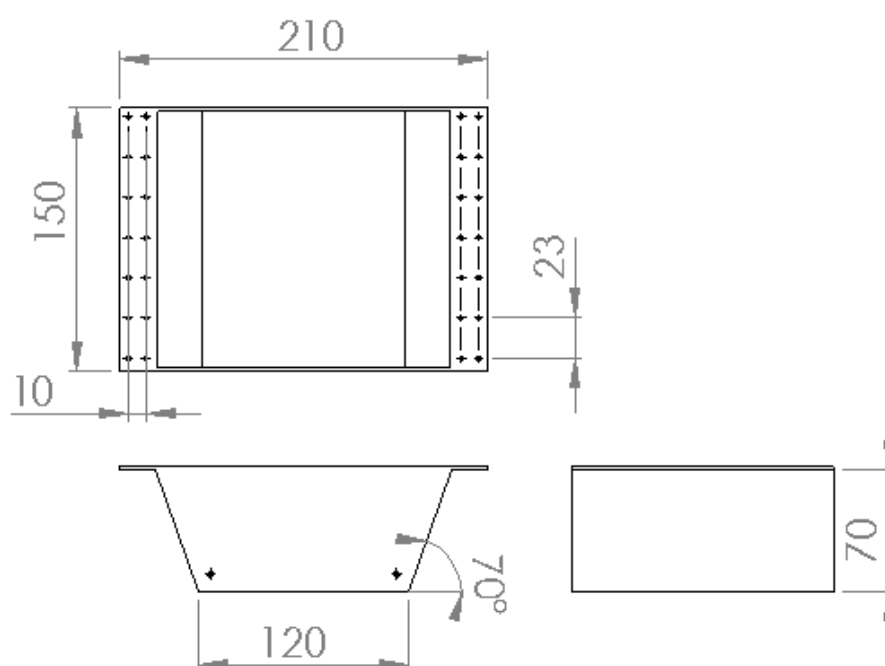


Ilustración 46. Vistas estándar de la base del chasis en milímetros.

Fuente: Elaboración Propia.

5.1.1.2. Soporte para el miniordenador Raspberry 3B.

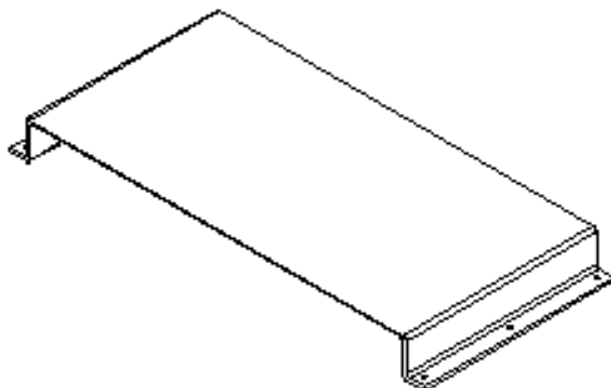


Ilustración 47. Vista 3D del soporte para el miniordenador Raspberry pi 3B.

Fuente: Elaboración Propia.

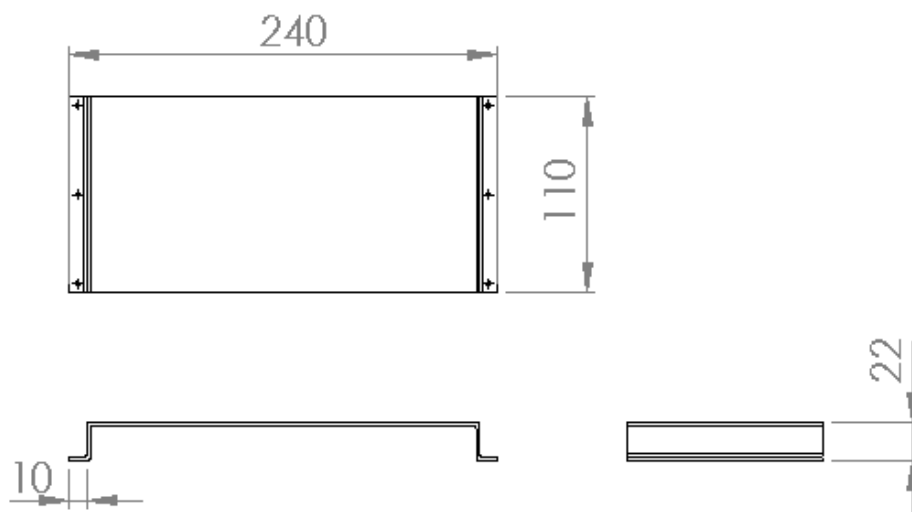


Ilustración 48. Vistas estándar del soporte para Raspberry pi 3B en milímetros.

Fuente: Elaboración Propia.

5.1.1.3. Soporte cámara.

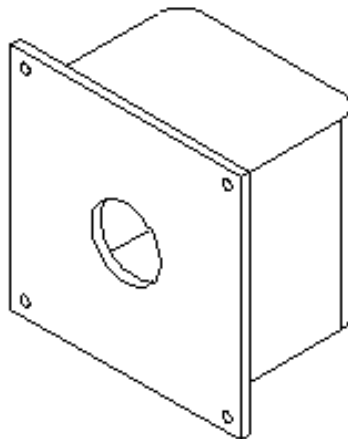


Ilustración 49. Vista 3D del soporte de la cámara de visión nocturna.

Fuente: Elaboración Propia.

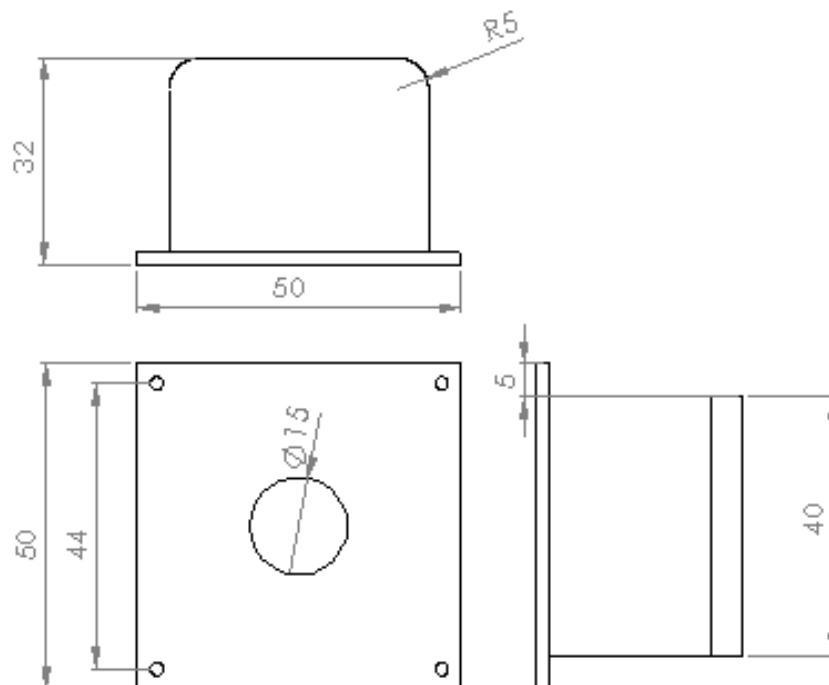


Ilustración 50. Vistas estándar del soporte de la cámara en milímetros.

Fuente: Elaboración Propia.

Tabla 10. Características técnicas y geométricas del soporte de la cámara.

| Característica | Valor | Unidad |
|----------------|-----------------------------|--------|
| Altura. | 32 | mm |
| Ancho. | 50 | mm |
| Longitud. | 50 | mm |
| Material. | Plancha de aluminio de 2mm. | mm |

Fuente: Elaboración Propia.

