Implementación de un sistema de navegación reactiva-social y telepresencia en el prototipo de un robot móvil diferencial

Espinoza Jaramillo, Ismael Nicolás y Zúñiga Navarrete, Christian Santiago

Departamento de Eléctrica, Electrónica y Telecomunicaciones

Carrera de Ingeniería en Electrónica, Automatización y Control

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

Ing. Loza Matovelle, David César, M.Sc.

3 de marzo del 2021

URKUND

Document Information

Analyzed document	TESIS_ESPINOZA_ZUÑIGA.pdf (D97228343)
Submitted	3/4/2021 4:19:00 PM
Submitted by	
Submitter email	dcloza@espe.edu.ec
Similarity	4%
Analysis address	dcloza.espe@analysis.urkund.com

Sources included in the report

SA	**documento_revision_urkund.pdf** Document documento_revision_urkund.pdf (D63716386)	8
SA	**Universidad de las Fuerzas Armadas ESPE / Tesis Alexis Carrera.pdf** Document Tesis Alexis Carrera.pdf (D38703363) Submitted by: wgaguilar@espe.edu.ec Receiver: wgaguilar.espe@analysis.urkund.com	3
W	URL: https://doi.org/10.3390/robotics8030076 Fetched: 3/4/2021 4:21:00 PM	3
SA	**Universidad de las Fuerzas Armadas ESPE / TESIS_ROBOT_DE_TELEPRESENCIA.pdf** Document TESIS_ROBOT_DE_TELEPRESENCIA.pdf (D40920343) Submitted by: dcloza@espe.edu.ec Receiver: dcloza.espe@analysis.urkund.com	1
SA	**Universidad de las Fuerzas Armadas ESPE / PROYECTO DE TITULACION SIMBA.pdf** Document PROYECTO DE TITULACION SIMBA.pdf (D40111557) Submitted by: dcloza@espe.edu.ec Receiver: dcloza.espe@analysis.urkund.com	13
SA	**Universidad de las Fuerzas Armadas ESPE / DISEÑO Y CONSTRUCCION DE UN PROTOTIPO DE PALTAFORMA ROBOTICA PARA APLICACIONES EN ...** Document DISEÑO Y CONSTRUCCION DE UN PROTOTIPO DE PALTAFORMA ROBOTICA PARA APLICACIONES EN ... (D36039053) Submitted by: dcloza@espe.edu.ec Receiver: dcloza.espe@analysis.urkund.com	3

Firma:

Firmado electrónicamente por:
DAVID CESAR
LOZA
MATOVELLE

Ing. Loza Matovelle, David César, M.Sc.

DIRECTOR

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

CERTIFICACIÓN

Certifico que el trabajo de titulación, "**Implementación de un sistema de navegación reactiva-social y telepresencia en el prototipo de un robot móvil diferencial**" fue realizado por los señores **Espinoza Jaramillo, Ismael Nicolás** y **Zúñiga Navarrete Christian Santiago** el cual ha sido revisado y analizado en su totalidad por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 5 de marzo de 2021

Firma:

Firmado electrónicamente por:
DAVID CESAR
LOZA
MATOVELLE

.....

Ing. Loza Matovelle, David César, M.Sc.

C. C. 1708661549

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

RESPONSABILIDAD DE AUTORÍA

Nosotros, **Espinoza Jaramillo, Ismael Nicolás y Zúñiga Navarrete, Christian Santiago**, con cédulas de ciudadanía n° 1722104716 y 1721225900, declaramos que el contenido, ideas y criterios del trabajo de titulación: **Implementación de un sistema de navegación reactiva-social y telepresencia en el prototipo de un robot móvil diferencial** es de nuestra autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Sangolquí, 5 de marzo de 2021

Firmas:

Espinoza Jaramillo, Ismael Nicolás

C.C.: 1722104716

Zúñiga Navarrete, Christian Santiago

C.C.: 1721225900

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

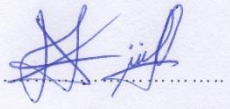
CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

AUTORIZACIÓN DE PUBLICACIÓN

Nosotros **Espinoza Jaramillo, Ismael Nicolás y Zúñiga Navarrete, Christian Santiago,** con cédulas de ciudadanía n° 1722104716 y 1721225900, autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **Implementación de un sistema de navegación reactiva-social y telepresencia en el prototipo de un robot móvil diferencial** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de nuestra responsabilidad.

Sangolquí, 5 de marzo de 2021

Firmas:



Espinoza Jaramillo, Ismael Nicolás
C.C.: 1722104716

Zúñiga Navarrete, Christian Santiago
C.C.: 1721225900

Dedicatoria

El proyecto realizado lo dedico a mi familia. A mis padres, porque su sacrificio ha sido una muestra de esfuerzo y responsabilidad día a día. Y su amor ha sabido sobrepasar las dificultades. A Pablo y David por la hermandad y camaradería. A mi abuelito Franklin, por siempre creer y estar orgulloso de sus nietos. A toda mi familia dedico mis logros. A mis amigos por su apoyo y consejo. De manera especial dedico el proyecto a un excelente ingeniero y sobre todo persona incomparable, Byron Zúñiga, aunque ya no esté entre nosotros.

Por último y más importante dedico el proyecto a Dios, porque de Él viene todo lo bueno.

Christian Santiago Zúñiga Navarrete

Dedico el trabajo a todos los miembros de familia, los cuales siempre serán mi mayor orgullo e inspiración, todo el esfuerzo y trabajo que he realizado ha sido pensando en ustedes. En especial a todas las mujeres que son parte de mi vida, mi madre, mi tía, mi abuela, mi bisabuela y mi nana. Gracias a su educación y cariño he logrado avanzar y convertirme en lo que soy. Espero algún día devolver todo lo que me han dado, los amo.

A mis compañeros, amigos y próximos colegas con los cuales hemos logrado y sobrellevado muchas cosas a lo largo de estos años en esta carrera.

Ismael Nicolás Espinoza Jaramillo

Agradecimiento

Primero agradezco al creador de todas las cosas, a Dios, por su misericordia de darme la salud, inteligencia y herramientas para enfrentar los retos que he encontrado en la vida. Agradezco a mi padre por enseñarme dedicación y responsabilidad desde pequeño. A mí madre por demostrar su amor en cada situación, preocupándose por su familia y buscando su bienestar. A mis hermanos por su amistad cuando surgía una situación difícil y su compañerismo al compartir la pasión por la automatización y la música.

Agradezco a mis amigos por su apoyo y cariño. A mis compañeros y colegas, por generar lazos dentro de situaciones complicadas presentes en el estudio de la ingeniería. Especialmente a un gran estudiante y compañero de Tesis, Ismael, por su dedicación y trabajo en equipo.

A los profesores que se han esforzado por enseñar y compartir su experiencia laboral. Al Ing. David Loza por su guía y comprensión durante la elaboración del proyecto.

Christian Santiago Zúñiga Navarrete

Ante todo, agradezco a Dios por brindarme salud, sensatez, inteligencia y por guiarme en cada travesía que me he propuesto durante toda mi vida.

A mi madre Paulina quien ha sido la mayor fuente de admiración y apoyo que tengo. Ha sido padre-madre y un modelo a seguir durante mi crecimiento y en donde a pesar de las adversidades que la vida nos ha impuesto siempre me ha brindado cariño y cuidado, levantándose a trabajar todos los días sin importar nada para darme lo mejor. De ti he aprendido la perseverancia y humildad.

A mi tía Anahí quien ha velado siempre por mi bienestar y cuidado de mí sin importar nada, has sido una fuente de inspiración y me has enseñado con tus acciones a ayudar los demás. Agradezco de todo corazón en todo lo que me has ayudado.

A mi abuela Angelita, mi bisabuela Inés y mi nana Cecilia quienes siempre ha visto por mi salud y bienestar, nunca me falto una muestra de cariño de su parte y me han inculcado muchos valores. Para mi tío Estuardo y todos sus hermanos por todo lo que me han dado y me han hecho sentir parte en todos los eventos y reuniones. A mi primo Sebastián quien ha sido más la imagen de un hermano de un hermano para mí y con quien siempre encontrare un amigo y alguien en quien confiar.

Quiero agradecer a todos mis compañeros y amigos con quien he pasado por muchas acontecimientos y procesos durante mi carrera. Cada ayuda y consejo lo agradezco desde el corazón y puedo decir con certeza que estos años no serían lo mismo sin ellos.

Un especial agradecimiento a todos los ingenieros con los que he recibido clases y a mi tutor, ya que gracias a ellos he logrado ampliar mi conocimiento y resolver muchos de los problemas que se han presentado durante mi carrera.

Ismael Nicolás Espinoza Jaramillo

Índice de contenidos

Índice de tablas

Índice de figuras

Resumen

En la actualidad los sistemas de telepresencia se han convertido en una herramienta de comunicación esencial entre personas. Lo cual se ha visto reflejado en diversas aplicaciones como teletrabajo, sistemas de vigilancia, telemedicina y teleducación. Partiendo de este principio, la inclusión de la telepresencia en la robótica social se ha vuelto clave para crear plataformas móviles que otorguen libertad de movimiento al operador remoto.

Para la interacción entre las personas y los robots móviles se han desarrollado sistemas de navegación reactiva que tomen en cuenta parámetros de comportamiento social. En el trabajo de investigación realizado se presenta el desarrollo de un sistema de navegación reactiva social y telepresencia en un prototipo de robot diferencial. El sistema tiene dos planificadores, global y local, encargados de generar la ruta de navegación en base a un mapa de costos, que posee una estructura por capas. La información del entorno se obtiene mediante el módulo de percepción, y la posición del robot por medio de un filtro EFK implementado en el módulo de localización. Para el comportamiento social, se utiliza un algoritmo de detección de piernas-personas que permite generar una zona prohibida alrededor del individuo durante la navegación. Mientras que, para la interacción entre la persona y el robot se utilizó el reconocimiento de un conjunto de gestos. Por último, el sistema de telepresencia y control remoto se estableció a través de WebRTC entre una aplicación móvil y el robot diferencial.

- Palabras clave:

- **ROBÓTICA SOCIAL**
- **NAVEGACIÓN REACTIVA**
- **TELEPRESENCIA**
- **WEBRTC**

Abstract

Nowadays telepresence systems have become an essential communication tool between people. Which has been reflected in various applications such as telecommuting, surveillance systems, telemedicine, and tele-education. Therefore, the inclusion of telepresence in the social robots has become an essential key into development of mobile platforms that grant freedom of movement to the remote operator.

For interaction between people and mobile robots have been developed multiple reactive navigation systems which consider social behavior parameters. The research work presents the development of a social reactive navigation and telepresence system in a differential mobile robot prototype. There are two planners into the developed system, global and local, in charge of generating the navigation path based on a cost map. The cost map has a layered structure. Environment information is obtained by a sensing module and the robot´s position is computed through a EFK filter implemented in the location module. For social behavior, a leg-person detection algorithm is used. The algorithm allows generating a forbidden zone around people during navigation. While, for the interaction between the person and the robot, the recognition of a set of gestures was used. Finally, telepresence and remote-control system were established through WebRTC between a mobile application and the differential robot.

- Key words:

- **SOCIAL ROBOTICS**
- **REACTIVE NAVIGATION**
- **TELEPRESENCE**
- **WEBRTC**

Capítulo I

Introducción

Dentro del capítulo 1 se plantea los antecedentes, justificación e importancia. Así como el alcance, objetivos y planteamiento del problema a resolver. Con la finalidad de determinar el estado actual de los trabajos previos en el país, identificar la problemática a resolverse, y la delimitación correspondiente del proyecto. Por último, el capítulo presenta una breve estructura del desarrollo de actividades correspondiente a cada capítulo.

Planteamiento del problema

En los robots móviles se trata de integrar capacidades que en cierta forma tratan de imitar a las del ser humano (vista, tacto, oído, etc.) y a los cuales, se los ha puesto a prueba en diferentes entornos. Tomando en cuenta los proyectos desarrollados especialmente dentro de la Universidad de las Fuerzas Armadas ESPE, se plantea como una necesidad la implementación de un robot móvil con capacidad de evitar colisiones en un entorno controlado y que integre un comportamiento social durante la navegación como las reglas mencionadas en (Pinter et al., 2017). El proyecto debe incluir:

- Un sistema de control configurado para navegación del robot en un área de trabajo.
- Un sistema de detección de objetos, configurado para detectar humanos cerca del robot.
- Reglas de comportamiento social configuradas para proporcionar instrucciones cuando se detecten personas cerca al robot.

La necesidad forma parte de un proyecto global enfocado en dar solución a la falta de comunicación entre ciertas personas con dificultades de desplazarse. También

pretende abordar la falta de robots con comportamiento social dentro de la Universidad de las Fuerzas Armadas ESPE.

Antecedentes

En la actualidad el mercado de la robótica ha crecido con gran rapidez debido a la amplia cantidad de campos en los cuales los robots han tenido acogida exigiendo nuevos diseños y mayores funcionalidades. El campo de robots para relaciones públicas ha sido uno de los sectores con mayor crecimiento como lo muestra (International Federation of Robotics, 2018). Donde se vendieron un total de 10.400 unidades en el año del 2017 siendo un 56% mayor al número de ventas respecto al anterior año donde la mayoría fueron robots de telepresencia para usos públicos. De igual manera el incremento de robots para uso doméstico aumentó un total de 25%. El mercado de la robótica ha crecido con mayor fuerza en Norte América, Asia y Europa e incluso en algunos países de latino América como lo ejemplifica (IFR Mobile Robots Market, s. f.).Por lo cual el desarrollo de robots móviles que sean capaces de realizar navegación reactiva, telepresencia, monitoreo y presenten mayores beneficios con la interacción social, ha crecido.

Durante los últimos años ha existido un gran avance dentro las investigaciones de robótica móvil. En los cuales se han aplicado diversas configuraciones, técnicas e implementaciones en robots móviles como se muestra en los siguientes proyectos:

a. El proyecto TERESA (Telepresence Reinforcement Learning Social Agent) desarrollado por la Universidad de Oxford el cual es un robot móvil social de telepresencia que permite el control remoto y supervisión de un operador de forma remota donde el robot se encuentra dentro de un entorno inteligente enfocado a la interacción humano máquina, como se muestra en la Figura 1.

Figura 1

Robot móvil diferencial TERESA dentro de conversación entre personas

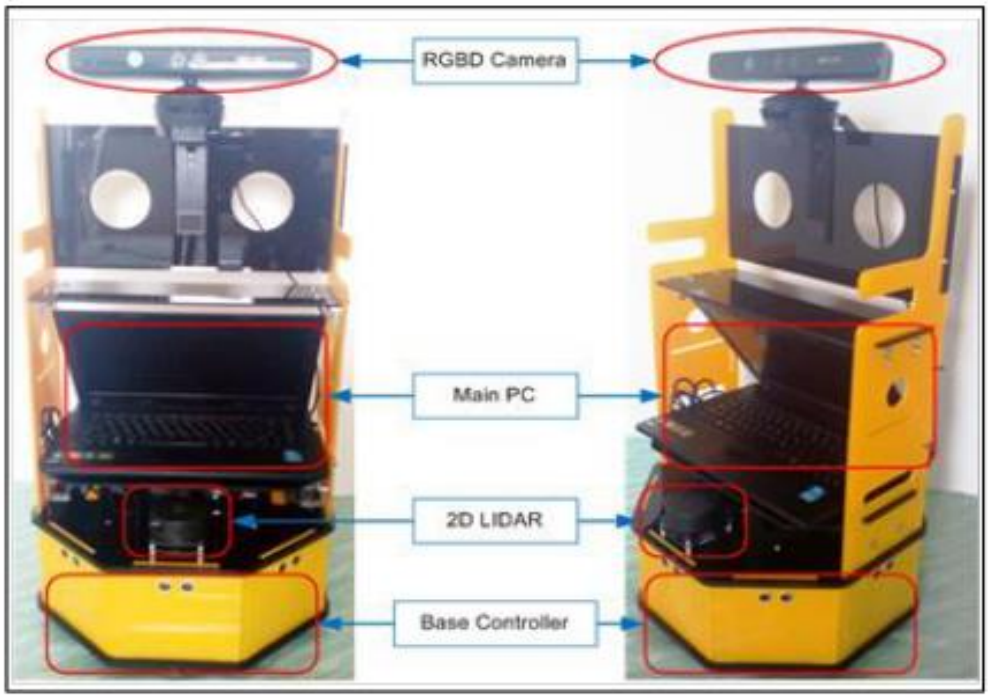
Nota: Tomando de *Social robot TERESA, (Oxford, 2017).*

El robot está compuesto de una pantalla para interacción con el cuidador, sensores y botones para interactuar con el mismo. Gracias a este robot se ha llevado a cabo múltiples publicaciones de investigación dentro de los campos de navegación, detección de objetos, navegación e interacción social, entre otros.

b. El proyecto realizado por (Priyandoko et al., 2018) consiste en una plataforma móvil capaz de seguir la las personas mediante detección de gestos utilizando un sistema de visión artificial potenciado a través de un sensor Kinect de Microsoft. Como se puede observar en la imagen este robot está compuesto por el sensor RGB-D, un computador como controlador general, un sensor lidar 2D y la base del controlador. El robot se muestra en la Figura 2.

Figura 2

Composición general del robot de seguimiento mediante gestos

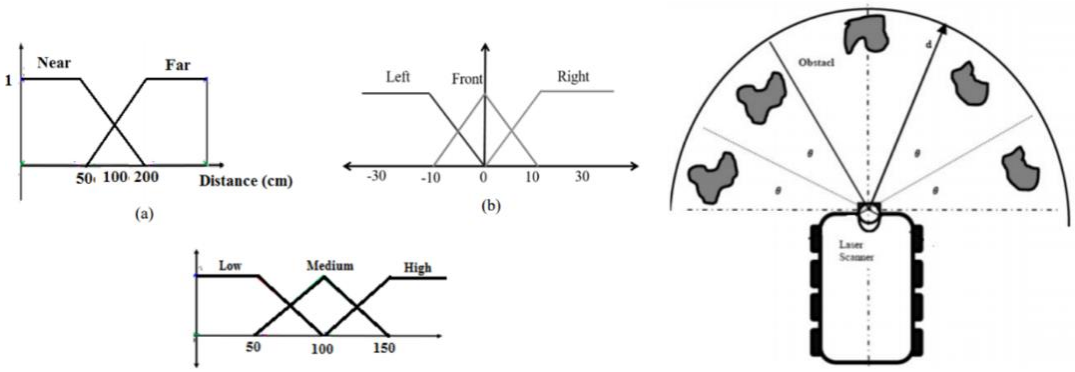


Nota: Tomado de *Human following on ROS framework,* (Priyandoko et al., 2018).

c. En el proyecto mostrado en (Najim Al-Din, 2017) se trata de la construcción e implementación mediante navegación reactiva de un robot móvil. El cual en conjunto con el uso de un controlador fuzzy y una red de sensores permitió una implementación de navegación autónoma. Se plantea un punto de partida y uno de llegada siendo esta la orden de navegación que cumplirá el robot a pesar de perturbaciones externar (objetos que obstruyen el paso). Para la creación de este control se implementaron las reglas mostradas en la Figura 3 y se las puso en práctica dentro de un entorno cerrado.

Figura 3

Entradas al controlador utilizados para control de distancias y entradas de ángulo para configuración del controlador fuzzy de velocidad del robot



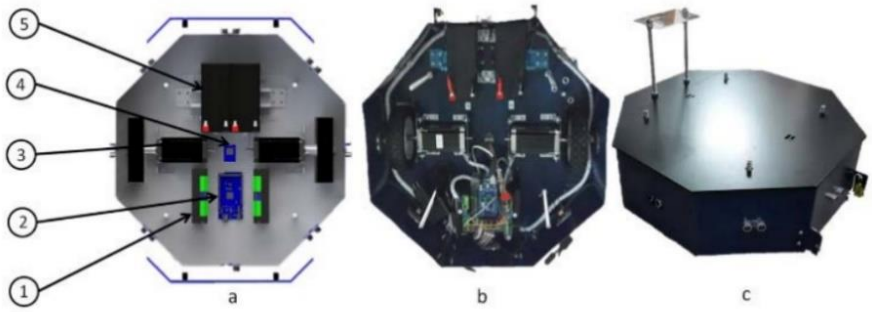
Nota: Tomado de *Reactive Mobile Robot Navigation using Fuzzy controller, (Najim Al-Din, 2017).*

Con la finalidad de mantener la línea de investigación dentro del país se toma como referencia los siguientes proyectos desarrollados en el Ecuador. Los proyectos fueron desarrollados durante los últimos años siendo en su mayoría proyectos de planificación de rutas, navegación mediante simulación o modelamiento matemático del robot. Por lo que dentro de los proyectos relacionados a navegación e implementación en robots móviles reales se tiene los siguientes ejemplos:

a. En el primer proyecto realizado por (Gaona et al., 2016) en la Universidad de las Fuerzas Armadas ESPE se muestra el diseño y construcción de un prototipo de robot móvil de configuración diferencial. El cual es capaz de ser teleoperado además de presentar sensores para odometría y navegación. El prototipo se encuentra realizado mediante materiales de bajo costo por lo que presenta un control básico implementado en el sistema operativo para robots (ROS) este prototipo se muestra en la Figura 4.

Figura 4

Representación del prototipo implementado mediante elementos de bajo costo

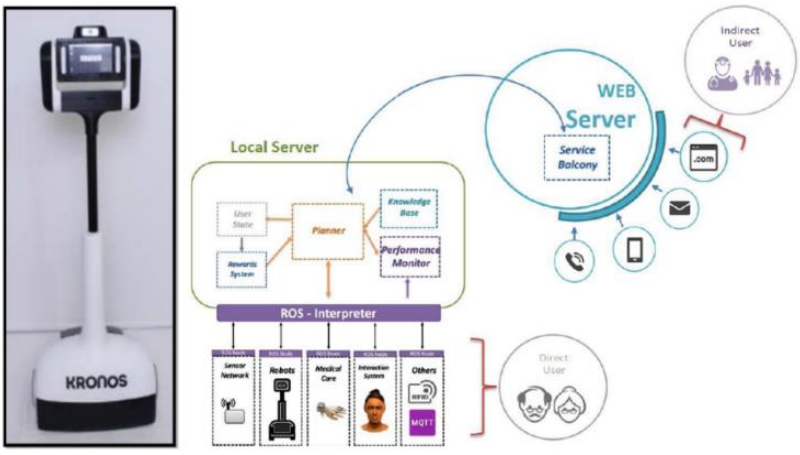


Nota: Tomado de *Diseño y construcción de una plataforma móvil para interiores capaz de realizar SLAM,* (Gaona et al., 2016).

b. El proyecto realizado por (Loza-Matovelle et al., 2019) consiste en la integración entre un robot móvil de telepresencia y una serie de sensores. Como se puede observar en Figura 5, el proyecto está orientado a la conexión entre IoT y robótica móvil el cuidado de personas mayores.

Figura 5

Robot de telepresencia KRONOS para cuidado de personas mayores



Nota: Tomado de *An architecture for the integration of robots and sensors for the care of the elderly in an Ambient Assisted Living Environment*, (Loza-Matovelle et al., 2019).

c. El proyecto realizado por (Quinaluisa & Toapanta, 2018) en la Universidad de las Fuerzas Armadas ESPE plantea la creación de un robot móvil SIMBA. El cual busca implementar un sistema de navegación reactiva mediante el uso de una red de sensores y utilización de SLAM dentro de un entorno controlado. El robot posee una configuración diferencial con dos ruedas tradicionales en el eje y una de tipo “castor” en el frente como se muestra en la Figura 6.

Figura 6

Robot móvil SIMBA para navegación reactiva

Nota: Tomado de *Implementación de un sistema de navegación reactiva autónoma basada en SLAM,* (Quinaluisa & Toapanta, 2018).

Justificación e importancia

Investigaciones previas sobre telepresencia han abordado los problemas de la interacción entre humanos en las áreas remotas. Las aplicaciones relacionadas con la comunicación se han centrado en conferencias multimedia, telemedicina y teleducación. Además, los robots de telepresencia se han desarrollado para superar la restricción de la posición fija del sistema de telepresencia y dar sentido de espacio físico para una mejor interacción. A diferencia de los dispositivos móviles (tabletas, smartphones) que no permiten esa capacidad de interacción y sentido de espacio compartido. Muchos

robots se han desarrollado para diversos fines, por ejemplo, asistencia o comunicación para el cuidado domiciliario de los ancianos (Tsai et al., 2007).

Por otro lado, la implementación de navegación reactiva-social en un robot le permite mayor robustez y flexibilidad. Como menciona (Arkin, 1990) la incorporación de diversos sensores ofrece más información al robot para poder tomar decisiones en entornos variables y con la presencia de personas. La integración de los sistemas de telepresencia y navegación reactiva-social en un robot proporcionarán diversas ventajas en diferentes campos. En el campo turístico, permitirá la supervisión y guía sin necesidad que el personal se desplace de un lugar a otro (Faber et al., 2009). En el campo de seguridad, será posible reconocer información del comportamiento de las personas (Andreasson et al., 2007) y finalmente los usuarios que empleen el robot para el cuidado de personas dependientes pueden hacer uso de este sistema (Ma & Quek, 2010). En todas las aplicaciones el robot podría interactuar con las personas, evitando problemas de la navegación social, como colisiones. incomodidad o posibles daños a las personas.

Como se mostró, la integración de navegación reactiva y telepresencia enfocado en el comportamiento social del robot es de utilidad en diferentes áreas de aplicación. Dado que, en el país, específicamente en la Universidad de las Fuerzas Armadas ESPE, no hay evidencia de la integración de navegación reactiva, telepresencia y comportamiento social en un robot. Se propone el diseño e implementación de un sistema de telepresencia y navegación reactiva-social para el prototipo de robot móvil del laboratorio de mecatrónica y sistemas dinámicos.

El sistema solventará el problema de comportamiento social del robot. Tomando en cuenta la navegación en un entorno cerrado y controlado, logrando evitar colisiones y siguiendo reglas sociales en la interacción con personas. Además, el proyecto

pretende establecer el sentido del espacio compartido a personas en ubicaciones separadas, a través del sistema de telepresencia.

Alcance del proyecto

En el proyecto se propone el diseño e implementación de un sistema de navegación reactiva-social y telepresencia en el prototipo de robot móvil de configuración diferencial (Andrango, 2020). Que se encuentra en el laboratorio de mecatrónica y sistemas dinámicos de la Universidad de las Fuerzas Armadas ESPE.

Partiendo de estos antecedentes se realizará una evaluación del estado y funcionalidad del prototipo de robot diferencial y ejecutar las modificaciones o correcciones necesarias sobre el mismo para orientarlo a un funcionamiento mediante navegación reactiva - social y telepresencia. El robot en el cual se realizará el proyecto lleva a cabo un mapeo y localización de su entorno controlado, mediante SLAM (Simultaneous Localization and Mapping). El prototipo inicial cuenta con los siguientes sensores:

- LIDAR (Light Detection and Ranging).
- Kinect.
- Encoders (incorporados en los motores de cada rueda).
- Acelerómetro, Giroscopio IMU MPU 6050.

Como actuadores tiene únicamente a los motores de cada rueda. Para la recepción y envío de las señales a cada sensor y actuador tiene una tarjeta de adquisición Arduino Mega. El Arduino envía y recibe los datos a una computadora Intel NUC i3, incorporada en el robot que cumple la función de procesador y controlador de todo el sistema. Para dar autonomía energética, el robot integra una batería recargable.

La programación del prototipo inicial está desarrollada en ROS (Robot Operating System) (Quigley et al., 2009). Por lo tanto, la mayor parte del sistema propuesto será

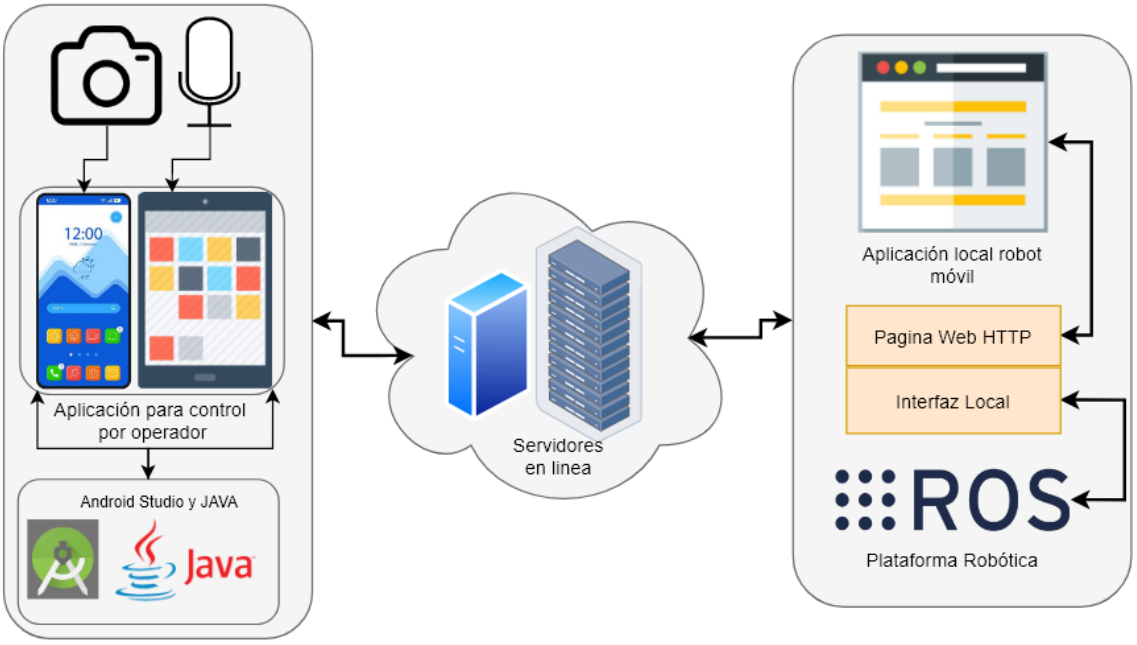
desarrollada en ROS para los parámetros de navegación y niveles de control para el robot diferencial. Después de la evaluación y corrección de prototipo inicial, se implementará una serie de sensores ultrasónicos colocados en una configuración de anillo en el cuerpo del robot. Los sensores en conjunto con el sensor lidar le permitirán al robot realizar navegación reactiva al detectar objetos y modificar su ruta, lo que se implementará dentro de un entorno controlado.

La navegación social se facilitará mediante la detección de personas para modificar el comportamiento del robot. Y por medio de gestos específicos, capturados con el sensor Kinect, se realizará el seguimiento de la persona detectada. Para la navegación social se tomará en cuenta las reglas de comportamiento social aplicadas a la robótica (Pinter et al., 2017).

Para la implementación de la navegación reactiva-social se basará en las patentes (Herzog et al., 2012; Pinter et al., 2017; Yulum et al., 2014). En (Yulum et al., 2014) y (Herzog et al., 2012) se hace referencia a una serie de parámetros necesarios dentro de la interfaz de control presentada al operador, para generar una herramienta que permita el monitoreo y la teleoperación del robot diferencial de forma simplificada mediante botones y gráficos. Mientras que en (Pinter et al., 2017) se muestra los parámetros necesarios para poder implementar una navegación social, donde el robot deberá incluir al menos un sistema de manejo, un sistema de control y un sistema de detección de objetos para poder seccionar el comportamiento del robot en dos grupos de reglas. Primero cuando se encuentre realizando una navegación reactiva sin la presencia de alguna persona y el segundo cuando la presencia de una o más personas sea detectada. En adición, se recomienda mantener en cuenta protocolos de interacción social humana (espacio personal, circulación segura a baja velocidad), para mantener una armonía durante la navegación del robot.

Figura 7

Arquitectura del sistema de telepresencia



Por último, para el desarrollo del sistema de telepresencia se contará con dos clientes y un servidor. El primer cliente está formado por una aplicación móvil implementada en un dispositivo inteligente (smartphone o tablet con sistema operativo Android) en el cual se presentará una serie de controles y botones que permitirán manejar de forma remota el robot, junto con la visualización del entorno en el cual se encuentra el prototipo. El segundo cliente será una interfaz local incorporada en el robot en el cual se podrá visualizar al teleoperador. El sistema estará compuesto por un servidor en internet, que obtendrá las transmisiones de video de ambos clientes y la serie de comandos generados por el operador para implementar la telepresencia y teleoperación. La arquitectura de este sistema de comunicación se muestra en la Figura 7.

Objetivos

Objetivo general

- Implementar un sistema de navegación reactiva-social y telepresencia en un prototipo de robot móvil diferencial.

Objetivos específicos

- Evaluar el estado del prototipo inicial y corregir posibles errores que este contenga.
- Implementar una red de sensores en el robot móvil para obtener la información del entorno controlado.
- Implementar el control de movimiento del robot en conjunto con un sistema de detección local que permita la navegación reactiva.
- Establecer la comunicación entre el servidor y los clientes para la implementación del sistema de telepresencia y teleoperación.
- Configurar la comunicación para la teleoperación del robot.
- Desarrollar una aplicación web que permita obtener la transmisión de video y operador para su visualización en el robot móvil.
- Desarrollar una aplicación móvil que permita controlar los movimientos del robot y obtener la transmisión de video del entorno del robot.
- Implementar el comportamiento social del robot mediante la evaluación del entorno y reglas sociales enfocadas a robótica.

Estructura del documento

El documento propone una estructura en seis capítulos donde se mostrará y describirá toda la información y desarrollo del trabajo a realizar en la plataforma robótica móvil dentro de un entorno controlado.

En el capítulo 1 se presenta las generalidades y precedentes que se tiene como antecedentes del proyecto, se definen los objetivos, la justificación del trabajo al igual que el alcance de este. El capítulo 2 muestra una visión general en cuanto al campo de la robótica móvil y conceptualización, en conjunto con el estado del arte respecto a investigaciones de los sistemas sobre navegación, telepresencia y comportamiento social.

En el capítulo 3 se presenta los subsistemas propuestos para el proyecto, así como las alternativas y selecciones de elementos de hardware y algoritmos, como solución para cada uno de los subsistemas. En el capítulo 4 se presenta el desarrollo del trabajo, además se describe la implementación del sistema propuesto. Las pruebas y resultados obtenidos con diferentes variantes del entorno del robot se presentan en el capítulo 5. Finalmente, en el capítulo 6 se detalla las conclusiones y recomendaciones obtenidas del trabajo realizado.

Capítulo II

Marco teórico

El capítulo presenta conceptos básicos de la robótica móvil y aplicaciones. También se menciona la clasificación en la robótica móvil y se analiza en especial los robots terrestres por ruedas. Se presenta los fundamentos de navegación y localización junto con las técnicas utilizadas en últimos proyectos. Así como el sistema de percepción necesario en un robot móvil. Se plantea un marco referencial para la navegación reactiva – social, incluyendo el comportamiento social y algunas estrategias de navegación utilizadas en la actualidad. Además, se incorpora el estado del arte sobre la telepresencia en robots móviles y estrategias para su implementación. Por último, se presenta algunos robots en el campo de aplicación que se implementará el sistema propuesto junto con una tabla comparativa con sus características principales.

Robótica móvil

En la clasificación de la robótica móvil se destaca a los robots de ruedas como la configuración más utilizada. “Esta rama es un campo de desarrollo e investigación de robots que poseen un sistema de movimiento propio, sean reprogramables e incluyan un sistema de control autónomo para poder llevar a cabo una determinada tarea dentro de un entorno específico” (Klančar et al., 2017). En el creciente desarrollo de la tecnología, la utilización de los robots móviles ha aumentado teniendo aplicaciones en diversos campos. Como lo muestra (Loza-Matovelle et al., 2019) la aplicación de la robótica móvil permitió un gran avance en campos como: manipulación y transporte de material en la industria, servicios de mantenimiento, exploración y reconocimiento, campo de construcción y servicios sociales.

En conjunto con el desarrollo de plataformas móviles dentro de este campo de investigación los principales temas de estudio en la actualidad según (Lazea & Lupu,

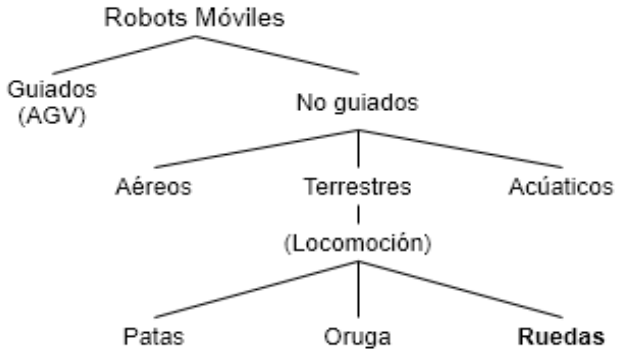
1997) están conformados por planeamiento de rutas y trayectorias, mapeo del entorno, estimación de posición y localización, evasión de obstáculos, implementación de sistemas de censado y control distribuido en tiempo real.

Clasificación de los robots móviles

Existe una gran diversidad de robots móviles. La Figura 8 ilustra una división basada en las características del vehículo. Se hace una primera distinción importante entre vehículos guiados y no guiados. Un vehículo guiado está restringido a un conjunto de caminos predefinidos en su área de trabajo. Estas rutas pueden indicarse mediante pistas, líneas magnéticas u ópticas, rutas de guía inductivas o una secuencia de movimientos almacenados en la memoria. El vehículo puede abandonar este camino en ninguna circunstancia. Los AGV (Vehículo de Guiado Automático) forman esta clase. Los vehículos no guiados no están restringidos a ninguna ruta guía predefinida.

Figura 8

Clasificación de los robots móviles



Nota: Tomado de *Aspects on path planning for mobile robots,* (Lazea & Lupu, 1997).

Los robots móviles terrestres se clasifican por el tipo de locomoción utilizado siendo estos mediante ruedas, por patas y orugas (Silva Ortigoza et al., 2007). Se considera que los robots móviles con ruedas tienen mayor facilidad de construcción y alta eficiencia en terrenos planos. Además, el equilibrio no suele ser un problema de

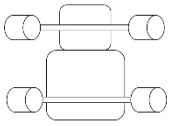
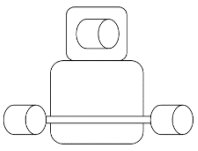
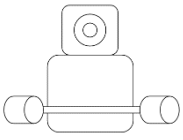
investigación en los diseños de robots con ruedas, porque los robots con ruedas casi siempre están diseñados para que todas las ruedas estén en contacto con el suelo en todo momento (Siegwart & Nourbakhsh, 2011).

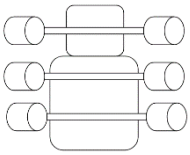
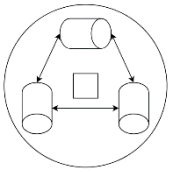
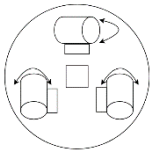
Robot móvil impulsado por ruedas

Existen diferentes configuraciones cinemáticas para los robots móviles con ruedas, estas dependen de la aplicación. De manera general se tienen las siguientes configuraciones: Ackerman, triciclo clásico, tracción diferencial, skid steer, síncrona y tracción omnidireccional (Ollero, 2001; Silva Ortigoza et al., 2007). Se presentan las diferentes configuraciones y su descripción en la Tabla 1.

Tabla 1

Configuraciones cinemáticas de los robots móviles con ruedas

Configuración	Descripción	Ejemplo
Ackerman 	Consta de dos ruedas motorizadas en la parte trasera, 2 ruedas de dirección en la parte delantera; la dirección debe ser diferente para las 2 ruedas para evitar resbalones.	Automóvil con tracción trasera
Triciclo clásico 	Consta de una rueda delantera que sirve para tracción y direccionamiento. El eje trasero, con dos ruedas laterales, es pasivo y sus ruedas se mueven libremente. Tiene gran movilidad, pero presenta problemas de estabilidad.	Neptune (Carnegie Mellon University)
Diferencial 	El direccionamiento y la tracción se consigue con la diferencia de velocidades de las ruedas laterales. Además, existe una o más ruedas para soporte. Es frecuente para robots para interiores	Hyperbot Chip

Configuración	Descripción	Ejemplo
Skid steer 	Se dispone varias ruedas en cada lado que actúan de forma simultánea. El movimiento resulta de combinar las velocidades de las ruedas de la izquierda y derecha.	Terregator
Síncrona 	Consiste en la acción simultánea de todas las ruedas, que giran de forma síncrona. La transmisión se consigue mediante coronas de engranaje o correas concéntricas.	RAM1
Omnidireccional 	Consta generalmente de tres o cuatro ruedas que permiten el movimiento omnidireccional del vehículo.	Carnegie Mellon Uranus

Nota: Tomado de *Introduction to autonomous mobile robots,* (Siegwart & Nourbakhsh, 2011).

De las configuraciones mostradas se considera que la configuración diferencial es la de mayor facilidad de implementar, debido a que su diseño minimiza el sistema de control a utilizar. Además, los existentes modelos cinemáticos proporcionan trayectorias perfectamente definidas lo que permite su uso en muchas de aplicaciones.

Robot móvil de configuración diferencial

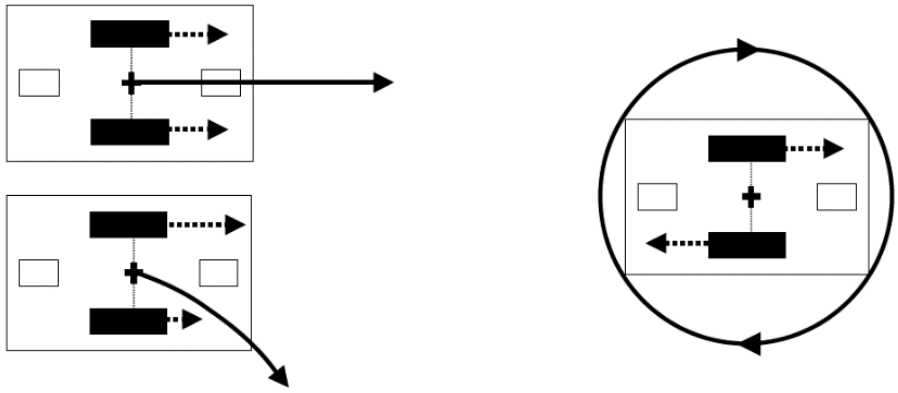
El robot móvil con ruedas de configuración diferencial consta de dos ruedas montadas, en un único eje, que son independientemente propulsadas y controladas, proporcionando tracción y direccionamiento (Bañó, 2003). La tracción y direccionamiento viene dado por la diferencia de velocidades de las ruedas laterales.

Cinemática del robot móvil diferencial

La cinemática del robot móvil diferencial pretende analizar la geometría del robot sin considerar las fuerzas que intervienen en los movimientos. El desplazamiento diferencial del robot se realiza mediante dos ruedas motrices, y ruedas pasivas para mantener el equilibrio (Bräunl, 2008). En la Figura 9 se observa el movimiento de conducción del robot móvil diferencial. Si ambos motores (ruedas motrices) funcionan a la misma velocidad, el robot se mueve hacia adelante o hacia atrás, si un motor funciona más rápido que el otro, el robot se mueve en una curva a lo largo del arco de un círculo, y si ambos motores funcionan a la misma velocidad en direcciones opuestas, el robot gira en el lugar.

Figura 9

Manejo y rotación del robot móvil diferencial



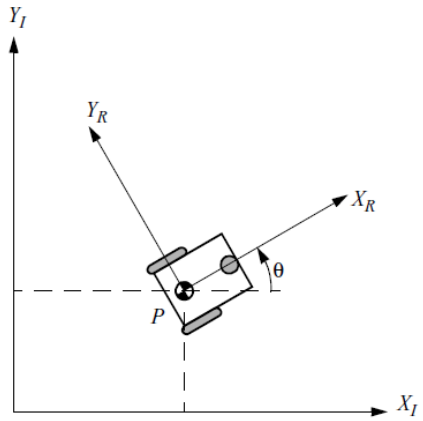
Nota: Tomado de *Embedded robotics (third edition): Mobile robot design and applications with embedded systems,* (Bräunl, 2008).

El movimiento del robot se analiza respecto a un plano horizontal fijo (X_1, Y_1) y un plano móvil (X_R, Y_R). El plano fijo se utiliza como referencia, llamado también plano global; y al plano móvil se lo llama también plano local (Siegwart & Nourbakhsh, 2011). En la Figura 10 los ejes permiten especificar matemáticamente la posición en el punto P, que corresponde al centro de giro de la configuración diferencial del robot. El punto P

se representa en el plano global por las coordenadas *x* e *y,* y la orientación se expresa por el ángulo θ, siendo este la diferencia entre el plano global y el plano local.

Figura 10

Plano global y local para la representación cinemática del robot



Nota: Tomado de *Introduction to autonomous mobile robots, (Siegwart & Nourbakhsh, 2011)*.

Navegación y localización

La navegación es la capacidad del robot para actuar en función de sus conocimientos y los valores de los sensores para alcanzar sus posiciones de objetivo de la manera más eficiente y confiable posible (Siegwart & Nourbakhsh, 2011). El éxito en la navegación requiere gestionar los cuatro componentes básicos de la navegación:

- Percepción del entorno.
- Posición en el entorno o localización.
- Cognición para decidir cómo actuar para lograr sus objetivos.
- Control de movimiento modulando sus salidas.

Por lo que, la localización es una de las tareas más importantes durante la navegación. Mediante la utilización de sensores se estima la posición y orientación relativa del robot junto con la elaboración de un mapa de su entorno para poder conocer

su posición. Además, cuando el robot se implementa en un entorno social es esencial poder conocer la posición de las personas con respecto al robot (Nehmzow, 2000; Siegwart & Nourbakhsh, 2011). Existen dos grupos de estrategias principales usadas en localización los cuales son: la localización relativa o local y la localización absoluta o global. Una alternativa utilizada en la actualidad es la utilización de técnicas de probabilidad para poder obtener una estimación por ambos métodos (González & Rodríguez, 2009).

Localización relativa

La localización absoluta es un conjunto de técnicas que determinan la posición del robot con respecto a un marco de referencia global. La metodología más conocida es la de GPS (Global Position System) que se basa en señales satelitales que otorgan la longitud, latitud y altitud de un objeto. En estos casos la posición del robot no depende del tiempo y de la posición inicial, por lo cual el error se mitiga cuando las mediciones están disponibles. Los principales problemas que presentan según (Bailey & Durrant-Whyte, 2006; Wang, 1988) son:

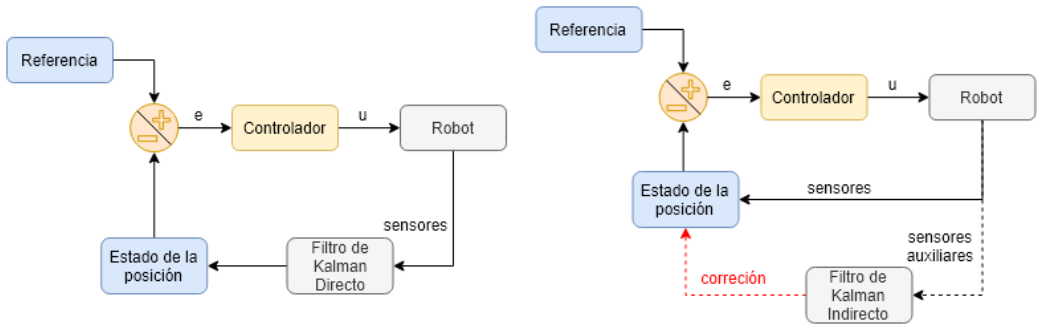
- Requiere de una costosa instalación de los marcadores en el área donde opera el robot.
- El robot móvil solo podrá navegar en el área en la que se encuentran los puntos de referencia.
- Entre puntos de referencia no se puede determinar la localización.
- Existe poca precisión en obtención de datos para plataformas móviles de pequeño tamaño.
- Posee un tiempo de muestreo relativamente grande en la mayoría de los casos mayor a 1 segundo.
- La señal de GPS se pierde dentro de espacios cerrados.

Localización probabilística

Las técnicas probabilísticas se basan en la estimación de la localización del robot móvil, utilizando datos de medición previos y conocimientos del entorno, disminuyendo el error obtenido. El método más utilizado es la técnica basada en el filtro de Kalman (Welch & Bishop, 2001). El cual es un algoritmo de procesamiento de datos recursivo utilizado para sistemas lineales. Se asume una distribución gaussiana y posee dos configuraciones posibles la directa e indirecta que utilizan la siguiente estructura mostrada en la Figura 11.

Figura 11

Estructuras utilizadas para filtros de Kalman directos e indirectos



Nota: Tomado de *A comparative study on encoding methods of local binary patterns for image segmentation,* (Wu et al., 2019).

No obstante, del gran uso del filtro de Kalman muchos de los problemas y sistemas son no lineales por lo cual existen modificaciones del método. Por ejemplo el EKF (Extended Kalman Filter) como se muestra en (Bailey & Durrant-Whyte, 2006), donde realiza una aproximación lineal de la respuesta del sistema usando la derivada en cada uno de los puntos de la función. Debido a esto se utiliza el jacobiano como se muestra en las siguientes ecuaciones (Cai & Zhao, 2006):

Donde, tomando en cuenta que el modelo del sistema y su jacobiano se representa por:

$$Sistema: \begin{cases} x_k = f(x_{k-1}, u_k) + w_k \\ y_k = g(x_k) + u_k \end{cases} \quad Jacobiano: \begin{cases} F = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1}, u_k} \\ G = \left.\frac{\partial g}{\partial x}\right|_{\hat{x}_k} \end{cases}$$

Obteniendo la linealización del sistema con la siguiente expresión:

$$Aproximación\ Lineal: \begin{cases} \Delta x_k \approx F\Delta x_{k-1} + w_k \\ \Delta y_k \approx G\Delta x_k + u_k \end{cases}$$

Una vez obtenida la aproximación lineal se puede aplicar el método de filtro de Kalman. A pesar de esto, los EKF poseen una serie de inconvenientes para su aplicación siendo estos:

- Los jacobianos pueden presentar un cálculo complejo.
- En caso de sistemas grandes presenta un gran costo computacional.
- EKF pueden trabajar únicamente con sistemas que poseen un modelo diferencial.
- EKF no es óptimo para sistemas no lineales de gran grado.

Otro algoritmo de localización es el filtro de partículas. Como se muestra en (Yatim & Buniyamin, 2015), el método consiste en la aproximación de expresiones matemáticas de alta complejidad con gran exactitud. Es decir, se intenta aproximar la distribución de la probabilidad mediante una distribución de partículas en el espacio de estados (Gallardo, 1999). Se utiliza la técnica del filtro de partículas cuando existen sistemas no lineales de gran tamaño y el método de linealización mediante derivadas no funciona. El principal objetivo es realizar un “seguimiento” sobre una variable de interés a través del tiempo. Se basa en construir una muestra de la PDF (Probability Density Function) compuesta por “partículas” cada una está compuesta por una posición y un peso. En donde el estimado de la variable de interés se obtiene mediante una suma de todos los pesos de las partículas (Rekleitis, 2003).

El filtro de partículas consta de dos partes la “predicción” y la “actualización”. La predicción se encuentra después de cada evento de cambio en donde las partículas se modifican de acuerdo con el modelo matemático del sistema y la segunda ocurre cuando se actualizan los pesos de cada partícula en función de los valores obtenidos en la nueva medición de su posición. Un algoritmo basado en el filtro de partículas es AMCL (Adaptative Monte Carlo Localization).

AMCL

El método de Monte Carlo MCL utiliza técnicas de muestreo rápido para representar la estimación de la posición del robot (Rubin, 1988). Cuando el robot se mueve o realiza medición del entorno, se aplica un nuevo muestreo de importancia para estimar la distribución posterior.

La idea principal del método es representar la estimación posterior *Bel(l)* mediante un conjunto de *N* muestras o partículas aleatorias ponderadas como:

$$S = \{s_i | i = 1 \dots N\}$$

Un conjunto de muestras constituye una aproximación discreta de una distribución de probabilidad. Las muestras en MCL son del tipo:

$$\langle \langle x, y, \theta \rangle, p \rangle$$

Donde $\langle x, y, \theta \rangle$ denota una posición del robot, y $p \geq 0$ es un factor de ponderación numérico, análogo a una probabilidad discreta. Se asume que:

$$\sum_{n=1}^{N} p_n = 1$$

Las dos fases del método son:

- **Movimiento del robot.** Cuando el robot se mueve, MCL genera *N* nuevas muestras que se aproximan a la posición del robot después del comando de

movimiento. Cada muestra se genera extrayendo aleatoriamente un grupo de datos del conjunto de muestras previamente calculado, con la probabilidad determinada por sus valores *p*. Conociendo que *l'* denota la posición de la muestra. El nuevo valor de *l* se genera con una muestra aleatoria única a partir de una probabilidad.

- **Lectura de sensores.** La información obtenida de los sensores se incorpora volviendo a ponderar el conjunto de muestras, de manera que implemente la regla de Bayes. Específicamente, se tiene que $\langle l, p \rangle$ es una muestra. Luego:

$$p \leftarrow \alpha \, P(s|l)$$

Donde *s* es la medición del sensor, y α es una constante de normalización para que la suma de los valores de *p* sea igual a 1. La incorporación de las lecturas del sensor se realiza típicamente en dos fases, una en la que *p* se multiplica por *P(s|l)* y otra en la que se normalizan los diversos valores de *p* (Fox et al., 1999).

La ventaja del método radica en la forma en que MCL coloca los recursos computacionales. Al muestrear en proporción a la probabilidad, MCL enfoca sus recursos computacionales en regiones con alta probabilidad. En la práctica, el número de muestras requerido para lograr una buena predicción varia drásticamente. Durante la localización global el robot desconoce su posición por lo tanto la estimación debe cubrir todo el espacio de estado. Pero durante el seguimiento de la posición el número de muestras se enfoca en un campo dimensional pequeño. Por lo tanto, se requiere muchas más muestras durante la localización global para aproximar la densidad con alta precisión, que las requeridas para el seguimiento de posición.

En el algoritmo AMCL se emplea un esquema de muestreo adaptativo (Koller & Fratkina, 1998). La idea es utilizar la divergencia de *P(l) y P (l*|s*)*, la estimación antes y después de la detección, para determinar los conjuntos de muestras. Específicamente,

los datos de movimiento y los datos del sensor se incorporan en un solo paso, y el muestreo se detiene cada vez que la suma de los pesos *p* supera un umbral. Si la posición pronosticada por la odometría es similar con la lectura del sensor, cada *p* individual es grande y el conjunto de muestra permanece pequeño.

Como resultado, AMCL utiliza muchas muestras durante la localización global cuando más se necesitan, mientras que el tamaño del conjunto de muestras es pequeño durante el seguimiento, cuando se conoce aproximadamente la posición del robot.

Percepción en un robot móvil

La tarea de mayor importancia en un sistema autónomo es adquirir conocimiento sobre su entorno. La información se adquiere tomando medidas usando varios sensores y luego extrayendo datos significativos de esas mediciones (Kucsera, 2006). Los sensores pueden ser clasificados en:

- Propioceptivos que miden valores internos como voltajes de batería, velocidad de la rueda.
- Exteroceptivos que adquieren información del entorno.

En la Tabla 2 presenta los principales sensores utilizados en los robots móviles.

Tabla 2

Clasificación de sensores en robot móviles

Propioceptivos	**Exteroceptivos**
Sensor de batería	Cámara
Sensor de temperatura	Sonar
Encoder	Sensor de distancia infrarrojo
IMU (acelerómetro, giroscopio)	Lidar

Encoder

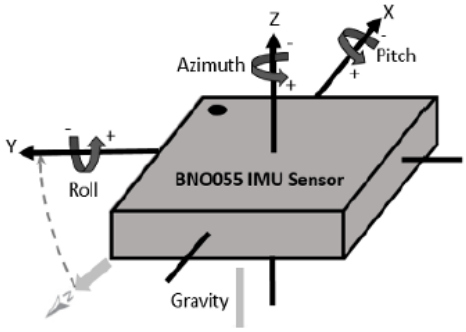
Un encoder óptico consiste en una fuente de iluminación, una rejilla fija que enmascara la luz, un disco de rotor con una rejilla óptica fina que gira con el eje, y detectores ópticos fijos. A medida que el rotor se mueve, la cantidad de luz que golpea los detectores ópticos varía en función de la alineación de las rejillas fijas y móviles. En función de la cantidad de luz que llega a los detectores ópticos se puede medir la velocidad angular y la posición del accionamiento del motor o mecanismo de dirección. Los encoders se utilizan como sensor de retroalimentación fundamental para controlar la posición o la velocidad de las ruedas (Bräunl, 2008; Everett, 1995).

Unidad de medida inercial (IMU)

La unidad de medida inercial se utiliza principalmente en dispositivos para medir la velocidad, la orientación y la fuerza gravitacional. Generalmente consiste en dos tipos de sensores acelerómetros y giroscopios. El acelerómetro se utiliza para medir la aceleración inercial. Mientras que el giroscopio mide la rotación angular. Ambos sensores suelen tener tres grados de libertad para medir desde tres ejes.

Figura 12

Descripción general de un sensor IMU



Nota: Tomado de *Reviews on Various Inertial Measurement Unit (IMU) Sensor Applications*, (Ahmad et al., 2013).

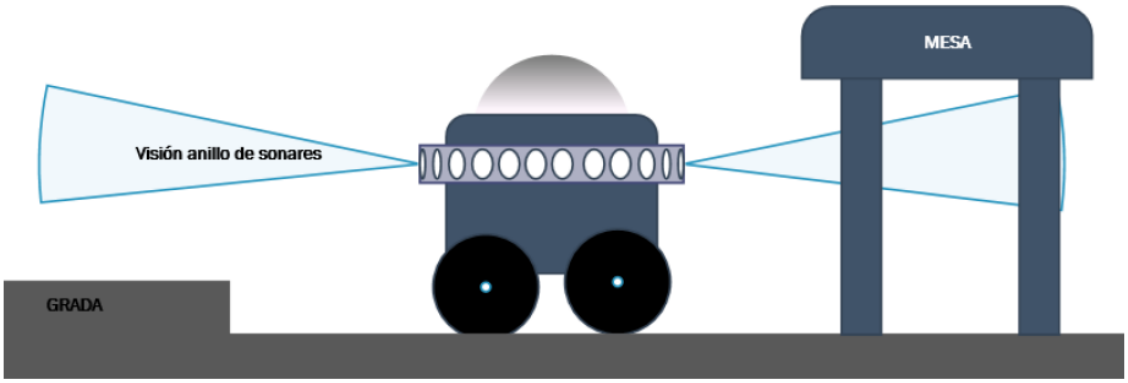
Algunos robots requieren datos lineales y angulares para sus movimientos; por lo tanto, la IMU es un buen sensor para obtener los datos deseados. (Ahmad et al., 2013).

Sonar

Sonar es un sensor utilizado para medir distancia utilizando el siguiente principio: se emite una señal acústica corta de aproximadamente 1 ms a una frecuencia de 50 kHz a 250 kHz y el tiempo se mide desde la emisión de la señal hasta que el eco vuelve al sensor. El tiempo de vuelo medido es proporcional al doble de la distancia del obstáculo más cercano en el cono del sensor. Debido al cono relativamente estrecho de estos sensores, una configuración típica para cubrir toda la circunferencia de un robot redondo requería 24 sensores, mapeando alrededor de 15 grados cada uno (Bräunl, 2008). La mayoría de los entornos cerrados pueden ser navegados exclusivamente por sonares, pero la corta gama de sensores y su tendencia a reflejarse significa que sólo pueden proporcionar datos de navegación confiables en ángulos y distancias limitados.

Figura 13

Aplicación del sonar en un robot móvil



Nota: Adaptado de *Designing mobile autonomous robot*, (Holland, 2004).

Además, los robots de interiores tienen algún tipo de sistema de sonda como parte de su capacidad de prevención de colisiones como presenta el robot “SR3

Security Robot" (Holland, 2004). En la Figura 13 se muestra una aplicación del anillo sonar en un robot móvil. Los problemas más significativos de los sensores son la reflexión y la interferencia. Pero también son un poderoso sistema de sensores, como se puede ver en el gran número de artículos publicados que muestran su aplicación principalmente como sistema anticolisiones (Barshan et al., 2000; Kuc, 2001).

Sensor lidar

Lidar es uno de los sensores más populares disponibles para la navegación en robots debido a las mejoras significativas que logra sobre el sensor ultrasónico, debido al uso de luz láser en lugar de sonido. Este tipo de sensor consiste en un transmisor que ilumina un objetivo con un has colimado y un receptor capaz de detectar el componente de luz que es esencialmente coaxial con el has transmitido. Los dispositivos producen una estimación de rango basada en el tiempo necesario para que la luz llegue al objetivo y regrese. Un mecanismo con un espejo barre el haz de luz para cubrir la escena requerida en un plano o incluso en tres dimensiones, utilizando un espejo giratorio. Las desventajas de los sistemas lidar son su costo, consumo de energía significativo, patrones de detección plana y complejidad mecánica (Siegwart & Nourbakhsh, 2011).

Sensor Kinect

Kinect es un dispositivo que integra una cámara RGB, un sensor de profundidad y una serie de giróscopos que aportan información sobre su orientación. El campo de visión de la cámara es de 57, 77º grados en horizontal y 45º en vertical. El rango de trabajo va desde los 1.2m hasta los 3.5m (Ruiz-Sarmiento et al., 2011). La aplicación inicial del Kinect fue para videojuegos, pero con el desarrollo e investigaciones se ha adaptado para aplicaciones dentro de la robótica móvil como mapeo, navegación, visión artificial, detección y reconocimiento de personas. Para obtener información del Kinect y

desarrollar aplicaciones basadas en interacción se hace uso de OpenNI en Linux. OpenNI es un framework de código abierto que permite la extracción de características del esqueleto como su posición y orientación.

Navegación reactiva – social

La navegación reactiva es una estrategia utilizada en los robótica móvil, que tienen la capacidad de responder a estímulos del entorno al evitar obstáculos (Ruiz-Sarmiento et al., 2011). La navegación reactiva pretende tratar el problema de movimiento autónomo y controlado en interiores como pasillos habitaciones, ingresos, etc. Pero se ha limitado en general en ambientes estáticos. La navegación reactiva se fundamenta en la detección de obstáculos a través de sus sensores, pudiendo ser estos basados en cámaras KINECT (Ruiz-Sarmiento et al., 2011), sonares (Elfes, 1987) o LIDAR (Liu et al., 2014).

Con la integración de robots en aplicaciones de servicio en interiores, se ha incrementado el desafío de la navegación autónoma en un entorno humano (Cheng et al., 2018). Es decir, sobre cómo hacer frente a obstáculos humanos dinámicos. La navegación en un entorno humano es la intersección entre la interacción humano-robot (HRI) y la planificación del movimiento del robot. "La interacción humano-robot (HRI) es el estudio de la dinámica de interacción entre humanos y robots" (Feil-Seifer & Matarić, 2009). Las propiedades que un robot necesita exhibir para navegar de una manera consciente para los humanos se pueden clasificar principalmente en:

- La comodidad es la ausencia de molestias y estrés para los humanos en interacción con los robots.
- La naturalidad es la similitud entre robots y humanos en patrones de comportamiento de bajo nivel.
- La sociabilidad es la afección a convenciones culturales explícitas de alto nivel.

Por lo tanto, es necesario que el robot móvil tenga presente en su navegación ciertas normas del comportamiento en un entorno humano.

Comportamiento social en la robótica

(Pinter et al., 2017), en la patente “Social behavior rules for a medical Telepresence robot”, se describe el comportamiento que debería tener un robot. El cual debe incluir un sistema de: navegación, control, detección de objetos, y reglas de comportamiento social. Las reglas de comportamiento social incluyen entre otras: distancia entre el robot y el humano, decisión para tomar la ruta hacia un punto específico, prioridad frente a colisión entre el humano y otros objetos, velocidad del robot para brindar seguridad y posicionamiento del robot con relación a las personas.

En (Lipman & Hall, 1970) se introduce el término espacio personal virtual llamado “proxemics”. El espacio personal virtual se refiere a la distancia que un robot debe mantener hacia las personas, esencialmente para evitar una incomodidad emocional del humano. En la Tabla 3 se muestran categorías y valores en el contexto de la distancia humano-robot.

Tabla 3

Distancias interpersonales proxémicas

Designación	Especificación	Reservado para
Distancia íntima	0 – 45 cm	Abrazar, susurrar
Distancia personal	45 – 120 cm	Amigos
Distancia social	1.2 – 3.6 m	Conocidos, extraños
Distancia pública	>3.6 m	Hablar en público

Nota: Tomado de *Human-Aware Robot Navigation: A survey*, (Kruse et al., 2013)*.*

Un estudio muestra que la distancia lateral (entre el robot y el humano) de 0.4m es la preferida en corredores. Pues una distancia muy lejana es antinatural (Pacchierotti et al., 2006). Mientras que un robot desplazándose en una habitación con personas a velocidad de 1m/s incomoda a los sujetos, pero a velocidad de 0.5m/s es aceptable

(Butler & Agah, 2001). Además, en (Young et al., 2011), un estudio muestra que las personas prefieren interactuar con el robot posicionado frente a ellos.

El enfoque más común para evitar molestias debido a la pequeña distancia es definir áreas alrededor de los humanos como funciones de costos o campos potenciales (Hansen et al., 2009). Dichas funciones son superiores a las soluciones que definen zonas prohibidas alrededor de los humanos (Huang et al., 2010), porque en espacios confinados puede ser necesario y útil que el robot se mueva muy cerca de una persona. Para planificar un camino suave alrededor de un humano, en (Pandey & Alami, 2010) se presenta un enfoque muy parecido a navegación puramente reactiva, utilizando pautas de proximidad dinámicas.

Además, se muestra que una manera de dar comodidad es no interferir con el humano si no es necesario. Lo cual se ha abordado en (Diego & Arras, 2011) evitando que el robot navegue en las áreas que causan posibles interferencias con otros, mientras realiza tareas como limpiar el hogar. Su enfoque es mantener un “mapa de accesibilidad espacial” que contiene probabilidades de actividades humanas en el entorno durante intervalos de tiempo. Por lo tanto, previas investigaciones de las características de un robot en un entorno social muestran que las principales capacidades, que el sistema del robot debe incluir, son (Kruse et al., 2013):

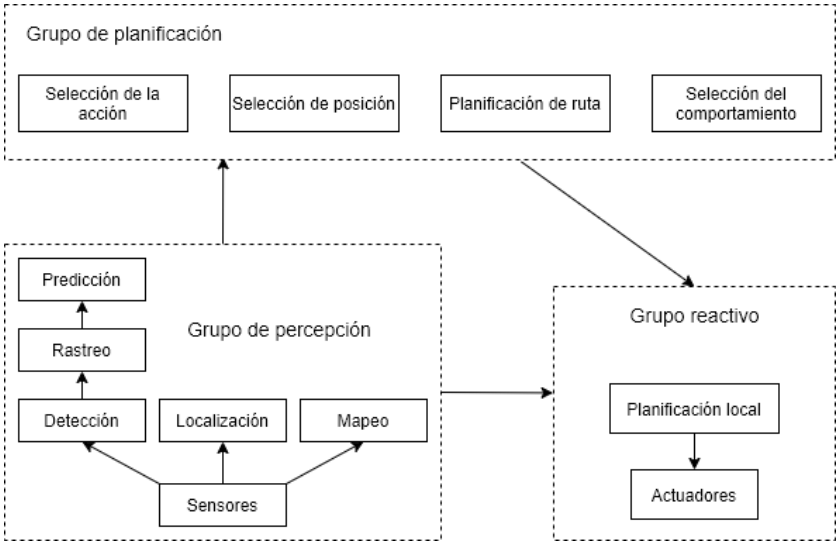
- Respetar las zonas personales.
- Respetar los espacios de accesibilidad.
- Evitar comportamientos despreciados culturalmente.
- Evitar movimientos erráticos o ruidos que causen distracción.
- Reducir la velocidad al acercarse a una persona.
- Enfoque desde el frente para la interacción explícita.
- Modular la dirección de la mirada.

Marco referencial para la navegación social

Un robot necesita principalmente dos componentes: reactivo para extraer características del entorno y premeditado para lidiar con la planificación del movimiento. La combinación de los componentes es llamada “arquitectura híbrida” o “arquitectura por capas” (Siciliano & Khatib, 2016). En la arquitectura por capas, la capa inferior es la reactiva. La capa inferior detecta el entorno y controla el movimiento para evitar situaciones no deseadas como colisiones o grandes desviaciones de un plan. Las capas superiores crean planes para la ruta y el comportamiento del robot.

Figura 14

Posible separación de las funcionalidades en módulos para navegación social



Nota: Adaptado de *Human-Aware Robot Navigation: A survey*, (Kruse et al., 2013).

En la figura Figura 14 se presenta las funcionalidades relevantes de un sistema de navegación en un entorno social (no es una arquitectura de software para navegación). La figura representa las funcionalidades de procesamiento de datos en módulos (recuadros) conectados mediante posibles flujos de datos (flechas). Las funcionalidades son divididas en tres grupos: percepción, planificación y actuación (grupo reactivo). El grupo sensorial se ocupa de generar información sobre el entorno

del robot y su propia posición. Para entornos con obstáculos dinámicos, el comportamiento de navegación del robot se puede mejorar al detectar obstáculos en movimiento, rastrearlos y proyectar su movimiento o intención futura independiente.

En el grupo de planificación, los módulos deciden qué acciones realizar en qué orden, a dónde ir para cada acción, cómo llegar allí y el comportamiento en el camino. Un flujo de control ejemplar es:

- El robot selecciona una acción (por ejemplo, ir hacia un humano).
- El robot selecciona una pose (por ejemplo, frente a un humano).
- El robot selecciona una ruta (por ejemplo, evita a un grupo de personas que hablan entre sí).
- El robot adapta un aspecto del comportamiento mientras se mueve en el camino (por ejemplo, reduciendo la velocidad para evitar el ruido cerca de las personas).

Por lo tanto, la selección de pose generalmente califica las posiciones en el espacio con respecto a la idoneidad para acciones determinadas. La planificación de rutas trata de encontrar una solución válida desde una posición actual a una posición de meta. También se llama planificación global. La planificación del comportamiento se ocupa de las modulaciones de los movimientos.

Cuando un robot se mueve, tiene la libertad de modular aspectos de comportamiento de bajo nivel, eligiendo diferentes velocidades, moviéndose de lado, haciendo una pausa, usando gestos o sonidos como señales. La planificación de la acción y la planificación del comportamiento son diferentes, ya que las primeras deciden sobre el "si" y el "qué" de los movimientos, y la segunda sobre el "cómo".

El grupo reactivo contiene todos los algoritmos de control que envían comandos a los motores del robot. Este grupo mantiene la seguridad y es responsable de producir movimientos suaves. El módulo de planificación local solo planifica una cierta distancia

o tiempo por delante. De modo que se pueda garantizar que un resultado esté disponible a tiempo mientras el robot se mueve a velocidades más altas entre los obstáculos en movimiento.

Estrategias de navegación

Se divide las estrategias de navegación social en cuatro grupos como lo propone (Cheng et al., 2018). Se clasifica en las siguientes estrategias:

- Planificadores reactivos.
- Planificadores predictivos.
- Estrategias basadas en modelos.
- Estrategias basadas en aprendizaje.

Planificadores reactivos

Un método sencillo, fácil de implementar y aplicar en situaciones reales es tener una trayectoria hacia el objetivo y si hay un obstáculo en su camino, simplemente evítelo como se ilustra en la Figura 15(a). El planificador reactivo puede garantizar la seguridad física de las personas cercanas ya que considera todos los casos de colisión. Pero puede causar molestias por su comportamiento antinatural.

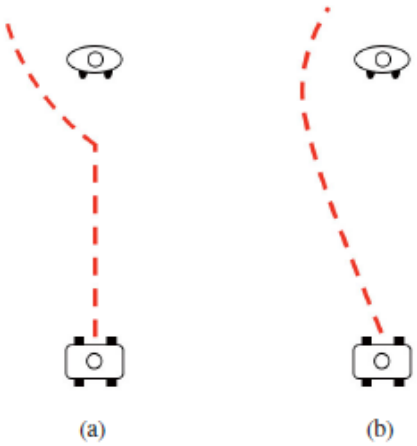
Para la estrategia reactiva se ha implementado avances dependiendo de la velocidad de movimiento del robot (Fiorini & Shiller, 1998). Sin embargo, existen problemas de oscilación cuando se aplica en un problema de navegación de múltiples agentes. Otro tipo de planificadores reactivos bien utilizados se basa en el Modelo de Fuerza Social (Helbing & Molnár, 1995). Las fuerzas provienen de la motivación interna, como dirigirse hacia la meta y el factor de distancia de las personas.

Planificadores predictivos

Algunas investigaciones (Aoude et al., 2013; Du Toit & Burdick, 2012) propusieron estimar las trayectorias futuras de los agentes como se ilustra en la Figura 15(b). El robot móvil estima los estados futuros de los otros agentes y realiza la acción recíproca. Sin embargo, en un escenario denso, cada ruta tendrá una posible colisión con algún agente. En este caso, el robot se detiene hasta que el entorno sea más seguro, lo que se conoce como "Problema de congelación del robot " (FRP).

Figura 15

(a) Planificación reactiva. El robot cambia su dirección cuando una persona aparece en el camino. (b) Planificación predictiva. El robot primero predice los estados futuros de la persona, luego reacciona para evitar una colisión por adelantado



Nota: Tomado de *Developing a telepresence robot for interpersonal communication with the elderly in a home environment*, (Tsai et al., 2007).

Una nueva estrategia es utilizar potencial de interacción para modelar la cooperación entre agentes (Trautman & Krause, 2010). Además, se ha integrado planificadores de predicción basados en: orientación espacial y predicción de velocidades (Vemula et al., 2017), inferencias probabilísticas de comportamiento (Mavrogiannis et al., 2017), protocolos y reglas sociales (Chung & Huang, 2012). Por

tanto, los planificadores predictivos se ajustan a las leyes de navegación humana, pero requieren muchos recursos computacionales. Los resultados de predicción no son siempre confiables. Requieren compensación cuando aparecen sesgos de predicción.

Estrategia basada en modelos

El proceso de navegación se basa en la toma de decisiones considerando la interacción espacial de los agentes. Los métodos para la estrategia basada en modelos pueden ser: basadas en modelo de mezcla gaussiana (GMM) (Sebastian et al., 2017), marcos sociales de interacción (Truong et al., 2017), marco de toma de decisiones de múltiples políticas (Cunningham et al., 2015), entre otras. Para garantizar la seguridad y ser coherente con las características de la navegación humana, el modelo de fuerza social (SFM) se utiliza y se extiende en muchos métodos (Chik et al., 2017; Zanlungo et al., 2017). La dependencia de la velocidad también se utiliza para reducir la incertidumbre de la trayectoria (Y. Chen et al., 2019). A pesar de que el método no necesita procesos de capacitación, necesita considerar mucha información social para que el robot se comporte más como una persona. El modelo depende la selección de parámetros, que es un proceso difícil y puede generar incertidumbre.

Estrategias basadas en aprendizaje

El desarrollo de una red neuronal profunda abre una nueva forma de hacer que la navegación del robot sea más segura y amigable. En función de las diferentes funciones, los modelos se pueden dividir en tres categorías: aprendizaje supervisado, aprendizaje de refuerzo profundo, métodos de aprendizaje de refuerzo inverso. Dentro del aprendizaje supervisado las investigaciones se han enfocado en el uso de redes neuronales convolucionales para predecir el mapa de costos (Perez-Higueras et al., 2018) o redes neuronales profundas para representar estados del entorno.

Las dos investigaciones buscan ayudar al robot a navegar socialmente. Dentro del aprendizaje supervisado las investigaciones se han enfocado en el uso de redes neuronales convolucionales para predecir el mapa de costos (Perez-Higueras et al., 2018) o redes neuronales profundas para representar estados del entorno. Las dos investigaciones buscan ayudar al robot a navegar socialmente (Li et al., 2018).

El aprendizaje de refuerzo profundo se ha utilizado para disminuir la computación de un modelo predictivo (Y. F. Chen et al., 2017). También se hizo uso el aprendizaje profundo para desarrollar un algoritmo de evasión de colisión (Long et al., 2018). El último método, basado en aprendizaje inverso, se ha propuesto para generar la función de costo y se utiliza en la planificación de rutas (Wulfmeier et al., 2017).

El modelo de red neuronal se entrena utilizando los datos de trayectoria real. Por lo tanto, reproduce el comportamiento humano en gran medida. Para el desarrollo de estrategia, el proceso de capacitación necesita una gran cantidad de datos, y no es posible generalizar su aplicación.

Arquitecturas de navegación

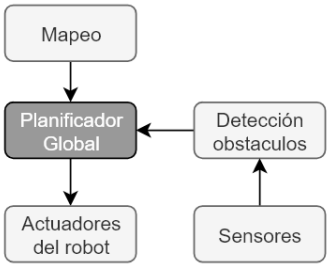
Según (Chik et al., 2016) para establecer un sistema de navegación social sólido, se utilizan diferentes configuraciones de planificadores globales, locales y modelos de predicción, formando de esta forma los siguientes tipos de arquitecturas.

Planificador global único

El modelo de arquitectura donde solo se utiliza un único planificador global en el cual únicamente se genera una ruta, con la información obtenida de un mapa previo. Cuando se detecta un obstáculo se debe construir una nueva ruta lo cual no presenta un rendimiento óptimo para navegación social, aumentando el costo computacional y tiempo de respuesta, posee la estructura mostrada en la Figura 16.

Figura 16

Arquitectura solo planificador global



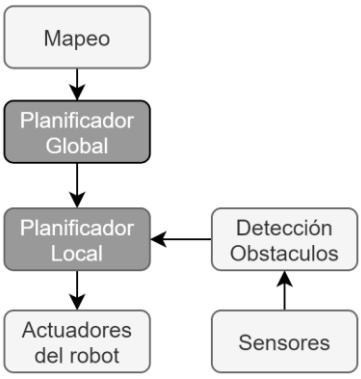
Nota: Tomado de *A review of social-aware navigation frameworks for service robot in dynamic human environments*, (Chik et al., 2016).

Planificador local y global

El modelo propuesto al implementar ambos planificadores en los cuales se refuerza la capacidad dinámica para evadir obstáculos al usar el planificador local y aumentando la capacidad de navegación independiente mediante el planificador global. Sin embargo, este método no es viable en circunstancias de movimiento o alta cantidad de personas, esta arquitectura posee la estructura mostrada en Figura 17.

Figura 17

Arquitectura con planificador global y local



Nota: Tomado de *A review of social-aware navigation frameworks for service robot in dynamic human environments*, (Chik et al., 2016).

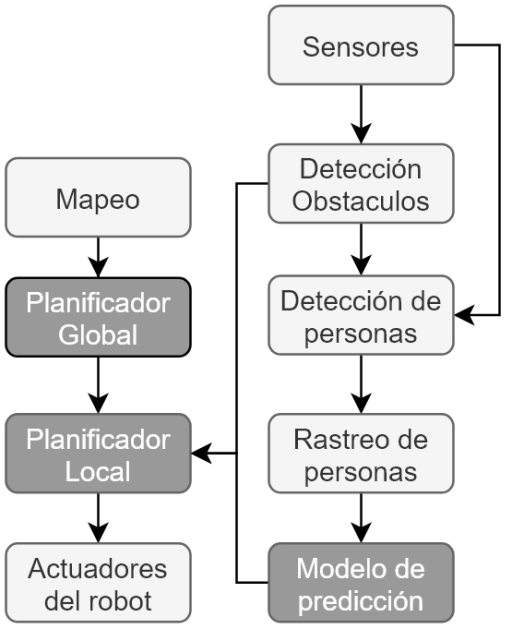
Planificador global con planificador predictivo

El modelo planteado en esta arquitectura se basa en la utilización de un planificador Global y Local y se añade una etapa de detección donde se separa a los obstáculos de las personas, lo cual permite al robot asignar una función de costo a las personas detectadas que represente el espacio que no puede invadir el robot, generando de esta forma un comportamiento social en aglomeraciones de personas, esta arquitectura posee la estructura mostrada en la Figura 18.

Dentro de estas arquitecturas existen diferentes tipos de planificadores globales y locales que son utilizados en la robótica, para entornos estáticos y dinámicos.

Figura 18

Arquitectura con planificador global y local con modelo de detección y seguimiento de personas



Nota: Tomado de *A review of social-aware navigation frameworks for service robot in dynamic human environments*, (Chik et al., 2016).

Planificadores globales

Los planificadores globales están encargos de otorgar una ruta óptima y segura libre de colisiones. Para lo cual es necesario tener una información como un mapa estático del entorno en el cual se va a llevar a cabo la navegación. Existen diferentes técnicas y métodos entre los principales se encuentra el Algoritmo A* y algoritmo Dijkstra.

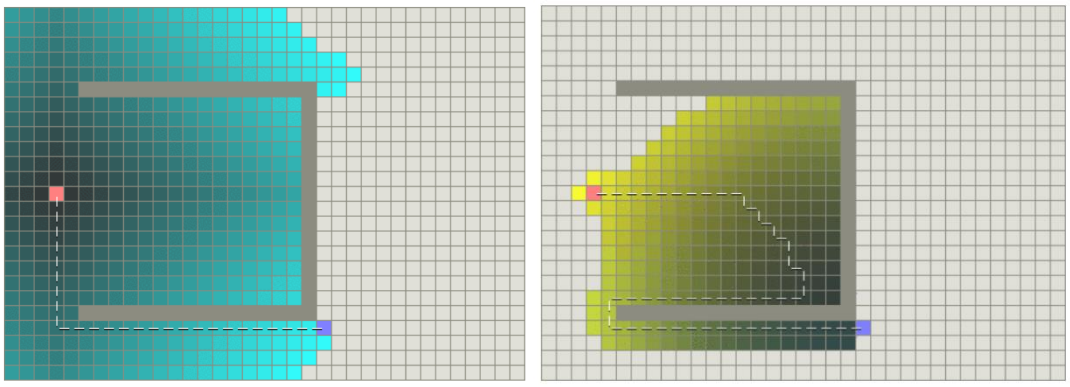
Algoritmo A*

Según (Fornth, 2012) el algoritmo se basa en una implementación flexible para la búsqueda de rutas ya que este compuesto de la suma de dos funciones de costo separadas : $f(n) = g(n) + h(n)$ y utiliza un mapa del entorno el cual se divide en una cuadricula y se toma en cuenta el punto de partida y el punto objetivo asignado.

La función $g(n)$ representa el costo exacto de la ruta desde el punto de partida hasta el vértice (n) o punto actual de iteración del algoritmo y $h(n)$ presenta el costo heurístico exacto desde el punto objetivo hasta el vértice o punto actual (n).

Figura 19

Ejemplo de planificación mediante solución exacta y solución heurística



Nota: Tomado de *Introducción a A *,* (Fornth, 2012).

Las funciones por separado presentan una forma distinta de encontrar el punto objetivo, donde $g(n)$ otorgará el camino de menor costo con respecto al punto inicial

(menor distancia recorrida) pero con mayor número de iteraciones y $h(n)$ una respuesta más rápida de la búsqueda, pero no el camino más eficiente como se muestra en la Figura 19. .

Por lo que gracias a la implementación conjunta el algoritmo A* muestra una respuesta con la ruta más corta entre el punto inicial y el punto final sin consumir mucho procesamiento y con menor tiempo de solución.

Algoritmo Dijkstra

El algoritmo es una versión simplificada del algoritmo A* en el cual únicamente se utiliza la función de costos $g(n)$ por lo que presenta como resultado la ruta más corta, pero presenta un gran costo computacional en entornos muy grandes y posee un tiempo mayor de implementación.

Planificadores locales

Este tipo de planificador es el encargo de otorgar una respuesta y modificación en la ruta inicial cuando se presenta un obstáculo dentro de un radio o marco menor alrededor del robot, el cual se actualiza de manera reiterativa durante la ruta principal.

Planificador DWA

En (Fox et al., 1997) se propone el algoritmo DWA, que es un planificador local que toma en cuenta las restricciones cinemáticas y los parámetros dinámicos del robot. El algoritmo DWA planifica la trayectoria libre de colisión en dos pasos. Primero, DWA reduce el espacio de búsqueda descartando esas velocidades no alcanzables. El paso tiene en cuenta tres conjuntos de velocidades: trayectorias circulares, velocidades admisibles y ventana dinámica. El segundo paso de DWA es maximizar una función objetivo eligiendo las velocidades posibles desde el paso uno.

Telepresencia

La telepresencia es una forma de comunicación en línea la cual implementa un sistema de video y audio que permite mantener contacto entre una o más personas que se encuentren en diferentes lugares geográficos. Otorga la sensación de mantener un encuentro cara a cara como lo explica (Minsky, 1980). La telepresencia fue implementada dentro del entorno laboral de oficinas para poder disminuir el gasto generado en servicios de comunicación y viajes. La primera tecnología implementada en los servicios de telepresencia fue un tipo de teléfono IP. Posteriormente se implementó la transferencia de archivos e imágenes y por último transmisiones de video en vivo (Cisco, 2015).

Robots móviles de telepresencia (MRP)

Dentro del campo de la robótica la implementación de los sistemas de telepresencia ha sido de gran importancia especialmente en ámbitos sociales como entornos laborales (entornos de oficina, educación), cuidados en la salud (supervisión de adultos, consultas médicas) y sistemas de exploración y seguridad (robots exploradores, unidades antiminas y de rescate). Gracias a la telepresencia y robótica móvil se logró otorgar un efecto de presencia en un espacio remoto (Beer & Takayama, 2011; Lee & Takayama, 2011) brinda ayuda de robots en nuevos campos de desarrollo e investigación.

Diseño de sistemas MRP

El diseño de los sistemas MRP depende principalmente de la aplicación objetivo de la plataforma. El principal diseño utilizado en la actualidad es el antropomórfico, el cual principalmente consta de tres partes distribuidas de la siguiente forma según (Kristoffersson et al., 2013):

- **Base móvil:** Consta prácticamente de todo el sistema de locomoción del MRP, extremidades y extensiones finales de cada una de las plataformas, la configuración más común utilizada es la diferencial (Gibstein, 2011; K Tsui & Desai, 2011).
- **Arreglo de sensores:** Es el método de conexión entre el MRP y el entorno exterior. Dentro de las mayorías de las plataformas móviles analizadas en (Kristoffersson et al., 2013) constan de sensores de odometría, acelerómetros, giroscopios, infrarrojos, y ultrasónicos.
- **Sistema de interacción:** Sistema encargado de mantener la interacción entre el operador y la persona que interactué mediante cámaras, arreglos de micrófonos y sensores de presencia.

El diseño de los sistemas MRP permite establecer un conjunto de tipos de interacción que ocurren simultáneamente. La primera es la interacción humano-robot la cual ocurre entre la persona y el MRP. La segunda ocurre entre el operador y la computadora durante el control de manejo del MRP. Por último, la tercera interacción se establece de forma indirecta entre el operador remoto y el usuario local. Es necesario mantener una comunicación fluida en cada una, manteniendo una transmisión de video sin latencias o grandes pérdidas de datos. Según (Tsai et al., 2007) los elementos básicos implementados dentro del sistema de telepresencia constan de los elementos y tecnologías mostrados en la Tabla 4.

Tabla 4

Elementos de diseño y tecnologías relacionadas a la implementación de MRP`s

Elementos de diseño	Tecnologías Relacionadas
Transmisión de datos	Transmisiones por RF e internet, tiempo de retardo y algoritmos de transmisión
Teleoperación	Entorno de control y navegación, interfaces de manejo

Elementos de diseño	Tecnologías Relacionadas
Sistema de censado	Transductores para interacción con el medio (presencia, ultrasónicos)
Sistema de locomoción	Actuadores, mecanismos de locomoción
Sistema de captura de video	cámaras, arreglos de micrófonos
Comportamiento autónomo	Sistema de control, reconocimiento del entorno

Nota: Tomado de *Developing a telepresence robot for interpersonal communication with the elderly in a home environment*, (Tsai et al., 2007).

Implementación del sistema de telepresencia

Dentro del desarrollo de los sistemas de telepresencia existen dos tipos. Uno correspondientes al comercial y el segundo al campo de la investigación. En donde en el primer aspecto se centra en el desarrollo de software. El segundo campo de desarrollo es referente a los prototipos de investigación. En el cual se implementan diferentes tipos de sistemas como los que se muestra en los siguientes trabajos:

- En el caso propuesto en (Tonin et al., 2018) se establece una comunicación mediante la utilización de un VoIP (Voice over Internet Protocol) comercial (Skype) para la comunicación entre el robot móvil y el computador del operador.
- Para el desarrollo del MRP mostrado en (Tsai et al., 2007) propone una implementación propietaria mediante la utilización de ADSL junto con WLAN utilizando un MDS (Mobile Data Server) compuesto por un controlador dedicado, una memoria de ROM y una tarjeta de red.
- El sistema propuesto en (Harmo et al., 2001) consta del uso de TCP/IP (Transmission Control Protocol/Internet Protocol) en conexión a una base de datos alojada en un servidor privado para disminuir las latencias y perdidas de datos.
- Cuando es necesario implementar un nivel de seguridad, pero mantener un bajo costo así como una transmisión establece se utiliza el sistema implementado en (Muralindran et al., 2011). En el cual mantiene el uso de TCP/IP pero se añade el

protocolo SSL-alike (Security Sockets Layer) para la seguridad en adición se propone una codificación y compresión de la transmisión de video.

- El sistema de MRP propuesto en (Melendez-Fernandez et al., 2017) consta de la implementación de un robot de telepresencia junto con una interfaz de control remota en la cual se utilizó los protocolos: HTTP, WebSocket y MQTT para su implementación. Dicha interfaz se encuentra alojada en una página web lo que otorgaba modularidad y compatibilidad con cualquier sistema de PC o dispositivos móviles. La interfaz se comunica con la plataforma mediante los protocolos: NodeJS y WEBRTC, entregando la información de control a la plataforma.

WEBRTC Web Real-Time Communication

Es una framework de código abierto y base fundamental del estándar RTC implementada por el equipo de desarrollo de Google (Sredojev et al., 2015). Dicha tecnología permite establecer una comunicación en tiempo real que establece un intercambio de datos, archivos, y transmisiones de video en tiempo real entre dos clientes. Es posible de utilizarse en diferentes lenguajes de programación y sistemas operativos ya que cuenta con diversas APIs para su compatibilidad.

Como se describe en (Suciu et al., 2020) la implementación de WEBRTC para el desarrollo de sistemas de telepresencia presenta varios beneficios debido a su conexión de punto a punto. Ya que otorga una encriptación única para el canal de transmisión brindando seguridad y robustez al sistema. No utiliza plugins o extensiones propietarias que ralentizan la transmisión de datos entre los clientes. WEBRTC se basa en tres módulos generales para el manejo de los datos estos son:

- **MediaStream:** Es el módulo que accede los dispositivos que entregan la información, teclado, cámaras y micrófonos.