5.1.1.4. Ruedas Mecanum.

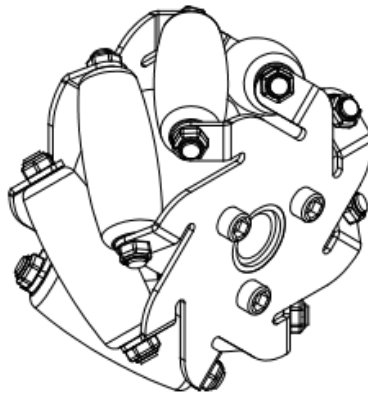


Ilustración 51. Vista 3D de la rueda Mecanum.

Fuente: Elaboración Propia.

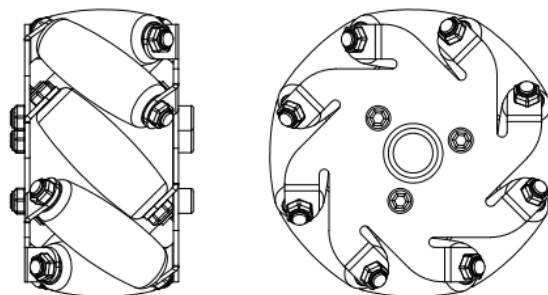


Ilustración 52. Vistas estándar de la rueda Mecanum.

Fuente: Elaboración Propia.

Tabla 11. Características técnicas y geométricas de las ruedas Mecanum.

| CARACTERÍSTICA | VALOR | UNIDAD |
|--------------------|-----------------------|--------|
| Diámetro exterior | 58 | mm |
| Ancho | 30 | mm |
| Capacidad de carga | 10 | Kg. |
| Peso | 86 | g. |
| Material | Aleación de aluminio. | - |

Fuente: Elaboración Propia.

5.1.1.5. Acople para motor y rueda.

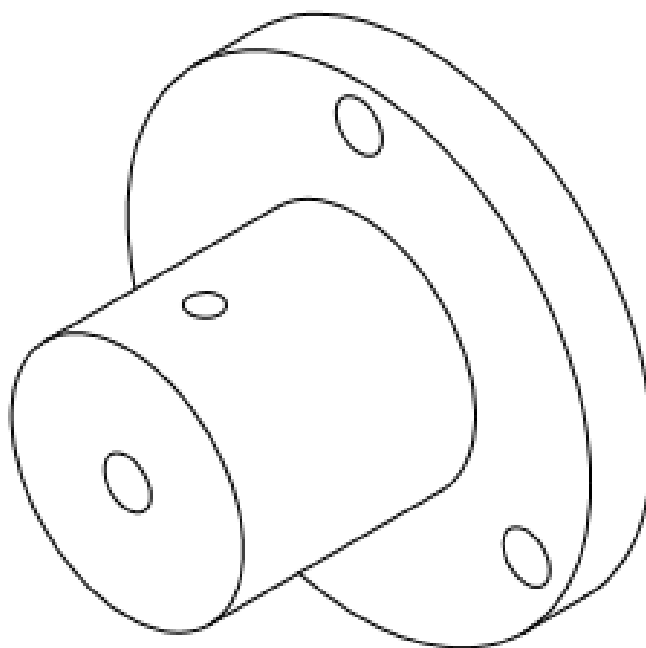


Ilustración 53. Vista 3D del acople.

Fuente: Elaboración Propia.

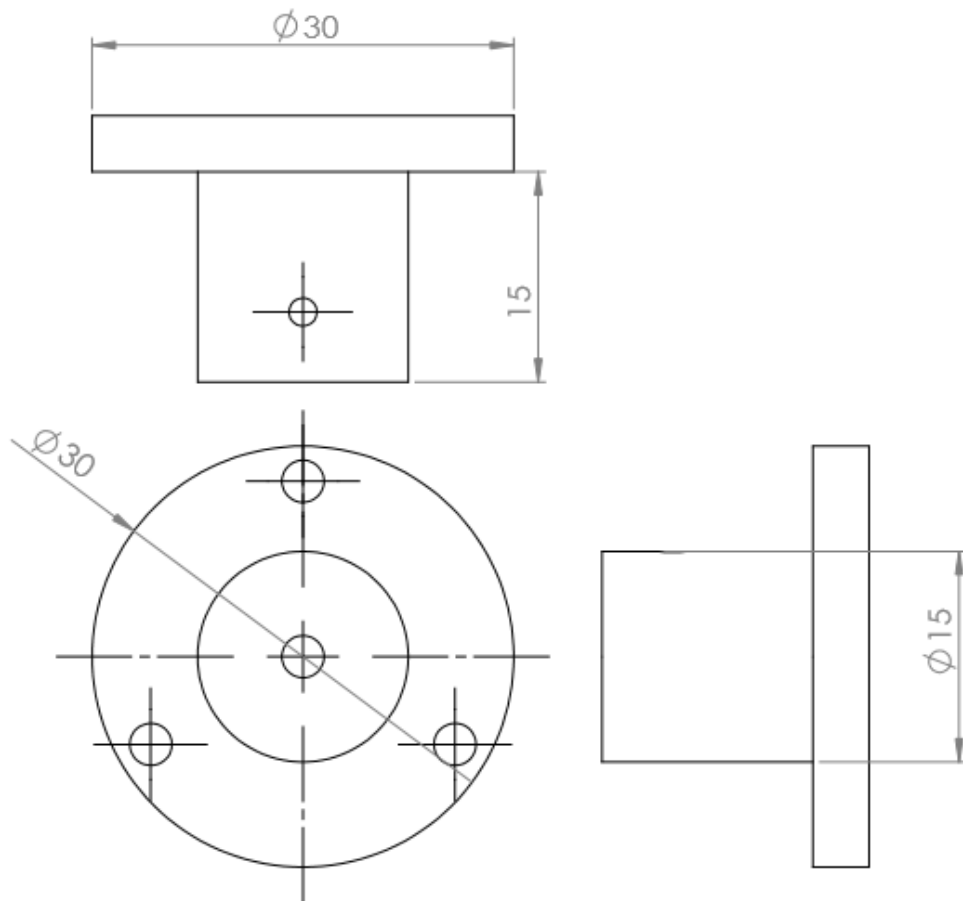


Ilustración 54. Vistas estándar del acople en milímetros.

Fuente: Elaboración Propia.

5.1.1.6. Pernos de sujeción.

Tornillo de acero 1/8 in.

Tabla 12. Especificaciones técnicas y geométricas de los pernos de sujeción.

| CARACTERÍSTICAS | VALOR | UNIDAD |
|-----------------|-------|--------|
| Diámetro cabeza | 6 | mm |
| Diámetro tuerca | 3.175 | mm |
| Longitud | 9.5 | mm |
| Cantidad | 160 | |

Fuente: Elaboración Propia.

5.1.2. Especificaciones técnicas y geométricas del sistema electrónico

5.1.2.1. Cámara No IR v2 Raspberry Pi Sony.

- Sensor. Sony IMX219 8Mpx.
- Resolución de fotos. 3238 x 2464.
- Resolución de videos. 1080p30, 720p60 y 640x480p90.
- Interfase. CSI.
- Dimensiones. 25mm x 23mm x 9mm.

5.1.2.2. Luz LED infrarroja.

- Potencia. 3 W.
- Tamaño. 2.8 x 2 x 1.4cm.