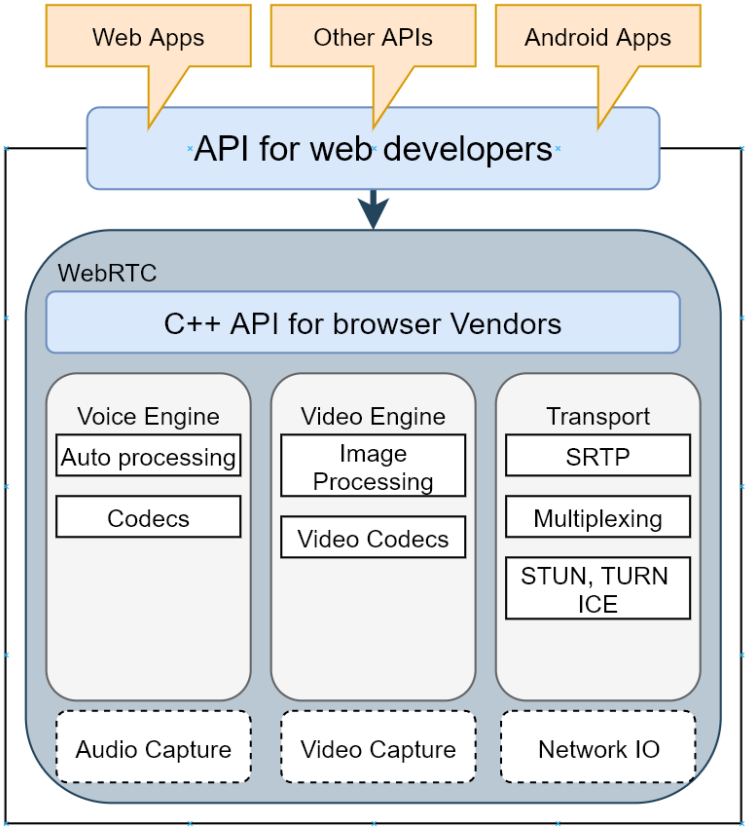
- **RTCPeerConnection:** Modulo encargado de las funciones de encriptación y ancho de banda.
- **RTCDataChannel:** Es el módulo encargado de enviar los datos de forma genérica.

Estructura de WEBRTC

La estructura de funcionamiento de WEBRTC se divide en dos capas generales. La primera hace referencia a la capa de desarrollo web para la creación de interfaces estas utilizan APIs para la conexión con la interfaz por motivos de seguridad.

Figura 20

Estructura de WebRTC



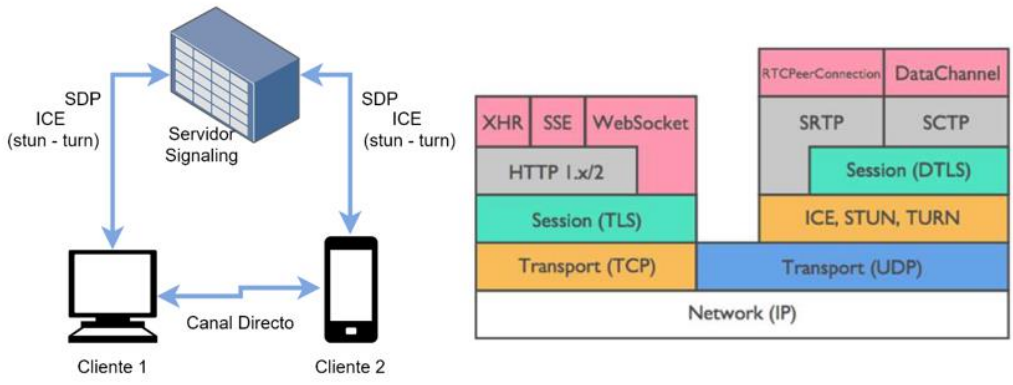
Nota: *Tomado de* WebRTC role in real-time communication and video conferencing, (Suciu et al., 2020).

La segunda capa es la encargada de comunicar los datos y pedidos de la API con las librerías y archivos base de WEBRTC, en donde se procesa los datos tanto de video, audio y transmisión de información. La estructura se pude ver en Figura 20, en donde el motor de procesamiento de voz se encarga de computar y decodificar el sonido capturado en cada cliente. El motor de video de igual manera se encarga de codificar y decodificar las imágenes recibidas en cada cliente mediante VP8 y por último el módulo de transporte es el encargado de obtener la información, encriptarla y transmitirla utilizando SRTP (Security Real-Time protocolo).

El funcionamiento general para un sistema de telepresencia mediante WEBRTC sigue el flujo indicado en la Figura 21. Para establecer la comunicación entre los dos clientes es necesario utilizar un servidor externo. Mediante el cual se intercambiarán los datos de red y media de cada cliente, así como la configuración del servidor STUN o TURN a utilizarse en caso de que los clientes se encuentren en redes remotas con sus respectivos firewalls.

Figura 21

Diagrama de conexión básica mediante WebRTC y protocolos utilizados



Nota: Tomado de *WebRTC role in real-time communication and video conferencing,* (Suciu et al., 2020).

Protocolos de WEBRTC

WEBRTC se basa en los tres conceptos de PeerConnection, DataChannel y MediaStream. En donde PeerConnection utiliza el protocolo ICE para establecer la conectividad entre los terminales en conjunto con los protocolos STUN y TURN para conexiones con clientes en redes NAT. Mientras que Datachannel para la transmisión de datos directa utiliza el protocolo UDP en conjunto del protocolo SCTP (Stream Control Transmission Protocol) y DLTS (Datagram Transport Layer Security) para la seguridad de transmisión. Un ejemplo del grupo de protocolos que se usan en WEBRTC se puede observar en la Figura 21.

Proyectos afines

El desarrollo de sistemas comerciales y de investigación en los MRP ha incursionado en diversos ámbitos utilizando diferentes configuraciones y propuestas como muestran (Kristoffersson et al., 2013) utilizando estos estudios de investigación y proyectos de plataformas comerciales, se escogió un grupo de sistemas MRP que presentan una serie de características similares a las propuestas en el presente trabajo y se realizó una identificación de los componentes y características esenciales con las que cuentan.

Robot VGo

Según (Katherine Tsui et al., 2013) VGo es un sistema de telepresencia que presenta una solución diferente frente a los sistemas de videoconferencia para poder brindar una sensación de presencia de forma 100% remota sin necesidad de interacción directa de las personas del entorno.

Figura 22

Ejemplo de Robot VGos en una reunión social

Nota: Tomado de *Design and development of two generations of semi-autonomous social telepresence robots*, (Katherine Tsui et al., 2013).

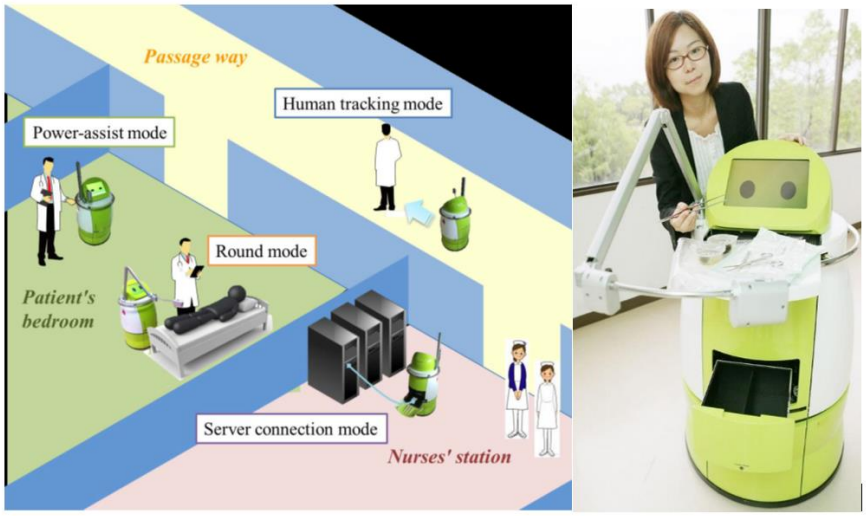
Es un robot móvil de configuración diferencial que cuenta con una pantalla giratoria en tilt y control remoto mediante teclado y ratón se muestra en la Figura 22. Fue diseñado para monitoreo de pacientes, estudiantes en procesos de recuperación y cuidados a personas mayores.

Robot de soporte medico Terapió

Es un robot de configuración diferencial diseñado para ser una estación de soporte móvil en salas de hospitales y clínicas. Además, puede desarrollar dos tareas que consisten en llevar equipamiento médico durante una ronda de revisión y de grabar información acerca del estado de salud de los pacientes en un historial electrónico (Tasaki et al., 2015). Consta de un mecanismo omnidireccional y un sistema de seguimiento de personas para seguir a las enfermeras y médicos. El robot es capaz de enviar y recibir información de otros lugares al conectarse a una estación mediante un cable de datos.

Figura 23

Modelo de circulación del robot general Terapió



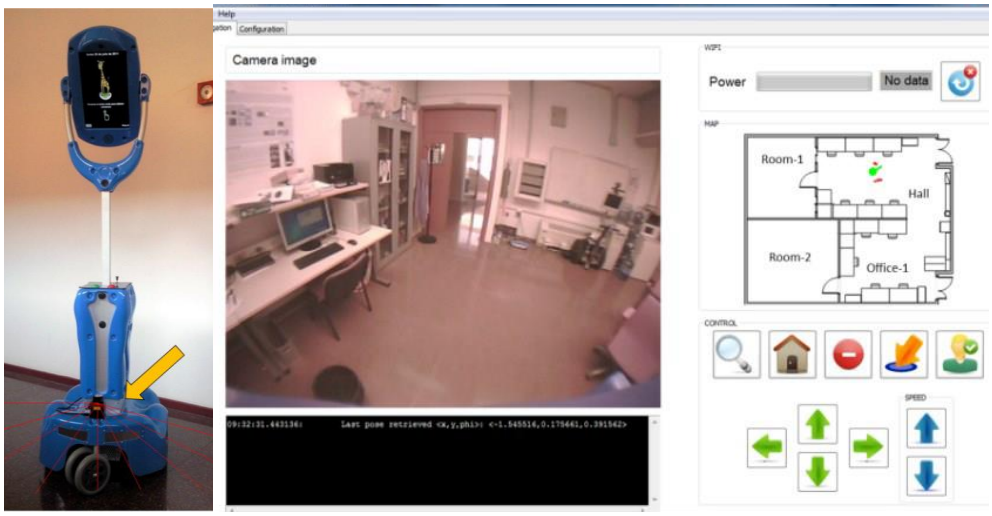
Nota: Tomado de *Prototype design of medical round supporting robot 'Terapio'*, (Tasaki et al., 2015).

Robot Giraff

Es un robot de configuración diferencial desarrollado para el cuidado de personas de la tercera edad en sus casas y en salas de hospitales. Posee dos tipos de funcionamiento uno remoto mediante el uso de un computador con conexión a internet y el segundo no necesita de conexión a internet y se comanda de forma local (Iarlori et al., 2018). El robot Giraff tiene un sistema de seguimiento de personas, conducción segura en entornos sociales, acoplamiento automático con su estación de carga, detección de obstáculos y puede mostrar la ubicación en tiempo real del robot al operador en un mapa (Gonzalez-Jimenez et al., 2012).

Figura 24

Modelo del robot Giraff e interfaz de monitoreo y control del robot



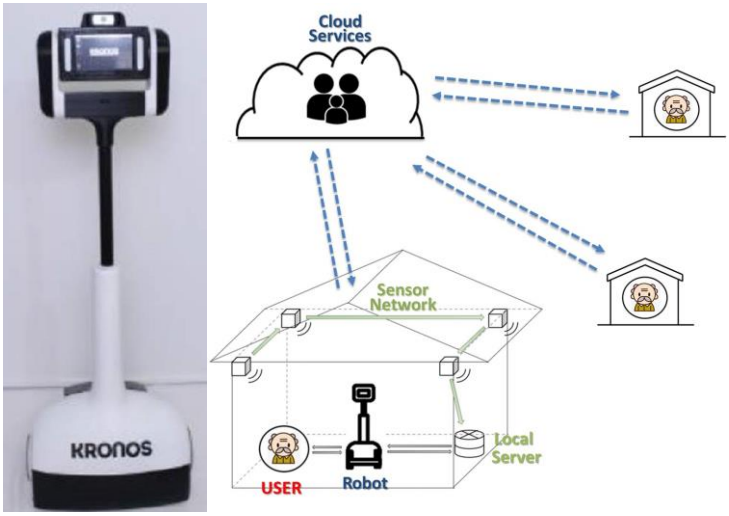
Nota: Tomado de *Technical improvements of the Giraff telepresence robot based on user's evaluation*, (Gonzalez-Jimenez et al., 2012).

Robot Kronos

Es un robot desarrollado para la interacción social dentro de un entorno inteligente con sensores para el cuidado de personas. Posee una configuración diferencial con un sistema de detección de obstáculos y un sistema de telepresencia que puede ser operado de dos formas. De forma remota a través de un centro web intermediario y de forma local utilizando una red privada (Loza-Matovelle et al., 2019). Está implementado mediante ROS para poder integrar las señales obtenidas del sistema de percepción, la información extra de los sensores colados en el entorno y datos de interés obtenidos de la video llamada para poder implementar un sistema de seguimiento de la cámara usando dos grados de libertad para mejorar el comportamiento social de la plataforma.

Figura 25

Modelo del robot Kronos y concepto previsto del sistema completo



Nota: Tomado de *An architecture for the integration of robots and sensors for the care of the elderly in an Ambient Assisted Living Environment*, (Loza-Matovelle et al., 2019).

Una vez obtenida la información técnica y de funcionamiento de cada uno de los proyectos analizados se creó la Tabla 5 en donde se muestran las características de cada uno de los sistemas.

Tabla 5

Tabla comparativa de características de los sistemas MRP analizados

Características	Robot Vgo	Robot Terapió	Robot Giraff	Robot Kronos
Configuración	Diferencial	Omnidireccional	Diferencial	Diferencial
Peso	8.16[kg]	80[kg]	65[kg]	35[kg]
Altura	121.9 [cm]	134.0[cm]	170[cm]	150[cm]
Duración batería	3 a 6 horas	2 a 7 horas	1.5 horas	
Control de navegación	ratón y teclado	botones	ratón y teclado	
Velocidad Máxima	1.2[m/s]	1.3 [m/s]	1.78[m/s]	1.5[m/s]

Características	Robot Vgo	Robot Terapió	Robot Giraff	Robot Kronos
Tamaño de pantalla	6 pulgadas	no tiene	13.3 pulgadas	7 pulgadas
Numero de cámaras	1 cámara	2 cámaras	1 cámara	1 cámara
Grados de libertad pantalla	1 grado	2 grados	1 grado	2 grados
Numero de altavoces	2 altavoces	1 altavoces	3 altavoces	1 altavoz integrado
Sensores incorporados	Sensores IR Sensor magnético Sensores fin de carrera micrófonos	Sensores IR Sensor de fuerza Sensores fin de carrea micrófonos	Sensores de color Sensores IR Sensores fin de carrera micrófonos	Sensores IR micrófonos
Medio de comunicación	WIFI	WIFI y LAN	WIFI	WIFI
Detección de obstáculos	SI	SI	SI	SI
Evasión de obstáculos	NO	SI	NO	SI
Estación de carga	NO	SI	SI	NO

Resumen

Los robots móviles son máquinas programables hechas para desplazarse por los medios como: aire, agua y tierra. La necesidad de dotar al robot móvil de autonomía surge de la gran variedad de campos en los que ha sido aplicado. La capacidad del robot para desplazarse en un entorno es la navegación. Sus principales componentes son: percepción del entorno, localización, planificación de ruta y control de sus salidas. La percepción del entorno en un robot móvil se realiza mediante sensores como: sonar,

lidar, encoders, entre otros. Lo que permite emular el sentido de la vista en los seres humanos. La cinemática del robot, junto con la resolución de la localización mediante odometría permite al robot situarse en un plano global, como punto de partida para la exploración de entornos cerrados y controlados. Por lo general es necesario implementar una corrección de posición dada por la odometría para mejorar su cálculo.

Se mostró diversos métodos y técnicas utilizadas para poder determinar la localización de los sistemas MRP donde se determina que la mejor estrategia consta de los métodos probabilísticos tomando en cuenta los datos obtenidos por los sensores de los sistemas. Se enumera varios métodos de optimización dependiendo de los sistemas a tratar en donde un filtro de Kalman representa la mejor opción para sistemas lineales y gaussianos mientras que para sistemas no lineales y dependiendo de la dificultad de linealización y complejidad se puede aplicar técnicas como EDF, UDF o filtro de partículas. Se ha definido al robot social como el ente cibernético que interactúa con las personas siguiendo comportamientos, patrones y normas sociales. Las propiedades que un robot social debe exhibir son comodidad, naturalidad y sociabilidad.

La navegación social es la intersección entre HRI (interacción humano-máquina) y la planificación de movimiento. Las funcionalidades de un sistema de navegación de un robot social se dividen en tres grupos: percepción, planificación y actuación o grupo reactivo. El grupo de percepción genera información del entorno y su propia posición. En el grupo de planificación se decide las acciones a realizar. Y el grupo reactivo controla las salidas y mantiene la seguridad. Las estrategias para navegación social se dividen en planificadores reactivos, predictivos y estrategias basadas en modelos y basadas en aprendizaje. Las características de cada una muestran que, los planificadores reactivos garantizan seguridad física, pero presentan comportamiento antinatural. El planificador predictivo permite estimar trayectorias futuras, pero con problemas de FRP. Los otros métodos se basan en parámetros para obtener

información del comportamiento humano. El uno se basa en modelos y el otro en aprendizaje (red neuronal) pero ambos requieren gran cantidad de datos.

La telepresencia permite transmisión de audio y video para una comunicación en diferentes lugares geográficos. La integración de telepresencia y robótica móvil (MRP) han otorgado un efecto de presencia en un espacio remoto. El diseño de sistemas MRP consta principalmente de base móvil, arreglo de sensores y sistema de interacción. En adición se enlistaron una serie de sistemas de telepresencia implementados en plataformas robóticas donde se puede observar que los métodos más utilizados eran mediante servicios VoIP como el caso de Skype e implementación por protocolo TCP y SSL utilizando servicios de servidores y bases de datos en la nube. Por último, se enumera una serie sistemas MRP implementados previamente junto con las características más importantes de cada uno.

Capítulo III

Diseño

El capítulo da a conocer el diseño del sistema de navegación reactiva social y telepresencia en el prototipo de robot, basado en el método de diseño concurrente (Ulrich & Eppinger, 2013). Se inicia identificando los requerimientos o necesidades del usuario y diseñadores. Luego, se definen las especificaciones que van acorde con dichas necesidades. Tanto las necesidades como especificaciones forman parte de la descomposición del robot en subsistemas. Se plantean alternativas de diseño para cada subsistema, para luego realizar la evaluación correspondiente. Posteriormente, se realiza la definición del diseño donde se da origen a la implementación de los componentes y algoritmos del sistema. Por último, se corrige posibles fallos y se mejora el diseño en la definición del concepto final.

Identificación de necesidades

Definir las necesidades asegura que el producto final del proyecto se enfoque en dichos parámetros, además de proporcionar una base de datos que justifique las especificaciones del producto. El conocer las necesidades permite a los diseñadores incluir todas las necesidades críticas del cliente o usuario. En la Tabla 6 se lista las necesidades identificadas para el proyecto. En base a estas necesidades descritas, se obtienen las especificaciones técnicas del sistema.

Tabla 6

Lista de necesidades para el sistema

#	Necesidad
1	Evitar obstáculos en el entorno
2	Navegación autónoma y manejo a distancia
3	Capacidad de realizar videollamadas con el operador
4	Interfaz gráfica intuitiva
5	Navegación silenciosa

#	Necesidad
6	Detección de personas
7	Velocidad de movimiento segura
8	Detección y rastreo de personas
9	Detección y evasión de obstáculos
10	Pantalla a una altura adecuada para comunicación

Definición de especificaciones

Una vez conocidas las necesidades del diseño se procede a establecer los parámetros de diseño del sistema. Los requerimientos y necesidades descritas se analizan como un valor medible físicamente. Por lo tanto, se establecen métricas que permitirán medir el grado de cumplimiento de los requerimientos. Las especificaciones se presentan en la Tabla 7.

Tabla 7

Especificaciones para las necesidades con su respectiva métrica

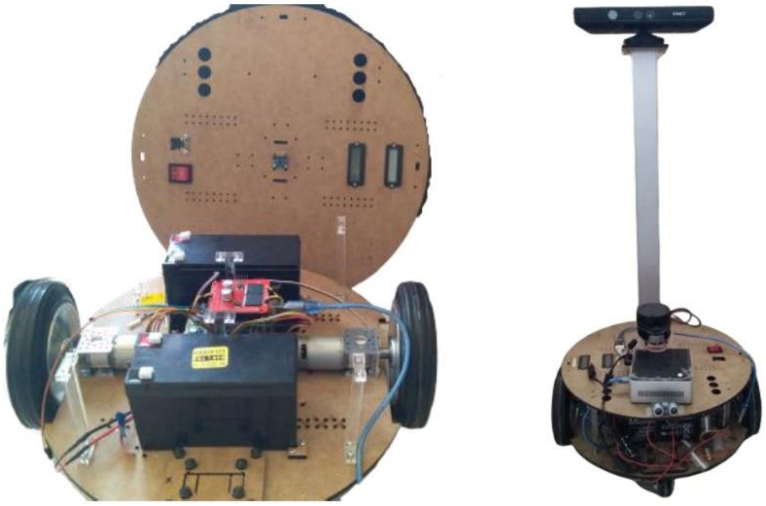
Necesidad	Métrica	Magnitud	Unidad
1,7,10	Número mínimo de sensores	12	-
3,5	Tamaño mínimo de la pantalla	17.78 (7´)	cm
3	Resolución mínima de calidad de video	480	pixeles
3,5	Cantidad máxima de botones en la interfaz de control	9	-
1, 2 ,6, 7, 9	Distancia de percepción	5	m
1, 8, 9	Velocidad promedio de desplazamiento	0.2	m/s
2	Voltaje requerido del robot	12	V
3	Número mínimo de micrófonos	1	-
6	Nivel máximo de decibelios	50	dB
2, 10	Distancia de desnivel	60	mm
12	Costo máximo de implementación	533	$
11	Altura máxima de la estructura	1.3	m

Descripción del prototipo de robot diferencial

Para poner en funcionamiento el sistema propuesto se utilizará el prototipo de robot móvil implementado en (Andrango, 2020), mostrado en la Figura 26. El cual es un robot de configuración diferencial, a base de materiales de bajo costo y utilizando ROS para la arquitectura de control de la plataforma, sus principales características se muestran en la Tabla 8.

Figura 26

Prototipo de robot diferencial a utilizarse



Nota: Tomado de Implementación de reglas de comportamiento social en una plataforma robótica de telepresencia, a través del reconocimiento de gestos, (Andrango, 2020).

Tabla 8

Características generales del prototipo de robot

Ítem	Características generales	Valor
1	Masa total del robot	12 [kg]
2	Velocidad máxima de traslación	0.5 [m/s]
3	Peso extra-máximo permitido	2 [kg]
4	Aceleración máxima del robot	0.2 [m/s^2]

Ítem	Características generales	Valor
5	Inclinación máxima para circulación	15 [grados]
6	Eficiencia de los motores	60 [%]
7	Consumo energético sección de control	10.7 [A]

El prototipo es capaz de realizar una navegación dentro de un entorno controlado y sin cambios. Además, detecta personas e identifica gestos mediante el sensor Kinect. Para poder controlar la plataforma se utilizó una interfaz gráfica de computador local, la cual permite utilizar tanto el reconocimiento de gesto y de forma manual ingresar el punto objetivo en rviz-ROS. Los recursos presentes en el prototipo constan de una serie de sensores, actuadores y elementos que se muestran en la Tabla 9.

Tabla 9

Elementos del prototipo

Ítem	Descripción elemento	Cantidad
1	Arduino mega Amiga 2560	1
2	Controlador Monster Shield	1
3	Sensor IMU MPU6050	1
4	Encoders efecto hall	2
5	Sensor RPLidar A1	1
6	Sensor Kinect modelo 1517	1
7	Baterías recargables de litio 7ah 12 VDC	2
8	Convertidor de voltaje DC-DC	1
9	Computador Intel NUC D54250WYKH	1
10	Motor de DC con caja reductora planetaria	2

El funcionamiento del prototipo al ser implementado por ROS utiliza paquetes generales para cada una de las acciones realizadas por el robot. Los principales paquetes corresponden a:

- Kobuki y Turtlebot: Paquetes generales para construcción de robots de configuración diferencial (Koubâa et al., 2016)
- RPLidar ROS: Paquete de conexión y transmisión de datos para el sensor RPLidar A1. (SHANGHAI SLAMTEC CO. LTD., 2009).
- Navigation_Stack y Move_Base: Paquetes para navegación de plataformas móviles y de configuración general (Lu & Marder-Eppstein, 2016).
- Openni_Tracker, Neuro Gesture_Kinect y Skeleton_markes: Paquetes utilizados para el reconocimiento de gestos y creación de marcadores de esqueleto (Goebel, 2015).

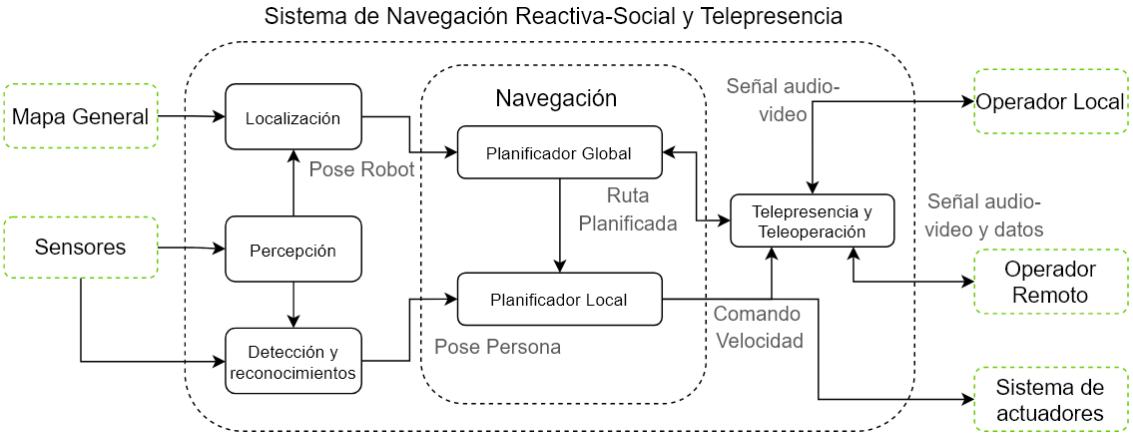
Partiendo del modelo de prototipo se implementará el sistema de navegación reactiva social y teleoperada del robot “socialbot”.

Definición de subsistemas y selección de alternativas

Una vez definidas las necesidades y las especificaciones del robot, se procede a descomponer el problema en subsistemas más simples. En cada subsistema se crea una descripción específica de los elementos necesarios para implementar todas las funcionalidades y características del sistema. Según (Chik et al., 2016; Sankar & Tsai, 2019; Talebpour et al., 2016) para poder implementar un sistema de navegación reactiva-social se debe tener un planificador global y local, tomando en cuenta que se posee un mapa previo del entorno de navegación y localización de la plataforma. Mientras que (Corke et al., 2012; Herring, 2013) proponen implementar un sistema de telepresencia de dos módulos. Uno para envío de comandos y otro para transmisión de audio y video. Por lo cual se propone el siguiente conjunto de subsistemas que se muestra en la Figura 27.

Figura 27

Subsistemas del sistema de navegación reactiva-social y telepresencia



En la siguiente sección, se detalla cada uno de los subsistemas. Además, se analiza alternativas de diseño y se evalúa las alternativas para seleccionar la más apropiada de acuerdo con la función de cada subsistema.

Subsistema 1: Percepción

El subsistema de percepción se ocupa de obtener los datos de todos los sensores utilizados en la plataforma (encoders, sensor IMU, sonares, sensor Kinect y lidar). La tarjeta de adquisición sirve para acondicionar las señales obtenidas y enviarlas a ROS mediante comunicación serial al publicar cada uno de los datos en tópicos específicos para utilizarlos en los siguientes subsistemas. Dentro del subsistema de percepción se presenta una serie de sensores implementados en el prototipo a utilizarse como es el caso del Kinect, el sensor LIDAR, y los sensores propioceptivos (IMU y encoders). Para poder implementar la navegación reactiva, es necesario sensores que permitan detectar obstáculos para evadirlos durante la navegación. Los criterios para análisis y evaluación de las alternativas se muestran en la Tabla 10.

Tabla 10

Criterios de diseño del subsistema de percepción

Criterio	Letra representativa
Costo	A
Distancia de detección	B
Ángulo de detección	C
Consumo energético	D
Confiabilidad frente al entorno	E

Nota: Las letras representativas de cada criterio serán utilizadas en las tablas del subsistema de percepción.

Una vez conocidos los criterios, se realiza una comparación entre cada uno de ellos. Se asigna mayor importancia (5), menor importancia (0.2) o igual importancia (1) en cada situación de comparación. La sumatoria de cada criterio es normalizada y se obtiene el peso ponderado definido que se presenta en la Tabla 11.

Tabla 11

Ponderación de criterios de evaluación

	A	B	C	D	E	Total	Peso ponderado definido
A		1	0.2	1	1	3.2	0.10
B	1		1	5	1	8	0.24
C	5	1		5	1	12	0.37
D	1	0.2	0.2		0.2	1.6	0.05
E	1	1	1	5		8	0.24
		Suma Total				32.8	1.00

Nota: La tabla muestra la comparación entre criterios para el subsistema de percepción, para definir una importancia a cada criterio. Las letras representan los criterios según se observa en la Tabla 10.

Las alternativas de sensores que permiten realizar navegación reactiva son las siguientes:

- **Sensor Kinect:** El sensor Kinect consta de una cámara RGB, un sensor de profundidad y una serie de giróscopos que aportan información sobre su orientación. El campo de visión de la cámara es de 57,77º grados en horizontal y 45º en vertical. El rango de trabajo va desde los 1.2m hasta los 3.5m. Las principales ventajas y desventajas del sensor Kinect son, los componentes se muestran en la Figura 28.

  Ventajas

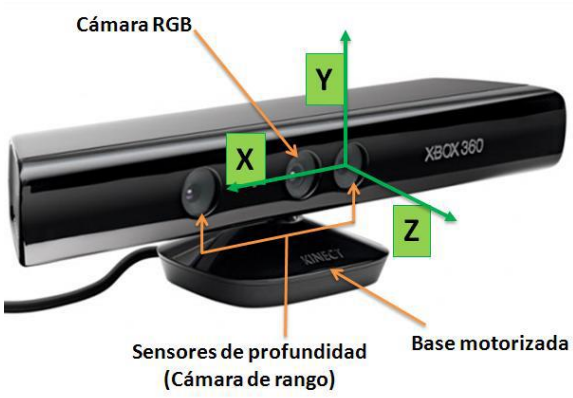
  - Proporciona imágenes RGB y rango de distancia.
  - Presenta una frecuencia de trabajo de 30 Hz.
  - Provee información tridimensional del entorno.

  Desventaja

  - Campo de visión limitado.
  - La distancia mínima de detección es 1.2 metros.

Figura 28

Ponderación de criterios de evaluación



- **Sensor LIDAR Omnidireccional 360º:** El sensor LIDAR es un sistema de medición láser triangular. El sistema puede realizar una exploración de 360 grados dentro del rango de 6 metros. Los datos de nube de puntos 2D

producidos se pueden usar en mapeo, localización y modelado de objetos/entorno.

Ventajas

- Alcanza a muestrear 360 puntos a una frecuencia de 5,5 Hz.
- Gran capacidad de medición en ambientes interiores.
- Rango de medición de 0.15 a 6 metros.

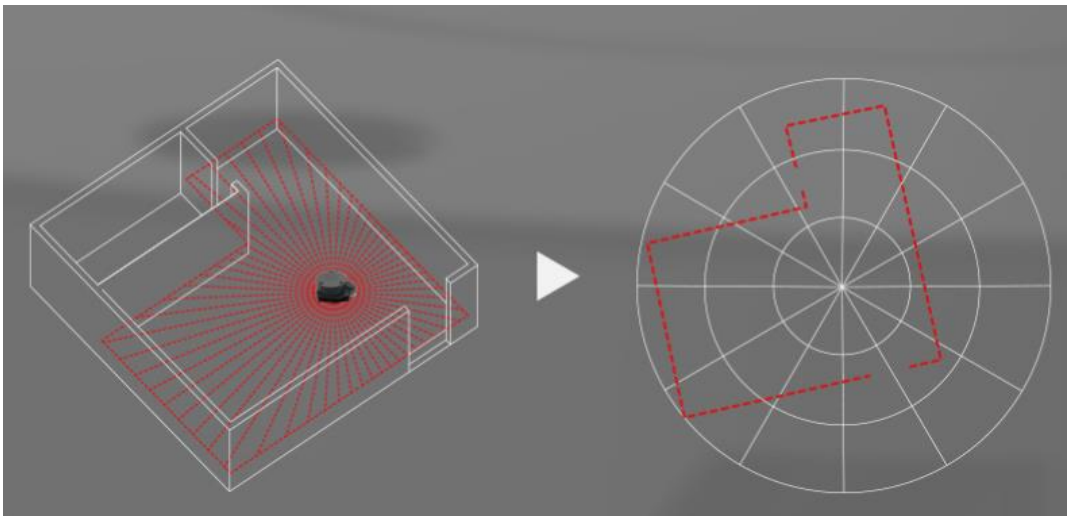
Desventaja

- Costo entorno a los 120 $.
- Consumo de energía significativo.

En la Figura 29 se presenta la gráfica del sensor Lidar en funcionamiento.

Figura 29

Alcance de detección del sensor lidar omnidireccional



- **Anillo de sensores ultrasónicos:** Es un tipo de sensor de proximidad que está basado en la emisión y reflexión de ondas acústicas emitidas, calcula la distancia con respecto al objeto en función al tiempo de retorno de la señal emitida, los pulsos emitidos son proporcionales a la anergia utilizada para su generación. Las principales ventajas y desventajas de los sonares son:

Ventajas

- Presentan una detección frente a cualquier superficie que no absorba el sonido
- EL rango de distancia es de 2 centímetros a 6 metros.
- Presenta una rápida respuesta de 8 [ms].
- Posee una vida útil muy larga porque no existe degradación de los componentes.

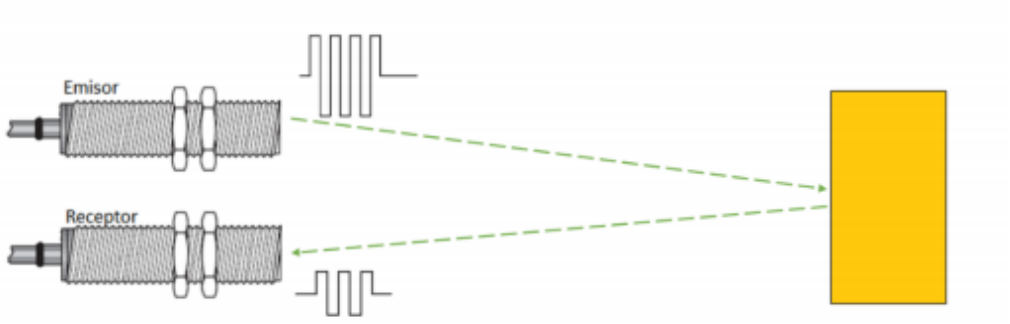
Desventaja

- Se necesita definir un área para cada sensor ultrasónico para que no tenga interferencia de otros sensores.
- La distancia de detección depende mucho de las características de los objetos alrededor.

En la Figura 30 se presenta el principio de funcionamiento del sonar.

Figura 30

Principio de funcionamiento de sensores ultrasónicos



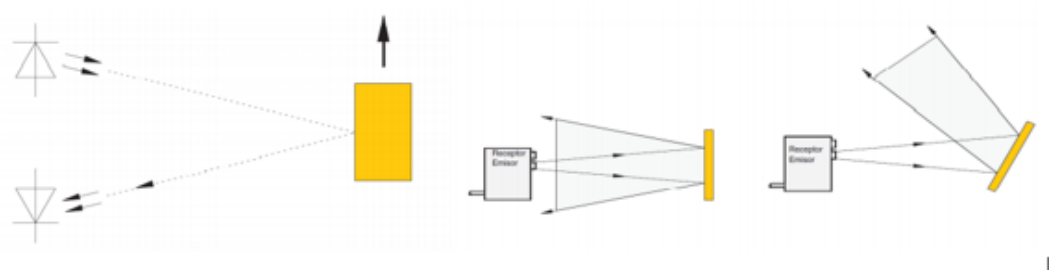
Nota: Tomado de *Estudio de los Sensores para la Detección de Obstáculos Aplicables a Robots Móviles*,

(Colomer, 2018).

- **Anillo de sensores ópticos:** Utilizan medios ópticos y electrónicos para detección de objetos. Para su funcionamiento hacen uso de luz infrarroja, siendo los más utilizados los LED debido a su duración y robustez.

Figura 31

Principio de funcionamiento de sensores ópticos y comportamiento frente diferentes ángulos



Nota: Tomado de *Estudio de los Sensores para la Detección de Obstáculos Aplicables a Robots Móviles*, (Colomer, 2018).

Los más recomendados para el uso en robótica móvil son los de reflexión debido a que tanto el emisor como receptor se encuentran instalados en el robot, las principales ventajas y desventajas de estos sensores son:

Ventajas

- Presentan una rápida respuesta con una frecuencia de conmutación de 2ms.
- Poseen una fácil implementación
- Poseen una gran vida útil aproximadamente de 100000 horas.

Desventaja

- La superficie para detectar debe ser perpendicular al plano del piso.
- La señal emitida se pierde al contacto con superficies transparentes.
- La distancia de detección depende de los objetos alrededor.

En la Figura 31 se presenta el principio de funcionamiento y comportamiento del sensor óptico. Se realiza la evaluación de cada alternativa con relación a los criterios planteados para el subsistema de percepción. En la Tabla 12 se presenta la evaluación de cada una de las alternativas con enfoque en el criterio de costo.

Tabla 12

Matriz de criterio de costo para las alternativas de percepción

Criterio: A	Kinect	Lidar	Sonares	Ópticos	Total	Peso relativo
Kinect		5	0.2	0.2	5.4	0.17
Lidar	0.2		0.2	0.2	0.6	0.02
Sonares	5	5		5	15	0.48
ópticos	5	5	0.2		10.2	0.33
		Suma Total			31.2	1.00

Nota: Se muestra la comparación para el sistema de percepción con el criterio de costo.

De la misma forma se procede para cada uno de los criterios de selección y se pueden observar los resultados parciales en la Tabla 13, Tabla 14, Tabla 15 y Tabla 16.

Tabla 13

Matriz de criterio de distancia de detección para las alternativas de percepción

Criterio: B	Kinect	Lidar	Sonares	Ópticos	Total	Peso relativo
Kinect		0.2	0.2	5	5.4	0.19
Lidar	5		1	5	11	0.39
Sonares	5	1		5	11	0.39
ópticos	0.2	0.2	0.2		0.6	0.02
		Suma Total			28	1.00

Nota: Se muestra la comparación para el sistema de percepción con el criterio de distancia de detección.

Tabla 14

Matriz de criterio de ángulo de detección para las alternativas de percepción

Criterio: C	Kinect	Lidar	Sonares	Ópticos	Total	Peso relativo
Kinect		0.2	0.2	0.2	0.6	0.02
Lidar	5		1	5	11	0.39
Sonares	5	1		5	11	0.39
ópticos	5	0.2	0.2		5.4	0.19
		Suma Total			28	1.00

Nota: La tabla muestra a comparación las alternativas para el sistema de percepción con el criterio de ángulo de detección.

Tabla 15

Matriz de criterio de consumo energético para las alternativas de percepción

Criterio: D	Kinect	Lidar	Sonares	Ópticos	Total	Peso relativo
Kinect		5	0.2	0.2	5.4	0.19
Lidar	0.2		0.2	0.2	0.6	0.02
Sonares	5	5		1	11	0.39
ópticos	5	5	1		11	0.39
		Suma Total			28	1.00

Nota. La tabla muestra a comparación las alternativas según el consumo energético.

Tabla 16

Matriz de criterio de confiabilidad para las alternativas de percepción

Criterio: E	Kinect	Lidar	Sonares	Ópticos	Total	Peso relativo
Kinect		5	5	5	15	0.48
Lidar	0.2		5	5	10.2	0.33
Sonares	0.2	0.2		5	5.4	0.17
ópticos	0.2	0.2	0.2		0.6	0.02
		Suma Total			31.2	1.00

Nota: La tabla muestra a comparación las alternativas según el criterio de confiabilidad.

Después de conocer los pesos relativos en base a la evaluación por criterios se procede a consolidar la selección de la mejor opción. Es decir, se multiplica los pesos relativos de cada alternativa por la ponderación definida de cada criterio. Se suma los pesos de cada alternativa obteniendo un valor total. La alternativa que tenga el máximo valor total será la mejor opción para implementar en el sistema de percepción. En la Tabla 17 se muestra los resultados en cada uno de los criterios.

Tabla 17

Matriz de resultados para las alternativas del subsistema de percepción

	A	B	C	D	E	Total	Orden de selección
Kinect	0.0169	0.0470	0.0078	0.0094	0.1173	0.1984	3
Lidar	0.0019	0.0958	0.1437	0.0010	0.0797	0.3222	2
Sonares	0.0469	0.0958	0.1437	0.0192	0.0422	0.3478	1
Ópticos	0.0319	0.0052	0.0706	0.0192	0.0047	0.1315	4

Nota: La tabla muestra el valor de cada alternativa por el peso ponderado de cada criterio para dar como resultado una sumatoria total de cada alternativa. El orden descendente de la sumatoria representa el orden de selección. Las letras representan los criterios según se observa en la Tabla 10.

De acuerdo con los valores mostrados la alternativa seleccionada es el anillo de sonares. Pero el sensor lidar y los sensores ultrasónicos tienen valores altos en la sumatoria de 0.48 y 0.52 respectivamente. Se puede realizar una implementación conjunta entre ambos sensores para obtener información del entorno y tener una navegación reactiva más robusta.

Subsistema 2: Detección de personas y reconocimiento de gestos

El subsistema de detección y rastreo obtiene la información del sensor RPLidar A1 y el sensor Kinect V1. Se rescata un arreglo de puntos e imágenes de la cámara RGB y de profundidad. Con dichos datos se utiliza un algoritmo de detección de

personas para poder generar la posición relativa con respecto a la posición del robot. Y un algoritmo de reconocimiento de gestos para la interacción del robot con la persona.

El reconocimiento de gestos se realiza con el sensor Kinect a partir de la detección de una persona y sus junturas. Se obtiene la imagen y mediante un algoritmo se compara los datos recogidos con librerías y base de datos de OpenNI. Una vez detectada la persona se reconoce gestos previamente guardados. En base a los gestos el robot realizará distintas acciones. Por otro lado, se implementa un algoritmo de detección de piernas en base a los datos obtenidos por el sensor RPlidar-A1. Este algoritmo posee un conjunto de datos entrenados que permite clasificar los puntos del sensor que corresponden a las piernas de una persona y generando de esta forma un tópico de posición. Las ventajas de la implementación de estos métodos son:

- El algoritmo de detección de piernas permite identificar una persona en cualquier dirección con respecto al robot.
- Modularidad de implementación mediante paquetes de ROS.
- Parámetros reconfigurables.
- Facilidad para reentrenamiento de datos.

Subsistema 3: Localización

El subsistema de localización es el encargado de generar la posición y orientación del robot en el mapa del entorno con respecto a un sistema de coordenadas referencial. Lo cual realiza a partir de los datos obtenidos de los encoders y el sensor IMU MPU650, para poder realizar esta función de localización se hace uso de algoritmos de creación y corrección de odometría en conjunto con la publicación de las transformadas entre el marco referencial y el marco base de la plataforma. En el subsistema de localización se presenta una única alternativa y solución debido a sus características. El algoritmo para localización que se utiliza es AMCL (Adaptative Monte

Carlo Localization). Es un método que usa el filtro de partículas para localización de un robot en un entorno mapeado y se puede utilizar en conjunto con un filtro de Kalman extendido para mejorar los valores de odometría. Las ventajas de la implementación del algoritmo son: la precisión ya que trabaja en un espacio continuo, y que está implementado en el prototipo del proyecto como un paquete de ROS.

Subsistema 4: Planificación global

El subsistema de planificación global se encarga de generar una ruta óptima, que presente la menor distancia y esté libre de obstáculos desde el punto de partida hasta el punto objetivo seleccionado. Se debe tener información sobre un mapa del entorno generado previamente. Para poder implementar la ruta dentro de un entorno social, el planificador toma en cuenta el espacio a respetar alrededor de cada una de las personas dentro del entorno, así como la detección de obstáculos fijos del mapa (columnas, escaleras, desniveles etc.). Dichas consideraciones se encontrarán contempladas dentro de su respectivo mapa de costos. En la planificación global para la navegación del robot en el entorno se contempla el algoritmo a cargo de construir la ruta. Por lo tanto, se realiza una selección del algoritmo a implementarse. Los criterios de selección del algoritmo de planificación global se presentan en la Tabla 18.

Tabla 18

Criterios de diseño del subsistema de planificación global

Criterio	Letra representativa
Costo de computo	A
Facilidad de implementación	B
Ruta más corta	C
Velocidad de convergencia	D

Nota: Las letras representativas de cada criterio serán utilizadas en las tablas del subsistema de planificación global.

En la Tabla 19 se presenta la ponderación de los criterios de evaluación.

Tabla 19

Ponderación de criterios de evaluación

	A	B	C	D	Total	Peso ponderado definido
A		1	0.2	1	2.2	0.14
B	1		1	1	3	0.20
C	5	1		1	7	0.46
D	1	1	1		3	0.20
	Suma Total				15.2	1.00

Nota: La tabla muestra a comparación entre criterios para el subsistema de planificador global, para definir una importancia a cada criterio. Las letras representan los criterios según se observa en la Tabla 18.

Las alternativas propuestas para la implementación del planificador global son 3 algoritmos de planificación de rutas. Se presenta una descripción de cada algoritmo junto con sus ventajas y desventajas.

- **Algoritmo A*:** Es un algoritmo de búsqueda determinístico que utiliza la distancia entre el nodo actual y el nodo objetivo. La función de costo del algoritmo A* está representado por: $f(n) = g(n) + h(n)$, en donde n es el nodo actual y h(n) es el costo estimado entre el nodo actual y el objetivo. El algoritmo tiene tres categorías de nodos: libres, ocupados y los no visitados. Se inicia tomando la posición de partida y se ingresan todas las posiciones libres de obstáculos y a las presentan obstáculos se los coloca en los nodos ocupados. Se analiza cada nodo libre alrededor del punto de partida junto con el costo que poseen con respecto al nodo objetivo. El nodo de menor valor es seleccionado como el siguiente nodo y se repite el ciclo hasta llegar al objetivo. Se presenta las ventajas y desventajas del algoritmo.

Ventajas

- Siempre devuelve la ruta más corta en ambientes de pequeño y mediano tamaño.
- Presenta varios módulos y paquetes para ROS.
- Presenta una rápida respuesta en entornos simples.
- Posee una fácil implementación

Desventaja

- Presenta un gran consumo de cómputo en espacios grandes.
- No considera la cinemática ni la dinámica del robot.

- **Algoritmo RRT (Rapidly-Exploring Random Trees):** Es un planificador global probabilístico con gran uso en planeamiento de rutas. Trata de encontrar una ruta desde un estado inicial (x_0) hasta un estado objetivo (x_f). Mediante la construcción de un árbol de exploración que cubre uniformemente todo el espacio libre de colisión dentro del entorno del robot, donde cada estado de transición hacia una nueva posición (x_{nueva}) es escogido siempre y cuando pertenezca a un conjunto de posibilidades libre de obstáculos en el mapa. La técnica presenta las siguientes ventajas y desventajas.

  Ventajas

  - Toma en cuenta la cinemática y dinámica del robot a utilizarse.
  - Puede ser utilizado como algoritmo para un planificador local.
  - Presenta una rápida respuesta.

  Desventaja

  - Presenta un gran consumo de cómputo en espacios grandes.
  - No siempre genera el camino más corto hacía el objetivo

- Puede quedar atrapado dentro de un bucle sin salida sino se toma en cuenta el número de interacciones.

- **Algoritmo Dijkstra:** Es un algoritmo para determinar caminos cortos. Es una versión simplificada del algoritmo A* eliminado la parte heurística de su función de costos g(n). Trabaja por etapas sin considerar nodos o posiciones futuras y utiliza la técnica “greedy”. La cual usa el principio que para que un camino sea optimo todos los caminos que contengan también deben ser óptimos. Por lo cual se analiza el costo de todos los nodos adyacentes al nodo de partida y se escoge el menor luego se hace lo mismo con el nuevo nodo hasta obtener el camino optimo hasta el nodo objetivo. La técnica presenta las siguientes ventajas y desventajas.

  Ventajas

  - Posee una fácil implementación en pseudocódigo
  - Presenta una rápida respuesta en entornos pequeños.
  - Posee una estructura sencilla

  Desventaja

  - Presenta un gran consumo de cómputo.
  - Uso ineficiente del espacio.
  - Presenta una respuesta tardía para entornos grandes.

La evaluación de cada alternativa con relación a los criterios planteados para el subsistema de planificación global se realiza siguiendo el mismo procedimiento que en el subsistema de percepción. Por lo tanto, se presenta únicamente la matriz de los resultados finales para el subsistema de planificación global. La Tabla 20 muestra los resultados totales de alternativa.

Tabla 20

Matriz de resultados para las alternativas del subsistema de planificación global

	A	B	C	D	Total	Orden de selección
A*	0.0700	0.0191	0.2228	0.0955	0.4075	1
RRT	0.0700	0.1592	0.0149	0.0955	0.3396	2
Dijkstra	0.0047	0.0191	0.2228	0.0064	0.2530	3

Nota: La tabla muestra el valor de cada alternativa por el peso ponderado de cada criterio para dar como resultado una sumatoria total de cada alternativa. Las letras representan los criterios según se observa en la Tabla 18.

De acuerdo con los valores mostrados la alternativa seleccionada es el algoritmo A* para el subsistema de planificación global.

Subsistema 5: Planificación local

El subsistema de planificación local se enfoca en evitar colisiones con los objetos presentes en el entorno que no han sido mapeados previamente. Además, el planificador local usa los datos de la ruta generada en el planificador global y los transforma en señales enviadas a los actuadores para generar el movimiento adecuado del robot. Por lo tanto, el subsistema de planificación local recibe datos de la ruta generada hacia el objetivo e información de los sensores sobre objetos cercanos, y tiene como salida la velocidad que debe seguir el robot durante su navegación. Dentro del planificador local es necesario un algoritmo para evitar colisiones y transformar la ruta generada en el planificador global en señales enviadas a los actuadores. Para la implementación de este subsistema se ha planteado una única alternativa debido a sus características y ventajas de uso.

- **Algoritmo DWA (Dynamic Window Approach):** Es un planificador local que toma en cuenta las restricciones cinemáticas y los parámetros dinámicos del robot. El algoritmo DWA planifica la trayectoria libre de colisión en dos pasos.

Primero, DWA reduce el espacio de búsqueda descartando esas velocidades no alcanzables. Este paso tiene en cuenta tres conjuntos de velocidades: trayectorias circulares, velocidades admisibles y ventana dinámica. Trayectorias circulares, (Vs) consiste en velocidades para el siguiente intervalo de tiempo que no se cruza con un obstáculo. Velocidades admisibles, (Va) representa un conjunto de velocidades que permite al robot frenar antes de llegar a un obstáculo. Mientras que la ventana dinámica, (Vd) consiste solo en velocidades que se pueden alcanzar dentro del siguiente intervalo de tiempo. El espacio de búsqueda (Vr) se restringe mediante la intersección de Vs, Va y Vd. El segundo paso de DWA es maximizar una función objetivo eligiendo las velocidades posibles en Vr desde el paso uno. La función objetivo es la siguiente:

$$G(v,w) = \sigma(\alpha * heading(v,w) + \beta * dist(v,w) + \gamma * vel(v,w))$$

Donde head (v, ω) mide el rumbo del robot con la posición objetivo, dist (v, ω) define la distancia más cercana al obstáculo detectado y vel (v, ω) representa la velocidad de la trayectoria, σ se usa para normalizar los pesos α, β y γ a [0,1].

Ventajas:

- Bajo requisitos de esfuerzo computacional.
- Guía al robot por un camino sin colisiones y reacciona rápidamente.
- Es utilizado con éxito en muchos escenarios del mundo real.
- Utilizado en el paquete de navegación del prototipo del robot.

Subsistema 6: Telepresencia y teleoperación

El subsistema de telepresencia es el encargado de establecer un canal de comunicación entre la plataforma y el operador remoto a través de una transmisión de video y audio, permitiendo entregar una señal de cada uno. El sistema subsistema también es el encargado de establecer un canal de comunicación de datos para

entregar comandos de navegación y puntos objetivo a la plataforma por parte del operador remoto. Para poder implementar la función de telepresencia es necesario un módulo de comunicación. Los componentes del módulo de comunicación permiten mantener una transmisión de audio y video continua entre le operador y el robot móvil. Los criterios para análisis y evaluación de las alternativas de componentes del módulo de transmisión de audio y video se muestran en la Tabla 21.

Tabla 21

Criterios de diseño componentes del subsistema de telepresencia

Criterio	Letra representativa
Costo	A
Interfaz de comunicación	B
Disponibilidad	C
Resolución de pantalla	D
Flexibilidad de implementación	E

En la Tabla 22 se presenta la ponderación de los criterios de evaluación a implementarse.

Tabla 22

Ponderación de criterios de evaluación

	A	B	C	D	E	Total	Peso ponderado definido
A		1	0.2	1	1	3.2	0.11
B	1		1	5	1	8	0.27
C	5	1		5	1	12	0.41
D	1	0.2	0.2		1	2.4	0.08
E	1	1	1	1		4	0.14
			Suma Total			29.6	1.00

Nota: La tabla muestra a comparación entre criterios para el subsistema de telepresencia, módulo de componentes, para definir una importancia a cada criterio. Las letras representan los criterios según se observa en la Tabla 21.

Las alternativas de componentes para implementar el subsistema de telepresencia son:

- **Monitor Externo:** Un monitor externo se basa en establecer una conexión directa de video y audio mediante un puerto mini HDMI. Otorga una implementación modular con el sistema, pero requiere utilizar una cámara y micrófono externo.

  Ventajas

  - Implementación modular.
  - Fácil mantenimiento de componentes.
  - Costo completo alrededor de $100.
  - Mejor calidad de video.

  Desventaja

  - Mayor cantidad de puertos para conexión.
  - Poca disponibilidad.
  - Componentes de mayor tamaño.

En la Figura 32 se presenta un ejemplo del conjunto de componentes necesarios en la alternativa de monitor externo.

Figura 32

Ejemplo de componentes para la alternativa de monitor externo

- **Tablet:** El grupo de componentes presenta una implementación completa de cámara, pantalla, micrófonos y parlantes. La tablet establecerá comunicación con el controlador principal mediante comunicación serial.

  Ventajas

  - Buena disponibilidad.
  - Posee batería integrada.
  - Utiliza únicamente un puerto de conexión.

  Desventaja

  - Costo entorno a los $120 y $145.
  - Mayor complicación en implementación.
  - No presenta un sistema modular

En la Figura 33 se presenta un ejemplo de tablet económica (Amazon Kindle 7) para la segunda alternativa.

Figura 33

Ejemplo de tablet como alternativa

La evaluación de cada alternativa con relación a los criterios planteados para el subsistema de telepresencia, módulo de componentes, se realiza siguiendo el mismo procedimiento que en el subsistema de percepción. Por lo tanto, se presenta

únicamente la matriz de los resultados finales para el módulo de componentes de telepresencia. La Tabla 23 muestra los resultados totales de cada alternativa.

Tabla 23

Matriz de resultados para las alternativas del subsistema de telepresencia - componentes

	A	B	C	D	E	Total	Orden
Monitor Externo	0.0042	0.1351	0.0156	0.0405	0.1299	0.3254	2
Tablet	0.1040	0.1351	0.3898	0.0405	0.0052	0.6746	1

Nota: La tabla muestra el valor de cada alternativa por el peso ponderado de cada criterio para dar como resultado una sumatoria total de cada alternativa. Las letras representan los criterios según se observa en la Tabla 21.

Como se puede observar la alternativa con mayor peso es la opción que integra todos los componentes en la tablet. A pesar de eso, dentro del rango de precio las cámaras integradas en la tablet presentan un bajo rendimiento y mala calidad de imagen por lo que se propone utilizar una cámara externa. La segunda selección de alternativas dentro del subsistema de telepresencia corresponde al método de transmisión (framework) de audio y video en la cual se tomó en cuenta los criterios mostrados en la Tabla 24.

Tabla 24

Criterios de diseño componentes del subsistema de telepresencia, método de transmisión

Criterio	Letra representativa
Dificultad de implementación	A
Flexibilidad	B
Latencia	C
Resolución	D
Seguridad de transmisión	E

En la Tabla 25 se presenta la ponderación de los criterios de evaluación.

Tabla 25

Ponderación de criterios de evaluación

	A	B	C	D	E	Total	Peso ponderado definido
A		1	0.2	1	1	3.2	0.11
B	1		1	5	1	8	0.27
C	5	1		5	1	12	0.41
D	1	0.2	0.2		1	2.4	0.08
E	1	1	1	1		4	0.14
			Suma Total			29.6	1.00

Nota: La tabla muestra a comparación entre criterios para el subsistema de telepresencia, método de transmisión, para definir una importancia a cada criterio. Las letras representan los criterios según se observa en la Tabla 21.

Como alternativas se proponen los siguientes métodos de transferencia de video mediante internet.

- **WEBRTC:** Es un framework de código abierto utilizado para establecer canales de comunicación en tiempo real para transmisiones de video y audio (MediaStream) y canales de transmisión de datos (DataChannel) a partir de la implementación de APIs propietarias. Su principal lenguaje de programación es JavaScript por lo que se puede crear aplicaciones web como interfaz de usuario. El método de transmisión es utilizado para comunicación remota entre dos puntos gracias a la utilización del protocolo ICE mediante servidores STUN y TURN que permiten establecer un canal de comunicación directo.

  Ventajas

  - Seguridad de transmisión de datos, se utiliza encriptación y protocolos de seguridad.

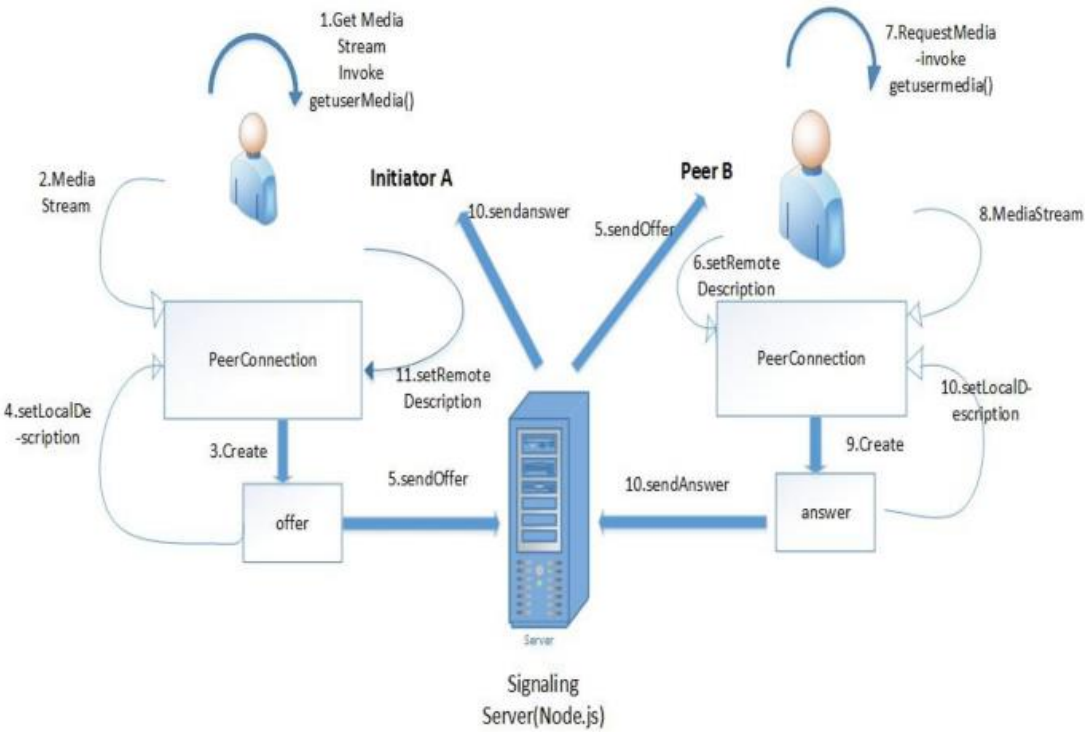
- Mínima latencia entre clientes.
- Resolución de transmisión configurable.
- Gran soporte y compatibilidad mediante APIs.
- Se puede crear varios canales de transmisión de datos.

Desventaja

- Necesidad de un servidor TURN para clientes remotos.
- Complejidad en la curva de aprendizaje.

Figura 34

Estructura básica de transmisión mediante WEBRTC



Nota: Tomado de *Performance Analysis of WebRTC and SIP-based Audio and Video Communication Systems*, (Fowdur et al., 2020).

Ventajas

- Trabaja con formatos de imagen livianos.

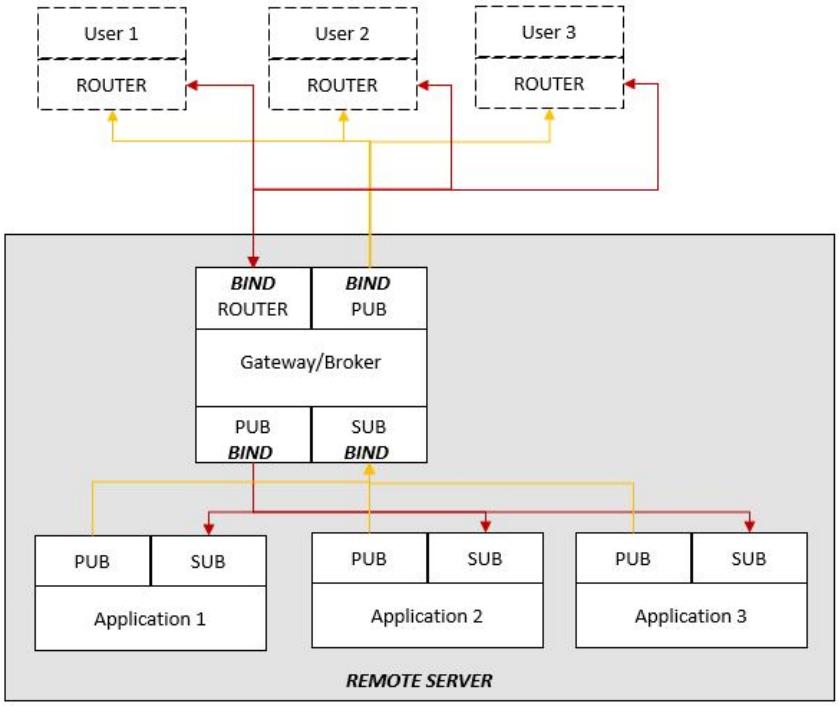
- Posee una rápida transmisión.
- Fácil de implementar usando librerías ZMQ.

Desventaja

- Se necesita habilitar las direcciones IP para transmisiones no locales.
- Menor seguridad en transmisión.
- Baja flexibilidad.
- Consumo computacional medio.

Figura 35

Estructura general mediante de ZMQ (ZeroMQ)



Nota: Tomado de *Pothole Visual Detection using Machine Learning Method integrated with Internet of Thing Video Streaming Platform*, (Rasyid et al., 2019).

- **Skype:** Se basa en la utilización de la herramienta comercial Skype y una seria de plugin programables mediante C++ o Python para poder establecer una

transmisión de video, este método se ha implementado en conexión con robots que utilizan ROS.

Ventajas

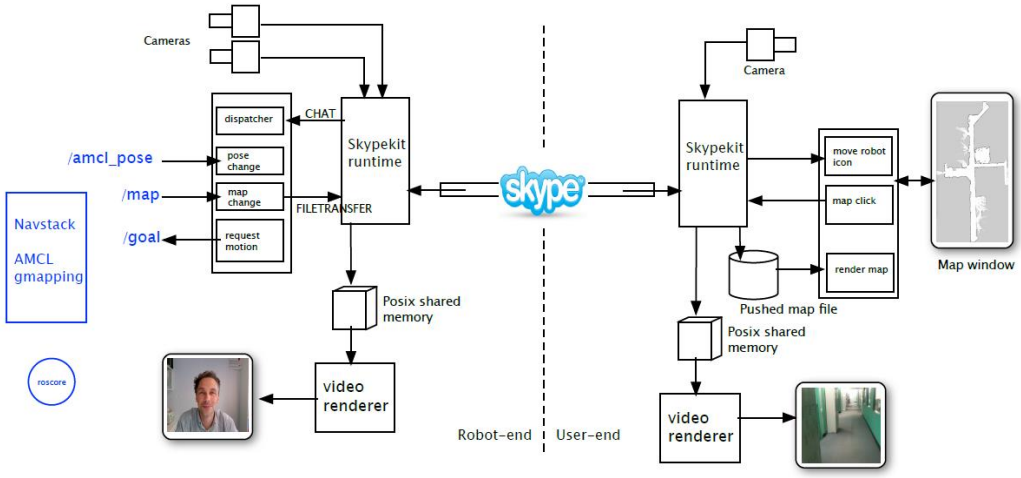
- Gran robustez y seguridad de transmisión.
- Únicamente se programa los plugins necesarios.
- Gran rendimiento en resoluciones medias.
- Fácil conexión con ROS.

Desventaja

- Poca cantidad de información.
- Baja flexibilidad de implementación.

Figura 36

Estructura de transmisión de video utilizando Skype



Nota: Tomado de *Skype: A communications framework for robotics,* (Corke et al., 2012).

La evaluación de cada alternativa con relación a los criterios planteados para el subsistema de telepresencia, método de transmisión de video, se realiza siguiendo el mismo procedimiento que en el subsistema de percepción. Por lo tanto, se presenta

únicamente la matriz de los resultados finales para el método de transmisión de telepresencia. La Tabla 26 muestra los resultados totales de cada alternativa.

Tabla 26

Matriz de resultados para las alternativas del subsistema de telepresencia – método de transmisión de audio y video

	A	B	C	D	E	Total	Orden
WEBRTC	0.052	0.059	0.196	0.052	0.109	0.468	1
ImageZMQ/ZMQ	0.003	0.035	0.013	0.027	0.013	0.092	3
Skype	0.052	0.176	0.196	0.002	0.013	0.440	2

Nota. La tabla muestra el valor de cada alternativa por el peso ponderado de cada criterio para dar como resultado una sumatoria total de cada alternativa. El orden ascendente de la sumatoria representa el orden de selección. Las letras representan los criterios según se observa en la Tabla 24. Como se puede observar la alternativa con mayor peso es el método de transmisión mediante WEBRTC. Utilizando APIs para su integración con aplicaciones Android y JavaScript y rosbrige para la comunicación con ROS.

Unidad de procesamiento y control

Los subsistemas anteriores requieren una interacción para poder formar en su conjunto todo el sistema. Por lo tanto, la interacción entre subsistemas y procesamiento de los datos adquiridos mediante los sensores se realizará en una unidad de procesamiento y control. El computador debe tener la capacidad suficiente para procesar los datos y controlar las funciones del robot. Las funciones incluyen: localización y navegación en el entorno, detección de personas y conectividad inalámbrica que permita la telepresencia. Debido a la autonomía requerida en el sistema, la unidad de procesamiento o computador irá montada en el robot requiriendo poco consumo energético.

Por lo tanto, se realiza una selección de alternativas para implementar la unidad de procesamiento y control en el prototipo. Los criterios de selección de la unidad de control se muestran en la Tabla 27.

Tabla 27

Criterios de selección para la unidad de procesamiento y control

Criterio	Letra representativa
Capacidad Computacional	A
Tamaño	B
Consumo energético	C
Costo	D
Conectividad	E

Nota. Las letras representativas de cada criterio serán utilizadas en las tablas de evaluación para las alternativas de adquisición de datos.

En la Tabla 28 se presenta la ponderación de los criterios de evaluación.

Tabla 28

Ponderación de criterios de evaluación

	A	B	C	D	E	Total	Peso ponderado definido
A		1	0.2	1	1	3.2	0.11
B	1		1	5	1	8	0.27
C	5	1		5	1	12	0.41
D	1	0.2	0.2		1	2.4	0.08
E	1	1	1	1		4	0.14
			Suma Total			29.6	1.00

Nota: La tabla muestra la comparación entre criterios para la unidad de control y procesamiento, para definir una importancia a cada criterio. Las letras representan los criterios según se observa en la Tabla 27.

Las alternativas propuestas para procesamiento y control son tres computadores. Se presenta a continuación una descripción de cada uno.

- **Raspberry Pi 4.0:** Es un miniordenador básico de bajo costo que presenta las siguientes características:
  - ARM Cortex- A72 @ 1.5GHz.
  - GPU VideoCore VI.
  - Consumo de 15,3 W.
  - Conectividad inalámbrica Wi-Fi.
  - 4 puertos USB (2x USB 3.0, 2x USB 2.0).
  - Dimensiones: 85 x 53 mm.
  - Precio: $ 80 (USA).

  La Raspberry 4.0 se muestra en la Figura 37.

Figura 37

Imagen de la Raspberry Pi 4.0

Nota: Tomado de *Raspberry Pi 4 Model B – Raspberry Pi*, (Raspberry pi Foundation, 2020).

- **Jetson Nano:** Es una microcomputadora para desarrollar aplicaciones de inteligencia artificial y robótica que cuenta con las siguientes características:
  - CPU Quad-core ARM57 @ 1.43GHz.
  - GPU 128-core Maxwell.

- Consumo de 5 a 15 W.
- Conectividad inalámbrica.
- 4 Puertos USB3.0.
- Dimensiones: 100 x 80 x 29 mm.
- Precio: $ 100 (USA).

La computadora Jetson Nano se muestra en la Figura 38.

Figura 38

Computadora Jetson Nano

Nota: Tomado de *NVIDIA Jetson Nano Developer Kit*, (NVIDIA, 2019).

- **Intel NUC i3:** Es un computador desarrollado por INTEL con las siguientes características:
  - Procesador i3 de 7ma generación.
  - Consumo de 65W.
  - Wifi incorporado.
  - 6 puertos USB.
  - Dimensiones: 114,3 x 50.8 x 114.3 mm.

- Precio: $ 265 (USA).

La computadora Jetson Nano se muestra en la Figura 39

Figura 39

Computador Intel NUC

Nota: Tomado de *Mini PC Intel® NUC*, (Intel Corporation, s. f.).

La evaluación de cada alternativa con relación a los criterios planteados para la unidad de control se realiza siguiendo el mismo procedimiento que en el subsistema de percepción. Por lo tanto, se presenta únicamente la matriz de los resultados finales. La Tabla 29 muestra los resultados totales de cada alternativa.

Tabla 29

Matriz de resultados para las alternativas de la unidad de Procesamiento y Control

	A	B	C	D	E	Total	Orden de selección
Raspberry Pi 4.0	0.0108	0.1425	0.0667	0.0142	0.0741	0.3082	2
Jetson Nano	0.1613	0.0741	0.1282	0.0074	0.0741	0.4451	1
Intel NUC	0.1613	0.0057	0.0051	0.0006	0.0741	0.2468	3

Nota: La tabla muestra el valor de cada alternativa por el peso ponderado de cada criterio para dar como resultado una sumatoria total de cada alternativa. Las letras representan los criterios según se observa en la Tabla 27.

Como se puede observar la alternativa con mayor peso es la microcomputadora Jetson Nano que cumplirá la función de procesar los datos y controlar la navegación del robot diferencial.

Resumen

El capítulo de diseño detalla inicialmente las necesidades que debe suplir el sistema. Basado en las necesidades se definen las especificaciones. Y partiendo de las características del prototipo inicial se realiza la división del sistema en cinco subsistemas. En cada subsistema se plantea alternativas y se elige la opción mejor puntuada de acuerdo con la matriz de selección y los criterios planteados.

El subsistema de percepción se encarga de recibir los datos del entorno e internos del robot, para llevarlos a la unidad de procesamiento y control. En este subsistema se incluye los sensores para realizar navegación reactiva como es el Lidar y un anillo de sensores ultrasónicos. Para conocer la localización del robot se puede utilizar los datos de los encoder y del sensor IMU. El subsistema de detección y rastreo hace uso de la información obtenido del Lidar para detectar a las personas en su entorno. Y para el reconocimiento de gestos utiliza librerías de OpenNI que reciben los datos de la Kinect.

En el subsistema de localización se utiliza el algoritmo AMCL para conocer la posición del robot en su entorno, y para mejorar el cálculo de odometría se plantea el uso de un filtro de Kalman. En los subsistemas de planificación global y local se eligen los algoritmos A* y DWA respectivamente. El algoritmo A* genera el camino desde el punto inicial al objetivo y el DWA genera la velocidad admisible del robot para no tener colisiones y evadir los obstáculos siguiendo al planificador global.

En el subsistema de telepresencia y teleoperación del robot se plantea como mejor opción de comunicación un módulo que incluya micrófono, cámara y pantalla táctil

para integrar en el robot. Para el método de transmisión de audio, video y datos se escoge WEBRTC, en integración con APIs para la aplicación móvil y comunicación con ROS. Finalmente, como unidad de control se escoge a la microcomputadora Jetson Nano debido a su bajo consumo de energía, gran capacidad para procesar algoritmos inteligentes y su conectividad con los sensores y actuadores.

Capítulo IV

Desarrollo e implementación

El capítulo describe la implementación de cada subsistema en el prototipo móvil diferencial. Se evalúa primero el prototipo en su funcionalidad y consumo de energía, y se solventan los problemas que intervengan en la implementación del sistema planteado. El apartado muestra también la modificación de la estructura física del robot. Y se explica la integración de nuevos sensores, tarjetas de adquisición y unidad de procesamiento.

El diseño del controlador de los motores se muestra junto con sus cálculos. Además, se describe cada algoritmo del sistema y su función en la navegación del robot. Se presenta la configuración para la comunicación mediante WEBRTC y finalmente se da a conocer la arquitectura del sistema implementado.

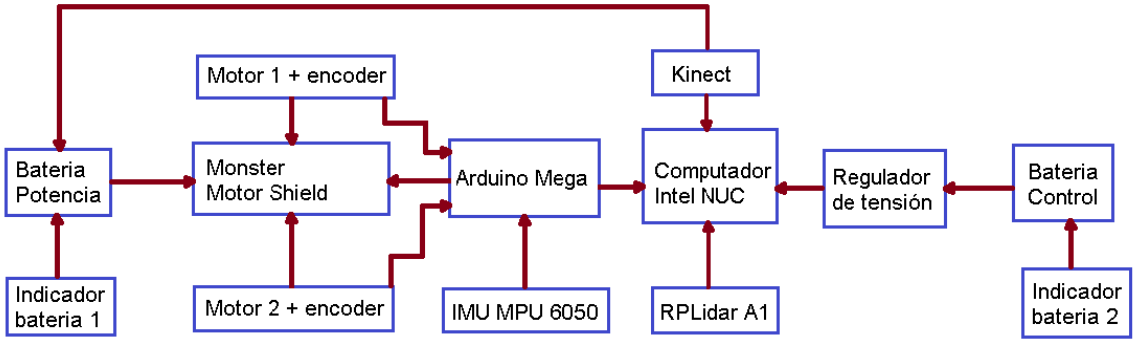
Evaluación del prototipo inicial

Para realizar la implementación del sistema de navegación y telepresencia se procede a evaluar el prototipo recibido inicialmente. Con la evaluación se conocerá las limitaciones y capacidades del robot diferencial inicial. Las limitaciones que influyan para realizar el sistema se solventarán.

Se presenta en la Figura 40, el diagrama electrónico de conexiones del prototipo para dar una guía de la comunicación y transferencia de datos entre sensores, actuadores, tarjeta de adquisición, unidad de procesamiento y controlador de motores.

Figura 40

Diagrama electrónico de conexiones del prototipo inicial



Nota: Tomado de *Implementación de reglas de comportamiento social en una plataforma robótica de telepresencia, a través del reconocimiento de gestos*, (Andrango, 2020).

Análisis energético

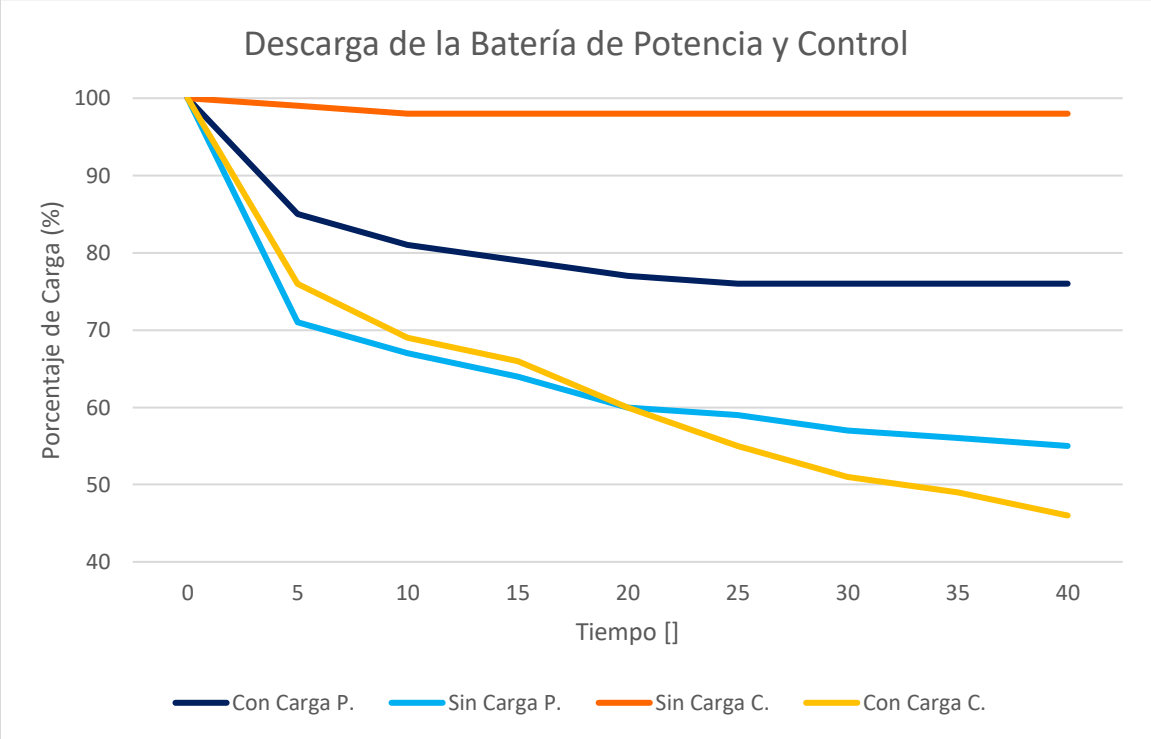
Como muestra la Figura 40, la alimentación de los elementos electrónicos se hace por medio de dos baterías de 12 VDC y 7Ah. Una batería alimenta el controlador para los motores y el sensor Kinect. La otra batería a través de un regulador de tensión a 12 Voltios alimenta al computador Intel NUC. La navegación autónoma del robot depende de la capacidad energética de las baterías para funcionar por determinado tiempo. Por lo tanto, se tomó datos del consumo durante 40 minutos para conocer la funcionalidad de las baterías. En la Figura 41 se muestra las gráficas de descarga de las baterías.

En la Figura 41 se observa la curva de descarga de la batería de potencia con carga (celeste) y sin carga (azul oscuro). Por la curva naranja se puede notar que la batería está degenerada por su uso, pues sin elementos que consuman energía la batería pierde más del 20% de su carga total. Mientras que, de la curva gris se puede observar en los primeros 5 minutos se descarga el 30% de la batería. En los minutos

posteriores, en determinados momentos la batería se descarga más rápido y es debido a que su consumo no es constante pues la corriente requerida por los motores depende de la velocidad a la que se necesita que avancen.

Figura 41

Curvas de descarga de baterías de potencia y control



En la Figura 41 se observa la curva de descarga de la batería de control con carga (amarillo) y sin carga (naranja). Por la curva azul se puede notar que la batería todavía conserva sus propiedades, pues sin elementos que consuman energía la batería pierde menos del 5% de su carga total. Mientras que, de la curva gris se puede observar en los primeros 5 minutos se descarga cerca del 30% de la batería. En los minutos posteriores, la batería se descarga más rápido que la batería de potencia debido a que el computador consume cerca de 7 Amperios y de manera constante. De acuerdo con la curva y a sus pruebas, la batería tiene un tiempo de duración de descarga de 1 hora.

Adquisición de datos

En (Andrango, 2020) se muestra que para adquirir los datos de los encoders y de la unidad de medición inercial (IMU) se utiliza una tarjeta de adquisición Arduino Mega y mediante los puertos USB del computador NUC se recibe los datos del sensor Kinect y del LIDAR. La adquisición de los datos del sensor Lidar se realiza con el paquete público “rplidar_ros” que implementa un nodo con la capacidad de convertir los datos adquiridos en un mensaje de ROS. Para controlar los actuadores (motores) se envía la señal PWM a través del microcontrolador Arduino Mega hacia el controlador de los dos motores.

Para el envío de datos entre el Arduino y la computadora se utiliza un puente serial realizado en Python (Joseph, 2015). La información enviada mediante el puente serial se convierte en tópicos que son utilizados en ROS para localización y navegación. Dadas las características de funcionamiento del Arduino Mega, los mensajes de datos de los encoders, IMU y motores se envían a un máximo de 14hz mediante el puente de comunicación. Tal frecuencia de transmisión es insuficiente para una navegación apropiada, presentando errores en convergencia con los paquetes ROS.

Además, la principal limitación en la transmisión de datos se debe a la caída prematura del puente serial entre el microcontrolador y la unidad de procesamiento NUC. Pues la comunicación tiene una duración aproximada de 6 minutos y 20 segundos, lo que impide la funcionalidad prolongada del prototipo.

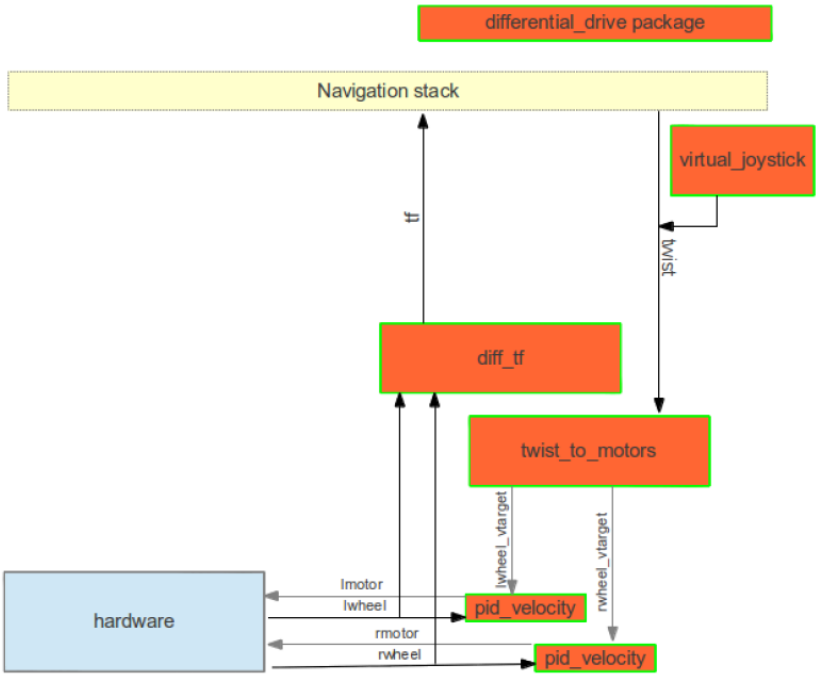
Localización y navegación

Para realizar la navegación en el entorno, el prototipo realiza primero un mapeo mediante SLAM (localización y mapeo simultáneos). Una vez generado y guardado el mapa el robot puede ubicarse y navegar en el entorno. El principio de localización y navegación de la plataforma robótica recibida está basado en el paquete

“differential_drive” desarrollado por (Stephan, 2012) y el proyecto realizado por (Joseph, 2015). El propósito de este paquete, basado en ROS, es proveer una interfaz para el control de robots con navegación diferencial. El diagrama de este paquete se muestra en la Figura 42.

Figura 42

Diagrama de nodos del paquete differential_drive en ROS



Nota: Tomado de *differential_drive - ROS Wiki*, (Stephan, 2012).

La localización se realiza en el nodo “diff_tf” que es el encargado de generar la odometría del robot. El nodo recibe los pulsos de los encoders y calcula la posición y velocidad del robot utilizando las ecuaciones de un robot diferencial. El cálculo de odometría depende de la distancia entre las ruedas, diámetro de la circunferencia de las ruedas y pulsos por revolución de los encoders.

Para tener mayor precisión entre la odometría calculada y la odometría real se debe incluir más sensores que aporten información de la ubicación del robot en su entorno. Pero en el proyecto de (Andrango, 2020) no se incluye los datos del sensor

inercial para el cálculo de la odometría. En la navegación, el nodo “twist_to_motors”, traduce las velocidades lineal y angular enviadas por el paquete de navegación de ROS en velocidades independientes para cada uno de los motores. De esta manera la velocidad para cada motor se puede control mediante el nodo “pid_velocity” utilizando como “set point” Los tópicos “lwheel_vtarget” y “rwheel_vtarget” enviados por el nodo “twist_to_motors”.

Para la planificación de trayectorias se utiliza el paquete de ROS “move_base”, desarrollado por (Lu & Marder-Eppstein, 2013). El paquete puede generar una trayectoria para navegación autónoma mientras el robot evita obstáculos, siempre que se dé una estimación o pose inicial del robot en un mapa. Para las tareas de navegación se utiliza los algoritmos de “global_planner” y “local_planner”. Para interpretar la información de los sensores y reconocer obstáculos se utiliza los algoritmos “global_costmap” y “local_costmap”. Todos los algoritmos permiten la configuración de determinados parámetros para lograr la navegación esperada por el usuario (Lu & Marder-Eppstein, 2013).

Conocidos los algoritmos y paquete utilizados en el prototipo inicial se procedió a evaluar la localización y navegación de la plataforma. Debido a la falta de datos para la localización en el entorno, se observó que el robot fácilmente perdía coherencia entre lo calculado y la ubicación real. Los datos se muestran en las pruebas de odometría del capítulo V.

La navegación del robot se puede realizar, pero por un corto tiempo (aproximadamente 5 min) por dos razones principales.

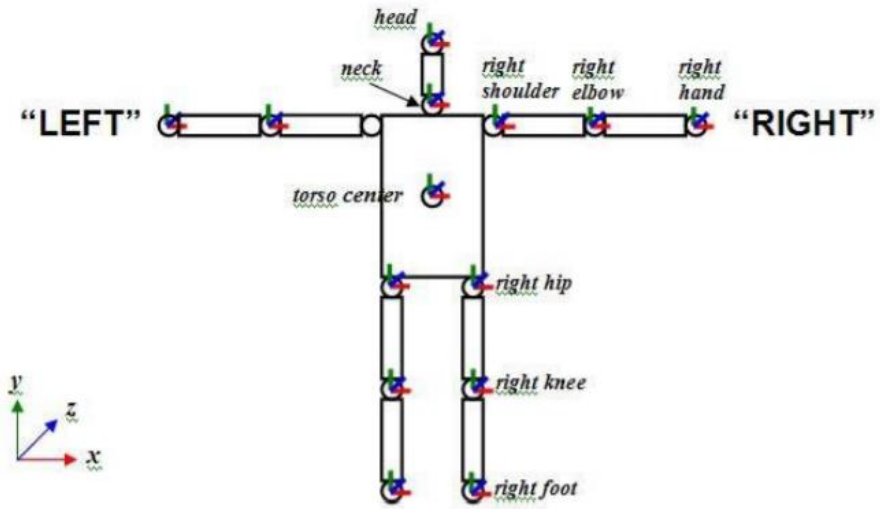
- La pérdida de comunicación entre la tarjeta de adquisición (Arduino Mega) y la unidad de procesamiento (Intel NUC) debido a la caída del puente serial.
- La falta de información para la correcta ubicación del robot en su entorno.

Reconocimiento de gestos

Para el reconocimiento de gestos en (Andrango, 2020) se utilizó el paquete “skeleton_markers”, mediante el cual se obtiene el dato de imagen y profundidad del sensor Kinect V1. Se basa en el uso de OpenNI, que provee APIs de código abierto para reconocimiento de gestos, voz, y movimientos del cuerpo para dispositivos de interacción natural. Primero se rescata los datos del sensor de profundidad y se procesa con la ayuda de la librería PCL.

Figura 43

Conjunto de articulaciones generadas al reconocer personas



Nota: Tomado de *Neural Network Based Gesture Recognition Robot*, (Yuan, 2017).

Al procesar los datos se tiene un conjunto de puntos localizados en un sistema de coordenadas definidos en los ejes x, y, z. Además, la librería provee una función básica de reconocimiento de personas y permite su posterior procesamiento.

Después de obtener la nube de puntos del sensor Kinect, los datos pasarán por el nodo “skeleton_marker” y proporcionarán un esqueleto humano con las coordenadas tridimensionales de cada articulación (Goebel, 2015). En la Figura 43 se muestra las articulaciones obtenidas al procesar los datos y reconocer el esqueleto humano.

Para modificar los gestos, se ha editado la base de datos creando 6 gestos con el paquete “neuro_gesture_kinect” (Parhar, 2015). El cual es un paquete de reconocimiento de gestos basado en redes neuronales. Utiliza una red neuronal Feed-Foward. Los datos que se recopilan son la posición de las articulaciones en el espacio 3D, con la ayuda del sensor Kinect. El paquete consta de 3 partes:

- Generación de un conjunto de datos de Kinect para el reconocimiento de gestos.
- Entrenamiento del conjunto de datos con una red neuronal Feed-Forward.
- Predecir los gestos que se dan como entradas.

La respuesta de la implementación del reconocimiento de gestos con la base de datos creada es satisfactoria como se muestra en las tablas de desempeño del algoritmo en (Andrango, 2020). Por lo que se procede a usar el algoritmo en la implementación del sistema de navegación social.

La descripción y evaluación del prototipo inicial permitió conocer las funciones que requieren una corrección o mejora. A continuación, se listan las correcciones y mejoras que se realizan en el prototipo, para que sea factible la integración del sistema de navegación social y telepresencia:

- Unidad de procesamiento con menor consumo energético para aumentar el tiempo de autonomía de la plataforma.
- Cambio del medio de comunicación entre la tarjeta de adquisición y la unidad de procesamiento.
- Aumento de una tarjeta de adquisición con mayor frecuencia de funcionamiento que la utilizada en el prototipo inicial (Arduino Mega).
- Integración de las medidas del sensor inercial y un filtro EKF para el cálculo de odometría.

Actualizaciones del prototipo

Después de evaluar el prototipo recibido y conocer las alternativas seleccionadas para cada subsistema, se procede a describir las actualizaciones realizadas tanto en hardware como en software en la plataforma. Inicialmente se actualizó el sistema operativo en el cual se desarrolla la implementación debido al cambio en la unidad de procesamiento y control del robot. La microcomputadora Jetson Nano solo permite la instalación del sistema operativo Ubuntu 18.04 LTS por lo que se actualizó el sistema de Ubuntu 16.04 a Ubuntu 18.04. El cambio de la versión del sistema operativo implicó también un cambio en la versión de ROS de Kinetic a Melodic, y la migración de los paquetes utilizados en el prototipo.

Para el subsistema de percepción se aumentó otra tarjeta de adquisición para poder procesar los datos de los encoders y el IMU a mayor frecuencia que el Arduino Mega. La tarjeta de adquisición Teensy 3.6 que se aumentó, mostrada en la Figura 44, trabaja a 180MHz, casi 10 veces mayor que los 16MHz de frecuencia de reloj del Arduino Mega. El Arduino cambió su función para recibir los datos del anillo de sonares seleccionado para reforzar la detección de obstáculos en la navegación reactiva y una posible expansión a un mayor número de sensores.