5.1.2.3. Raspberry Pi 3 B.

- CPU. 1.2GHz Quadcore ARM Cortex-A53.
- Juego de instrucciones. 64 bits.
- Unidad de procesamiento gráfico (GPU). Broadcom Video Core IV (250 MHz).
- Memoria RAM. 1GB (compartidos con la GPU).
- Alimentación. 5V/2.5 A (3.5 W) vía micro USB.
- Conexiones. 4 x USB 2.0. 1 x Salida audio mini Jack 3.5mm. 1 x Salida audio/vídeo HDMI. 1 x Micro USB (Alimentación). 1 x RJ45 10/100 Ethernet.
- Interfase de cámara. CSI.
- Interfase para display. DSI.
- Dimensiones. 85.6 mm x 56.5mm.

5.1.2.4. Microcontrolador Arduino Mega 2560 R3.

- Microcontrolador. ATmega2560.
- Chip USB. ATmega16U2.
- Voltaje de Operación. 5V.
- Voltaje de alimentación. 6-20V (7-12V recomendado).
- Pines digitales I/O. 54 (15 salidas PWM).
- Entradas analógicas. 16.
- Corriente máxima entrada/salida. 40mA.
- Memoria FLASH. 256K.
- Memoria SRAM. 8K.
- Memoria EEPROM. 4K.
- Velocidad de reloj. 16MHz.

5.1.2.5. Motor de DC Mini Gear /Core 6v - 750 rpm.

- Voltaje: 6 V.
- N_I (vacío): 750 RPM.
- I_I (vacío): 120mA.
- I_s (Stall): 3.5 A.
- T_s (Stall): 2.8 Kg.cm
- Dimensiones: 45 x 15 x 10 mm

5.1.2.6. Driver Puente H L298N.

- Chip: L298N.
- Canales: 2 (soporta 2 motores DC o 1 motor PAP).
- Voltaje lógico: 5 V.

- Voltaje de potencia: 5 V – 35 VDC.
- Consumo de corriente lógico: 0 a 36mA.
- Capacidad de corriente: 2A (picos de hasta 3A).
- Potencia máxima: 25W
- Peso: 30 gramos.
- Dimensiones: 43 x 43 x 27mm.

5.1.2.7. Batería SAKO

- Voltaje: 6V.
- Capacidad eléctrica: 4Ah.
- Medidas: 9.8 x 6.8 x 4.5 cm.

5.1.2.8. Adaptador inalámbrico USB Tp-Link.

- Velocidad: Hasta 150 Mbps.
- Ganancia de la antena: 2 dBi.
- Banda de frecuencia: 2.4 GHz.
- Compatibilidad con sistemas operativos: Windows. 2000, XP, Vista, Win7, Linux, Mac, Raspberry Pi.
- Peso: 0.070 kg.

5.1.2.9. Cable plano flexible CSI para cámara Raspberry Pi.

- Longitud: 50cm.

5.1.3. Especificaciones técnicas y geométricas del sistema de control.

5.1.3.1. Creación de un servidor de transmisión de video en vivo.

Las funciones anteriormente mencionadas nos ayudarán a implementar el video streaming en Python. El método que se utiliza se conoce como “Motion JPEG” transmite una secuencia de imágenes en formato JPEG independientes. Otorga una baja latencia y es muy utilizado en cámaras IP, a pesar de que la calidad no es la mejor. La aplicación para el video streaming consta de 2 programas que se encuentran disponibles en “GITHUB”, el primero de ellos está desarrollado en Python/FLASK e importa una clase “Camera” que se encarga de proveer la secuencia de cuadros. La aplicación consta de dos rutas, `@app. route ('/')` genera el servidor web “index.html” (HTML) que generará el video streaming y define una función generadora de cuadros en formato JPEG (`“gen(camera)”`).

`@app. route ('/video_feed')`, genera la respuesta de transmisión generando las imágenes que se mostrarán en la página web. El navegador mantendrá actualizado automáticamente el elemento de imagen al mostrar la secuencia de imágenes JPEG (contiene una respuesta multiparte). El segundo programa es index.html define el servidor web, y cuenta con un atributo “src” que contiene la URL de la ruta `@app. route ('/video_feed')` que es el generador de video streaming.

Se puede acceder al video streaming a través del servidor web, conectando todos los componentes y ejecutando el programa appCam.py desde el terminal de Raspbian, solamente colocando la dirección IP de la Raspberry pi en cualquier navegador. El código de las aplicaciones utilizadas se encontrará en el Anexo 01.

5.1.3.2. Control de la locomoción.

5.1.3.2.1. Programa Arduino.

Los motores de DC necesitan ser controlados para darle movimiento al robot móvil, por ello se desarrolla un programa en el entorno de Arduino. Como el robot utiliza ruedas Mecanum, se agregan más direcciones de movimiento que en los vehículos convencionales.

La placa Arduino estará monitoreando el puerto serial constantemente, a la espera de alguna señal de entrada. Cada movimiento tiene asociado un código que activa los motores específicos para generar la movilidad en esa dirección. Para ello hemos asociado el código respectivo de cada movimiento a un dígito numérico. Cuando llegue un dígito numérico a través del puerto serial de Arduino, este ejecutará el código asociado a ese número, generando el movimiento.

- FORWARD: "1"
- BACKWARD: "2"
- RIGHT: "3"
- LEFT: "4"
- GIRO HORARIO: "5"
- GIRO ANTIHORARIO: "6"
- STOP: "7"

El código debe cargarse a la placa Arduino Mega antes de su uso. El código utilizado se encuentra en el Anexo 02.

Para generar todos los movimientos, los motores eléctricos deben girar como se muestran en las siguientes imágenes, para cada caso.

- **Arriba.** El giro de todas las ruedas debe ser horario.

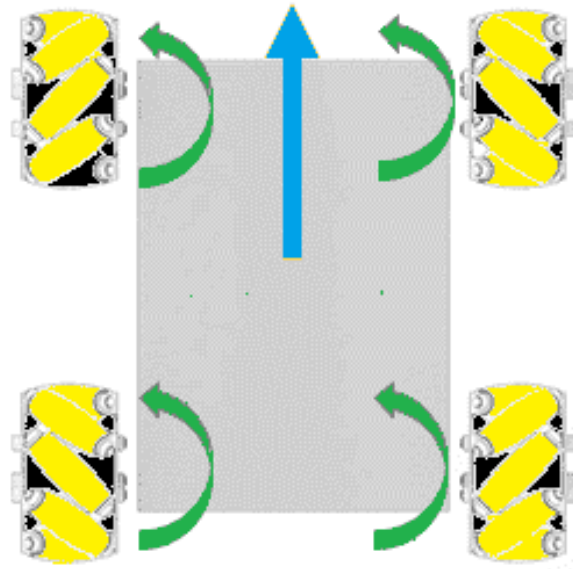


Ilustración 55. Giro hacia arriba con ruedas Mecanum.

Fuente: Elaboración Propia.

- **Abajo.** El giro de todas las ruedas debe ser antihorario.

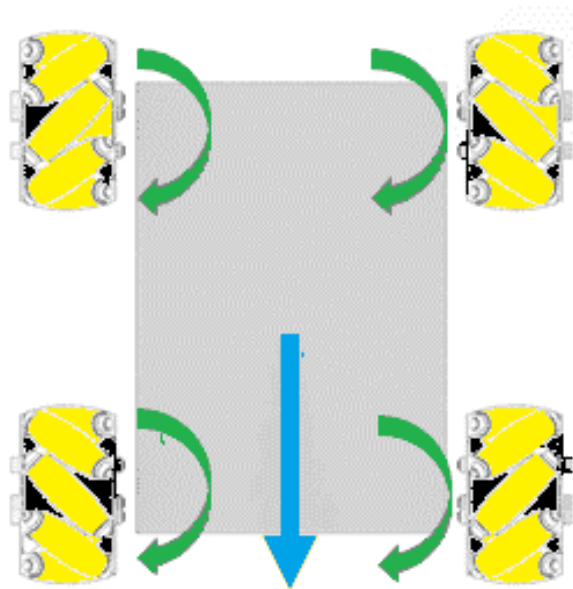


Ilustración 56. Giro hacia abajo con ruedas Mecanum.