Figura 44

Tarjeta de desarrollo Teensy 3.6

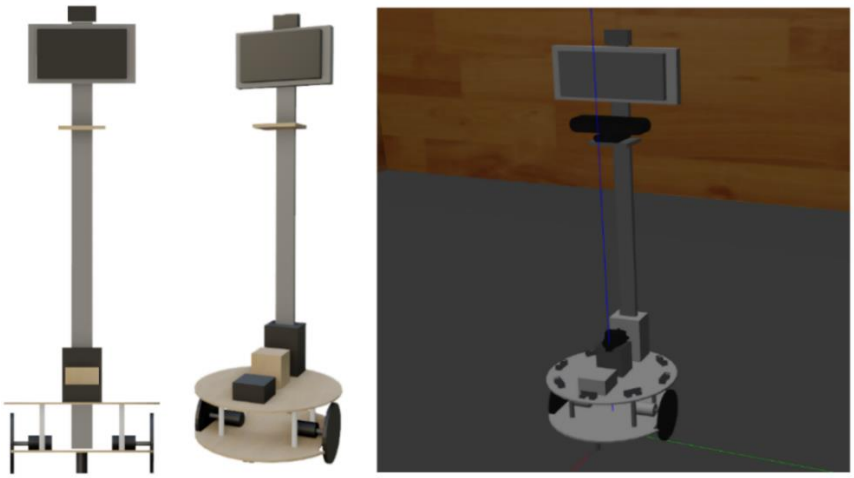
Nota: Tomado de *Teensy® 3.6 Development Board*, (PJRC, s. f.)

Modelo de la plataforma

Para la implementación del sistema propuesto se utilizó una serie de modificaciones físicas al prototipo inicial. Entre estas modificaciones se encuentran los soportes tanto para los sensores ultrasónicos y el soporte para el sensor Kinect y la cámara WEB. Junto con estos soportes se añadió un perfil de aluminio de 110 [cm] de altura, 5.5 [cm] de ancho y 2.7 [cm] de profundidad.

Figura 45

A-B) Modelo 3D implementado en Bender, C) Modelo en simulación de Gazebo



El soporte se implementó para poder colocar la pantalla, cámara y sensor Kinect como se muestra en la Figura 45, brindando al robot una altura máxima de 130 [cm] para colocar la pantalla táctil a una altura que sea confortable para la persona frente al robot.

En las Figura 46, Figura 47 y Figura 48 se muestra la implementación de las modificaciones físicas del robot. En la Figura 46 se observa el soporte para el perfil que sostiene a los elementos superiores. También se observa los soportes elaborados en acrílico y MDF para los ocho sonares. En la Figura 47 se presenta los elementos de la parte superior. En orden descendente está la cámara USB con micrófono integrado, la

pantalla táctil y el sensor Kinect. Cada uno tiene un soporte elaborado en MDF y la pantalla está unida al perfil con un suporte VESA metálico. Se muestra la estructura física completa en la Figura 48 vista lateral y frontal.

Figura 46

Estructura de la base del robot

Figura 47

Estructura de la parte superior del robot

Figura 48

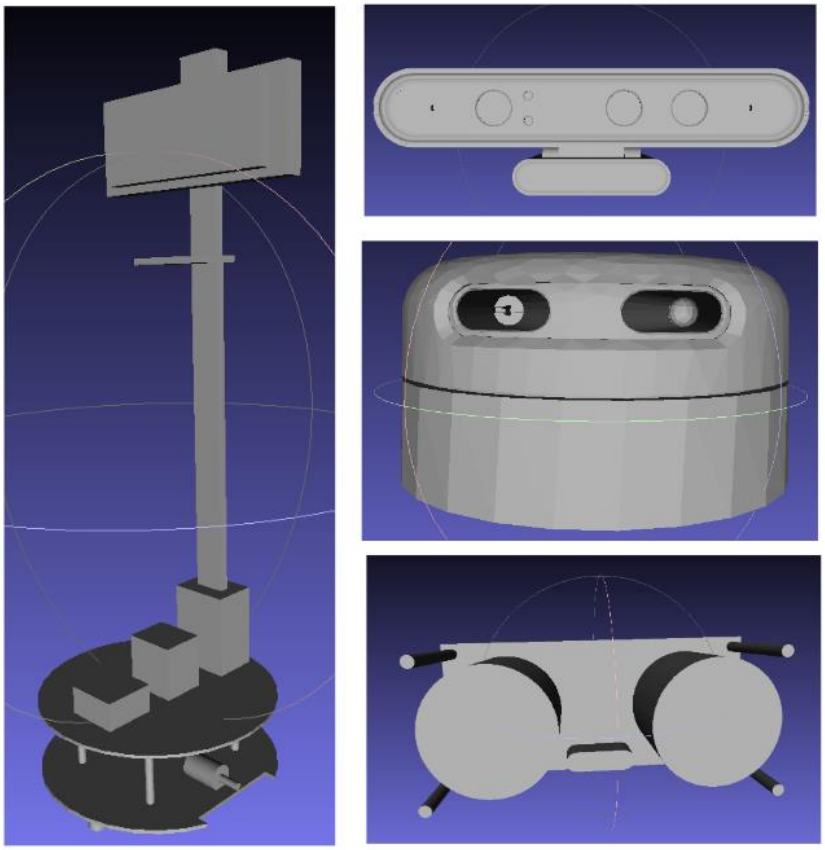
Estructura completa del robot

Gracias a las herramientas y librerías brindadas por ROS se implementó un modelo 3D de la plataforma mediante el formato URDF en la cual se representó la descripción física, cinemática, dinámica y el modelo de colisión y visual del robot. El modelo consta de dos partes en general. Los eslabones (links) que describen partes o secciones de cuerpo rigió con parámetros de colisión e inercia, los elementos están vinculados al robot mediante transformadas que indican su posición y orientación. Y las

uniones (junturas) que son las encargadas de conectar los eslabones y especificar si la conexión podrá tener movimiento o será estática. Para el desarrollo de los elementos virtuales se utilizó el software de diseño 3D Blender versión 3.01 mediante el cual se exporto los modelos visuales como se indica en la Figura 49.

Figura 49

Modelos visuales 3D generados en Blender



En adicción a los modelos físicos se implementó la simulación de sensores IMU, encoders, motores, sensor Lidar y Kinect V1 mediante plugins de Gazebo con el cual se crea una conexión y envió de mensajes a ROS permitiendo crear una simulación de la plataforma móvil como se observar en Figura 50.

Figura 50

Ejemplo de simulación del modelo 3D en RVIZ

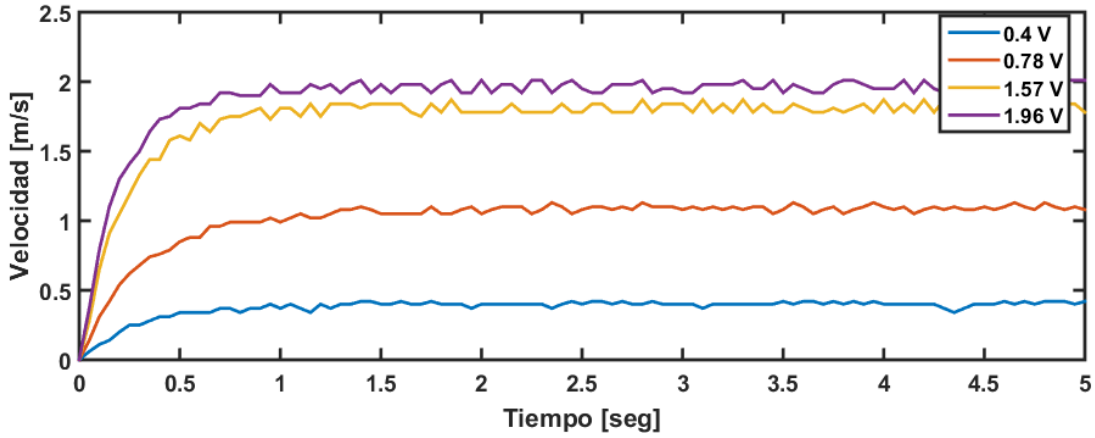
Diseño del controlador de motores

El control de bajo nivel es el control sobre los actuadores del robot, es decir sobre los motores de las ruedas. El controlador, por lo tanto, será un PID que actuará sobre los motores, los cuales tienen una entrada de voltaje y salida de velocidad. Para realizar el controlador primero se identificó el comportamiento de los motores con una entrada de voltaje determinada. Se tomó datos con el microcontrolador de la salida de cada motor para diferentes valores de entrada de voltaje escalón con un tiempo de muestreo de 50 ms. Se explica el proceso de identificación del motor derecho indicando que se sigue el mismo proceso para el motor izquierdo.

La Figura 51 muestra el comportamiento del motor de la rueda derecha del robot. La curva en azul representa la velocidad del motor con una entrada de 0.4 V. La curva en rojo representa la velocidad con una entrada de 0.78 V. La curva en amarillo representa la velocidad con una entrada de 1.57 V. Y la curva en morado representa la velocidad con una entrada de 1.96 V.

Figura 51

Curvas de velocidad del motor derecho para diferentes entradas escalón



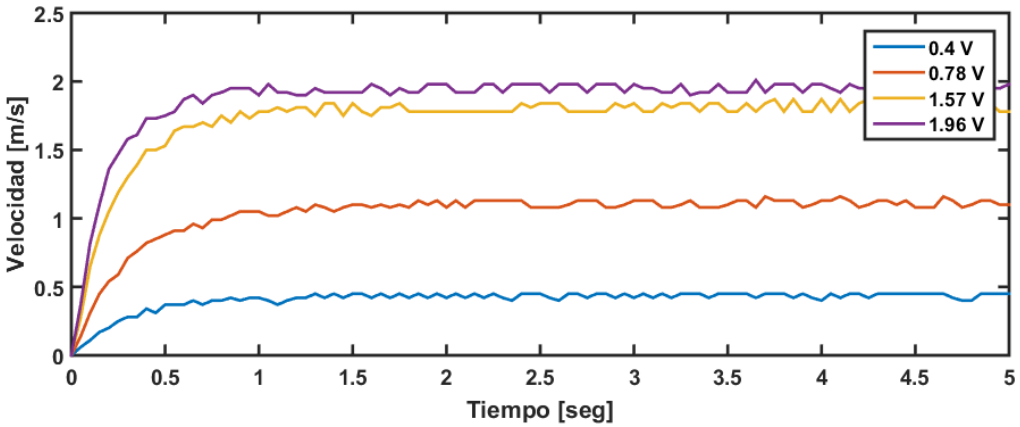
Se realizó la identificación del motor derecho con la función de Matlab “arx”. La función permite obtener una estimación del modelo de la planta en tiempo discreto ingresando el vector de entrada, de la salida, tiempo de muestreo y el orden del polinomio requerido. En este caso se configuró para encontrar una estimación de segundo orden, por lo tanto, la función de transferencia encontrada para el motor derecho fue:

$$Gzr = \frac{0.1947 \cdot z - 0.1313}{z^2 - 1.497 \cdot z + 0.5419} \qquad (1)$$

La estimación del modelo tiene un ajuste de 88.46% con los datos medidos por lo que se procede a usar la función de transferencia encontrada para sintonizar el controlador. El mismo proceso se aplicó para el motor izquierdo por lo que se indica en la Figura 52 el comportamiento del motor de la rueda izquierda del robot.

Figura 52

Curvas de velocidad del motor izquierdo para diferentes entradas escalón



Utilizando la función de estimación de Matlab se obtuvo la siguiente función que representa al motor izquierdo

$$Gzl = \frac{0.1785 \cdot z - 0.1098}{z^2 - 1.463 \cdot z + 0.5125} \qquad (2)$$

La estimación del modelo tiene un ajuste de 89.62% con los datos medidos por lo que se procede a usar la función de transferencia encontrada para sintonizar el controlador. La sintonización o diseño del controlador se realiza con el método de Ziegler-Nichols partiendo de la ecuación característica de la planta y reemplazando los valores de la siguiente función de transferencia. Sabiendo que la planta es de segundo orden:

$$Gz = \frac{b_1 \cdot z + b_2}{z^2 + a_1 \cdot z + a_2} \qquad (3)$$

Al estar en lazo cerrado para encontrar los valores críticos de ganancia y período se iguala la ecuación característica a 0.

$$\Delta(z) = z^2 + a_1 \cdot z + a_2 + Ku \cdot (b_1 \cdot z + b_2) = 0 \qquad (4)$$

Y al saber que los polos de la función tienen una parte real e imaginaria, siendo estos:

$$Z_{1,2} = \exp(\pm j\omega_o \cdot T) = \cos(\omega_o \cdot T) \pm \mathrm{j} \cdot \sin(\omega_o \cdot T) \quad (5)$$

Se hace:

$$\alpha = \cos(\omega_o \cdot T) \quad (6)$$

$$\beta = \sin(\omega_o \cdot T) \quad (7)$$

Por identidad trigonométrica:

$$\beta^2 = 1 - \alpha^2 \quad (8)$$

Se reemplaza 4, 5 y 6 en 3 obteniendo la siguiente ecuación:

$$\Delta(\alpha + j\beta) = (\alpha + j\beta)^2 + a_1 \cdot (\alpha + j\beta) + a_2 + Ku \cdot (b_1 \cdot (\alpha + j\beta) + b_2) \quad (9)$$

Igualando a cero cada uno de los términos real e imaginario de la ecuación se tiene:

$$\alpha^2 - \beta^2 + a_1 \cdot \alpha + a_2 + Ku \cdot b_1 \cdot \alpha + Ku \cdot b_2 = 0 \quad (10)$$

$$2 \cdot \alpha \cdot \beta + a_1 \cdot \beta + Ku \cdot b_1 \cdot \beta = 0 \quad (11)$$

Reemplazando 7 en 9 se obtiene que:

$$Ku = \frac{1 - a_2}{b_2} \quad (12)$$

Y hallando el valor de alfa de la ecuación 10 se tiene:

$$\alpha = -\frac{1}{2} \cdot (Ku \cdot b_1 + a_1) \quad (13)$$

De la frecuencia natural se deduce el período crítico que es:

$$Tu = \frac{2 \cdot \pi \cdot T}{\cos^{-1}(\alpha)} \quad (14)$$

Las ecuaciones (12), (13) y (14) se utilizan para encontrar la ganancia y período crítico de cada motor reemplazando los valores de su función de transferencia.

Entonces se encuentra que los valores críticos para el motor de la derecha son:

$$Ku_r = \frac{1 - 0.5419}{0.1313} = 3.48 \quad (15)$$

$$\alpha_r = -\frac{1}{2} \cdot (3.48 \cdot 0.1947 - 1.497) = 0.4097 \quad (16)$$

$$Tu_r = \frac{2 \cdot \pi \cdot 0.05}{cos^{-1}(0.4097)} = 0.273 \quad (17)$$

Y los valores críticos del motor de la izquierda son:

$$Ku_r = \frac{1 - 0.5125}{0.1098} = 3.66 \quad (18)$$

$$\alpha_r = -\frac{1}{2} \cdot (4.44 \cdot 0.1785 - 1.463) = 0.3352 \quad (19)$$

$$Tu_r = \frac{2 \cdot \pi \cdot 0.05}{cos^{-1}(0.3352)} = 0.2556 \quad (20)$$

Se reemplaza los valores críticos en las fórmulas de sintonización por Ziegler-Nichols obtenida de (Patki et al., 2013).

Tabla 30

Parámetros de sintonización del controlador por Ziegler-Nichols

Coeficiente	PID
Ganancia proporcional (Kp)	0.6 Ku
Tiempo integral (Ti)	0.5 Ti
Tiempo derivativo (Td)	0.125 Tu

Nota: Recuperado de *Design and Implementation of Discrete Augmented Ziegler-Nichols PID Controller*, (Patki et al., 2013).

Aplicando la fórmula de la Tabla 30 se obtienen los siguientes coeficientes del controlador del motor derecho:

$$Kp = 2.08 \quad (21)$$

$$Ti = 0.136 \quad (22)$$

$$Td = 0.034 \quad (23)$$

Y los coeficientes del controlador del motor izquierdo son:

$$Kp = 2.2 \quad (24)$$

$$Ti = 0.12 \quad (25)$$

$$Td = 0.032 \quad (26)$$

Se conoce que la sintonización por Ziegler-Nichols plantea un controlador con un sobre impulso menor al 25% y un tiempo de establecimiento relativamente corto. Pero para la aplicación en los motores del robot se requiere afinar los parámetros para tener un movimiento suave sin reacciones bruscas provocadas por el sobre impulso y un tiempo de establecimiento máximo de 0.5 segundos.

Por lo tanto, tomando como referencia los coeficientes calculados, se procede a afinar los coeficientes del controlador variando los valores y verificando el funcionamiento con cada uno de los motores hasta obtener un comportamiento adecuado para la navegación. Se ajustó los mismos valores para los controladores, dado que se utiliza un paquete que recibe similares coeficientes en ambos controladores y debido a la similitud en el comportamiento de los motores. Los valores finales para los coeficientes de los controladores son los siguientes:

$$Kp = 1.8 \quad (27)$$

$$Ti = 0.2 \quad (28)$$

$$Td = 0.03 \quad (29)$$

En la Figura 53 se muestra la respuesta de los motores sin contacto con el suelo después de aplicar los controladores. Y la Figura 54 muestra la respuesta de los motores al estar el robot sobre el suelo. En las gráficas la línea roja muestra la velocidad enviada por los comandos de teleoperación, simulando una señal escalón. Y la curva azul representa la velocidad lineal de la plataforma, obtenida por el cálculo del encoder de ambos motores.

Figura 53

Respuesta de los motores sin contacto con el suelo

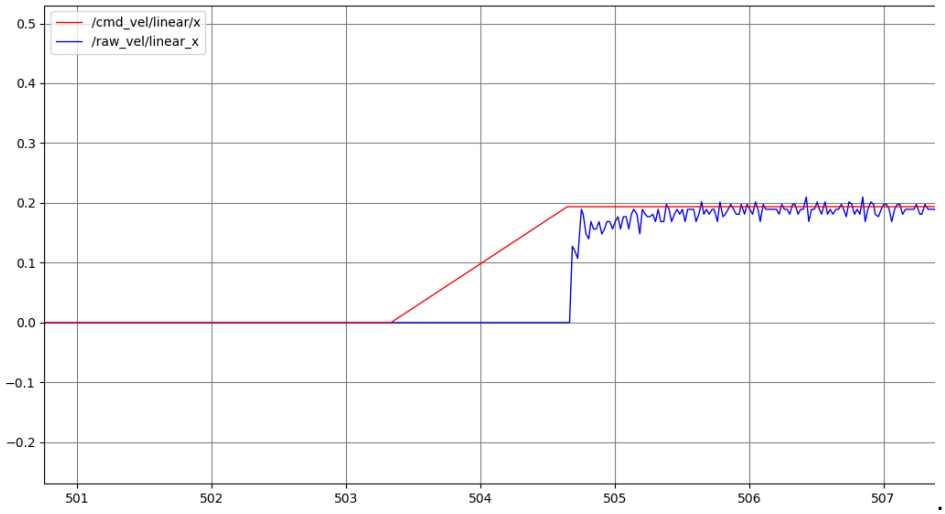
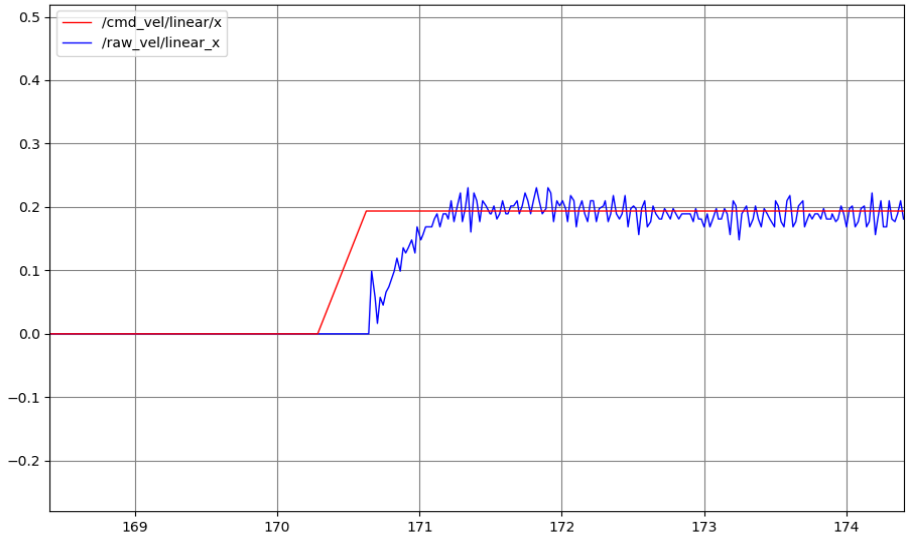


Figura 54

Respuesta de los motores sobre el suelo



Percepción

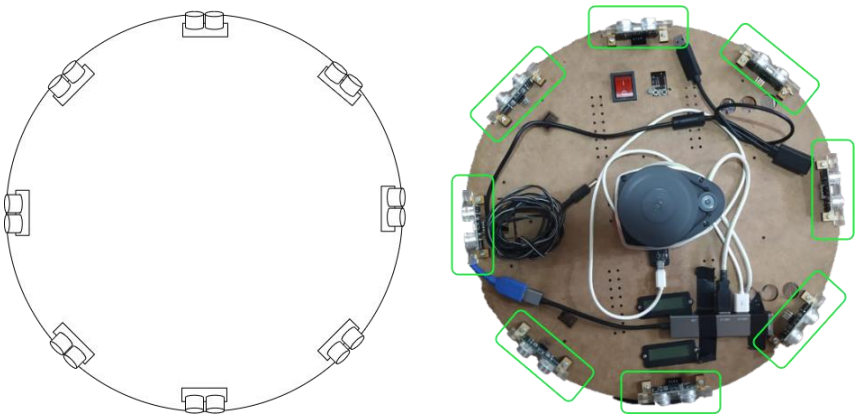
En la implementación correspondiente a la percepción se indica la integración del anillo de sonares, el cambio en comunicación entre las tarjetas de adquisición, la

unidad de procesamiento (Jetson Nano), y la programación de la tarjeta Teensy. Cada uno se describe a continuación en el orden mencionado.

Se implementó un anillo de ocho sensores ultrasónicos en la circunferencia de la plataforma del robot a una altura de 19,5 centímetros. Los sensores ultrasónicos permiten conocer su entorno en 160º a su alrededor. Los datos de los sensores se utilizarán para reforzar la información para evadir obstáculos y también para cubrir espacios ciegos para el Lidar como es la parte posterior al robot donde se encuentra obstruyendo el soporte de la pantalla. En la Figura 55 se muestra la disposición de los sensores y su integración en la plataforma. Los sensores fueron colocados en soportes de MDF y acrílico diseñados para acoplar la base con los sonares.

Figura 55

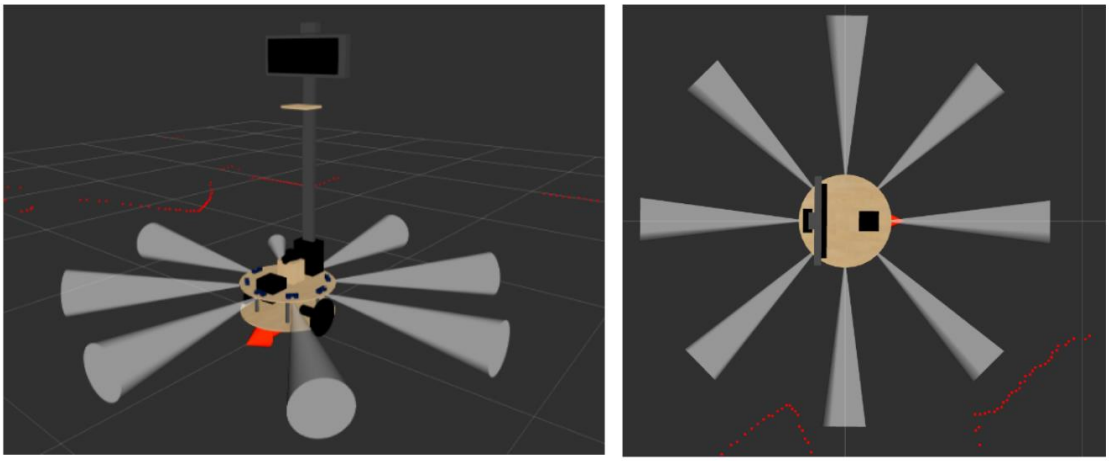
Disposición e integración de los sonares en la plataforma



Para la adquisición de los datos de los sonares se dedicó a la tarjeta Arduino Mega que envía todos los datos en un solo arreglo o vector. Posteriormente el vector datos de la medición de los sonares se divide en 8 mensajes correspondientes a cada sensor que se incluirá en las fuentes de percepción del entorno para detección de obstáculos. El dato obtenido de cada uno de los sensores se representaba en ROS como un mensaje del tipo “range” como se muestra en la Figura 56.

Figura 56

Señal del anillo de sensores implementado vista lateral – vista superior



Dado que la comunicación entre las tarjetas de adquisición y el microcomputador mediante el puente de comunicación serial tenía una corta duración de funcionamiento, se cambió por un puente utilizando el paquete “rosserial”. El cual implementa un protocolo de comunicación para envolver mensajes serializados de ROS estándar y multiplexar varios tópicos y servicios a través de un dispositivo de caracteres, como un puerto serie (Ferguson, 2009). Particularmente, el paquete “rosserial_arduino” permite la integración del protocolo con Arduino IDE. De tal forma se programó la tarjeta Teensy y el Arduino Mega para comunicarse con la unidad de procesamiento y control.

Para la programación de la tarjeta Teensy se utilizó una herramienta de código abierto compatible con ROS. Linorobot es la herramienta que facilita la integración de sensores y actuadores, y su comunicación con la plataforma ROS. Linorobot contiene un código que permite la programación de la tarjeta de adquisición. El firmware requiere la configurando los parámetros correspondientes a la estructura y componentes del robot, además de los coeficientes del controlador de motores y frecuencias de funcionamiento. Los parámetros se muestran en la Tabla 31.

Tabla 31

Parámetros de configuración en el firmware de linorobot

Parámetro	Descripción	Valor
COUNTS_PER_REV	Pulsos por revolución de los encoders	1288
WHEEL_DIAMETER	Diámetro de la rueda [metros]	0.1572
PWM_BITS	Resolución PWM [bits]	10
LF_WHEELS_DISTANCE	Distancia entre ruedas [metros]	0.3935
K_P, K_I, K_D	Coeficientes del controlador	1.8, 0.2, 0.03

Linorobot utiliza los parámetros mencionados para ejecutar acciones sobre los motores. La tarjeta Teensy recibe las velocidades objetivo tanto lineal como angular, las transforma en velocidades para cada uno de los motores y realiza el seguimiento al valor de entrada con la participación de los controladores. Además, con los datos de los encoders y los parámetros configurados en la tarjeta Teensy se calcula la velocidad lineal y angular del robot. La velocidad calculada se envía al microcomputador mediante “rosserial” para conocer la ubicación del robot en su entorno.

Otro dato enviado al computador es aquel obtenido del sensor inercial (MPU6050). El sensor inicialmente se comunica mediante el protocolo serial I2C con la tarjeta de adquisición. En la arquitectura de la comunicación la tarjeta Teensy es maestro y el sensor MPU6050 esclavo. Posteriormente los datos se envían desde la tarjeta de adquisición al computador como un mensaje de ROS que incluye aceleración lineal y velocidad angular; pero estos datos serán procesados y calibrados en la unidad de procesamiento y control. La frecuencia de envío de los datos es diferente para cada sensor. La frecuencia para el controlador de los motores es de 20Hz debido al tiempo de muestreo del diseño del controlador. La frecuencia para el sensor inercial es de 50Hz y para los encoders es de 20Hz, igual que los controladores.

Con respecto al sensor Lidar y el sensor Kinect se conectan directamente a la tarjeta Jetson Nano por los puertos USB y su frecuencia se limita al máximo de cada

sensor. Se utiliza los mismos paquetes y herramientas que el prototipo inicial, para adquirir y procesar los datos de ambos sensores. Por lo tanto, se utiliza el paquete “rplidar_ros” para adquirir los datos del Lidar y para el sensor Kinect se hace uso del paquete “skeleton_markers”.

En el Anexo A se muestra el esquema de la conexión entre los sensores, actuadores, tarjetas de adquisición y unidad de procesamiento mediante un diagrama de bloques. En el Anexo E se muestra el diagrama electrónico de los elementos implementados en la plataforma.

Localización

La localización del robot se basa principalmente en la odometría calculada en base a los datos de los sensores. Y al navegar se complementa con la comparación de los datos del lidar y el mapa del entorno. Para el cálculo de odometría se utiliza los datos de los encoder (estimación de odometría) y del sensor inercial IMU. Las velocidades lineal y angular obtenidas por los encoders enviadas a través de la tarjeta Teensy se procesan en el nodo “lino_base_node”. A la salida del nodo se obtiene un mensaje de tipo “odom” que contiene posición, orientación, velocidad lineal y angular. El mensaje es recibido en un filtro de odometría que será explicado posteriormente.

El mensaje correspondiente al sensor inercial recibido en la microcomputadora primeramente es procesado en el nodo “imu_calib”. El paquete “imu_calib” contiene herramientas para normalizar y aplicar una calibración a las medidas de los datos del IMU. La salida de “imu_calib” es un mensaje que contiene la velocidad angular en radianes por segundo (rad/s), y aceleración lineal en metros sobre segundos al cuadrado (m/s^2) cumpliendo el estándar REP-103 de ROS (Tully Foote & Mike Purvis, 2010). Después de ser calibradas, las mediciones de velocidad angular y aceleración lineal del sensor inercial pasan a través de un filtro. El filtro está contenido en el paquete

“imu_filter_madgwick” que se utiliza para fusionar datos sin procesar de dispositivos IMU (Madgwick et al., 2011). Fusionando velocidades angulares, aceleraciones y lecturas magnéticas (opcional) de un dispositivo IMU genérico en un cuaternión de orientación. Y publica los datos fusionados sobre el tópico “imu/data”.

Los datos procesados del IMU y de los encoders son recibidos en el filtro de odometría. El filtro se implementa en el paquete “robot_localization”, que es un paquete de nodos de estimación de estado no lineal (Moore & Stouch, 2016). El objetivo del filtro del nodo “ekf_localization_node” es estimar en tres dimensiones: la posición (x, y, z), la orientación (roll, pitch, yaw) y sus correspondientes velocidades; de un robot móvil a lo largo del tiempo. El cálculo de la odometría del robot se obtiene de una función de transición de estado no lineal, y el ruido del proceso. Estos valores dependen de un modelo de sensor no lineal que mapea el estado en el espacio de medición y el ruido de medición normalmente distribuido.

En el proceso se incluye un paso de corrección. En la corrección se hace el cálculo de ganancia de Kalman que usa la matriz de observación y medición de covarianza. La ganancia de Kalman sirve para actualizar el vector de estados y la matriz de covarianza. De tal forma se realiza la estimación de odometría basado en la fusión de las mediciones de los sensores.

Los parámetros de configuración del nodo “ekf_localization_node” son: la frecuencia de estimación del filtro, tiempo de lectura de entradas (medición de los sensores), publicación de transformada, límites de aceleración y desaceleración. Además, los valores utilizados de cada sensor se especifican en una matriz que contiene posición, orientación, sus velocidades y aceleración lineal.

En la Tabla 32 se muestra los parámetros utilizados para cada sensor. Siendo 1: utilizado y 0: no utilizado.

Tabla 32

Configuración del vector de los sensores de entrada al filtro EKF

Parámetro	Sensor	
	Odometría (encoders)	IMU
x [m]	1	0
y [m]	0	0
z [m]	0	0
ϕ [rad]	0	0
θ [rad]	0	0
ψ [rad]	1	1
x' [m/s]	1	0
y' [m/s]	1	0
z' [m/s]	0	0
ϕ′ [rad/s]	0	0
θ′ [rad/s]	0	0
ψ′ [rad/s]	1	1
ax [m/s^2]	0	0
ay [m/s^2]	0	0
az [m/s^2]	0	0

Nota: Modificado de *A Generalized Extended Kalman Filter Implementation for the Robot Operating System*, (Moore & Stouch, 2016).

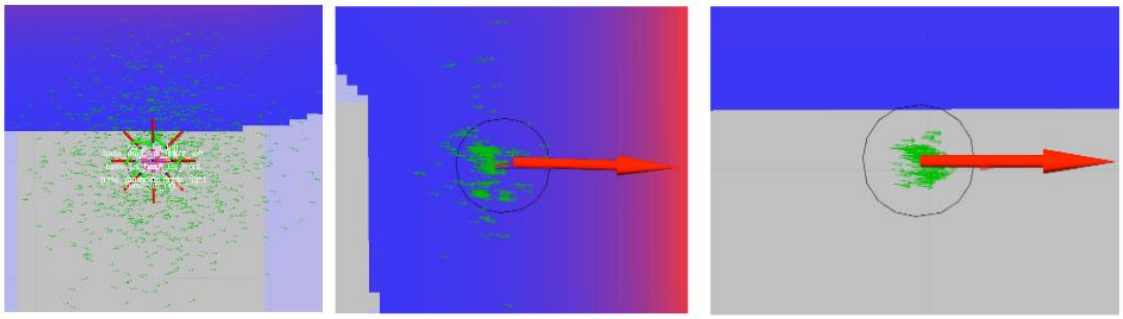
El paquete “ekf_localization_node” permite modificar la covarianza del ruido del proceso mediante una matriz. Pero debido a que la covarianza del ruido del proceso puede ser difícil de ajustar para una aplicación dada, se colocaron los valores por defecto del nodo.

Para la localización global, cuando el robot está en movimiento en el entorno, se incluye el algoritmo AMCL. Se implementa el algoritmo con el paquete “amcl” que recibe los datos del sensor lidar y del mapa del entorno para compararlos y estimar la posición de acuerdo con pesos calculados. El algoritmo utiliza muchas partículas o muestras

cuando no hay similitud del muestreo del lidar con el mapa. Y cuando se conoce aproximadamente la posición del robot el conjunto de muestras es pequeño. En la Figura 57 se presenta las gráficas que muestran el número de partículas y su dispersión inicial, sus movimientos y después de moverse en su entorno localizándose en el mapa.

Figura 57

Muestra de dispersión de puntos: a) inicial b) al empezar la navegación c) después de navegación



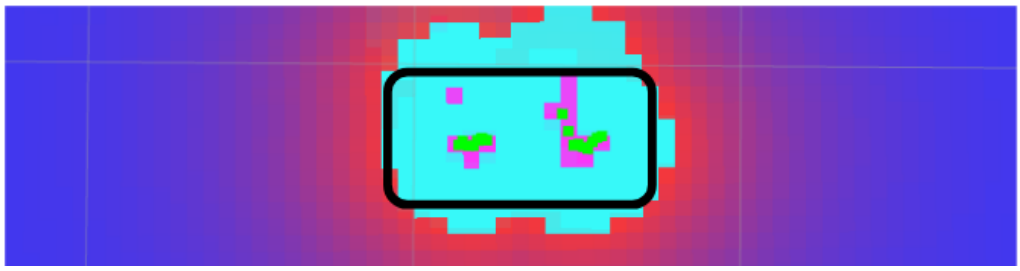
Detección de personas y reconocimiento de gestos

En el subsistema se utilizó los datos de ubicación y detección por láser del sensor lidar y los datos de imagen RGB-profundidad obtenidos del sensor Kinect V1. Con estos parámetros se crearon dos nodos para la detección de personas y reconocimiento de gestos.

En la detección de personas se utiliza como base dos algoritmos. Uno de detección de piernas basado en el algoritmo propuesto por (Arras et al., 2007; Lu & Smart, 2013) y otro que crea un objeto tipo persona (people), a partir de los datos obtenidos por el sensor lidar. Para poder detectar piernas se identifica grupos de puntos del dato “laser_scan”, con forma de arco que presenten una separación mínima entre cada uno de ellos como muestra la Figura 58.

Figura 58

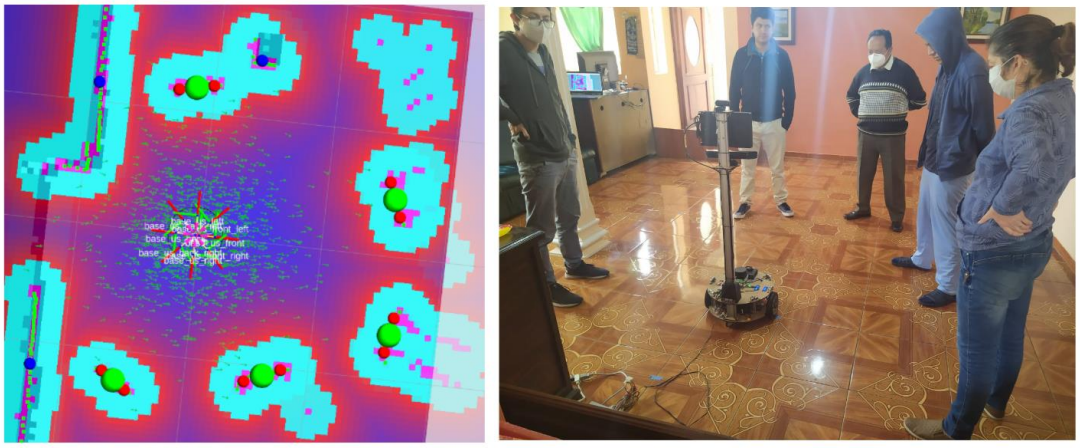
Piernas detectadas (puntos verdes) mediante el algoritmo y el sensor Lidar



Se reentrenó el algoritmo de detección de piernas mediante el método utilizado en (Leigh et al., 2015). El cual implementa un clasificador de bosque aleatorio de la librería de OpenCV3 para tener un mejor desempeño con el sensor lidar integrado en la plataforma.

Figura 59

Ejemplo de personas creadas por el detector de piernas

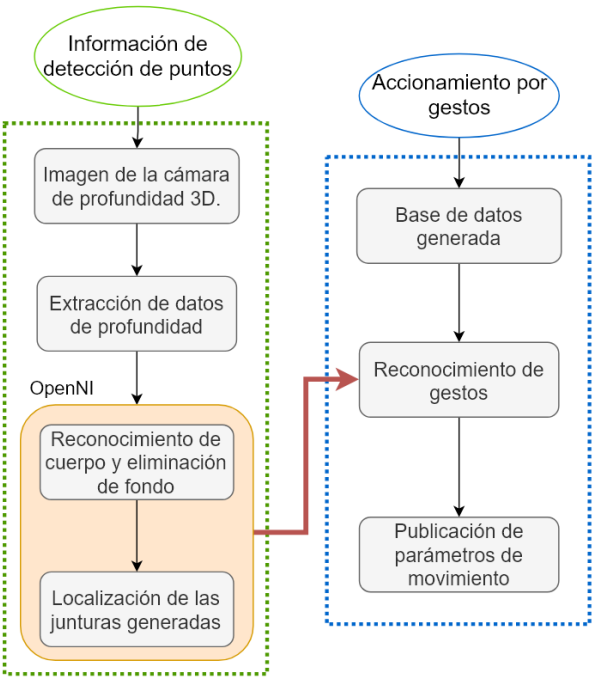


En el entrenamiento se utilizó el robot físico con personas dentro de un entorno controlado. Se realizó un total de 34 pruebas, siendo 20 de ellas positivas y 14 negativas. Donde los datos positivos mostraban a la persona caminado y colocándose en posiciones dentro del entorno en un área específica; y en las muestras negativas se colocaron objetos como sillas y mesas dentro del entorno.

Los datos se recopilaron mediante la herramienta de ROS “rosbag” que permite guardar datos de tópicos para su posterior entrenamiento. Para la creación del objeto tipo persona se tomó la salida del detector de piernas, en donde si existe dos posibles agrupaciones con una distancia máxima de 0.38 [cm] entre ellas, y una fiabilidad mayor al 65% se crea el objeto como se muestra en la Figura 59. En la detección y reconocimiento de gestos se reutilizó los paquetes “skeleton_markers” y “neuro_gesture_kinect” utilizados en (Andrango, 2020), en conjunto con la base de datos generada. El diagrama de funcionamiento del reconocimiento de gestos y su acción sobre los movimientos del robot se muestra en la Figura 60.

Figura 60

Diagrama de funcionamiento para el reconocimiento de gestos



Nota: Modificado de *Development of gesture recognition-based serious games,* (He et al., 2012).

Para la interacción con la plataforma se seleccionó 5 de los 6 gestos extrínsecos entrenados. Dos instrucciones incluyen movimiento (avanzar, hola) y tres gestos son

estáticos (derecha, izquierda y detenerse) como se muestra en la Figura 61. Las respuestas a estos gestos se implementaron mediante un puente en Python y en conexión al subsistema de navegación.

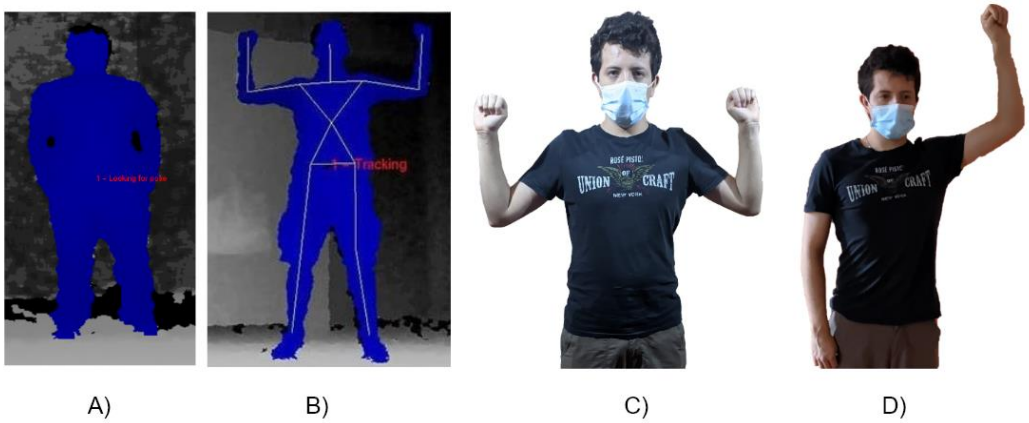
Figura 61

Gestos implementados en el subsistema de detección y reconocimiento

Para iniciar el sistema de detección de gestos, la persona que va a interactuar con la plataforma debe de iniciar la calibración mediante la posee “psi” (pose default generada por “skeleton_markers”). Posteriormente realizará la posee de detección con la mano derecha sobre el hombro como se muestra en Figura 62.

Figura 62

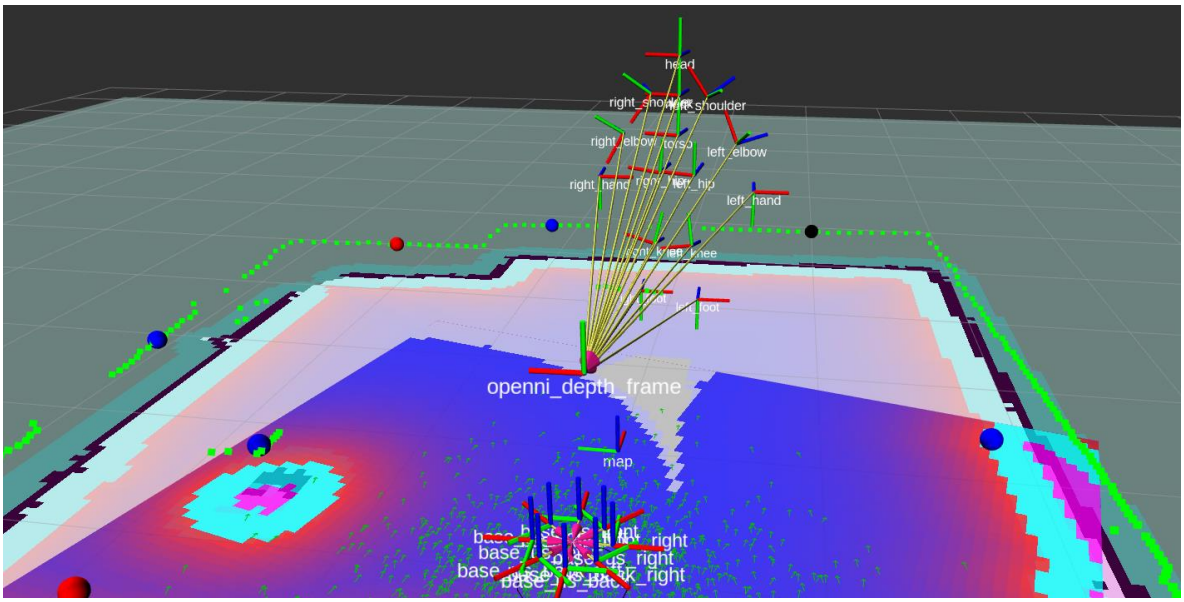
A) Pose previa - B y C) Posee “Psi” – D) Señal de detección



Una vez establecida la detección de la persona por el nodo “skeleton_markers”. Se obtiene el conjunto de las transformadas de ubicación para cada una de las partes del cuerpo identificadas con respecto al eslabón del sensor Kinect V1 como se muestra en la Figura 63.

Figura 63

Detección y seguimiento de junturas y creación de transformadas de la persona



Navegación

El subsistema utiliza el paquete de navegación por defecto de ROS “move_base” desarrollado por (Lu & Marder-Eppstein, 2013), compatible con plataformas móviles diferenciales y holonómicas. La cual está compuesta por la estructura mostrada en la Figura 64. La estructura utiliza como parámetros de ingreso el mapa estático del entorno, valores de odometría y sensores externos, además del paquete AMCL para la corrección de odometría a partir de los datos de tipo “laser_scan”. Dentro de los algoritmos de este paquete se utilizan:

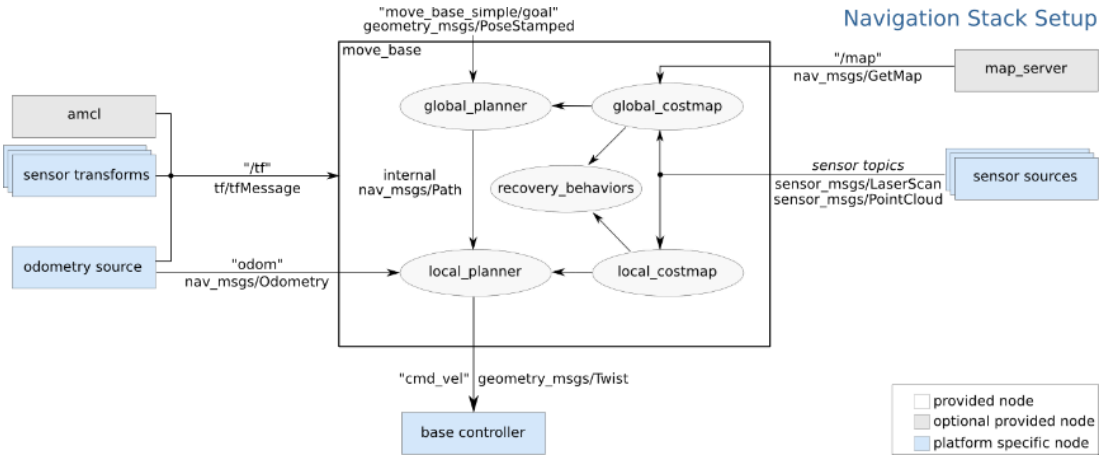
- **Planificador global:** Algoritmo encargado de generar la planificación global de la ruta.

- **Planificador local:** Algoritmo encargado de generar la planificación dentro del entorno cercano al robot.
- **Mapa de costos local:** Datos de costos obtenidos por los sensores de la plataforma, sensores ultrasónicos, LIDAR, sensor Kinect V1.
- **Mapa de costos global:** Datos de costos estáticos generados por los obstáculos detectados en el SLAM.

Para el prototipo propuesto se utilizará el algoritmo A* como planificador global y el algoritmo DWA como planificador local. Se usará el paquete AMCL y el dato de odometría obtenida por el filtro de Kalman extendido como parámetros de localización.

Figura 64

Configuración estándar utilizada en el paquete "move_base"



Nota: Tomado de *move_base ROS package*, (Lu & Marder-Eppstein, 2013).

Planificador global

El planificador global está configurado con un algoritmo A*, que se implementó mediante el paquete nativo de ROS "move_base" utilizando el módulo "nav_core::BaseGlobal" basados en el trabajo realizado por (Brock & Khatib, 1999). Para poder configurar el planificador se utilizarán los parámetros indicados en la Tabla 33.

Tabla 33

*Parámetros para configuración de navegador A**

Parámetro	Descripción
Allow_unknown	Crea plan a través de lugares no descubiertos
Use_dijkstra	Escoge algoritmo entre Dijkstra o A*
Use_quadratic	Permite calcular la siguiente posición de forma cuadrática
Use_grid_path	Permite planificar a través de los límites del mapa
Old_nav_behavior	Permite establecer una navegación igual a NAVFN

Una vez utilizado los parámetros se consigue la configuración de navegación indicada en la imagen Figura 65, tal comportamiento se obtiene en adición al valor de costo de navegación que se muestra en la siguiente ecuación:

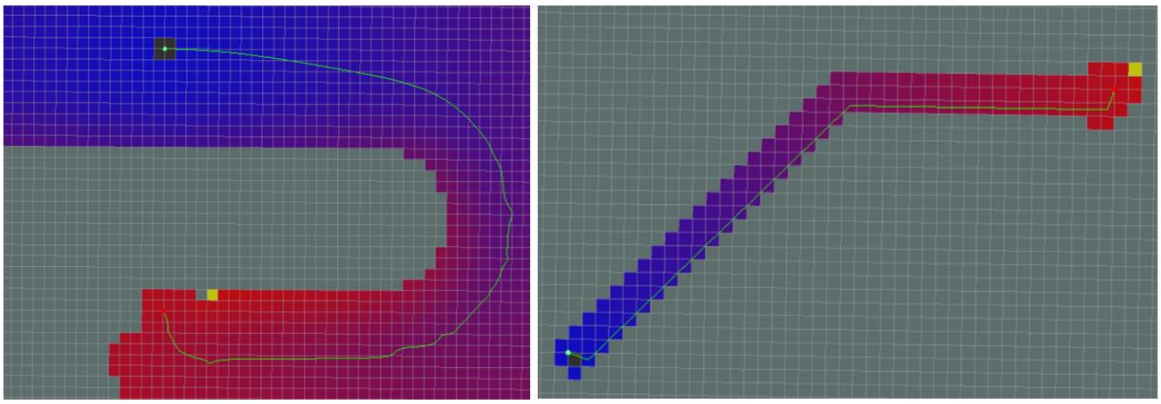
$$costo = costo\ neutral + factor\ costo + valor\ mapa\ costo \quad (30)$$

- **Costo neutral:** Costo obtenido a partir de las dimensiones físicas de la plataforma móvil.
- **Factor de costo:** Factor de escalado para el costo de obstáculos.
- **Valor mapa de costo:** Costo de cada obstáculo en referencia a la resolución utilizada para el mapa de costos.

Los parámetros se configuran de manera manual en relación con el comportamiento del robot en el entorno físico, así como la cantidad de costo que se desee implementar al planificador frente a los obstáculos y parámetros sociales.

Figura 65

*Comportamiento con planificador Dijkstra contra el planificador A**



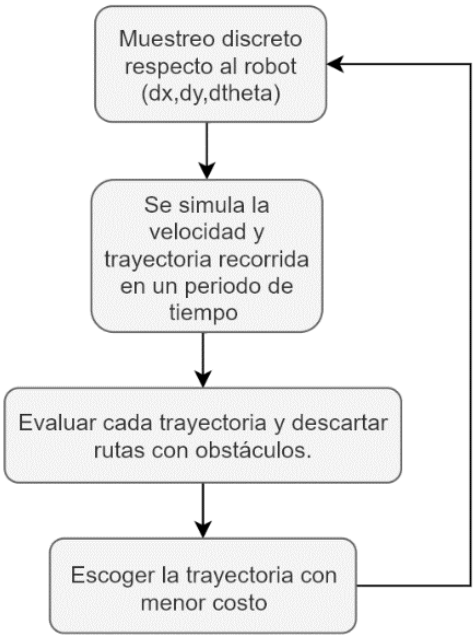
Nota: Tomado de *ROS Navigation Tuning Guide*, (Zheng, 2017).

Planificador local

El planificador local al utilizar el modelo de algoritmo DWA de ROS propuesto por (Fox et al., 1997), busca generar un valor de velocidad lineal y angular (v, w), que represente una trayectoria circular tomando en cuenta la posición actual del robot y sus dimensiones físicas durante un intervalo de tiempo determinado. El algoritmo DWA únicamente publicará los valores de velocidad que estén dentro de la ventana de acción obtenida por el tiempo simulado de la trayectoria. El funcionamiento simplificado se muestra en la Figura 66.

Figura 66

Diagrama de funcionamiento del planificador DWA



Nota: Adaptado de *ROS Navigation Tuning Guide*, (Zheng, 2017).

El planificador posee varios parámetros de configuración, los cuales se modificaron y optimizaron a partir de las características físicas y de procesamiento de la plataforma móvil. Los principales parámetros se muestran en la Tabla 34.

Tabla 34

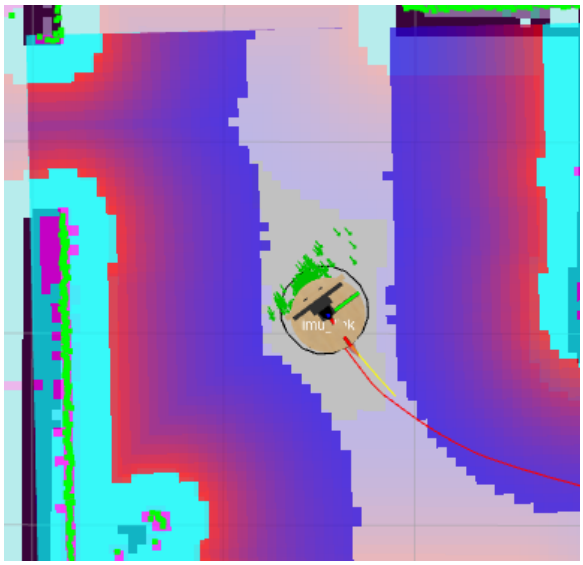
Parámetros para configuración del planificador DWA

Parámetro	Descripción
Sim_time	Parámetro de tiempo a calcular el planificador
Vx_samples	Cantidad de muestras en el eje x al robot
Vth_samples	Cantidad de muestras angulares a tomar en cuenta
Simulation granularity	Resolución del planificador en función del entorno.
Cost_value	Costo por evaluar para cada planificación creada.
Tolerance_values	Valor de tolerancia con relación al punto objetivo de navegación.

Dentro del planificador local también se configuró los parámetros físicos de navegación de la plataforma como valores límites de velocidad, aceleración, sentido de navegación y parámetros de frecuencia del controlador. En la Figura 67 se muestra la representación de la salida del planificador DWA en la línea amarilla.

Figura 67

Funcionamiento del planificador DWA (color amarillo)



Mapas de costos

Los mapas de costos son utilizados como nodo de conexión entre los datos obtenidos por los sensores y los planificadores de navegación, los mapas de costos son representados mediante un arreglo bidimensional. El costo, en el paquete “move_base” de forma predetermina está representada por una distribución gaussiana bidimensional defina como se muestra en la siguiente ecuación.

$$f(\hat{x},\hat{y}) = A\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \quad (31)$$

Donde:

- Posición x: El valor inicial desde el que se toma en cuenta el cálculo del costo de obstáculo en el eje x.

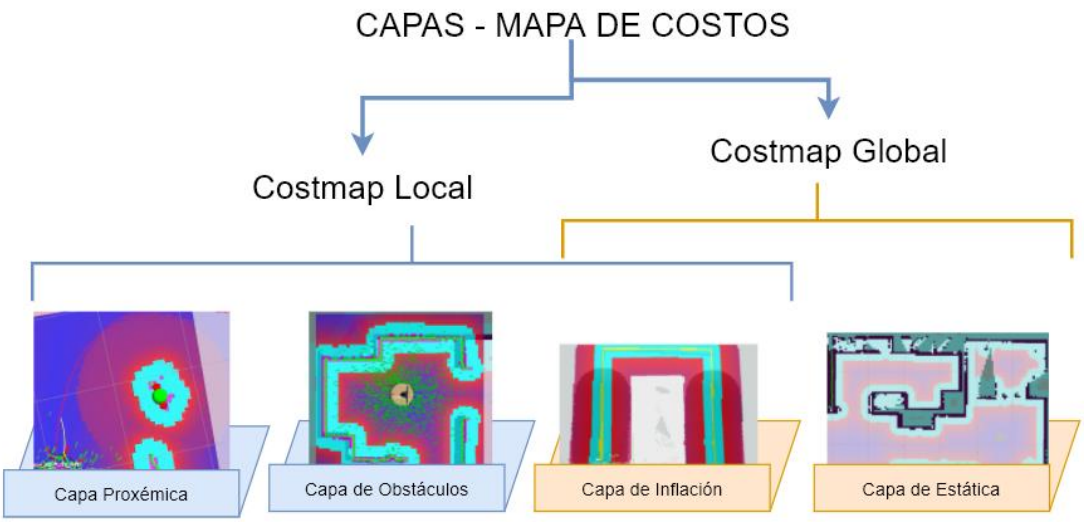
- Posición y: El valor inicial desde el que se toma en cuenta el cálculo del costo de obstáculo en el eje y.
- Amplitud A: Constante de trayectoria a través del obstáculo.
- Varianza σ: El valor máximo que se tomará como desviación desde el centro del obstáculo.

El nodo genera los parámetros de inflación dentro del entorno y establece el comportamiento de navegación de la plataforma. Dentro del mapa de costos se establece el contorno de la plataforma representado en ROS como un arreglo de la forma $[x_0, y_0], [x_1, y_1], [x_2, y_2], [x_3, y_3]$. El arreglo muestra los puntos del contorno de la plataforma, el vector es utilizado para calcular el círculo circunscrito, que es usado como área a tomarse en cuenta durante la navegación.

La configuración de mapa de costos implementada se basa en la estructura propuesta por (Lu et al., 2014). Donde se propone una división por capas para cada uno de los elementos a tomar en cuenta en la navegación. La estructura de capas utilizada se muestra en la Figura 68.

Figura 68

Estructura de mapa de costos utilizados



Los mapas de costos implementados están tienen las siguientes características.

- **Capa proxémica:** Es la encarga de generar un mapa de costos circular con la distancia mínima a la cual puede estar la plataforma de una persona. En base a una distribución gaussiana desde la posición de la persona detectada.
- **Capa de obstáculos:** Entrega la información de la ubicación de los obstáculos que no se encontraban registrados en el SLAM del entorno.
- **Capa de Inflación:** Genera el espació o distancia entre la posibilidad de planificación de la plataforma y el circulo circunscrito calculado del robot.
- **Capa estática:** Es la capa de costos generada en base a la información del entorno obtenida durante el SLAM, este mapa delimita la circulación de la plataforma y posee un costo menor en referencia a la capa de obstáculos.

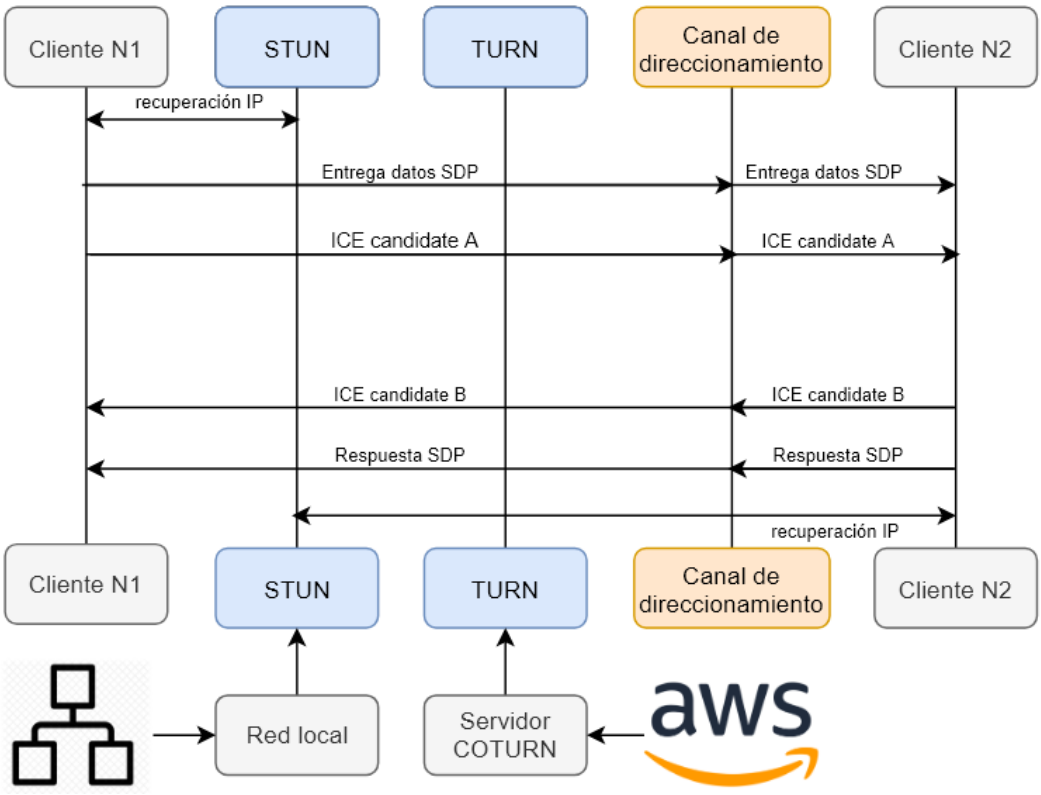
Telepresencia

Dentro del sistema de telepresencia se implementó una comunicación mediante un canal de video/audio y un canal de datos utilizando el framework de WebRTC. Con el cual se implementó dos clientes y un servidor COTURN para poder establecer conexiones remotas. El sistema posee la configuración propuesta en la Figura 7. En donde se muestra la comunicación entre un cliente local implementado en una página web en la plataforma móvil, y el cliente remoto desarrollado en una aplicación móvil Android, mediante el software de desarrollo Android Studio 4.1.1.

Se creó una comunicación mediante WebSockets para reconocer los parámetros de conexión y envió de datos de cada cliente para poder establecer una conexión directa y sus respectivos canales de información. La conexión sigue el funcionamiento mostrado en Figura 69.

Figura 69

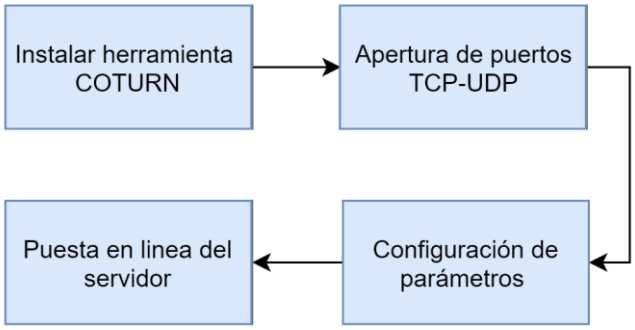
Funcionamiento de la comunicación mediante WebSockets



Para poder establecer la comunicación es necesario utilizar los protocolos STUN o TURN. Se utiliza el protocolo STUN para poder conectar ambos clientes cuando la plataforma móvil puede rescatar la dirección IP del operador remoto. Por otra parte, se implementó un servidor TURN que para cuando los clientes se encuentren entre firewalls, redes NAT o en diferente lugar geográfico, mediante la herramienta COTURN y un computador con Linux en el servidor de AWS. Para poder crear el servidor se realizó los pasos mostrados en la Figura 70, configurando los parámetros de seguridad y puertos en el panel de control de AWS.

Figura 70

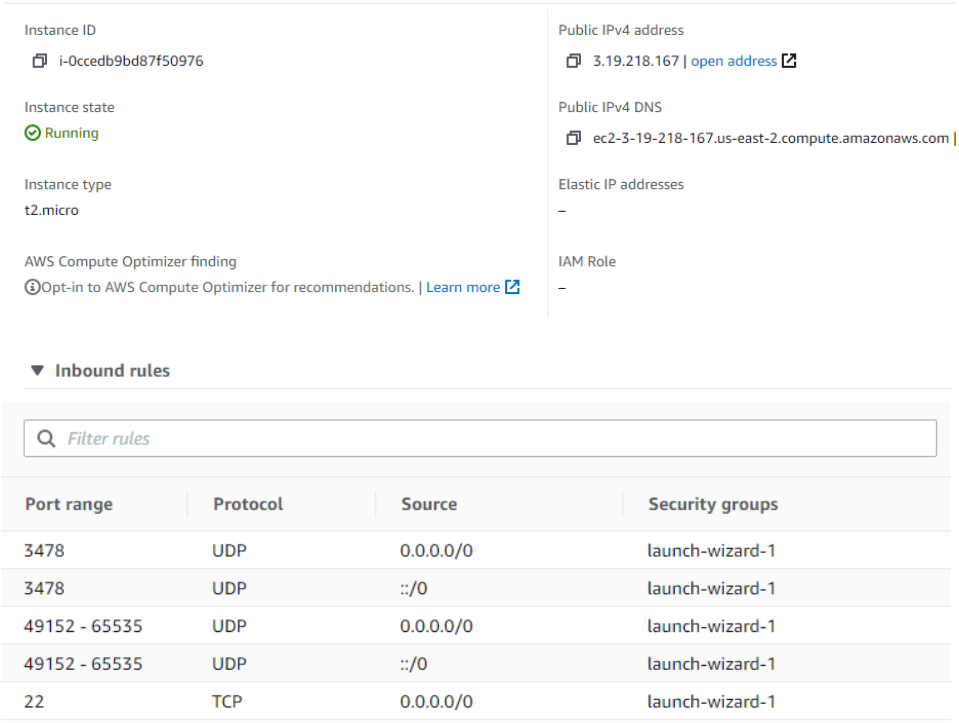
Instrucciones de configuración del servidor COTURN



Una vez realizado las configuraciones dentro del servidor, es esencial abrir los puertos de conexión en el panel de control de AWS y asignar una dirección IP publica para su conexión directa como se muestra en Figura 71.

Figura 71

Estado del servidor y dirección IP pública y configuración de puertos en AWS

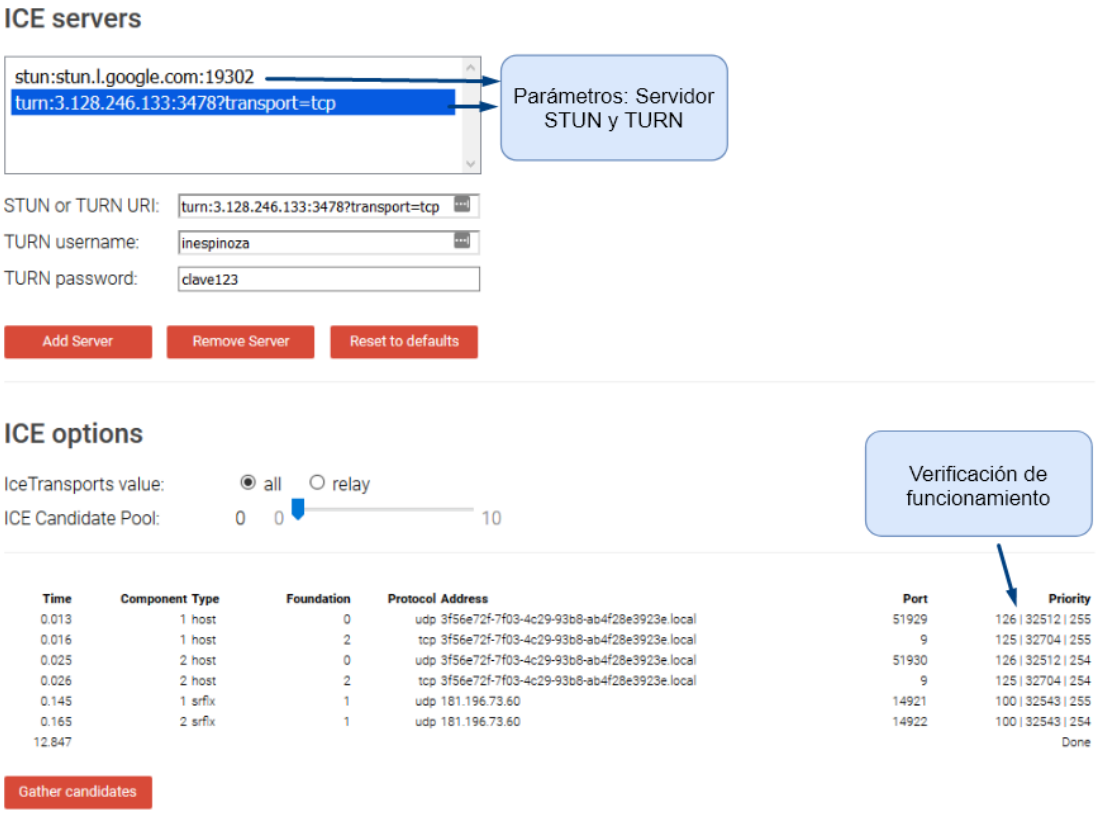


Para comprobar la conectividad con el servidor TURN y su compatibilidad con WebRTC se utilizó la herramienta online trickle ICE implementada por (Ben Wright

et al., 2016). En la cual se ingresa los datos de conexión de cada protocolo y entrega un dato de confirmación y el valor de latencia como se muestra en la Figura 72.

Figura 72

Verificación del estado de los servidores STUN y TURN



Aplicación WEB (cliente local)

Se desarrolló una aplicación WEB utilizando JavaScript, CSS y HTML en la cual se implementó un interfaz de usuario que contiene un menú de selección y una interfaz de conexión para el funcionamiento de telepresencia. La implementación con el sistema ROS se realizó mediante el paquete “rosbridge-server/library” desarrollado por (Mace & Toris, 2017).

Rosbridge establece una interfaz en JSON con ROS permitiendo conectar nodos de suscripción y publicación mediante JavaScript utilizando una conexión local mediante el protocolo TCP y WebSocket.

Figura 73

Estructura de mensajes utilizados en "rosbridge"

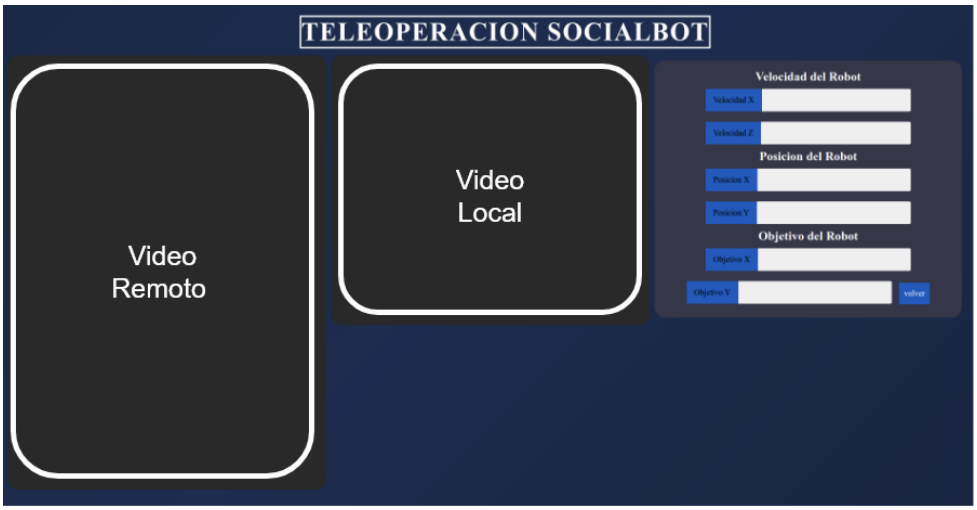
Nota: Tomado de *Rosdbridge v2.0 Protocol Specifications*, (Mace & Toris, 2018).

El paquete utiliza un protocolo específico para la creación y estructura de datos de subscripción y publicación mostrada en (Mace & Toris, 2018). Para la implementación del sistema se utilizaron las estructuras de mensajes mostrados en la Figura 73. La interfaz de comunicación para el control y a de telepresencia se muestra en la Figura 74, dentro de la interfaz se tiene tres secciones enfocadas en:

- **Video Remoto:** En esta sección se muestra el video-audio obtenido de la aplicación Android del operador remoto.
- **Video Local:** Se encuentra el video-audio obtenido de la cámara web utilizada en la plataforma móvil.
- **Parámetros de información:** En esta sección se indican los parámetros de posición en (x, y) del robot, así como su punto objetivo de navegación y su velocidad.

Figura 74

Interfaz gráfica implementada para el control de telepresencia



Dentro de la aplicación se implementó de igual manera una interfaz para poder monitorear los parámetros, configurar los valores del controlador PID, visualizar el comportamiento en Rviz y monitorear y publicar tópicos de manera local. La estructura de esta interfaz se muestra en la Figura 75.

Figura 75

Interfaz gráfica implementada para el monitoreo

Para poder acceder a la aplicación web el proyecto fue publicado en la plataforma de desarrollo en la nube HEROKU (Heroku, 2007), Lo que permite acceder a la aplicación desde cualquier dispositivo que tenga compatibilidad con navegadores web. El funcionamiento implementado se muestra en el diagrama de flujo del Anexo B.

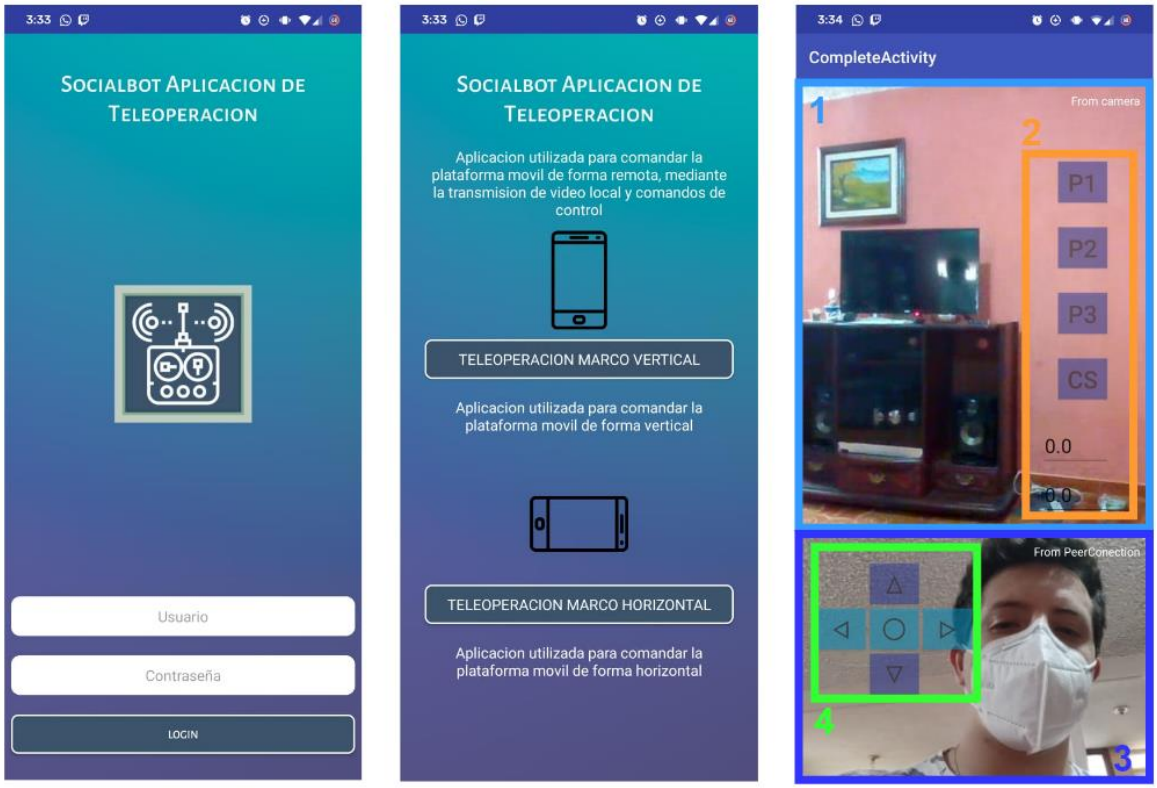
Aplicación Android (cliente remoto)

El desarrollo de aplicación Android está basado en la API desarrollada por (Sredojev et al., 2015), La aplicación consta de tres interfaces de usuario. La primera es una página de acceso que requiere de usuario y contraseña. La segunda interfaz corresponde a un menú de selección de orientación. Y la tercera muestran la sección principal para establecer la comunicación y envió de datos, como se muestra en Figura 76. En la Figura 76 imagen C, muestra la interfaz de operación general en la cual se encuentra separado los botones y marcos de la siguiente forma:

- **1 - video remoto:** muestra la imagen obtenida de la aplicación web ejecutada en la plataforma móvil.
- **2 - botones navegación autónoma:** muestra un total de 4 botones y dos entradas numéricas; mediante las cuales se puede enviar un punto objetivo fijo (botones P1-P2-P3) o una posición (x, y) personalizada (botón CS y entradas numéricas).
- **3 - video local:** muestra la imagen rescatada de la cámara frontal del dispositivo móvil.
- **4 - botones teleoperación:** Esta sección consta de cinco botones utilizados para poder controlar la plataforma a una velocidad fija de 0.25 (m/s): adelante (↑), izquierda (←), derecha (→), retroceder (↓) y parar (○).

Figura 76

A) Interfaz de ingreso B) Menú de selección C) Interfaz operación en Vertical



El funcionamiento del código implementado en la aplicación móvil se muestra en el Anexo C.

Arquitectura del sistema

El sistema de navegación reactiva social este compuesto por todos los subsistemas antes mostrados y en conjunto poseen la distribución y arquitectura mostrado en la Figura 77, implementada en ROS mediante nodos.

Figura 77

Arquitectura de la plataforma móvil

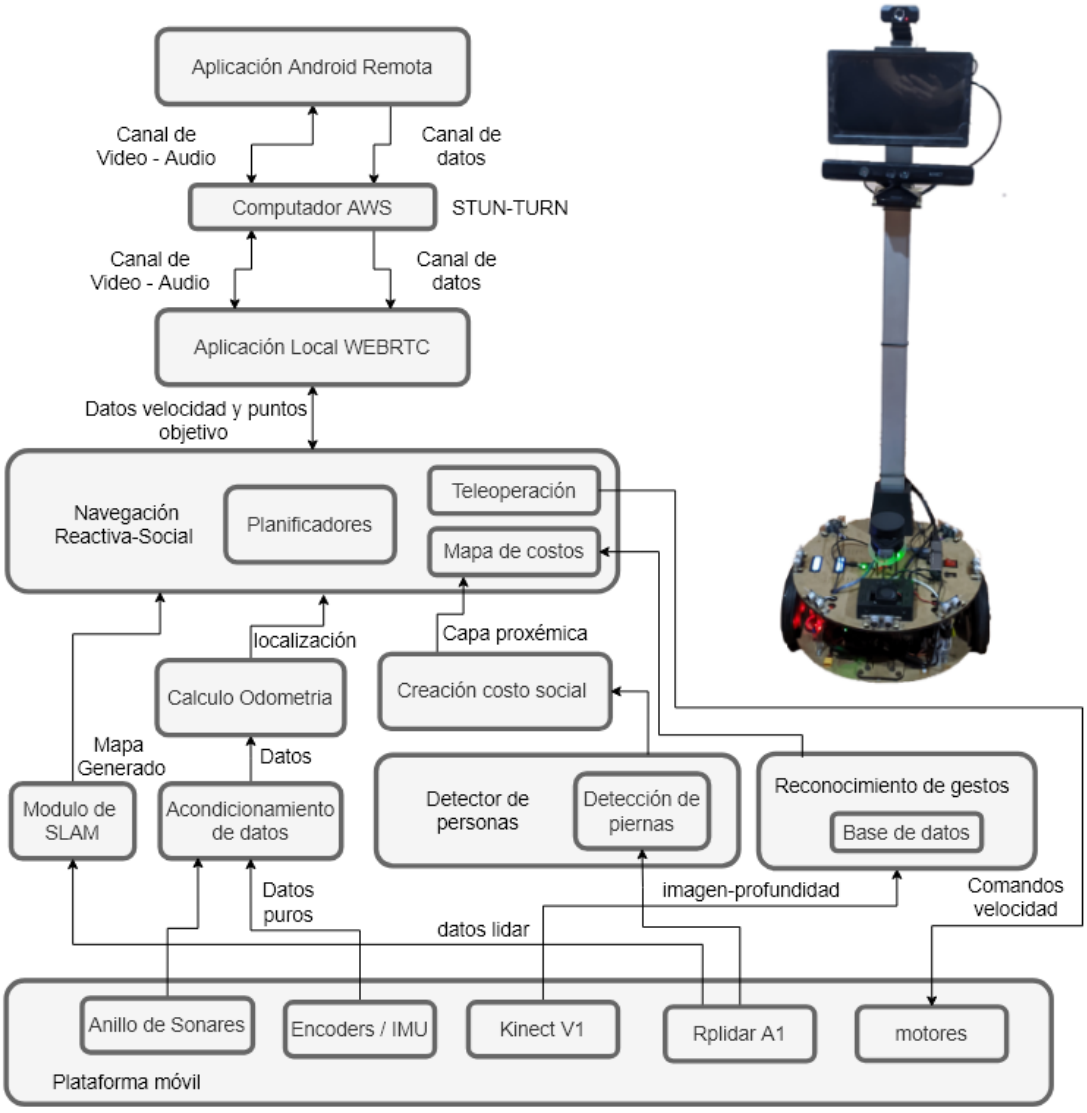
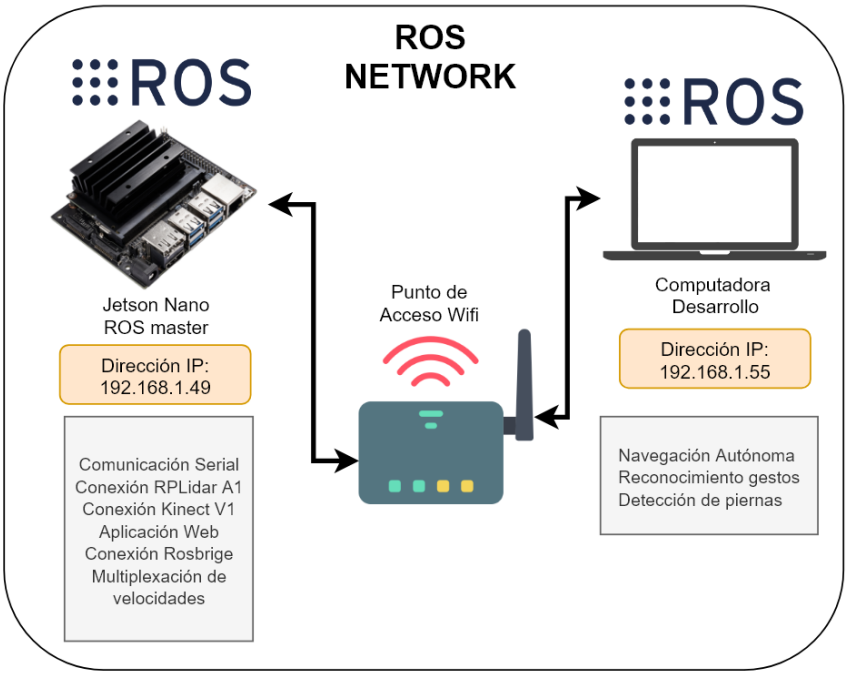


Figura 78

Distribución de nodos mediante ros network



Resumen

En el capítulo se presentó la evaluación del prototipo inicial para conocer las correcciones que era necesario realizar. Una vez descritas las mejoras, se explica la implementación del sistema. Se expone las modificaciones en la estructura del robot y la integración de elementos. Se muestra la explicación del uso y reentrenamiento del algoritmo de detección de piernas y personas. Además, el funcionamiento del algoritmo de reconocimiento de gestos y su uso para la interacción con personas. Para la localización del robot se explica la integración del IMU y el filtro EKF. En navegación se describe los parámetros utilizados en los planificadores y el mapa de costos por capas. Para el sistema de telepresencia se implementó una comunicación punto a punto mediante WebRTC. Se utilizó un servidor STUN para redes locales y servidor TURN implementado en AWS para redes remotas. La comunicación se estableció entre una aplicación web para la interfaz local y una aplicación Android para el operador remoto.

Capítulo V

Pruebas y resultados

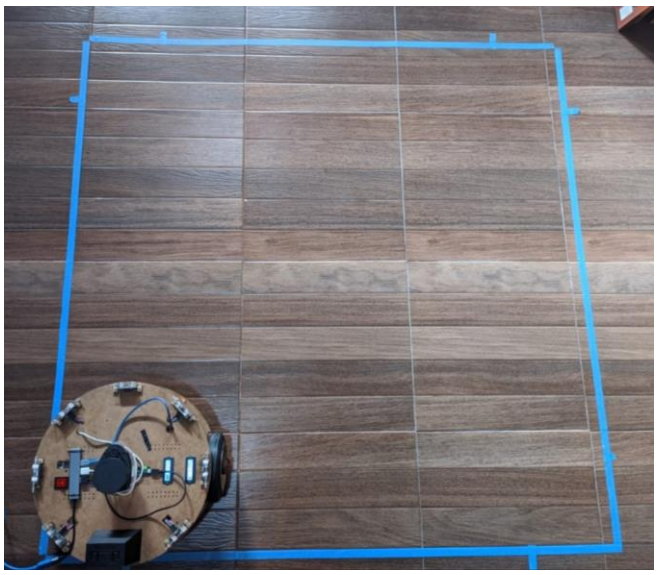
El capítulo muestra las pruebas realizadas en el sistema y los resultados obtenidos de las diferentes pruebas. Se describe las pruebas de odometría en el prototipo inicial y el prototipo actual. Además, se muestra la evaluación de los algoritmos de detección de piernas-personas y reconocimiento de gestos. En navegación se realiza pruebas en diversos entornos. Los tres entornos de prueba son: vacío, con objetos y con personas. Por último, se realizaron las pruebas en la transmisión de audio y video del sistema de telepresencia y teleoperación.

Pruebas de odometría

El error en odometría se encontró realizando el experimento “Bidirectional Square Path”, llamado University of Michigan Benchmark (UMBmark) (Borenstein & Feng, 1995). Se utilizó una trayectoria cuadrada bidireccional de 1x1 [metro] como se muestra en la marca azul de la Figura 79.

Figura 79

Trayectoria para pruebas de odometría

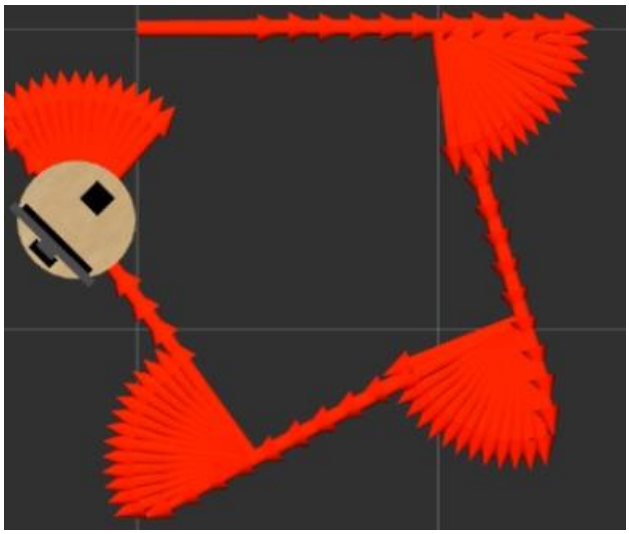


Se realizó la prueba UMBmark cinco veces en cada dirección (horaria y antihoraria), con el movimiento del robot a una velocidad moderada de 20 cm/s.

Primero se realizó la prueba con la configuración del prototipo inicial. Los únicos sensores para calcular la odometría fueron los encoders de cada motor. En la Figura 80 se presenta un ejemplo del movimiento del robot en la trayectoria cuadrada.

Figura 80

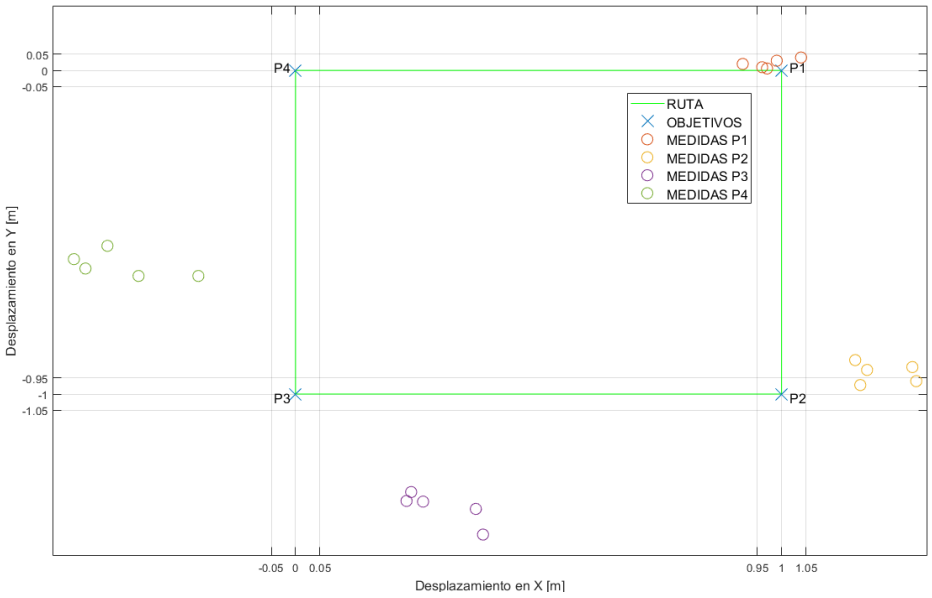
Representación del movimiento del prototipo inicial en pruebas de odometría



La Figura 81 muestra la gráfica que representa la dispersión de los movimientos del robot en la trayectoria cuadrada en dirección horaria. Donde la línea verde representa la trayectoria cuadrada planteada. Las “x” azules son los puntos objetivos marcados de la trayectoria. Y los círculos rojos, amarillos, morados y verdes representan las medidas en los puntos P1, P2, P3 y P4 respectivamente.

Figura 81

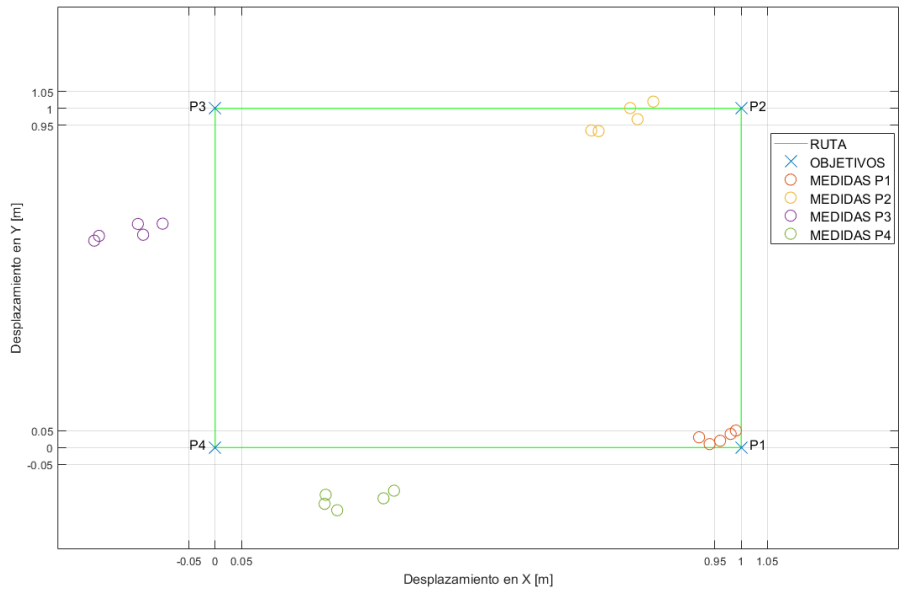
Representación de la medición de odometría en dirección horaria en el prototipo inicial



Así mismo, se realizó la medición en el sentido antihorario y en la Figura 82 se muestra su representación.