Fuente: Elaboración Propia.

- **Derecha.** El giro de las ruedas 1 y 4 debe ser antihorario, y el giro de las ruedas 2 y 3 debe ser horario.

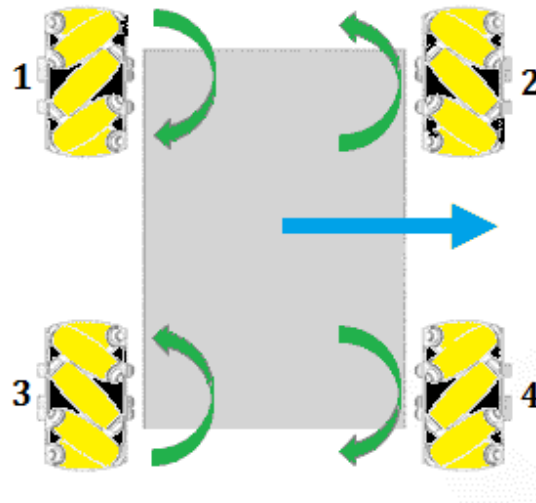


Ilustración 57. Giro a la derecha con ruedas Mecanum.

Fuente: Elaboración Propia.

- **Izquierda.** El giro de las ruedas 1 y 4 debe ser horario, y el giro de las ruedas 2 y 3 debe ser antihorario.

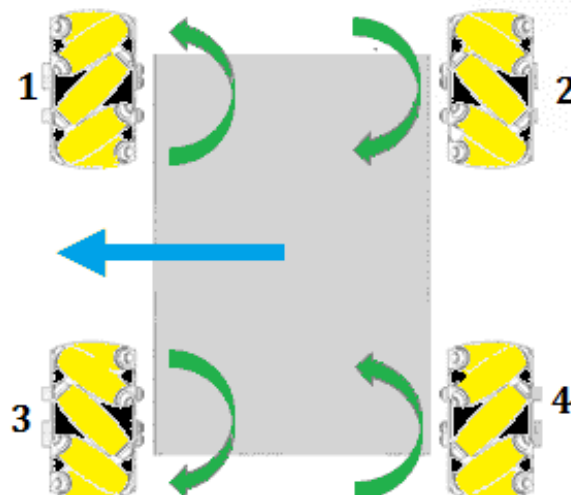


Ilustración 58. Giro a la izquierda con ruedas Mecanum.

Fuente: Elaboración Propia.

- **Giro horario sobre su propio eje.** Las ruedas a nuestra izquierda tienen que girar en sentido horario. Mientras que las ruedas a nuestra derecha tienen que girar en sentido antihorario.

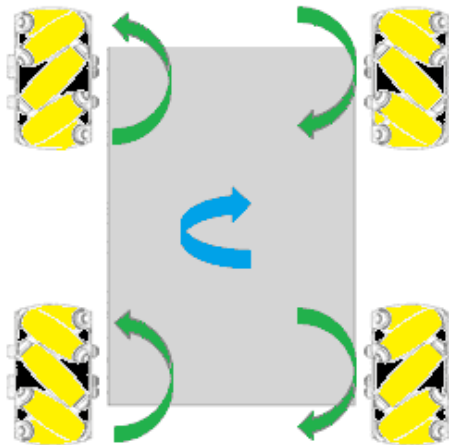


Ilustración 59. Giro horario con ruedas Mecanum.

Fuente: Elaboración Propia.

- **Giro antihorario sobre su propio eje.** Las ruedas a nuestra izquierda tienen que girar en sentido antihorario. Mientras que las ruedas a nuestra derecha tienen que girar en sentido horario.

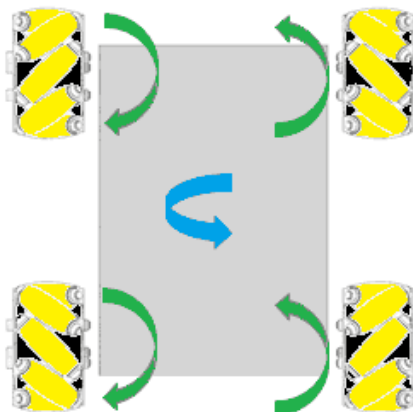


Ilustración 60. Giro antihorario con ruedas Mecanum.

Fuente: Elaboración Propia.

- **Parar.** Todos los motores deben estar desactivados.

5.1.3.2.2. GUI con Python.

La biblioteca “Tkinter” permite la creación de GUIs de manera sencilla, solamente tenemos que importarla para hacer uso de ella. La GUI consta de los comandos de dirección y los comandos de giro.

A. Comandos de dirección.

- **FORWARD.** Al presionar esta tecla se enviará un dígito “1” hacia Arduino, donde contiene un código asociado que activará los motores específicos para generar el movimiento hacia arriba.
- **BACKWARD.** Al presionar esta tecla se enviará un dígito “2” hacia Arduino, donde contiene un código asociado que activará los motores específicos para generar el movimiento hacia abajo.
- **RIGHT.** Al presionar esta tecla se enviará un dígito “3” hacia Arduino, donde contiene un código asociado que activará los motores específicos para generar el movimiento hacia la derecha.
- **LEFT.** Al presionar esta tecla se enviará un dígito “4” hacia Arduino, donde contiene un código asociado que activará los motores específicos para generar el movimiento hacia la izquierda.
- **STOP.** Al presionar esta tecla se enviará un dígito “7” hacia Arduino, donde contiene un código asociado que desactivará todos los motores para generar la detención de los motores.

B. Comandos de giro.

- GH. Al presionar esta tecla se enviará un dígito “5” hacia Arduino, donde contiene un código asociado que activará los motores específicos para generar el giro horario del robot sobre su propio eje.
- GA. Al presionar esta tecla se enviará un dígito “6” hacia Arduino, donde contiene un código asociado que activará los motores específicos para generar el giro antihorario del robot sobre su propio eje.



Ilustración 61. Interfaz de usuario para el control de la locomoción.

Fuente: Elaboración Propia.

5.1.3.3. Sistema de detección de intrusos.

Se utiliza una aplicación desarrollada en Python y disponible en GitHub. Esta aplicación permite enviar un correo de Gmail, al detectar una señal de entrada en alguno de los pines GPIO de la Raspberry Pi 3B. La incorporación de la librería “smtplib” permite realizar esta funcionalidad si agregamos nuestra cuenta de Gmail.

Cuando se detecta un intruso automáticamente el sensor PIR genera una señal de salida por el pin “OUTPUT”, esta señal es recepcionada en los pines GPIO en los cuales está conectado. Al momento en que Raspberry Pi recibe la señal generada se enviará un correo de Gmail notificando cuál de los sensores ha sido activado. El código utilizado se encuentra disponible en el anexo 03.

Los costos involucrados en la construcción del prototipo son los siguientes:

Tabla 13. Costos involucrados en la construcción.

| N ° | Partes | Unidades | Cantidad. | (S/.) /unidad | Parcial (S/) |
|-------------|-----------------------------------------------------|----------------|-----------|------------------|--------------|
| 1. | Diseño mecánico. | | | | |
| 1.1. | Chasis. | m ² | 0.08 | 83 | 6.8 |
| 1.2. | Soporte de cámara. | m ² | 0.02 | 83 | 1.7 |
| 1.3. | Soporte para Raspberry 3B. | m ² | 0.1 | 83 | 0.8 |
| 1.4. | Ruedas Mecanum. | Unidad. | 4 | 245 | 980 |
| 1.5. | Tornillo de acero Stove Bold 1/8". | Unidad | 160 | 0.03 | 4.8 |
| 1.6. | Acople. | Unidad | 4 | | |
| 1.7. | Soldadura | Kg. | 0.1 | 12 | 1.2 |
| 1.8. | Mano de obra | Dia | 1 | 100 | 100 |
| 2. | Diseño del sistema electrónico y de control. | | | | |

Fuente: Elaboración Propia.

Tabla 14. Costos involucrados en la construcción (Continuación).

| | | | | | |
|-------------|-----------------------------------|---------|----|-----|-----|
| A | Visión nocturna | | | | |
| 2.1. | Cámara NOIR Sony Raspberry Pi. | Unidad. | 01 | 165 | 165 |
| 2.2. | Cable CSI. 50cm. | Unidad. | 01 | 16 | 16 |
| 2.3. | Leds infrarrojos. | Unidad. | 02 | 8 | 16 |
| B | Control | | | | |
| 2.4. | Raspberry Pi 3B | Unidad. | 01 | 220 | 220 |
| 2.5. | Arduino Mega 2560 | Unidad. | 01 | 70 | 70 |
| | Sensores | | | | |
| 2.6. | PIR | Unidad. | 04 | 10 | 40 |
| C | Sistema de potencia. | | | | |
| 2.7. | Batería eléctrica 6V 4Ah SAKO. | Unidad. | 02 | 35 | 70 |
| 2.8. | Drivers H L298N. | Unidad. | 02 | 15 | 30 |
| 2.9. | Motores de DC JSUMO. | Unidad. | 04 | 60 | 240 |

Fuente: Elaboración Propia.

Tabla 15. Costos involucrados en la construcción (Continuación).

| D | Comunicación. | | | | |
|--------------|---------------------------------------------------------------|---------|----|--------------|---------------|
| 2.10. | Wifi. 150m. Tp-Link. | Unidad. | 01 | 60 | 60 |
| 2.11. | Cables de conexión sensores PIR. 20cm. Hembra / Hembra. | Unidad | 20 | 0.3 | 6 |
| 2.12. | Cables de conexión motores DC / drivers. | Unidad | 20 | 0.3 | 6 |
| 2.13. | Cables USB. 100cm. | Unidad | 01 | 11 | 11 |
| | | | | Total | 1945,3 |

Fuente: Elaboración Propia.

5.2. Recomendaciones.

El miniordenador Raspberry Pi incorporado tiene capacidades que aún no han sido utilizada del todo, por ello en el futuro se recomienda buscar incorporar algoritmos de procesamiento de imágenes y visión artificial. Mediante estos sistemas el reconocimiento de intrusos se facilitará más.

El robot móvil solo está diseñado para movilizarse por terrenos planos, en el futuro debe buscarse incorporar un sistema de amortiguamiento para dar mayor capacidad de superación de obstáculos. La inspección y vigilancia se realiza en muchos campos, como la minería, hospitales, calles de la ciudad, centrales eléctricas, etc. En estos lugares las condiciones del terreno no siempre son las adecuadas.

REFERENCIAS.

- Acevedo, J. J., Arrue, B. C., Maza, I., & Ollero, A. (2012). Cooperative Large Area Surveillance with a Team of Aerial Mobile Robots for Long Endurance Missions. *Journal of Intelligent & Robotic Systems* 2012 70:1, 70(1), 329–345. <https://doi.org/10.1007/S10846-012-9716-3>
- Aracil, R., Balaguer, C., & Armada, M. (2008). Robots de servicio. *Revista Iberoamericana de Automática e Informática Industrial RIAI*, 5(2), 6–13. [https://doi.org/10.1016/S1697-7912\(08\)70140-7](https://doi.org/10.1016/S1697-7912(08)70140-7)
- Araújo, A., Portugal, D., Couceiro, M. S., Sales, J., & Rocha, R. P. (2014). Desarrollo de un robot móvil compacto integrado en el middleware ROS. *Revista Iberoamericana de Automática e Informática Industrial RIAI*, 11(3), 315–326. <https://doi.org/10.1016/J.RIAI.2014.02.009>
- Bora, R., de La Pinta, J. R., Alvarez, A., & Maestre, J. M. (2013). Integration of service robots in the smart home by means of UPnP: A surveillance robot case study. *Robotics and Autonomous Systems*, 61(2), 153–160. <https://doi.org/10.1016/J.ROBOT.2012.10.005>
- Calle Rojas, M. A., & Ullauri Guerrero, G. F. (2012). *Diseño y construcción de un robot móvil teleoperado para asistencia en operaciones de alto riesgo en el Cuerpo de Bomberos*. <http://dspace.ups.edu.ec/handle/123456789/6318>
- Garcia, E., Jimenez, M. A., de Santos, P. G., & Armada, M. (2007). The evolution of robotics research. *IEEE Robotics and Automation Magazine*, 14(1), 90–103. <https://doi.org/10.1109/MRA.2007.339608>
- Karnavas, Y. L., Chasiotis, I. D., & Stravourellis, D. N. (2018). A practical BLDC motor design procedure for diver propulsion vehicle applications. *Proceedings - 2018 23rd International Conference on Electrical Machines, ICEM 2018*, 513–519. <https://doi.org/10.1109/ICELMACH.2018.8507046>
- Kusztor, A., Raud, L., Juel, B. E., Nilsen, A. S., Storm, J. F., & Huster, R. J. (2019). Sleep deprivation differentially affects subcomponents of cognitive control. *Sleep*, 42(4). <https://doi.org/10.1093/SLEEP/ZSZ016>

- Lee, H.-T., Lin, W.-C., Huang, C.-H., & Huang, Y.-J. (2011). Wireless indoor surveillance robot. *SICE Annual Conference 2011*, 2164–2169.
- López, J., Pérez, D., Paz, E., & Santana, A. (2013). WatchBot: A building maintenance and surveillance system based on autonomous robots. *Robotics and Autonomous Systems*, 61(12), 1559–1571.
<https://doi.org/10.1016/J.ROBOT.2013.06.012>
- Meghana, S., Nikhil, T. v., Murali, R., Sanjana, S., Vidhya, R., & Mohammed, K. J. (2017). Design and implementation of surveillance robot for outdoor security. *RTEICT 2017 - 2nd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, Proceedings, 2018-January*, 1679–1682.
<https://doi.org/10.1109/RTEICT.2017.8256885>
- Mittal, S., & Rai, J. K. (2017). Wadoro: An autonomous mobile robot for surveillance. *1st IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems, ICPEICES 2016*.
<https://doi.org/10.1109/ICPEICES.2016.7853652>
- Avila, S. (2016). Implicaciones Del Trabajo Nocturno y/o Trabajo Por Turnos Sobre La Salud. *Medicina Legal de Costa Rica*, 33(1), 70–78.
https://www.scielo.sa.cr/scielo.php?pid=S1409-00152016000100070&script=sci_abstract&tlng=es
- Siegwart, R., Nourbakhsh, I. R., & Scaramuzza, D. (2011). Locomotion. *Introduction to Autonomous Mobile Robots*, 1–12. <https://0-ieeeexplore-ieee-org.oasis.unisa.ac.za/xpl/ebooks/bookPdfWithBanner.jsp?fileName=6277379.pdf&bkn=6267528&pdfType=chapter>
- Thakur, D., Likhachev, M., Keller, J., Kumar, V., Dobrokhodov, V., Jones, K., Wurz, J., & Kaminer, I. (2013). Planning for opportunistic surveillance with multiple robots. *IEEE International Conference on Intelligent Robots and Systems*, 5750–5757. <https://doi.org/10.1109/IROS.2013.6697189>

ANEXOS

ANEXO 01.