Figura 82

Representación de la medición de odometría en dirección antihoraria en el prototipo inicial



Para representar el error de odometría se considera el centro de gravedad del grupo de medidas en cada punto. Por lo tanto, se parte obteniendo los errores absolutos en ambos sentidos para cada una de las medidas en posición (x, y) como se muestra en las ecuaciones (32) y (33):

$$\epsilon x = |x - x_{calc}| \tag{32}$$

$$\epsilon y = |y - y_{calc}| \tag{33}$$

De los errores se obtiene el promedio en x, y en cada uno de los puntos P1, P2, P3 y P4.

$$x_{h/a} = \frac{1}{n}\sum_{i}^{n} \epsilon x i_{h/a} \tag{34}$$

$$y_{h/a} = \frac{1}{n}\sum_{i}^{n} \epsilon y i_{h/a} \tag{35}$$

Donde:

h/a: es prueba horaria o antihoraria

n: es número de muestras medidas en cada prueba

Los desplazamientos absolutos de los centros de gravedad desde cada punto son dados por:

$$r_{h/a,Pk} = \sqrt{(x_{h/a,Pk})^2 + \left(y_{h/a,Pk}\right)^2} \tag{36}$$

Donde:

Pk: es cada uno de los puntos de la trayectoria. k=1,2, 3 o 4

Se presentan en la Tabla 35 el resumen los cálculos de error basados en los desplazamientos absolutos de los centros de gravedad de cada punto.

Tabla 35

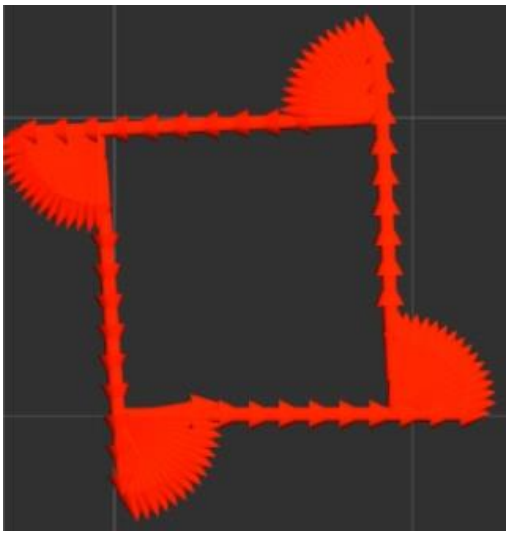
Resumen del error en odometría en el prototipo inicial

Puntos en la trayectoria	**Desplazamiento absoluto del centro de gravedad en odometría**	
	Sentido Horario (r_h) [m]	Sentido Antihorario (r_a) [m]
P1	0.031	0.052
P2	0.217	0.416
P3	0.459	0.445
P4	0.650	0.305

Finalmente se elige el valor más grande del desplazamiento de los centros de gravedad como la medida de precisión odométrica (E_{max}). Por lo tanto, la precisión odométrica del prototipo inicial es de 65.0 cm. El mismo procedimiento se realizó con la plataforma después de implementar los sensores, tarjetas y algoritmos mostrados en el capítulo anterior. Se procede a indicar las representaciones correspondientes y los datos obtenidos de la prueba. La Figura 83 muestra un ejemplo del movimiento del robot en la trayectoria cuadrada.

Figura 83

Representación del movimiento del prototipo en pruebas de odometría



En la Figura 84 y la Figura 85 se muestra la representación de las mediciones en cada los puntos de la trayectoria en sentido horario y antihorario respectivamente.

Figura 84

Representación de la medición de odometría en dirección horaria en el prototipo

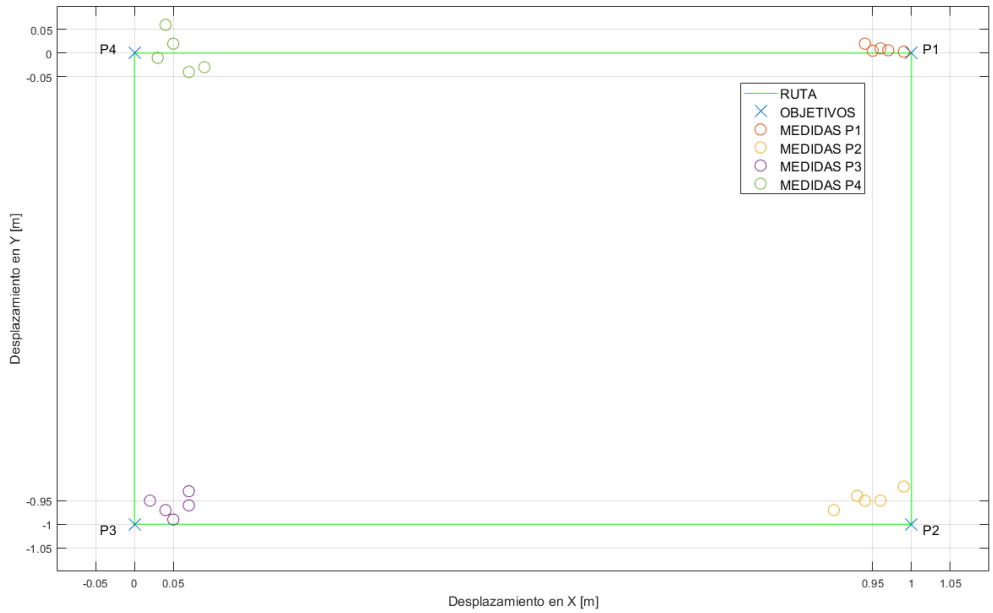
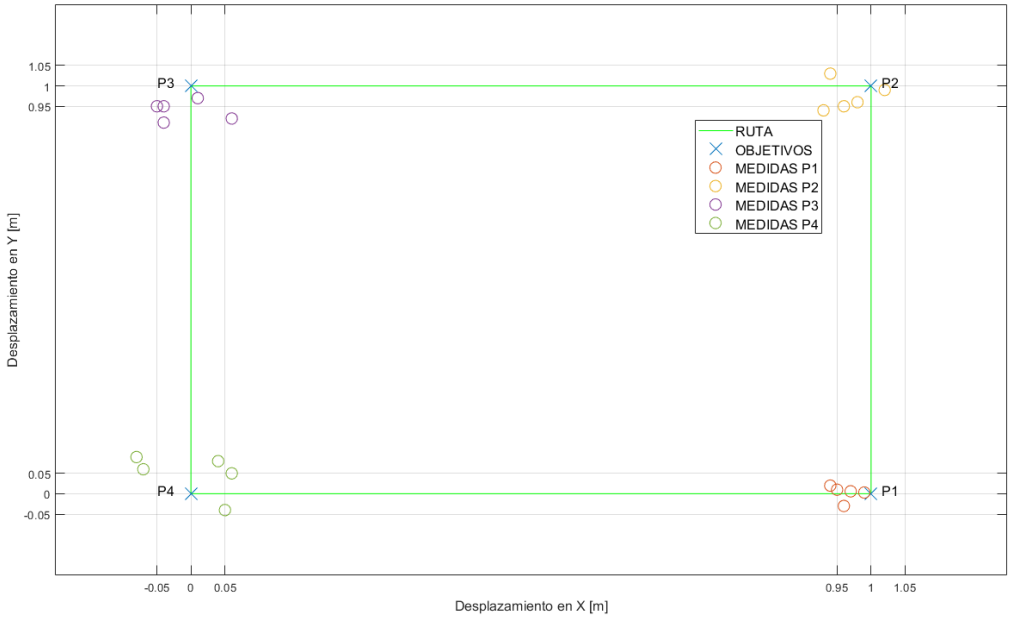


Figura 85

Representación de la medición de odometría en dirección antihoraria en el prototipo



Se presentan en la Tabla 36 el resumen los cálculos de error basados en los desplazamientos absolutos de los centros de gravedad de cada punto.

Tabla 36

Resumen del error de odometría en el prototipo

Puntos en la trayectoria	**Desplazamiento absoluto del centro de gravedad en odometría**	
	Sentido Horario (r_h) [m]	Sentido Antihorario (r_a) [m]
P1	0.039	0.041
P2	0.064	0.056
P3	0.069	0.072
P4	0.077	0.087

El valor más grande de la Tabla 36 representa la precisión odométrica del prototipo y es de 8.7 cm. Se presenta en la Tabla 37 la comparativa de la precisión odométrica del prototipo inicial y del prototipo después de la implementación del sistema de navegación reactiva social y telepresencia.

Tabla 37

Comparación de odometría entre el prototipo inicial y el prototipo actual

	Prototipo inicial	**Prototipo actual**
Precisión odométrica $(\boldsymbol{E_{max}})$	65 cm	8.7 cm

Además, se realizó una prueba de odometría en línea recta. Se movió a la plataforma a través de una trayectoria recta de 3 metros para conocer el error entre la odometría real y la calculada. La Figura 86 muestra la trayectoria seguida por el robot marcado en azul en el piso. La velocidad del robot fue de 20 cm/s.

Figura 86

Trayectoria recta seguida por el robot

El robot recorrió la trayectoria recta 10 veces y los datos obtenidos están presentes en la Tabla 38.

Tabla 38

Datos obtenidos de la prueba de odometría en trayectoria recta

Iteración	Posición final en x [m]	Posición final en y [m]
1	3.09	0.05
2	3.02	0.026
3	3.10	0.021
4	3.11	0.001
5	3.12	0.06
6	3.11	0.054
7	3.09	0.03
8	3.10	0.025
9	3.11	0.04
10	3.095	0.015

Dado que los puntos objetivos reales son 3 metros en x y 0 metros en y (3,0), se obtiene el error para cada medición y se calcula el promedio. Del promedio de error en x, y, se obtiene el módulo y se calcula el error porcentual. El error promedio en x es de 9.45 cm y el promedio del error absoluto en y es 3.18 cm. El módulo de los errores en x,

y es 9.97 cm, lo que equivale a un error porcentual de 3.32%. Cabe recalcar que el error es obtenido de una trayectoria recta.

Detector de piernas

Para las pruebas del algoritmo de detección de piernas se tomó en cuenta la confiabilidad generada (porcentaje de certeza), la cantidad de detecciones y falsos positivos. Para la prueba se utilizó un total de 30 datos en los cuales se empleó como constante la distancia máxima entre cada punto de $5.00[cm]$. Como parámetro variable se toma la cantidad de puntos necesarios para detectar una pierna desde un valor de 2 a 6 puntos. Para cada número de puntos se realizaron cinco iteraciones. La distancia utilizada entre el robot y la persona fue de $1.50[m]$ donde este valor se tomó a partir de los siguientes parámetros:

$$D_{prueba} = D_{social} + R_{robot} + D_{maniobra} \quad (37)$$

- **Distancia de prueba (D_{prueba}):** Distancia a la cual se realizará la recolección de datos para el algoritmo.
- **Distancia social (D_{social}):** Distancia máxima a la cual debe estar el robot de la persona $1.00\ [m]$.
- **Radio del robot (R_{robot}):** Radio del robot para tomar en cuenta la cinemática de la plataforma $0.215\ [m]$.
- **Distancia de maniobra ($D_{maniobra}$):** Distancia mínima para giro del robot $0.285\ [m]$.

En la Tabla 39 se muestra los valores obtenidos de las pruebas realizadas. De cada prueba se obtiene el promedio de confiabilidad, las lecturas correctas y los falsos positivos.

Tabla 39

Resumen de resultado del nodo de detección de piernas

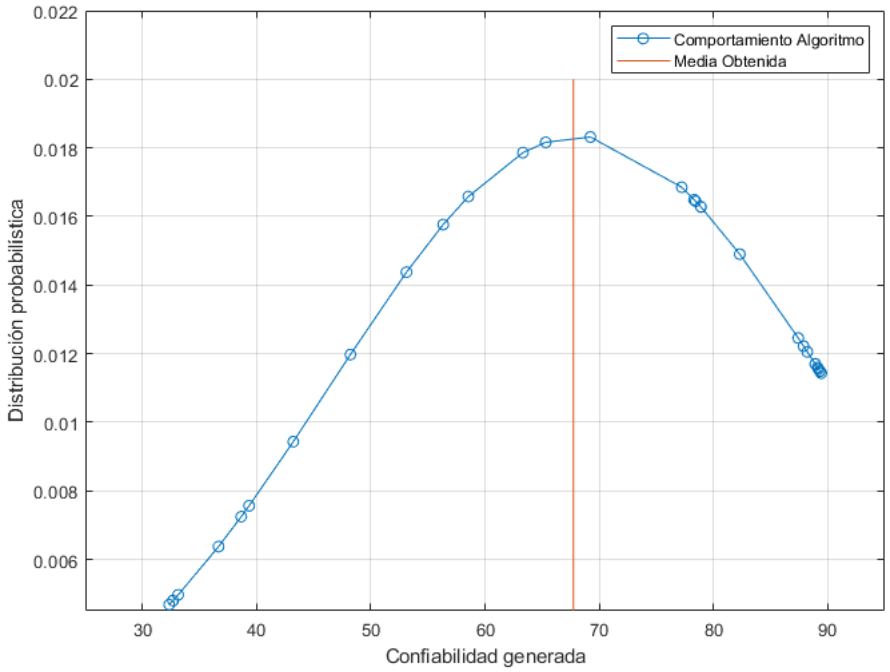
# puntos	confiabilidad	lecturas correctas	falsos positivos
2	40.111 %	21	21
3	84.477 %	22	7
4	90.032 %	23	2
5	80.202 %	16	3
6	49.146 %	8	2

Nota: En la toma de datos se utilizaron a 3 personas (6 piernas).

El comportamiento del algoritmo se muestra en la Figura 87. Se puede observar que cuando existe una identificación correcta de una pierna utilizando un número de puntos entre 3 y 5 se obtiene una confiabilidad mayor al 67.35%. Por otro lado, al utilizar un número de puntos fuera del rango anterior se obtiene un valor de confiabilidad menor al 50.00%.

Figura 87

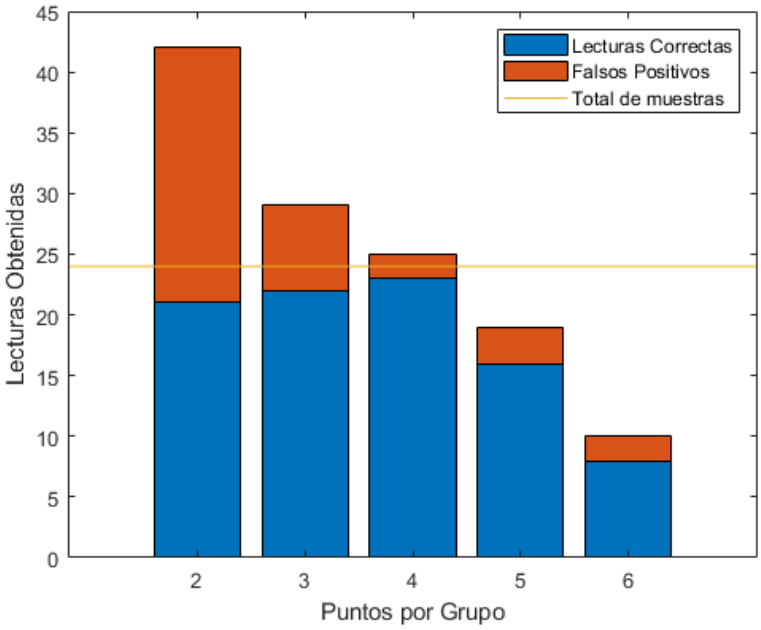
Confiabilidad obtenida por el detector de piernas



La comparación entre lecturas correctas y falsos positivos se muestra en la Figura 88. Se puede ver que el mejor parámetro para la cantidad de puntos corresponde a 4 ya que presenta la menor cantidad de falsos positivos y la mejor respuesta del algoritmo.

Figura 88

Lecturas y Falsos Positivos obtenidas en el detector de piernas

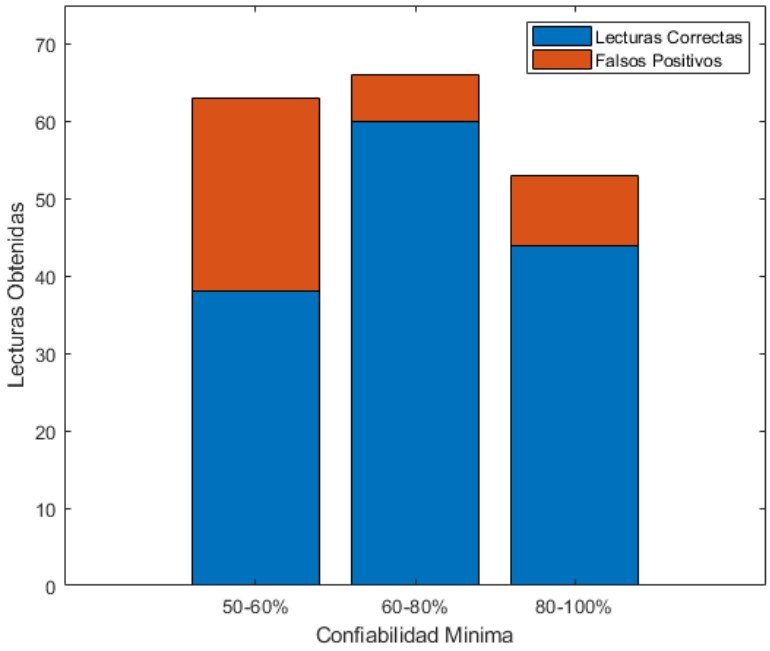


Estimación de posición de personas

Para la toma de datos sobre la generación de personas se utiliza de base la confiablidad obtenida del detector de piernas y la distancia máxima que puede existir entre dos piernas. Para la prueba estuvo presente una persona en el entorno. Se tomó un total de 60 datos con un parámetro fijo de separación de piernas máximo de $38\ [cm]$. De todas las muestras se dividieron en tres secciones la primera correspondiente a una confiablidad mínima entre el 50-60%, la segunda entre 60-80% y la tercera para un valor de 80-100%. La distancia de prueba fue de $1.50\ [m]$ al igual que el detector de piernas.

Figura 89

Lecturas correctas y falsos positivos del algoritmo de detección de personas



Como se muestra en la Figura 89, la mejor respuesta del algoritmo se obtiene al utilizar una confiabilidad entre 60-80% generando 56 lecturas correctas y solo 4 falsos positivos. Por lo cual se implementó una confiabilidad mínima del 65% basado en el comportamiento del detector de la Figura 87. El resumen de los datos se muestra en la Tabla 40.

Tabla 40

Parámetros obtenidos del generador de personas

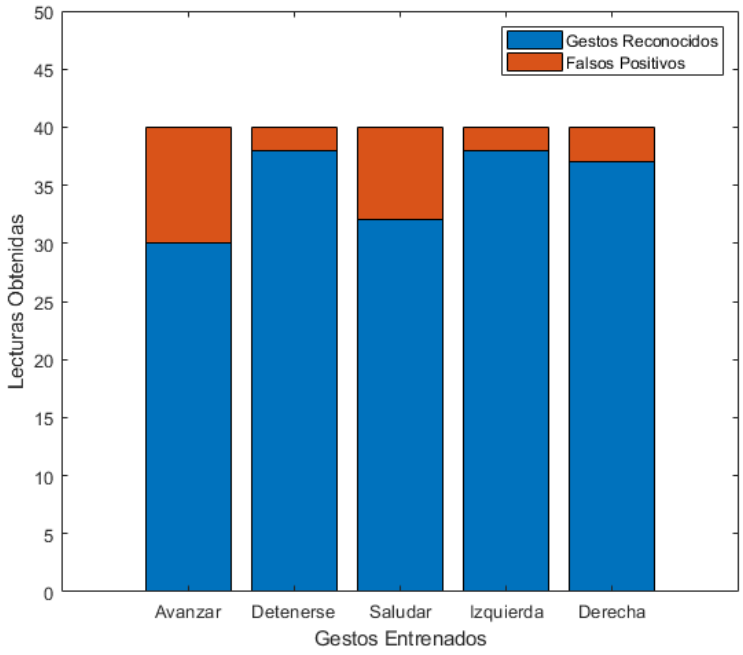
Confiabilidad	Lecturas correctas	Falsos positivos	Exactitud %	Precisión %
50-60%	38	25	43.182	60.317
60-80%	60	6	83.333	90.909
80-100%	44	9	70.968	83.019

Pruebas del reconocimiento de gestos

Para el nodo de reconocimiento de gestos se implementó una prueba con cinco gestos implementados en (Andrango, 2020) dichos gestos son “avanzar, detenerse, avanzar, izquierda y derecha”. La representación de los gestos se muestra en la Figura 61. Para esta prueba se tomó un total de 40 iteraciones por cada gesto. Se probaron las lecturas correctas y los falsos positivos como se muestra en la Figura 90.

Figura 90

Lecturas y falsos positivos del nodo de reconocimiento de gestos



En los resultados se pude observar que los gestos de “Avanzar” y “Saludar” son los gestos que presentan mayor cantidad de falsos positivos debido a son gestos con movimiento para su detección. Cuando la persona que está siendo detecta se pone en diagonal o de lado altera. Los puntos de cada unión del cuerpo se chocan, generan una lectura errónea que confunde la detección con otro gesto como se muestra en la Tabla 41. Por lo tanto, los gestos estáticos presentan una mayor precisión generando una exactitud de reconocimiento general del 87.5%.

Tabla 41

Matriz de confusión del reconocimiento de gestos

Gestos	**Reconocimiento**				
	Saludar	Avanzar	Detenerse	Izquierda	Derecha
Saludar	32	3	2	0	3
Avanzar	2	30	6	2	0
Detenerse	1	1	38	0	0
Izquierda	2	0	0	38	0
Derecha	2	1	0	0	37
Precisión	82.1%	85.7%	82.6%	95%	92.5%

Pruebas de navegación

Las pruebas de navegación integran pruebas en un entorno vacío, entorno con objetos que son obstáculos para comprobar la navegación reactiva del robot y entorno con personas para conocer el comportamiento social del sistema. Las pruebas se desarrollaron en un ambiente controlado. Un entorno basado en la disponibilidad del espacio en el interior de una casa. El entorno utilizado fue previamente mapeado por el robot para generar un mapa en 2D que servirá para ubicar al robot y sus puntos objetivo. En todas las pruebas realizados el robot tuvo una velocidad lineal máxima de 0.2m/s. La velocidad angular máxima fue de 0.45 rad/s. Además, se limitó el movimiento del robot para desplazarse únicamente hacia adelante. El movimiento hacia atrás genera una pérdida de las propiedades sociales del robot: comodidad y naturalidad.

Descripción del entorno de navegación

La Figura 91 muestra imágenes del entorno utilizado para la navegación. Se puede observar que cuenta con un piso de baldosa, paredes que limitan el área de movimiento del robot y, además una columna interna que será un obstáculo permanente para el robot. Para dar a conocer las medidas del entorno se presenta el plano del

Anexo D que muestra un perímetro global de $20.18\ [m]$. Pero el área útil donde podrá moverse el robot es aproximadamente de $18.16\ [m^2]$ debido a los muebles que contiene el entorno.

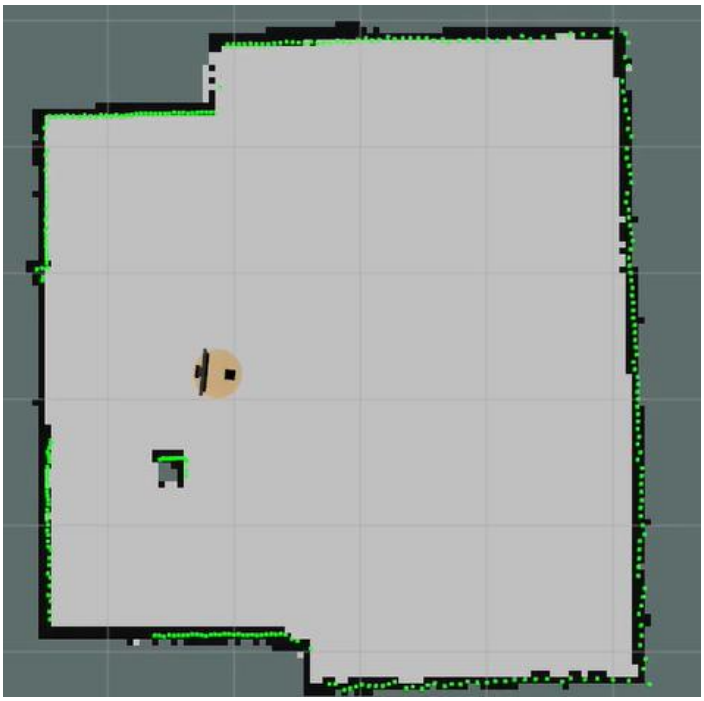
Figura 91

Entorno para la navegación del robot

Con el sensor Lidar se generó el mapa en 2D para poder navegar en el entorno conociendo la localización del robot y los objetivos dentro del mapa. La Figura 92 muestra el mapa obtenido del entorno.

Figura 92

Mapa del entorno para la navegación del robot



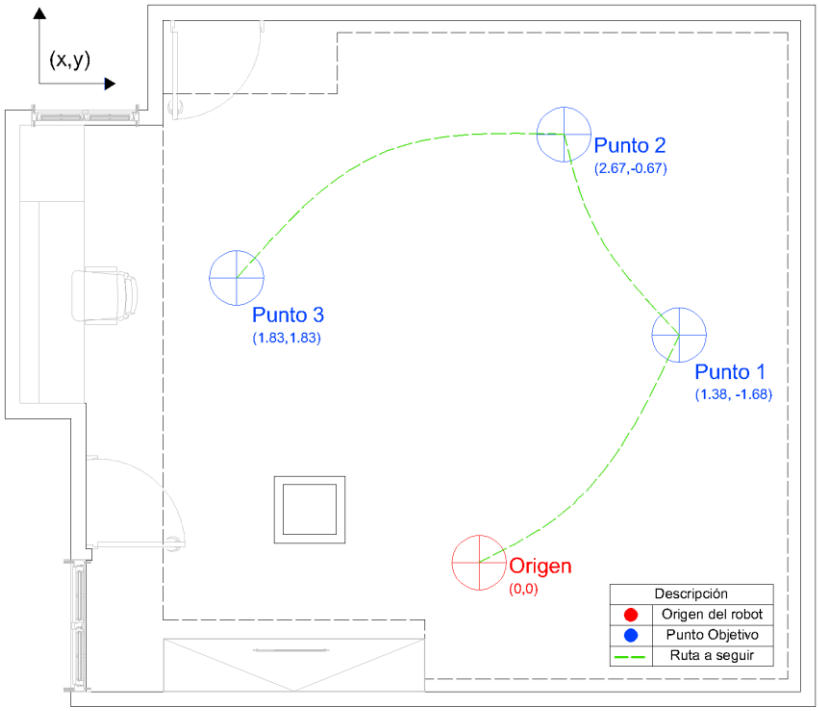
Para las pruebas respectivas, en el entorno estarán presentes una persona estática y dos objetos inertes que se presentarán como obstáculos para la navegación del robot. Además, en el entorno deberá existir conexión a internet para realizar la comunicación con el operador remoto.

Pruebas en el entorno vacío

Las primeras pruebas realizadas fueron con el robot en el entorno vacío. El robot se desplazó a tres puntos establecidos en el mapa. La trayectoria seguida por el robot se encuentra representada en la Figura 93.

Figura 93

Representación de la ruta del robot en el entorno vacío



La prueba en el entorno vacío se repitió tres veces. En todas las repeticiones el robot logró llegar a cada uno de los puntos, sin realizar maniobras de recuperación (“rotate_recovery”) debido a que siempre tuvo ruta que seguir. El tiempo promedio de navegación hasta llegar al objetivo y el error en odometría para cada uno de los puntos a los que llegó el robot se presentan en la Tabla 42.

Tabla 42

Datos de las pruebas en entorno vacío

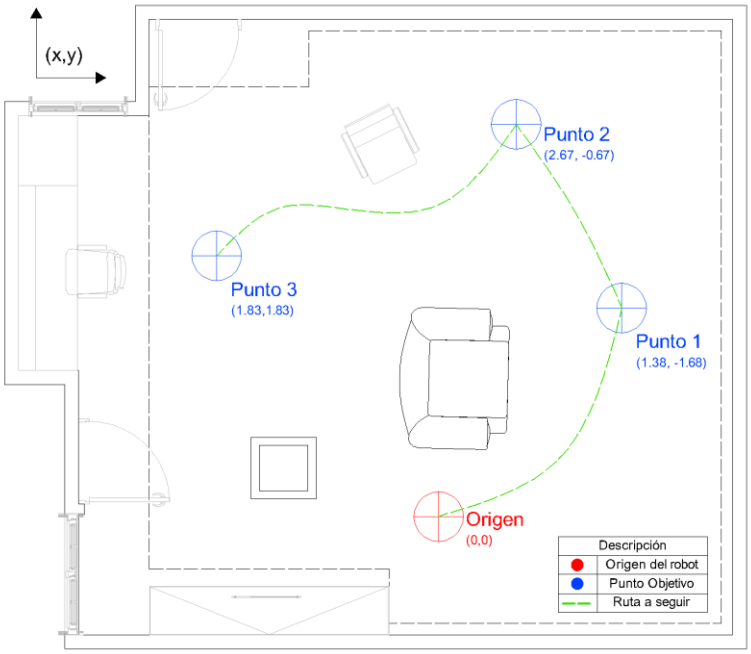
Puntos (x,y)	Tiempo promedio de navegación [s]	Error promedio [cm]
P1 (1.38, -1.64)	14.77	3.72
P2 (2.97, -0.67)	13.11	4.81
P3 (1.83, 1.83)	19.14	8.35

Pruebas de navegación con obstáculos

Para la prueba con obstáculos se añadió un sillón y una silla al entorno. Al igual que la prueba en entorno vacío se estableció tres puntos como objetivos para el robot. La prueba se repitió tres veces.

Figura 94

Representación de la ruta del robot en el entorno con obstáculos



La trayectoria seguida por el robot y los puntos se muestran en Figura 94. Se observa también la posición de los obstáculos en el entorno. Para la prueba con obstáculos se muestra el tiempo promedio de navegación y el módulo del error en odometría para para punto. La Tabla 43 muestra los datos mencionados.

Tabla 43

Datos de las pruebas en navegación reactiva

Puntos (x,y)	Tiempo promedio de navegación [s]	Error promedio [cm]
P1 (1.38, -1.64)	24.45	17.69
P2 (2.97, -0.67)	23.81	20.46
P3 (1.83, 1.83)	25.5	21.61

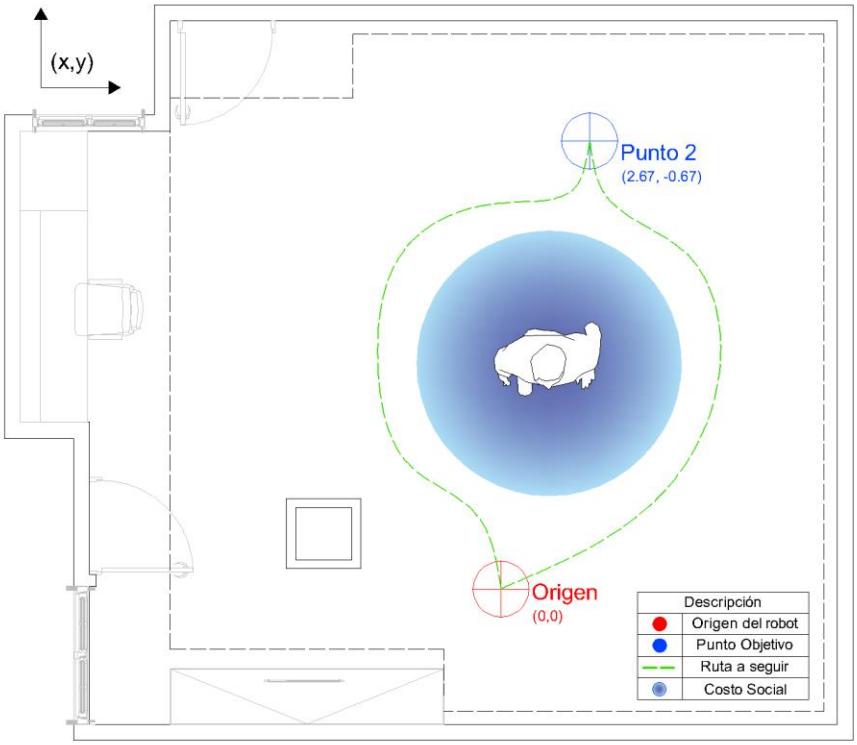
En cada una de las pruebas el robot evadió los obstáculos sin colisión. La distancia a la que más cerca pasó el robot de los obstáculos fue de 16,2 cm. Y la distancia más lejana fue de 25,21 cm.

Pruebas de navegación con personas

Para comprobar el comportamiento social y la aplicación de las normas sociales en la navegación del robot se realizó la prueba con una persona en el entorno. Dada la limitación de espacio en el entorno se planteó dos puntos para mover al robot.

Figura 95

Representación de la ruta del robot en el entorno con una persona



Se realizó tres repeticiones de la prueba obteniendo los tiempos de navegación hacia cada punto y el error en odometría. Para evidenciar el comportamiento social se midió la distancia más cercana entre el robot y la persona. En la Figura 95 se muestra la trayectoria seguida por el robot durante las pruebas y las posiciones a donde se

trasladó. En la Tabla 44 se presenta los datos obtenidos de la prueba de navegación social.

Tabla 44

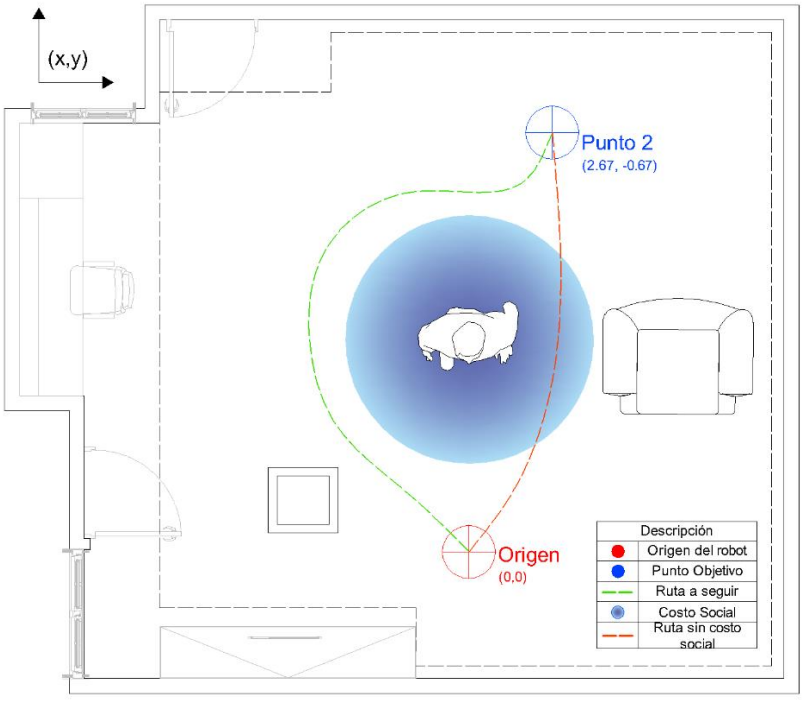
Datos de las pruebas en navegación social

Puntos (x,y)	Tiempo promedio de navegación [s]	Error promedio [cm]
P1 (2.97, -0.67)	22.11	9.21
P2 (0, 0)	21.96	5.92

De acuerdo con los parámetros configurados para la navegación el robot debe mantener la distancia con la persona en un rango entre 60 cm y 80 cm. La distancia más cercana del robot a la persona durante las pruebas fue de 72 cm. En las pruebas de navegación social se incluyó obstáculos inertes y la persona para conocer el comportamiento del robot en un entorno variado. En la Figura 96 se muestra la posición del obstáculo y la persona en el entorno.

Figura 96

Representación de la ruta del robot en el entorno con una persona y obstáculos



Se realizaron dos pruebas. Una prueba incluía la capa social encargada de dar mayor distancia entre el robot y la persona. Y la otra prueba percibía a la persona como otro objeto inerte. La Figura 96 muestra el comportamiento del robot cuando se toma a la persona como un obstáculo cualquiera. La Tabla 45 muestra la diferencia en la distancia mínima medida entre el robot y la persona. Con comportamiento reactivo se toma a la persona como objeto y en social se discrimina a la persona detectada.

Tabla 45

Comparación en navegación social y reactiva

Comportamiento	Distancia mínima a la persona [cm]
Reactivo	22
Social	72

Para concluir las pruebas de navegación se presenta en la Tabla 46 una comparación de los tiempos de navegación y error de localización en el entorno vacío y con objetos presenten.

Tabla 46

Tiempos de navegación y error de localización en diferentes entornos

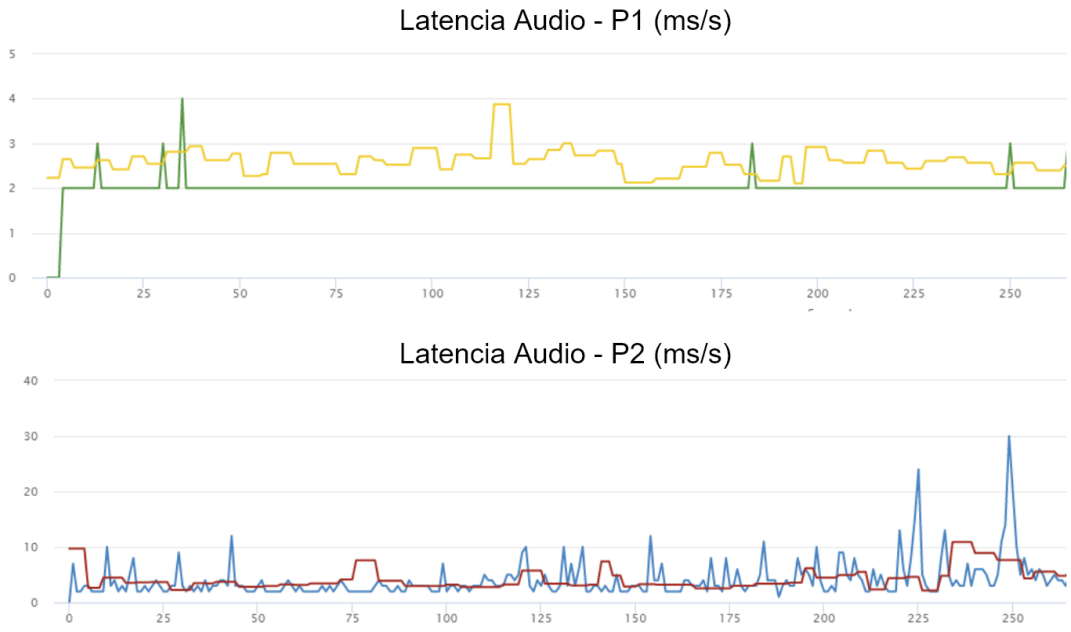
Entorno	Tiempo navegación [s]	Error de localización [cm]
Vacío	47.02	8.35
Con obstáculos	73.77	21.61

Pruebas de telepresencia y teleoperación

Para la prueba de telepresencia se realizaron dos tomas de datos de una duración de 6 minutos. Se realizaron entre la aplicación android y la interfaz web. La primera prueba se realizó en una misma red de internet (P1) y la segunda aplicación se realizó mediante una conexión remota (P2). Para ambas pruebas se utilizó una velocidad de internet local de 20 Mb/s de descarga y subida.

Figura 97

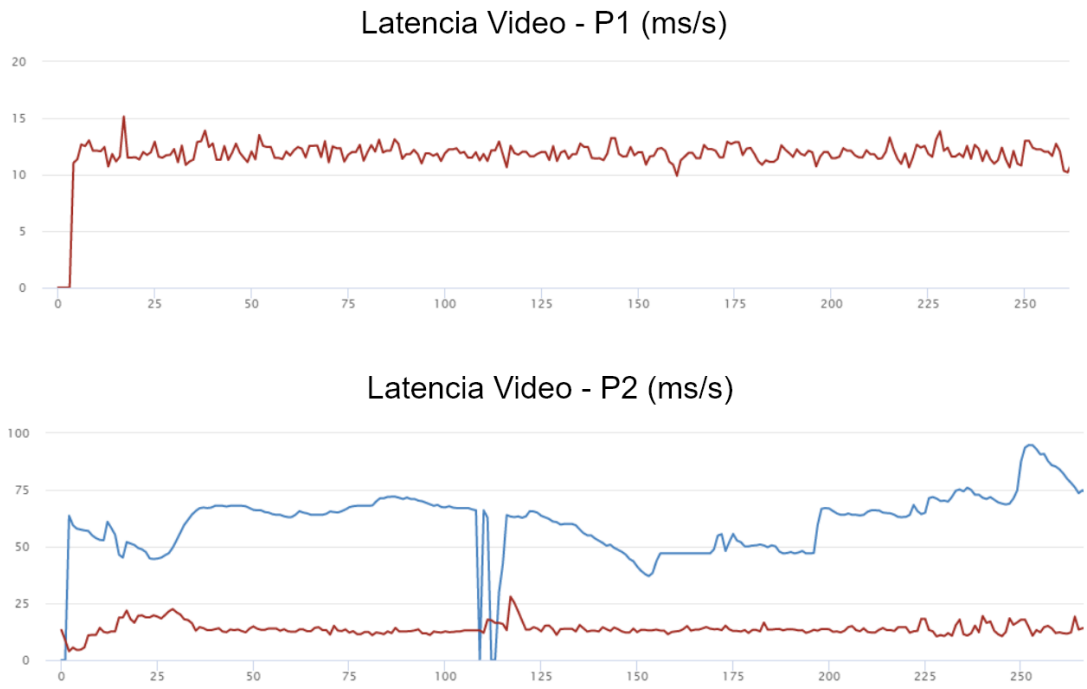
Resultado de latencia de audio entre las dos pruebas de telepresencia



Como se muestra en la Figura 97 los valores de latencia de audio obtenidos en la prueba en la misma red de internet muestran un valor máximo de $4[ms]$ mientras que en la prueba en redes distintas muestra una latencia máxima de $30\ [ms]$. En la Figura 98 se muestra las comparaciones de latencia de video obtenidos en ambas pruebas donde se muestra un valor máximo de $15\ [ms]$ en la red local y un valor de $92\ [ms]$ en la comunicación mediante red remota.

Figura 98

Resultado de latencia de video entre las dos pruebas de telepresencia



En la Figura 99 se muestra la comparación en el rendimiento de transmisión de frames entre la prueba uno y dos en donde se muestra un valor de $\max: 31[fps] - min: 29[fps]$ y en la prueba un valor de $\max: 65\ [fps] - min: 10[fps]$ lo que genera una baja en la calidad de imagen, debido a la velocidad del internet en cada punto.

Figura 99

Resultado de los cuadros por segundo entre las dos pruebas de telepresencia

Todos los valores obtenidos de las dos pruebas se muestran en la Tabla 47. Se puede ver que existe mayor estabilidad en los cuadros por segundo en la prueba en la misma red. Pero los valores de latencia muestran que existe un control del robot viable de forma remota.

Tabla 47

Resumen de parámetros obtenidos en las pruebas de telepresencia

Parámetros	Prueba local	Prueba Remota
Latencia Audio	2 - 4 [ms/s]	5-30 [ms/s]
Latencia Video	11 - 15[ms/s]	26 - 92[ms/s]
Cuadros por Segundo	29- 31 [fps]	10 - 65 [fps]
Paquetes perdidos	0%	1.6%
Caídas de Conexión	0	1

Resumen

En el capítulo se presentó las pruebas realizadas a la plataforma las cuales fueron implementadas en un entorno interior controlado. Se realizaron dos pruebas de odometría la primera en una trayectoria cerrada y la segunda en una trayectoria recta.

Las pruebas detección de piernas, así como el reconocimiento de gestos se realizaron con varios sujetos a una distancia de 1.50[m]. Dicha distancia se reutilizo para las pruebas de estimación de posición de una persona. Las pruebas de navegación se implementaron en tres circunstancias en un entorno vacío, uno con obstáculos y el ultimo con obstáculos y una persona. Por ultimo las pruebas de telepresencia muestran los parámetros de latencia y tasa de cuadros por segundo durante la transmisión remota y local. Los resultados más relevantes de cada una de las pruebas se muestran en la Tabla 48.

Tabla 48

Resultados obtenidos

Pruebas	Resultados
Odometría	Precisión odométrica: inicial = 65.7[cm], actual = 8.7[cm] Área recorrida = un metro cuadrado
Detección piernas	Confiabilidad = 90.03%, puntos mínimos = 4
Estimación posición persona	confiablidad mínima = 65%, exactitud = 83.33% y precisión = 90.09%
Reconocimiento de gestos	Exactitud general = 87.5%, precisión general = 87.57%
Navegación	Tiempo: entorno obstáculos = 73.77[s], entorno vacío = 47.03[s] Distancia mínima: robot-persona = 72[cm], robot-obstáculo = 16.2[cm]
Sistema de Telepresencia	Latencia de video: local = 11.5[ms/s], remota =56.3[ms/s] Paquetes perdidos: local = 0%, remota = 1.6% Caída máxima de FPS: local = 2[fps] , remota = 15[fps]

Capítulo VI

Conclusiones y recomendaciones

El capítulo expone las conclusiones obtenidas durante el desarrollo del trabajo de investigación. Además, se presentan recomendaciones y posibles trabajos futuros a partir del sistema implementado.

Conclusiones

En el trabajo de investigación se realizó el diseño e implementación de un sistema de navegación reactiva-social y telepresencia en un prototipo de robot diferencial. El sistema incluye normas sociales de navegación y es posible controlar a la plataforma de forma remota. Las normas sociales de navegación incluyen: no invadir la zona personal del individuo, movimiento a una velocidad que no afecte el estado del ser humano y navegación solo hacia adelante.