A. CÓDIGO FLASK/PYTHON (appCam.py)

```
#Importar librerías necesarias.

from flask import Flask, render_template, Response

from camera_pi import Camera

app = Flask(__name__)

@app.route('/') # Ruta número 01.

def index(): # Definimos "index" como Servidor Web "html".

    """Video streaming home page."""

    return render_template('index.html')

def gen(camera): # Definimos una función generadora.

    """Video streaming generator function."""

    while True: # Bucle para la obtención constante de cuadros de la cámara.

        frame = camera.get_frame()

        yield (b'--frame\r\n'

               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')

@app.route('/video_feed') # Ruta número 02.

def video_feed(): # Definición de la función para generar video streaming.

    """Video streaming route. Put this in the src attribute of an img tag."""
```


#Tomamos los cuadros generados por el bucle “while” y generamos una función de respuestas multiparte para actualizar constantemente los cuadros.

```
return Response(gen(Camera()),
                mimetype='multipart/x-mixed-replace; boundary=frame') #
if __name__ == '__main__':
    app.run(host='0.0.0.0', port =80, debug=True, threaded=True)
```

B. Servidor Web (index.html)

```
<!--
index.html
-->
<html>
<head>
<title>Robot Móvil de Vigilancia ----- video Streaming</title>
<link rel="stylesheet" href='../static/style.css'/>
<style>
    body {
        text-align: center;
    }
</style>
</head>
<body>
<h1> Robot Móvil de Vigilancia ----- Streaming </h1>
<h3></h3>
```

```
<hr>

    <h3>        Return        to        main        page        ==>        <a
href="http://192.168.1.13/"class="button">RETURN</a></h3>

    <hr>

    <p> Lambayeque@2020  </p>

</body>

</html>
```

ANEXO 02.

A. Programa Arduino.

```
//Control de los motores de DC.

// variables utilizadas.

const int pinesmotor01A = 51;
const int pinesmotor01B = 49;
const int pinesmotor02A = 47;
const int pinesmotor02B = 45;
const int pinesmotor03A = 50;
const int pinesmotor03B = 48;
const int pinesmotor04A = 46;
const int pinesmotor04B = 44;

void setup()
{
    Serial.begin(9600); // Iniciar comunicación serial.

    pinMode(53, OUTPUT); // Establecer pin 53 como salida.
    pinMode(49, OUTPUT); // Establecer pin 49 como salida.
```

```

pinMode(47, OUTPUT); // Establecer pin 47 como salida.
pinMode(45, OUTPUT); // Establecer pin 45 como salida.
pinMode(50, OUTPUT); // Establecer pin 50 como salida.
pinMode(48, OUTPUT); // Establecer pin 48 como salida.
pinMode(46, OUTPUT); // Establecer pin 46 como salida.
pinMode(44, OUTPUT); // Establecer pin 44 como salida.

}

void loop()
{
  if (Serial.available()>0)
  {
    int dato = Serial.read(); // Iniciar lectura si la comunicación serial está activa.

    if (dato == '1') // movimiento hacia arriba.
    {
      //Motor 01.
      digitalWrite(pinesmotor01A,HIGH);
      digitalWrite(pinesmotor01B,LOW);

      //Motor 02.
      digitalWrite(pinesmotor02A,HIGH);
      digitalWrite(pinesmotor02B,LOW);

      //Motor 03.
      digitalWrite(pinesmotor03A,HIGH);

```

```
digitalWrite(pinesmotor03B,LOW);  
  
//Motor 04.  
  
digitalWrite(pinesmotor04A,HIGH);  
digitalWrite(pinesmotor04B,LOW);  
  
delay(20);  
}
```

```
if (dato == '2')// movimiento hacia abajo.  
{  
  
    //Motor 01.  
  
    digitalWrite(pinesmotor01A,LOW);  
    digitalWrite(pinesmotor01B,HIGH);  
  
    //Motor 02.  
  
    digitalWrite(pinesmotor02A,LOW);  
    digitalWrite(pinesmotor02B,HIGH);  
  
    //Motor 03.  
  
    digitalWrite(pinesmotor03A,LOW);  
    digitalWrite(pinesmotor03B,HIGH);  
  
    //Motor 04.  
  
    digitalWrite(pinesmotor04A,LOW);  
    digitalWrite(pinesmotor04B,HIGH);;  
    delay(20);  
}
```

```
if (dato == '3') // movimiento hacia la derecha.
```

```
{
```

```
    //Motor 01.
```

```
    digitalWrite(pinesmotor01A,LOW);
```

```
    digitalWrite(pinesmotor01B,HIGH);
```

```
    //Motor 02.
```

```
    digitalWrite(pinesmotor02A,HIGH);
```

```
    digitalWrite(pinesmotor02B,LOW);
```

```
    //Motor 03.
```

```
    digitalWrite(pinesmotor03A,HIGH);
```

```
    digitalWrite(pinesmotor03B,LOW);
```

```
    //Motor 04.
```

```
    digitalWrite(pinesmotor04A,LOW);
```

```
    digitalWrite(pinesmotor04B,HIGH);
```

```
    delay(20);
```

```
}
```

```
if (dato == '4') // movimiento hacia la izquierda.
```

```
{
```

```
    //Motor 01.
```

```

digitalWrite(pinesmotor01A,HIGH);
digitalWrite(pinesmotor01B,LOW);

//Motor 02.

digitalWrite(pinesmotor02A,LOW);
digitalWrite(pinesmotor02B,HIGH);

//Motor 03.

digitalWrite(pinesmotor03A,LOW);
digitalWrite(pinesmotor03B,HIGH);

//Motor 04.

digitalWrite(pinesmotor04A,HIGH);
digitalWrite(pinesmotor04B,LOW);
delay(20);
}

if (dato == '5') // movimiento horario.
{
    //Motor 01.

    digitalWrite(pinesmotor01A,HIGH);
    digitalWrite(pinesmotor01B,LOW);

    //Motor 02.

    digitalWrite(pinesmotor02A,LOW);
    digitalWrite(pinesmotor02B,HIGH);

```

```

//Motor 03.

digitalWrite(pinesmotor03A,HIGH);

digitalWrite(pinesmotor03B,LOW);

//Motor 04.

digitalWrite(pinesmotor04A,LOW);

digitalWrite(pinesmotor04B,HIGH);

delay(20);

}

if (dato == '6') //movimiento antihorario.
{
    //Motor 01.

    digitalWrite(pinesmotor01A,LOW);

    digitalWrite(pinesmotor01B,HIGH);

    //Motor 02.

    digitalWrite(pinesmotor02A,HIGH);

    digitalWrite(pinesmotor02B,LOW);

    //Motor 03.

    digitalWrite(pinesmotor03A,LOW);

    digitalWrite(pinesmotor03B,HIGH);

    //Motor 04.

    digitalWrite(pinesmotor04A,HIGH);

    digitalWrite(pinesmotor04B,LOW);

    delay(20);

}

```

```
if (dato == '7') //parar giro continuo.
{
    //Motor 01.
    digitalWrite(pinesmotor01A,LOW);
    digitalWrite(pinesmotor01B,LOW);
    //Motor 02.
    digitalWrite(pinesmotor02A,LOW);
    digitalWrite(pinesmotor02B,LOW);
    //Motor 03.
    digitalWrite(pinesmotor03A,LOW);
    digitalWrite(pinesmotor03B,LOW);
    //Motor 04.
    digitalWrite(pinesmotor04A,LOW);
    digitalWrite(pinesmotor04B,LOW);
    delay(20);
}
}
}
```


B. GUI (interfaz de usuario).

#GUI Python.

#Librerías.

import tkinter as tk

from tkinter import ttk

import serial, time

import threading

#configuración Arduino.

arduino = serial.Serial('COM4', baudrate = 9600, timeout=1)

time.sleep(2)

class PasosFrame(tk.Frame):

#pestaña principal.

def __init__(self, *args, **kwargs):

 super().__init__(*args, **kwargs)

#widgets.

#Etiqueta título 01.

self.L1 = tk.Label(self)

self.L1["text"] = ("COMANDOS DE DIRECCIÓN")

self.L1["bg"]="black")

self.L1["fg"]="white")

```
self.L1.place(x=120, y=30)
```

```
#Etiqueta título 02.
```

```
self.L2 = tk.Label(self)
```

```
self.L2["text"] = ("GIRO HORARIO")
```

```
self.L2["bg"]="black")
```

```
self.L2["fg"]="white")
```

```
self.L2.place(x=420, y=30)
```

```
#Etiqueta título 03.
```

```
self.L3 = tk.Label(self)
```

```
self.L3["text"] = ("GIRO ANTIHORARIO")
```

```
self.L3["bg"]="black")
```

```
self.L3["fg"]="white")
```

```
self.L3.place(x=600, y=30)
```

```
#Botón Adelante.
```

```
self.gAd = tk.Button(self, text="FORWARD", command=self.continuo_ad,  
width=10, bg="orange",font=("Arial", 10),fg="white")
```

```
self.gAd.pack()
```

```
self.gAd.place(x=150, y=100)
```

```
#Botón Atrás.
```

```
self.gAt = tk.Button(self, text="BACKWARD", command=self.continuo_at,  
width=10, bg="orange",font=("Arial", 10),fg="white")
```

```
self.gAt.pack()
```

```
self.gAt.place(x=150, y=200)
```

```
#Botón Derecha.
```

```
self.gDr = tk.Button(self, text="RIGHT", command=self.continuo_de,  
width=10, bg="orange", font=("Arial", 10), fg="white")
```

```
self.gDr.pack()
```

```
self.gDr.place(x=250, y=150)
```

```
#Botón Izquierda.
```

```
self.glz = tk.Button(self, text="LEFT", command=self.continuo_iz, width=10,  
bg="orange", font=("Arial", 10), fg="white")
```

```
self.glz.pack()
```

```
self.glz.place(x=50, y=150)
```

```
#Botón Giro Horario.
```

```
self.gGh = tk.Button(self, text="GH", command=self.continuo_Hr, width=5,  
bg="orange", font=("Arial", 10), fg="white")
```

```
self.gGh.pack()
```

```
self.gGh.place(x=440, y=150)
```

```
#Botón Giro Antihorario.
```

```
self.gGa = tk.Button(self, text="GA", command=self.continuo_AHr, width=5,  
bg="orange", font=("Arial", 10), fg="white")
```

```
self.gGa.pack()
```

```
self.gGa.place(x=630, y=150)
```

```
#Botón Stop.
```

```
self.gAs = tk.Button(self, text="STOP", command=self.parar_g, width=10,  
bg="orange",font=("Arial", 10),fg="white")
```

```
self.gAs.pack()
```

```
self.gAs.place(x=150, y=150)
```

```
# Funciones #
```

```
#Giro continuo Adelante.
```

```
def continuo_ad(self):
```

```
    arduino.write(b'1')
```

```
#Giro continuo Atrás.
```

```
def continuo_at(self):
```

```
    arduino.write(b'2')
```

```
#Giro continuo a la derecha.
```

```
def continuo_de(self):
```

```
    arduino.write(b'3')
```

```
#Giro continuo a la izquierda.
```

```
def continuo_iz(self):
```

```
    arduino.write(b'4')
```

```

#Giro horario.

def continuo_Hr(self):

    arduino.write(b'5')


#Giro antihorario.

def continuo_AHr(self):

    arduino.write(b'6')


#Para giro continuo.

def parar_g(self):

    arduino.write(b'7')


#Configuración de pestañas.

class Interface(ttk.Frame):

    def __init__(self, principalV):

        super().__init__(principalV)

        #título principal de ventana.

        principalV.title("ROBOT MÓVIL DE VIGILANCIA NOCTURNA")

        #objeto notebook, definimos tamaño de la ventana.

        self.notebook = ttk.Notebook(self,width=800, height=400)

        #configuración de pestaña1.

        self.paso_frame = PasosFrame(self.notebook)

        self.notebook.add(

            self.paso_frame,    text="LOCOMOCIÓN    DEL    ROBOT    MÓVIL",

padding=10)

```

```
self.notebook.pack()
```

```
self.pack()
```

```
#####
```

```
principalV = tk.Tk()
```

```
principalV.geometry('800x400')
```

```
principalV.resizable(width= False, height= False)
```

```
app = Interface(principalV)
```

```
app.mainloop()
```

ANEXO 03.

```
#Importar librerías.
```

```
import RPi.GPIO as GPIO
```

```
import time
```

```
import smtplib
```

```
#Definición de pines GPIO para los sensores.
```

```
pir_sensor01 = 4
```

```
pir_sensor02 = 17
```

```
pir_sensor03 = 27
```

```
pir_sensor04 = 22
```

```
#Definir Pines como entradas.
```

```
GPIO.setmode(GPIO.BOARD)
```

```
GPIO.setup(pir_sensor01, GPIO.IN)
```

```
GPIO.setup(pir_sensor02, GPIO.IN)
```

```
GPIO.setup(pir_sensor03, GPIO.IN)
```

```

GPIO.setup(pir_sensor04, GPIO.IN)

#Definir estado inicial de los sensores.

current_state01 = 0

current_state02 = 0

current_state03 = 0

current_state04 = 0


try:

    while True:

        time.sleep(0.1)

        current_state01 = GPIO.input(pir_sensor01)

        current_state02 = GPIO.input(pir_sensor02)

        current_state03 = GPIO.input(pir_sensor03)

        current_state04 = GPIO.input(pir_sensor04)


        if current_state01 == 1:

            print("GPIO pin %s is %s" % (pir_sensor01, current_state01))

            GPIO.output(piezo,True)

            time.sleep(1)

            GPIO.output(piezo,False)

            FROM = "adol123gustavo@gmail.com"


            #Lista de correos a enviar.

            TO = ['micorreo1@gmail.com','micorreo2@gmail.com']

            TOstr = 'adol123gustavo@gmail.com'

```

```

server = smtplib.SMTP('smtp.gmail.com',587)

server.ehlo()

server.starttls()

server.ehlo


#Es necesario logearse con el servidor utilizando tu cuenta
micorreo1@gmail.com y tu clave.

server.login(FROM,'123456')

# Crear el encabezado del correo.

header = 'To:' + TOstr + '\n' + 'From: ' + FROM + '\n' + 'Subject: SENSOR
PIR DE LA CARA FRONTAL ACTIVADO. \n'

print header

# Unir el encabezado con el mensaje...

msg = header + '\n INTRUSO DETECTADO, VERIFICAR EL ESTADO
DEL AMBIENTE. \n'

# Una vez que se haya pasado lo anterior ahora si a enviar...

server.sendmail(FROM,TO,msg)

print "Listo!"

# Cerramos sesión ...

server.quit()

time.sleep(5)


if current_state02 == 1:

    print("GPIO pin %s is %s" % (pir_sensor02, current_state02))

    GPIO.output(piezo,True)

```



```

time.sleep(1)

GPIO.output(piezo,False)

FROM = "adol123gustavo@gmail.com"

#Lista de correos a enviar.

TO = ['micorreo1@gmail.com','micorreo2@gmail.com']

TOstr = 'adol123gustavo@gmail.com'

server = smtplib.SMTP('smtp.gmail.com',587)

server.ehlo()

server.starttls()

server.ehlo


#Es necesario logearse con el servidor utilizando tu cuenta
micorreo1@gmail.com y tu clave.

server.login(FROM,'123456')

# Crear el encabezado del correo.

header = 'To:' + TOstr + '\n' + 'From: ' + FROM + '\n' + 'Subject: SENSOR
PIR DE LA CARA TRASERA ACTIVADO. \n'

print header

# Unir el encabezado con el mensaje ...

msg = header + '\n INTRUSO DETECTADO, VERIFICAR EL ESTADO
DEL AMBIENTE. \n'

# Una vez que se haya pasado lo anterior ahora si a enviar ...

server.sendmail(FROM,TO,msg)

print "Listo !"

# Cerramos sesión ...