Para el desarrollo del sistema, se partió de la evaluación del prototipo inicial. Del cual se analizó la autonomía energética mostrando que la batería con menor capacidad tiene una duración de 1 hora y 20 minutos. La autonomía energética de la plataforma aumentó al integrar la unidad de procesamiento Jetson Nano que tiene menor consumo que la computadora Intel NUC. Para la transmisión de datos entre el Arduino Mega y la unidad de control se utilizaba un puente de comunicación serial mediante Python. Lo que provocaba caídas de comunicación repentinas inhabilitando el control de la plataforma.

Las correcciones implementadas en el prototipo iniciaron con la integración de la tarjeta de adquisición Teensy 3.6. La tarjeta permitió una frecuencia de funcionamiento 10 veces mayor que el Arduino Mega. Se utilizó el firmware implementado en (Jimeno, 2012). Lo que permitió una comunicación serial mediante el paquete “ros serial”(Ferguson, 2009), logrando solucionar las caídas de comunicación.

Para la navegación reactiva se integró un anillo de ocho sensores ultrasónicos alrededor de la plataforma. El anillo de sonares posee un ángulo de detección total de 160º. Lo que permite obtener un sistema más robusto en la evasión de obstáculos del entorno. Para la implementación de la telepresencia se añadió una pantalla táctil de 10,7 pulgadas, y una cámara USB con micrófono integrado. Ambos elementos fueron colocados en un soporte de aluminio. Por lo tanto, la altura total del robot es de 1.30 metros, brindando comodidad para la interacción social.

Por medio de ROS y Gazebo se implementó un modelo virtual del robot y se simuló el funcionamiento de sus sensores y actuadores. Lo que permitió probar el sistema diseñado sin necesidad de la plataforma física. La simulación brindó una idea del movimiento del robot en su entorno y las variaciones al configurar los parámetros de navegación.

Se diseñó un controlador para los motores al notar que la documentación del prototipo no incluía la identificación de los actuadores y sintonización de un controlador. En el diseñó se obtuvo un controlador PID sintonizado por el método de Ziegler-Nichols. Posteriormente fue afinado e implementado en los motores de la plataforma.

El cálculo de odometría se implementó mediante un Filtro de Kalman Extendido. El filtro recibió los datos de los encoders y del sensor inercial MPU 6050 añadido a la plataforma. Los resultados de odometría se obtuvieron del análisis de la prueba UMBmark bidireccional realizada en una trayectoria cerrada. Se mostró una mejoría de la precisión odométrica inicial de 65 cm a 8.7 cm en una trayectoria cuadrada de $1[m^2]$.

Para el reconocimiento de gestos se utilizó el paquete “skeleton_markers” (Goebel, 2015) y los gestos desarrollados en (Andrango, 2020). Se confirma el comportamiento del algoritmo para los gestos entrenados teniendo una exactitud del 87.5 % al analizar un total de 200 muestras procesadas. Cada gesto se relacionó con un movimiento del robot y así se hizo perceptible la acción de la plataforma de acuerdo

con el gesto realizado por la persona. Debido a la configuración del paquete “neuro_gesture_kinect” (Parhar, 2015) se debe esperar 3.2 segundos para que el nodo de reconocimiento realice la acción deseada.

La detección de piernas se realizó con el paquete “leg_detector” (Lu & Smart, 2013). La frecuencia del sensor lidar, la distancia de detección y el entorno del entrenamiento difieren de los utilizados en el paquete. Por lo que se realizó el reentrenamiento del algoritmo con datos positivos (personas caminado dentro del entorno) y datos negativos (elementos con geometría similar a una pierna). Se obtuvo una precisión del 90.032% en el reconocimiento al utilizar un mínimo de 4 puntos por grupo a una distancia de 1.5[m].

Para la planificación de ruta se utilizó un planificador global y uno local. El planificador global fue el algoritmo A* que se encargó de generar la ruta desde la ubicación actual hacia el objetivo. Y como planificador local se utilizó el algoritmo DWA que generó las velocidades permisibles para navegar sin colisiones. Para obtener el costo del movimiento del robot dentro de su entorno se utilizó la estructura por capas propuesta por (Lu et al., 2014). Por lo que, se logró discriminar los objetos inertes como obstáculos, de las personas. A las personas se les asignó una zona prohibida para brindar comodidad y mostrar sociabilidad durante la navegación.

En las pruebas de navegación se utilizó una velocidad lineal de 0.2 m/s y una velocidad angular de 0.45 rad/s. En el entorno vacío se logró obtener un tiempo de 47.03 segundos de navegación entre los tres objetivos mostrados en la Tabla 46. Mientras que durante la navegación reactiva se obtuvo un tiempo de 73.77 segundos. El tiempo de navegación incrementa debido a que la plataforma disminuye su velocidad y altera su ruta para no entrar dentro del área asignada para cada obstáculo.

Durante la navegación social la distancia mínima entre el robot y una persona fue de 72 cm. El valor supera la distancia propuesta de 60 cm, evitando entrar en la

zona personal del individuo. Por otro lado, la distancia mínima entre un obstáculo y la plataforma fue de 16.2 cm. Al implementar una navegación en la cual el robot no pueda moverse hacia atrás se logró obtener un movimiento natural y confortable para las personas del entorno durante la navegación. La velocidad de navegación de 0.2 m/s también mantiene la tranquilidad de la persona en el entorno, pues velocidades mayores generan una sensación de inseguridad en las personas.

El sistema de telepresencia se basa en una comunicación punto a punto entre la plataforma móvil y el operador mediante el framework WebRTC. La cual se establece al conectar ambos puntos mediante una conexión socket utilizando un servidor STUN público. Y un servidor TURN implementado en AWS para conexiones remotas. Por lo tanto, se logró implementar un sistema de comunicación responsivo al obtener un valor medio de latencia en video de 11.5 ms cuando el operador se encuentra dentro de la misma red y un valor de 56.3 ms cuando se encuentra en una red remota. Lo que permite al operador controlar la plataforma mediante la visualización del video del entorno. Por otro lado, al utilizar el framework se logró obtener una transmisión estable de 30 fps con una caída máxima de 2 cuadros por segundo en la red local. Mientras que en la red remota se obtuvo una caída máxima 15 cuadros por segundo como se puede ver en la Figura 99. Se utilizó una resolución de 640x480 pixeles.

Recomendaciones

La estructura actual de la plataforma puede ser mejorada mediante un análisis de vibraciones. Junto con el reposicionamiento de los elementos estructurales para no obstruir el campo de detección de los sensores y brindar más estabilidad a la estructura.

Es recomendable cambiar la fuente de la alimentación de la plataforma, por dos fuentes de litio de gel. Para entregar una descarga de voltaje más estable y permitir la entrega mayor corriente en el arranque aumentando la autonomía de la plataforma.

Aunque el controlador diseñado e implementado permite navegar y llegar a los objetivos dentro del entorno, se aconseja añadir niveles de control lo que permitirá mejorar la respuesta del robot frente a perturbaciones en las llantas, y minimizar errores durante la navegación y localización en entornos con superficies irregulares.

Para mejorar el cálculo de odometría, es necesario colocar al IMU en el centro de masa del robot y en una posición fija. Además, incluir un magnetómetro para corregir el crecimiento progresivo del dato de giro del sensor inercial. Redimensionar la caja reductora de los motores para obtener mayor resolución y menor ruido en el dato de los encoders.

Debido a la emergencia sanitaria el sistema de navegación se implementó en un espacio interior de un hogar, con parámetros específicos del detector de piernas para este entorno. Por lo cual para mejorar el funcionamiento del algoritmo en entornos de mayor tamaño es recomendable reconfigurar los parámetros utilizados.

Se recomienda implementar el sistema de telepresencia en un lugar con mayor velocidad de internet para disminuir la latencia y aumentar la resolución del video en la llamada. Implementar el servidor TURN en un computador local para mejorar la velocidad de transmisión y no utilizar un servicio externo como AWS.

Aumentar el número de unidades de procesamiento a bordo de la plataforma. En donde la primera se encargue del cálculo de localización, detección de piernas y navegación mientras la segunda unidad se encargue del reconocimiento e interacción mediante gestos, así como la conexión con WebRTC.

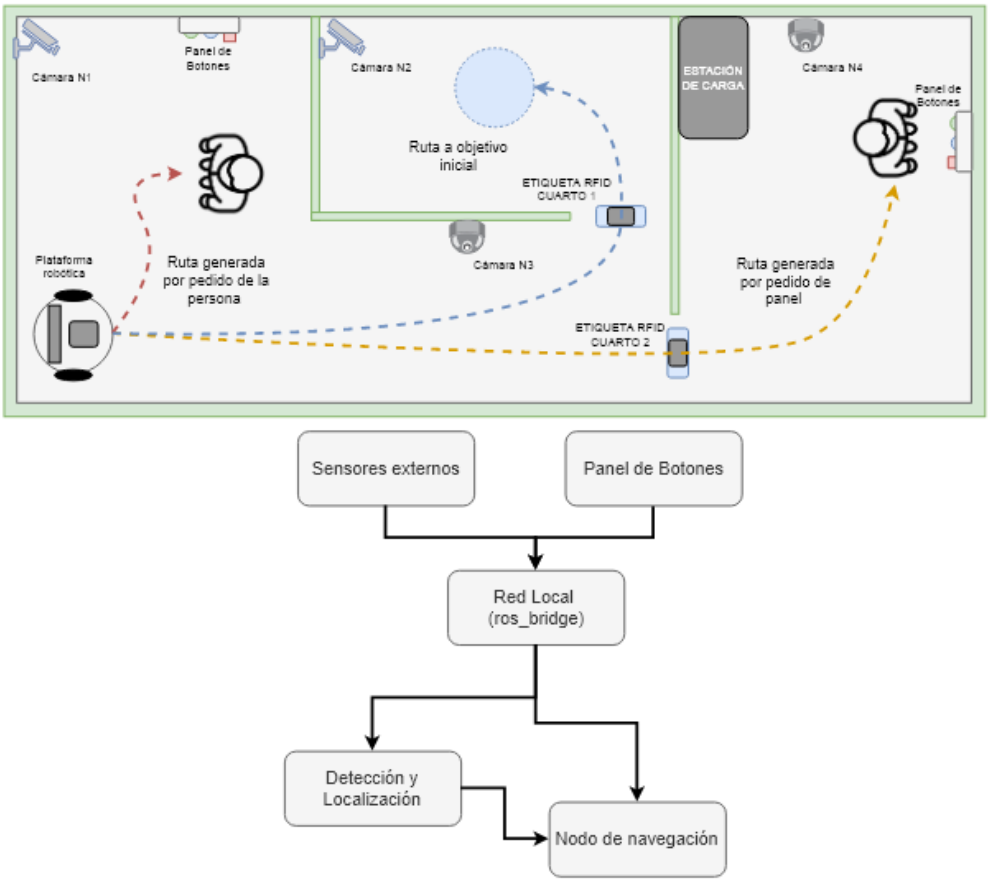
Además, para tener mayor compatibilidad con paquetes de ROS se recomienda actualizar el sensor Kinect. Ya que el sensor integrado en la plataforma, Kinect V1, se encuentra sin soporte en actualidad. Los sensores que podrían reemplazarlo son: sensor Asus Xtion 2 (ASUS, s. f.), Intel RealSense Camera (Intel, s. f.) o Kinect V2 (Microsoft, s. f.).

Trabajos futuros

A partir del sistema de navegación implementado se propone desarrollar un entorno inteligente. El cual debe incluir sensores exteroceptivos tales como: cámaras y lectores de código. Las cámaras permitirán la detección de objetos y personas independientemente de la posición y movimiento de la plataforma. Al incluir códigos específicos para diferentes áreas del entorno se podrá modificar el comportamiento del robot dependiendo del área en la que se encuentra. La integración de cámaras permitirá transmitir la imagen del entorno al operador remoto, mediante WebRTC. Por lo que, el operador podrá tener información para mayor control de los movimientos del robot un esquema general se muestra en la Figura 100.

Figura 100

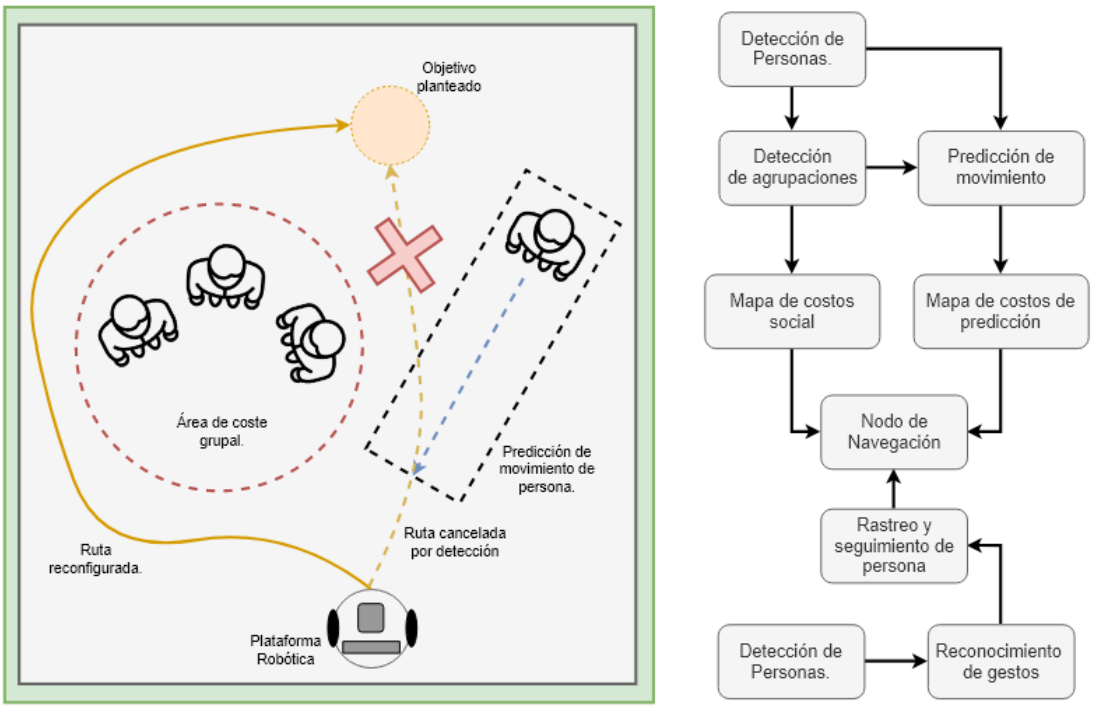
Esquema ejemplo para la implementación en un entorno inteligente



El siguiente paso en el comportamiento del robot en un entorno social es depender del movimiento de las personas. Por lo que se propone implementar un modelo de predicción de movimiento permitiendo a la plataforma tener un comportamiento dinámico durante la navegación. Además, se puede implementar un reconocimiento de grupos para poder realizar una navegación en lugares concurridos, como centros comerciales y museos. Por otro lado, se puede utilizar el reconocimiento de gestos para realizar el seguimiento a un individuo sin entrar en su espacio personal, un ejemplo de esta propuesta se muestra en la Figura 101.

Figura 101

Esquema ejemplo para la mejora de detección de personas y predicción de movimiento



Referencias bibliográficas

Ahmad, N., Ghazilla, R. A. R., Khairi, N., & Kasi, V. (2013). Reviews on Various Inertial Measurement Unit (IMU) Sensor Applications. *International Journal of Signal Processing Systems*, 256-262. https://doi.org/10.12720/ijsps.1.2.256-262

Andrango, F. J. (2020). *Implementación de reglas de comportamiento social en una plataforma robótica de telepresencia, a través del reconocimiento de gestos*. Universidad de las Fuerzas Armadas ESPE.

Andreasson, H., Magnusson, M., & Lilienthal, A. (2007). Has something changed here? Autonomous difference detection for security patrol robots. *IEEE International Conference on Intelligent Robots and Systems*, 3429-3435. https://doi.org/10.1109/IROS.2007.4399381

Aoude, G. S., Luders, B. D., Joseph, J. M., Roy, N., & How, J. P. (2013). Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns. *Autonomous Robots*, *35*(1), 51-76. https://doi.org/10.1007/s10514-013-9334-3

Arkin, R. C. (1990). Integrating behavioral, perceptual, and world knowledge in reactive navigation. *Robotics and Autonomous Systems*, *6*(1-2), 105-122. https://doi.org/10.1016/S0921-8890(05)80031-4

Arras, K. O., Mozos, Ó. M., & Burgard, W. (2007). Using boosted features for the detection of people in 2D range data. *Proceedings - IEEE International Conference on Robotics and Automation*, *April*, 3402-3407. https://doi.org/10.1109/ROBOT.2007.363998

ASUS. (s. f.). *Xtion 2 - ASUS*. Recuperado 3 de marzo de 2021, de https://www.asus.com/ch-en/Networking-IoT-Servers/Smart-Home/Security-Camera/Xtion-2/

Bailey, T., & Durrant-Whyte, H. (2006). Simultaneous localization and mapping (SLAM):

Part II. *IEEE Robotics and Automation Magazine*, *13*(3), 108-117. https://doi.org/10.1109/MRA.2006.1678144

Bañó, A. A. (2003). Análisis y Diseño del Control de Posición de un Robot Móvil con Tracción Diferencial. *Escola tecnica superior enginyeria*, 167.

Barshan, B., Ayrulu, B., & Utete, S. W. (2000). Neural network-based target differentiation using sonar for robotics applications. *IEEE Transactions on Robotics and Automation*, *16*(4), 435-442. https://doi.org/10.1109/70.864239

Beer, J. M., & Takayama, L. (2011). Mobile remote presence systems for older adults: Acceptance, benefits, and concerns. *HRI 2011 - Proceedings of the 6th ACM/IEEE International Conference on Human-Robot Interaction*, *11*(HRI), 19-26. https://doi.org/10.1145/1957656.1957665

Ben Wright, Jan Ivar Bruaroey, & Mirko Bonadei. (2016). *Trickle ICE*. Trickle ICE checker. https://webrtc.github.io/samples/src/content/peerconnection/trickle-ice/

Borenstein, J., & Feng, L. (1995). UMBmark: A Benchmark Test for Measuring Odometry Errors in Mobile Robots. *Mobile Robots X*, *2591*, 113-124. https://doi.org/10.1117/12.228968

Bräunl, T. (2008). *Embedded robotics (third edition): Mobile robot design and applications with embedded systems* (2.ª ed.). Springer. https://doi.org/10.1007/978-3-540-70534-5

Brock, O., & Khatib, O. (1999). High-speed navigation using the global dynamic window approach. *Proceedings - IEEE International Conference on Robotics and Automation*, *1*(May), 341-346. https://doi.org/10.1109/robot.1999.770002

Butler, J. T., & Agah, A. (2001). Psychological effects of behavior patterns of a mobile personal robot. *Autonomous Robots*, *10*(2), 185-202. https://doi.org/10.1023/A:1008986004181

Cai, Z., & Zhao, D. (2006). Unscented Kalman filter for non-linear estimation. *Geomatics*

and Information Science of Wuhan University, *31*(2), 180-183.

Chen, Y. F., Liu, M., Everett, M., & How, J. P. (2017). Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. *Proceedings - IEEE International Conference on Robotics and Automation*, 285-292. https://doi.org/10.1109/ICRA.2017.7989037

Chen, Y., Liu, M., & Wang, L. (2019). RRT∗ Combined with GVO for real-time nonholonomic robot navigation in dynamic environment. *2018 IEEE International Conference on Real-Time Computing and Robotics, RCAR 2018*, 479-484. https://doi.org/10.1109/RCAR.2018.8621737

Cheng, J., Cheng, H., Meng, M. Q. H., & Zhang, H. (2018). Autonomous Navigation by Mobile Robots in Human Environments: A Survey. *2018 IEEE International Conference on Robotics and Biomimetics, ROBIO 2018*, 1981-1986. https://doi.org/10.1109/ROBIO.2018.8665075

Chik, S. F., Yeong, C. F., Su, E. L. M., Lim, T. Y., Duan, F., Tan, J. T. C., Tan, P. H., & Chin, P. J. H. (2017). Gaussian pedestrian proxemics model with social force for service robot navigation in dynamic environment. *Communications in Computer and Information Science*, *751*, 61-73. https://doi.org/10.1007/978-981-10-6463-0_6

Chik, S. F., Yeong, C. F., Su, E. L. M., Lim, T. Y., Subramaniam, Y., & Chin, P. J. H. (2016). A review of social-aware navigation frameworks for service robot in dynamic human environments. *Journal of Telecommunication, Electronic and Computer Engineering*, *8*(11), 41-50.

Chung, S. Y., & Huang, H. P. (2012). Incremental learning of human social behaviors with feature-based spatial effects. *IEEE International Conference on Intelligent Robots and Systems*, 2417-2422. https://doi.org/10.1109/IROS.2012.6385852

Cisco, C. (2015). *Journey to Collaboration - Cisco*. https://www.cisco.com/c/en/us/solutions/collateral/enterprise/cisco-on-

cisco/journey-to-collaboration.html

Colomer, J. (2018). Estudio de los Sensores para la Detección de Obstáculos Aplicables a Robots Móviles. *Universitat Oberta de Catalunya*, 24-63.

Corke, P., Findlater, K., & Murphy, E. (2012). Skype: A communications framework for robotics. *Australasian Conference on Robotics and Automation, ACRA*, 3-5.

Cunningham, A. G., Galceran, E., Eustice, R. M., & Olson, E. (2015). MPDM: Multipolicy decision-making in dynamic, uncertain environments for autonomous driving. *Proceedings - IEEE International Conference on Robotics and Automation*, *2015-June*(June), 1670-1677. https://doi.org/10.1109/ICRA.2015.7139412

Diego, G., & Arras, T. K. O. (2011). *Please do not disturb! Minimum interference coverage for social robots*. 1968-1973. https://doi.org/10.1109/iros.2011.6094867

Du Toit, N. E., & Burdick, J. W. (2012). Robot motion planning in dynamic, uncertain environments. *IEEE Transactions on Robotics*, *28*(1), 101-115. https://doi.org/10.1109/TRO.2011.2166435

Elfes, A. (1987). Sonar-Based Real-World Mapping and Navigation. *IEEE Journal on Robotics and Automation*, *3*(3), 249-265. https://doi.org/10.1109/JRA.1987.1087096

Everett, H. R. (1995). Sensors for Mobile Robots. En *Sensors for Mobile Robots*. A.K. Peters. https://doi.org/10.1201/9781439863480

Faber, F., Gonsior, C., Bennewitz, M., Joho, D., Eppner, C., Schreiber, M., Görög, A., & Behnke, S. (2009). The humanoid museum tour guide Robotinho. *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, 891-896. https://doi.org/10.1109/ROMAN.2009.5326326

Feil-Seifer, D., & Matarić, M. J. (2009). Human Robot interaction (HRI). En *Encyclopedia of Complexity and Systems Science* (pp. 4643-4659). Springer New York. https://doi.org/10.1007/978-0-387-30440-3_274

Ferguson, M. (2009). *rosserial - ROS Wiki*. http://wiki.ros.org/rosserial

Fiorini, P., & Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *International Journal of Robotics Research*, *17*(7), 760-772. https://doi.org/10.1177/027836499801700706

Fornth, A. (2012). *Introducción a A **. http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html

Fowdur, T. P., Ramkorun, N., & Chiniah, P. K. (2020). Performance Analysis of WebRTC and SIP-based Audio and Video Communication Systems. *SN Computer Science*, *1*(6), 362. https://doi.org/10.1007/s42979-020-00380-z

Fox, D., Burgard, W., Dellaert, F., & Thrun, S. (1999). Monte Carlo Localization: efficient position estimation for mobile robots. *Proceedings of the National Conference on Artificial Intelligence*, 343-349.

Fox, D., Burgard, W., & Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, *4*(1), 23-33. https://doi.org/10.1109/100.580977

Gallardo, D. (1999). Aplicación Del Muestreo Bayesiano En Robots Móviles : Estrategias Para Localización Y Estimación De Mapas Del Entorno. En *Tesis Doctoral*. Universidad de Alicante.

Gaona, A., Loza, D., Byron, C., & Segura, L. (2016). Diseño y construcción de una plataforma de robot móvil teleoperada a bajo costo para interiores. *ESPE*, *1*.

Gibstein, M. (2011). CES coverage anybots QB telepresence robot hands on. *Tech Tracker Tracking Today's Tech*, *12*.

Goebel, P. (2015). *skeleton_markers - ROS Wiki*. http://wiki.ros.org/skeleton_markers

Gonzalez-Jimenez, J., Galindo, C., & Ruiz-Sarmiento, J. R. (2012). Technical improvements of the Giraff telepresence robot based on users' evaluation. *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, 827-832. https://doi.org/10.1109/ROMAN.2012.6343854

González, R., & Rodríguez, F. (2009). Comparative study of localization techniques for mobile robots based on indirect kalman filter. *Proceedings of IFR Int.*, *ii*, 253-258.

Hansen, S. T., Svenstrup, M., Andersen, H. J., & Bak, T. (2009). Adaptive human aware navigation based on motion pattern analysis. *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, 927-932. https://doi.org/10.1109/ROMAN.2009.5326212

Harmo, P., Halme, A., Pitkänen, H., Virekoski, P., Halinen, M., & Suomela, J. (2001). Moving Eye - Interactive Telepresence Over Internet with a Ball Shaped Mobile Robot. *IFAC Proceedings Volumes*, *34*(9), 481-486. https://doi.org/10.1016/s1474-6670(17)41754-x

He, G. F., Park, J. W., Kang, S. K., & Jung, S. T. (2012). Development of gesture recognition-based serious games. *Proceedings - IEEE-EMBS International Conference on Biomedical and Health Informatics: Global Grand Challenge of Health Informatics, BHI 2012*, 922-925. https://doi.org/10.1109/BHI.2012.6211738

Helbing, D., & Molnár, P. (1995). Social force model for pedestrian dynamics. *Physical Review E*, *51*(5), 4282-4286. https://doi.org/10.1103/PhysRevE.51.4282

Heroku. (2007). *Cloud Application Platform | Heroku*. https://www.heroku.com/home

Herring, S. C. (2013). Telepresence robots for academics. *Proceedings of the ASIST Annual Meeting*, *50*(1), 2011-2014. https://doi.org/10.1002/meet.14505001156

Herzog, J. C., Whitney, B., Wang, Y., Jordan, C. S., & Pinter, M. (2012). *SERVER CONNECTIVITY CONTROL FOR TELE-PRESENCE ROBOT*.

Holland, J. M. (2004). Designing mobile autonomous robots. *Choice Reviews Online*, *42*(03), 42-1580-42-1580. https://doi.org/10.5860/choice.42-1580

Huang, K. C., Li, J. Y., & Fu, L. C. (2010). Human-oriented navigation for service providing in home environment. *Proceedings of the SICE Annual Conference*, 1892-1897.

Iarlori, S., Superiore, S., Anna, S., Benettazzo, F., Iarlori, S., Ferracuti, F., Giantomassi, A., Ortenzi, D., Freddi, A., Monteriù, A., Innocenzi, S., Capecci, M., Ceravolo, M. G., & Longhi, S. (2018). Robot Interface Design: The Giraff Telepresence Robot for Social Interaction. *Research Trends in Media Informatics*, *426*(April), 39-46. https://doi.org/10.1007/978-3-319-18374-9

IFR Mobile Robots Market. (s. f.). *Mobile Robots Market Size, Growth, Trend and Forecast to 2023* | *MarketsandMarkets*. Recuperado 12 de febrero de 2020, de https://www.marketsandmarkets.com/Market-Reports/mobile-robots-market-43703276.html

Intel. (s. f.). *Intel RealSense Computer Vision - Depth and Tracking cameras*. Recuperado 3 de marzo de 2021, de https://www.intelrealsense.com/

Intel Corporation. (s. f.). *Mini PC: Intel NUC*. Recuperado 28 de febrero de 2021, de https://www.intel.la/content/www/xl/es/products/boards-kits/nuc/mini-pcs.html

International Federation of Robotics. (2018). Executive Summary World Robotics 2018 Service Robots. *World Robotic Report - Executive Summary*, 11-16.

Jimeno, J. M. (2012). *linorobot Autonomous mobile robot*. https://linorobot.org/2017/

Joseph, L. (2015). *Simulate a mobile robot Learning Robotics Using Python* (Vol. 44, Número 8). Packt Publishing.

Klančar, G., Zdešar, A., Blažič, S., & Škrjanc, I. (2017). *Wheeled Mobile Robotics: From Fundamentals Towards Autonomous Systems* (Buttheworth-heineman (ed.)). Elsevier.

Koller, D., & Fratkina, R. (1998). Using Learning for Approximation in Stochastic Processes. *In Proceedings of the International Conference on Machine Learning (ICML)*, 287-295. https://doi.org/10.1.1.47.7474

Koubâa, A., Sriti, M. F., Javed, Y., Alajlan, M., Qureshi, B., Ellouze, F., & Mahmoud, A. (2016). Turtlebot at Office: A Service-Oriented Software Architecture for Personal

Assistant Robots Using ROS. *Proceedings - 2016 International Conference on Autonomous Robot Systems and Competitions, ICARSC 2016*, 270-276. https://doi.org/10.1109/ICARSC.2016.66

Kristoffersson, A., Coradeschi, S., & Loutfi, A. (2013). A review of mobile robotic telepresence. *Advances in Human-Computer Interaction*, *2013*. https://doi.org/10.1155/2013/902316

Kruse, T., Pandey, A. K., Alami, R., & Kirsch, A. (2013). Human-aware robot navigation: A survey. *Robotics and Autonomous Systems*, *61*(12), 1726-1743. https://doi.org/10.1016/j.robot.2013.05.007

Kuc, R. (2001). Pseudoamplitude scan sonar maps. *IEEE Transactions on Robotics and Automation*, *17*(5), 767-770. https://doi.org/10.1109/70.964675

Kucsera, P. (2006). Sensors for mobile robot systems. *AARMS TECHNOLOGY*, *5*(4), 645-658.

Lazea, G., & Lupu, A. (1997). Aspects on path planning for mobile robots. *Intensive Course on TEMPUS M-JEP 11467*.

Lee, M. K., & Takayama, L. (2011). «Now, I have a body»: Uses and social norms for mobile remote presence in the workplace. *Conference on Human Factors in Computing Systems - Proceedings*, *CH11*, 33-42. https://doi.org/10.1145/1978942.1978950

Leigh, A., Pineau, J., Olmedo, N., & Zhang, H. (2015). Person tracking and following with 2D laser scanners. *Proceedings - IEEE International Conference on Robotics and Automation*, *2015-June*(June), 726-733. https://doi.org/10.1109/ICRA.2015.7139259

Li, M., Jiang, R., Ge, S. S., & Lee, T. H. (2018). Role playing learning for socially concomitant mobile robot navigation. *CAAI Transactions on Intelligence Technology*, *3*(1), 49-58. https://doi.org/10.1049/trit.2018.0008

Lipman, A., & Hall, E. T. (1970). The Hidden Dimension. *The British Journal of Sociology*, *21*(3), 353. https://doi.org/10.2307/589150

Liu, J., Jayakumar, P., Stein, J. L., & Ersal, T. (2014). A multi-stage optimization formulation for MPC-based obstacle avoidance in autonomous vehicles using a LIDAR sensor. *ASME 2014 Dynamic Systems and Control Conference, DSCC 2014*, *2*. https://doi.org/10.1115/DSCC2014-6269

Long, P., Fanl, T., Liao, X., Liu, W., Zhang, H., & Pan, J. (2018). Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning. *Proceedings - IEEE International Conference on Robotics and Automation*, 6252-6259. https://doi.org/10.1109/ICRA.2018.8461113

Loza-Matovelle, D., Verdugo, A., Zalama, E., & Gómez-García-Bermejo, J. (2019). An architecture for the integration of robots and sensors for the care of the elderly in an Ambient Assisted Living Environment. *Robotics*, *8*(3). https://doi.org/10.3390/robotics8030076

Lu, D. V., Hershberger, D., & Smart, W. D. (2014). Layered costmaps for context-sensitive navigation. *IEEE International Conference on Intelligent Robots and Systems*, 709-715. https://doi.org/10.1109/IROS.2014.6942636

Lu, D. V., & Marder-Eppstein, E. (2013). *move_base ROS package*. http://wiki.ros.org/move_base

Lu, D. V., & Marder-Eppstein, E. (2016). *navigation - ROS Wiki*. ROS. http://wiki.ros.org/navigation

Lu, D. V., & Smart, W. D. (2013). Towards more efficient navigation for robots and humans. *IEEE International Conference on Intelligent Robots and Systems*, 1707-1713. https://doi.org/10.1109/IROS.2013.6696579

Ma, X., & Quek, F. (2010). Development of a child-oriented social robot for safe and interactive physical interaction. *IEEE/RSJ 2010 International Conference on*

Intelligent Robots and Systems, IROS 2010 - Conference Proceedings, 2163-2168. https://doi.org/10.1109/IROS.2010.5652715

Mace, J., & Toris, R. (2017, enero 30). *rosbridge_suite - ROS Wiki*. http://wiki.ros.org/rosbridge_suite

Mace, J., & Toris, R. (2018, febrero 13). *Rosdbridge v2.0 Protocol Specifications*. https://github.com/RobotWebTools/rosbridge_suite/blob/develop/ROSBRIDGE_PROTOCOL.md

Madgwick, S. O. H., Harrison, A. J. L., & Vaidyanathan, R. (2011). Estimation of IMU and MARG orientation using a gradient descent algorithm. *IEEE International Conference on Rehabilitation Robotics*, 32. https://doi.org/10.1109/ICORR.2011.5975346

Mavrogiannis, C. I., Blukis, V., & Knepper, R. A. (2017). Socially competent navigation planning by deep learning of multi-agent path topologies. *IEEE International Conference on Intelligent Robots and Systems*, *2017-Septe*, 6817-6824. https://doi.org/10.1109/IROS.2017.8206601

Melendez-Fernandez, F., Galindo, C., & Gonzalez-Jimenez, J. (2017). A web-based solution for robotic telepresence. *International Journal of Advanced Robotic Systems*, *14*(6), 1-19. https://doi.org/10.1177/1729881417743738

Microsoft. (s. f.). *Kinect - Windows app development*. Recuperado 3 de marzo de 2021, de https://developer.microsoft.com/en-us/windows/kinect/

Minsky, M. (1980). Telepresence OMNI Magazine. *OMNI*.

Moore, T., & Stouch, D. (2016). A generalized extended Kalman filter implementation for the robot operating system. *Advances in Intelligent Systems and Computing*, *302*, 335-348. https://doi.org/10.1007/978-3-319-08338-4_25

Muralindran, M., Brendan, K. T. T., Vigneswaran, R., Thayabaren, G., & Iftikhar, M. (2011). A design methodology for video transmission of controlling an internet

based noninterventional mobile robot for orthopedic surgeon (OTOROB). *Proceedings of the 2nd Kuwait Conference on e-Services and e-Systems, KCESS'11*, *February 2016*. https://doi.org/10.1145/2107556.2107560

Najim Al-Din, M. (2017). *Reactive Mobile Robot Navigation Using Fuzzy Controller Aggressive driving recognition View project rediction and Measurement of Surface Mounted Permanent Magnet Motor Performance with Soft Magnetic Composite and Laminated Steel Stator Cores View project*.

Nehmzow, U. (2000). Navigation. En *Mobile Robotics: A Practical Introduction* (1.ª ed., pp. 87-151). Springer-Verlag London. https://doi.org/10.1007/978-1-4471-3392-6_5

NVIDIA. (2019). *NVIDIA Jetson Nano Developer Kit | NVIDIA Developer*. NVIDIA. https://developer.nvidia.com/embedded/jetson-nano-developer-kit

Ollero, A. (2001). *Robótica; manipuladores y robots móviles - Marcombo, S.A. (ediciones técnicas)*. Marcombo S.A.

Oxford, U. web page. (2017). *Robot | TERESA Offical Web Page*. https://whirl.cs.ox.ac.uk/teresa/robot/

Pacchierotti, E., Christensen, H. I., & Jensfelt, P. (2006). Evaluation of passing distance for social robots. *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, 315-320. https://doi.org/10.1109/ROMAN.2006.314436

Pandey, A. K., & Alami, R. (2010). A framework towards a socially aware mobile robot motion in human-centered dynamic environment. *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, 5855-5860. https://doi.org/10.1109/IROS.2010.5649688

Parhar, T. (2015). *neuro_gesture_kinect - ROS Wiki*. http://wiki.ros.org/TanvirParhar/neuro_gesture_kinect

Patki, V., Sonawane, D. N., & Ingole, D. D. (2013). Design and implementation of discrete augmented Ziegler-Nichols PID control. *Communications in Computer and*

Information Science, *296 CCIS*(1), 262-268. https://doi.org/10.1007/978-3-642-35864-7_37

Perez-Higueras, N., Caballero, F., & Merino, L. (2018). Learning Human-Aware Path Planning with Fully Convolutional Networks. *Proceedings - IEEE International Conference on Robotics and Automation*, 5897-5902. https://doi.org/10.1109/ICRA.2018.8460851

Pinter, M., Lai, F., Sanchez, D. S., Ballantyne, J., Roe, D. B., Wang, Y., Jordan, C. S., Taka, O., & Wong, C. W. (2017). *Social behavior rules for a medical telepresence robot*.

PJRC. (s. f.). *Teensy® 3.6 Development Board*. Recuperado 26 de febrero de 2021, de https://www.pjrc.com/store/teensy36.html

Priyandoko, G., Wei, C. K., & Achmad, M. S. H. (2018). HUMAN FOLLOWING ON ROS FRAMEWORK A MOBILE ROBOT. *SINERGI*, *22*(2), 77. https://doi.org/10.22441/sinergi.2018.2.002

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., & Ng, A. (2009). *ROS: an open-source Robot Operating System*.

Quinaluisa, E., & Toapanta, J. (2018). *IMPLEMENTACIÓN DE UN SISTEMA DE NAVEGACIÓN AUTÓNOMO BASADO EN SLAM Y NAVEGACIÓN REACTIVA*. Universidad de las Fuerzas Armadas ESPE.

Raspberry pi Foundation. (2020). *Raspberry Pi 4 Model B – Raspberry Pi*. Raspberry Pi. https://www.raspberrypi.org/products/raspberry-pi-4-model-b/

Rasyid, A., Albaab, M. R. U., Falah, M. F., Panduman, Y. Y. F., Yusuf, A. A., Basuki, D. K., Tjahjono, A., Budiarti, R. P. N., Sukaridhoto, S., Yudianto, F., & Wicaksono, H. (2019). Pothole Visual Detection using Machine Learning Method integrated with Internet of Thing Video Streaming Platform. *2019 International Electronics Symposium (IES)*, 672-675. https://doi.org/10.1109/ELECSYM.2019.8901626

Rekleitis, I. M. (2003). A particle filter tutorial for mobile robot localization. *International Conference on Robotics and*, *3*(TR-CIM-04-02), 1–36.

Rubin, D. B. (1988). Using the SIR Algorithm to Simulate Posterior Distributions. *Bayesian Statistics*, *3*, 395-402.

Ruiz-Sarmiento, J. R., Galindo, C., Gonzalez-Jimenez, J., Blanco, J. L., & Teatinos, C. De. (2011). Navegación Reactiva de un Robot Móvil usando Kinect. *Actas ROBOT 2011*.

Sankar, S., & Tsai, C.-Y. (2019). ROS-Based Human Detection and Tracking from a Wireless Controlled Mobile Robot Using Kinect. *Applied System Innovation*, *2*(1), 5. https://doi.org/10.3390/asi2010005

Sebastian, M., Banisetty, S. B., & Feil-Seifer, D. (2017). Socially-aware navigation planner using models of human-human interaction. *RO-MAN 2017 - 26th IEEE International Symposium on Robot and Human Interactive Communication*, *2017-Janua*, 405-410. https://doi.org/10.1109/ROMAN.2017.8172334

SHANGHAI SLAMTEC CO. LTD. (2009). *Rplidar a1*. www.slamtec.com

Siciliano, B., & Khatib, O. (2016). *Springer handbook of robotics*. https://doi.org/10.1007/978-3-319-32552-1

Siegwart, R., & Nourbakhsh, I. R. (2011). Introduction to autonomous mobile robots. *Choice Reviews Online*, *49*(03), 49-1492-49-1492. https://doi.org/10.5860/choice.49-1492

Silva Ortigoza, R., García Sánchez, J., Barrientos Sotelo, V., Molina Vilchis, M., Hernández Guzmán, V., & Silva Ortigoza, G. (2007). Una panorámica de los robots móviles. *Télématique*, *6*(3), 1-14.

Sredojev, B., Samardzija, D., & Posarac, D. (2015). WebRTC technology overview and signaling solution design and implementation. *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics,*

MIPRO 2015 - Proceedings, *May*, 1006-1009. https://doi.org/10.1109/MIPRO.2015.7160422

Stephan, J. (2012). *differential_drive - ROS Wiki*. http://wiki.ros.org/differential_drive

Suciu, G., Stefanescu, S., Beceanu, C., & Ceaparu, M. (2020). WebRTC role in real-time communication and video conferencing. *GIoTS 2020 - Global Internet of Things Summit, Proceedings*. https://doi.org/10.1109/GIOTS49054.2020.9119656

Talebpour, Z., Navarro, I., & Martinoli, A. (2016). On-board human-aware navigation for indoor resource-constrained robots: A case-study with the ranger. *2015 IEEE/SICE International Symposium on System Integration, SII 2015*, 63-68. https://doi.org/10.1109/SII.2015.7404955

Tasaki, R., Kitazaki, M., Miura, J., & Terashima, K. (2015). Prototype design of medical round supporting robot «Terapio». *Proceedings - IEEE International Conference on Robotics and Automation*, *2015-June*(June), 829-834. https://doi.org/10.1109/ICRA.2015.7139274

Tonin, L., Leeb, R., Tavella, M., Perdikis, S., & Mill, R. (2018). *The role of shared-control in BCI-based telepresence*. *1*, 2-5.

Trautman, P., & Krause, A. (2010). Unfreezing the robot: Navigation in dense, interacting crowds. *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, 797-803. https://doi.org/10.1109/IROS.2010.5654369

Truong, X. T., Yoong, V. N., & Ngo, T. D. (2017). Socially aware robot navigation system in human interactive environments. *Intelligent Service Robotics*, *10*(4), 287-295. https://doi.org/10.1007/s11370-017-0232-y

Tsai, T. C., Hsu, Y. L., Ma, A. I., King, T., & Wu, C. H. (2007). Developing a telepresence robot for interpersonal communication with the elderly in a home environment. *Telemedicine and e-Health*, *13*(4), 407-424. https://doi.org/10.1089/tmj.2006.0068

Tsui, K, & Desai, M. (2011). Exploring use cases for telepresence robots. *ACM/IEEE International Conference on Human-Robot Interaction*, *11*, 11-18.

Tsui, Katherine, Norton, A., Brooks, D., McCann, E., Medvedev, M., & Yanco, H. (2013). Design and development of two generations of semi-autonomous social telepresence robots. *IEEE Conference on Technologies for Practical Robot Applications, TePRA*, 0-5. https://doi.org/10.1109/TePRA.2013.6556360

Tully Foote, & Mike Purvis. (2010, octubre 7). *Standard Units of Measure and Coordinate Conventions*. https://www.ros.org/reps/rep-0103.html

Ulrich, K. T., & Eppinger, S. D. (2013). Diseño y desarrollo de productos. En *Diseño y desarrollo de productos* (5ta ed.). McGraw-Hill. https://doi.org/10.1017/CBO9781107415324.004

Vemula, A., Muelling, K., & Oh, J. (2017). Modeling cooperative navigation in dense human crowds. *Proceedings - IEEE International Conference on Robotics and Automation*, 1685-1692. https://doi.org/10.1109/ICRA.2017.7989199

Wang, C. M. (1988). Location estimation and uncertainty analysis for mobile robots. En IEEE (Ed.), *IEEE International Conference on Robotics and Automation* (1.ª ed., pp. 1231–1235). IEEE.

Welch, G., & Bishop, G. (2001). An introduction to the kalman filter. ACM Press. *SIGGRAPH*, *12*(1), 121.

Wu, C. H., Lai, C. C., Lo, H. J., & Wang, P. Sen. (2019). A comparative study on encoding methods of local binary patterns for image segmentation. *Smart Innovation, Systems and Technologies*, *128*, 277-283. https://doi.org/10.1007/978-3-030-04585-2_33

Wulfmeier, M., Rao, D., Wang, D. Z., Ondruska, P., & Posner, I. (2017). Large-scale cost function learning for path planning using deep inverse reinforcement learning. *The International Journal of Robotics Research*, *36*(10), 1073-1087.

https://doi.org/10.1177/0278364917722396

Yatim, N. M., & Buniyamin, N. (2015). Particle filter in simultaneous localization and mapping (Slam) using differential drive mobile robot. *Jurnal Teknologi*, *77*(20), 91-97. https://doi.org/10.11113/jt.v77.6557

Young, J. E., Kamiyama, Y., Reichenbach, J., Igarashi, T., & Sharlin, E. (2011). How to walk a robot: A dog-leash human-robot interface. *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, 376-382. https://doi.org/10.1109/ROMAN.2011.6005225

Yuan, B. (2017). *Neural Network Based Gesture Recognition Robot*.

Yulum, W., Charles, J., Michael, C., Marco, P., Kevin, H., Daniel, S., Cody, H., Withney, B., Fuji, L., Kelton, T., Rauhunt, E., & Cheuk, W. W. (2014). *INTERFACING WITH A MOBILE TELEPRESENCE ROBOT*.

Zanlungo, F., Yücel, Z., Ferreri, F., Even, J., Morales Saiki, L. Y., & Kanda, T. (2017). Social Group Motion in Robots. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10652 LNAI*, 474-484. https://doi.org/10.1007/978-3-319-70022-9_47

Zheng, K. (2017). ROS Navigation Tuning Guide. *arXiv*, *June 2017*.

Anexos

Anexo A: Diagrama de bloques general

Anexo B: Diagrama de flujo de la aplicación web

Anexo C: Diagrama de flujo de la aplicación móvil

Anexo D: Plano del entorno

Anexo E: Diagrama electrónico

Anexo F: Manual de usuario de la plataforma móvil