```

```

server.quit()

time.sleep(5)

if current_state03 == 1:

    print("GPIO pin %s is %s" % (pir_sensor03, current_state03))

    GPIO.output(piezo,True)

    time.sleep(1)

    GPIO.output(piezo,False)

    FROM = "adol123gustavo@gmail.com"

    #Lista de correos a enviar.

    TO = ['micorreo1@gmail.com','micorreo2@gmail.com']

    TOstr = 'adol123gustavo@gmail.com'

    server = smtplib.SMTP('smtp.gmail.com',587)

    server.ehlo()

    server.starttls()

    server.ehlo

    #Es necesario logearse con el servidor utilizando tu cuenta
micorreo1@gmail.com y tu clave.

    server.login(FROM,'123456')

    # Crear el encabezado del correo.

    header = 'To:' + TOstr + '\n' + 'From: ' + FROM + '\n' + 'Subject: SENSOR
PIR DE LA CARA LATERAL DERECHA ACTIVADO. \n'

    print header

```

```

# Unir el encabezado con el mensaje ...

msg = header + '\n INTRUSO DETECTADO, VERIFICAR EL ESTADO
DEL AMBIENTE. \n'

# Una vez que se haya pasado lo anterior ahora si a enviar ...

server.sendmail(FROM,TO,msg)

print "Listo !"

# Cerramos sesión ...

server.quit()

time.sleep(5)


if current_state04 == 1:

    print("GPIO pin %s is %s" % (pir_sensor04, current_state04))

    GPIO.output(piezo,True)

    time.sleep(1)

    GPIO.output(piezo,False)

    FROM = "adol123gustavo@gmail.com"

    #Lista de correos a enviar.

    TO = ['micorreo1@gmail.com','micorreo2@gmail.com']

    TOstr = 'adol123gustavo@gmail.com'

    server = smtplib.SMTP('smtp.gmail.com',587)

    server.ehlo()

    server.starttls()

    server.ehlo

```

#Es necesario logearse con el servidor utilizando tu cuenta micorreo1@gmail.com y tu clave.

```
server.login(FROM,'123456')
```

Crear el encabezado del correo.

```
header = 'To:' + TOstr + '\n' + 'From: ' + FROM + '\n' + 'Subject: SENSOR  
PIR DE LA CARA LATERAL IZQUIERDA ACTIVADO. \n'
```

```
print header
```

Unir el encabezado con el mensaje ...

```
msg = header + '\n INTRUSO DETECTADO, VERIFICAR EL ESTADO  
DEL AMBIENTE. \n'
```

Una vez que se haya pasado lo anterior ahora si a enviar ...

```
server.sendmail(FROM,TO,msg)
```

```
print "Listo !"
```

Cerramos sesión ...

```
server.quit()
```

```
time.sleep(5)
```

except KeyboardInterrupt:

```
pass
```

finally:

```
GPIO.cleanup()
```



UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
FACULTAD DE INGENIERIA MECANICA Y ELECTRICA
SECRETARIA DOCENTE



UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
FACULTAD DE INGENIERIA MECANICA ELECTRICA
CERTIFICO que el presente documento es copia
fidel de su original que he tenido a la vista

Ing. Oscar Mendez Cruz
SECRETARIO DOCENTE

ACTA DE SUSTENTACION DE TESIS



En la ciudad de Lambayeque siendo las 10:00 horas del día
07 DE ENERO del año 2020, en la Sala de Sustentaciones de la
Facultad de Ingeniería Mecánica y Eléctrica de la Universidad Nacional Pedro Ruiz Gallo de
Lambayeque, se reunieron los miembros de Jurado, designados mediante Resolución
N° 303-2019-D-FIME, de fecha 31-12-2019, conformado por:

M. SC JUAN TUMIALAN HINOSTROZA : PRESIDENTE
ING. CARLOS YUPANQUI RODRIGUEZ : SECRETARIO
ING. OSCAR MENDEZ CRUZ : MIEMBRO
M. SC CARLOS A. CHAMBERGO LARREA : ASESOR

Se recibió la Tesis titulada:

"DISEÑO DE UN ROBOT MOVIL PARA REALIZAR TAREAS
DE VIGILANCIA NOCTURNA EN LA FACULTAD DE
INGENIERIA MECANICA DE LA UNPREG"

Presentado por su autor, Bachiller: GONZALES LOPEZ GUSTAVO ADOLFO

Finalizada la sustentación, el sustentante respondió acertadamente las preguntas formuladas por el
jurado examinador procediendo a deliberar.

Probando la Tesis con el calificativo: NOTA APROBADO

Quedando el sustentante apto para obtener el Título Profesional de Ingeniero Mecánico Electricista,
debiendo cumplir antes con las exigencias de las normas legales vigentes.

Siendo las 10:50 horas del mismo día se da por concluido el acto académico,
firmando la presente acta el Jurado respectivo:

PRESIDENTE

SECRETARIO

MIEMBRO

ASESOR

CONSTANCIA DE APROBACIÓN DE ORIGINALIDAD DE TESIS

Yo, Aníbal Jesús Salazar Mendoza, Asesor de tesis del trabajo de investigación, del estudiante Gustavo Adolfo Gonzales López

Titulada:

DISEÑO DE UN ROBOT MÓVIL PARA REALIZAR TAREAS DE VIGILANCIA NOCTURNA EN LA FACULTAD DE INGENIERÍA MECÁNICA DE LA U.N.P.R.G, luego de la revisión exhaustiva del documento constato que la misma tiene un índice de similitud de 19% verificable en el reporte de similitud del programa Turnitin.

El suscrito analizó dicho reporte y concluyo que cada una de las coincidencias detectadas no constituyen plagio. A mi leal saber y entender la tesis cumple con todas las normas para el uso de citas y referencias establecidas por la Universidad Nacional Pedro Ruiz Gallo.

Lambayeque, 09 de Agosto del 2023



Aníbal Jesús Salazar Mendoza

DNI: 16720249

ASESOR



GUSTAVO A. Gonzales

DNI: 73129143

Autor.

19%

INDICE DE SIMILITUD

20%

FUENTES DE INTERNET

2%

PUBLICACIONES

9%

TRABAJOS DEL
ESTUDIANTE

FUENTES PRIMARIAS

1

hdl.handle.net

Fuente de Internet

3%

2

documentop.com

Fuente de Internet

3%

3

es.wikipedia.org

Fuente de Internet

2%

4

appswl.elsevier.es

Fuente de Internet

2%

5

blog.infranetworking.com

Fuente de Internet

1%

6

academica-e.unavarra.es

Fuente de Internet

1%

7

gissellastar.blogspot.com

Fuente de Internet

1%

8

www.edmundoptics.es

Fuente de Internet

1%

9

issuu.com

Fuente de Internet

1%

Anibal Jesus Salazar Mendoza

Asesor