

Universidad Nacional de La Plata

Facultad de Ingeniería

2024



Trabajo de Desarrollo - Proyecto Andino

Introducción a la Robótica

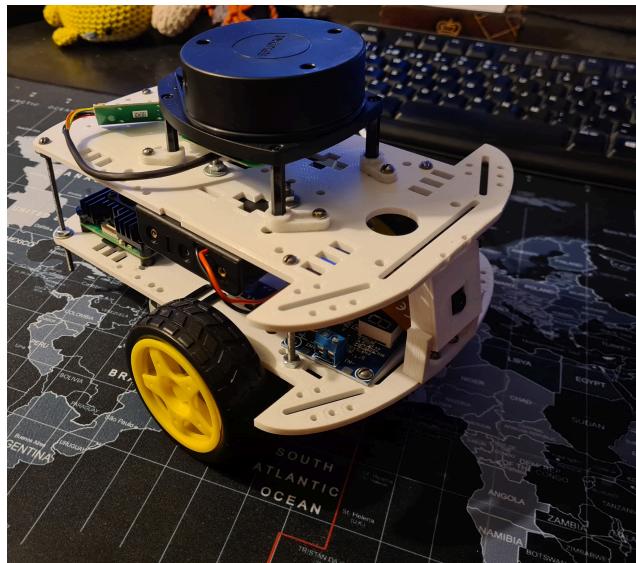
Calvo, Matias

Petrigh, Lautaro

Introducción

El proyecto andino consta de un robot autónomo diferencial de código abierto (github.com/Ekumen-OS/andino), desarrollado por la empresa Ekumen, el robot cuenta con un módulo de locomoción, dirigido por un Arduino Nano, el cual consiste de 2 ruedas traccionadas por motores con encoders, y una rueda libre para estabilidad, además se cuenta con un sensor IMU.

El segundo módulo es el de mapeo y localización, el cual consiste por una cámara V2 de raspberry pi y un sensor lidar. Este último módulo y la indicaciones enviadas al arduino nano son manejadas por el middleware ROS el cual corre en una Raspberry Pi.



Modificaciones de la Cátedra

Considerando que el proyecto fue desarrollado en el exterior, donde componentes electrónicos son más accesibles, la cátedra debió acudir a elementos sustitutos por razones de accesibilidad.

Los cambios más apreciables de la arquitectura fueron:

- Los motores no tenían encoders integrados, por lo que se instalaron aparte (Lo que el firmware del proyecto fue modificado).
- La alimentación a través de 3 baterías de 4.2V cada una.
- Raspberry pi 5, el proyecto planteaba el uso de la raspberry pi 4.

Estas modificaciones implican una reorganización en la disposición de algunos elementos que no afectan la funcionalidad del robot.

Modificaciones de Software: github.com/Petrigh/andino

Desafíos Encontrados

La mayor cantidad de problemas fueron encontrados a la hora de seguir la documentación del proyecto, y la alimentación y compatibilidad con la Raspberry Pi 5

Documentación de Andino

Algunas advertencias de cómo configurar el proyecto en caso de que deba ser levantado desde 0.

Versionado

El manejo de versiones del proyecto lo hacen a través de ramas de github, en lugar de utilizar releases.

Aun así, hasta Diciembre de 2024, la rama *jazzy* compatible con ROS Jazzy Jalisco tiene errores que fueron copiados desde la rama *humble*, **Recomendamos seguir las instrucciones de la rama jazzy con precaución, algunos links o comandos utilizan ROS Humble Hawksbill.**

Setup

En caso de que se configure el proyecto desde 0, recomendamos utilizar los pasos indicados en el directorio ([andino/andino.hardware](#)) de la rama deseada, tener en cuenta las recomendaciones de la sección anterior.

La razón de comenzar con ese directorio es porque especifica cómo lograr que los periféricos se autoconfiguren cada vez que se conectan, haciendo el flujo de trabajo menos tedioso.

Luego se puede seguir con el directorio deseado para comenzar el uso del proyecto físico ([/andino#usage](#)) o por simulación ([/andino#simulation](#)).

Uno de los cambios realizados fue la del uso de encoders con discos impresos en 3D, el equipo que implementó la solución tenemos el firmware del Arduino Nano modificado en ([Petrich/andino/andino.firmware](#))

Raspberry Pi 5

Esta SBC tiene una gran capacidad de procesamiento, pero todo poder conlleva una gran responsabilidad

Alimentación

Tiene picos de consumo altos, los cuales no son abastecidos por transformadores convencionales (Diciembre 2024). En especial si se conectan varios periféricos. Durante la implementación del proyecto no funcionó la alimentación a base de pilas, fuente de pc, transformador de 5V 5A, fuente de laboratorio.

Debimos acudir a una fuente de 100W con adaptador USB-C, esta fuente si bien indica un output de 5V 3A, los cuales no son utilizados constantemente (Suelen ser 5V 0.5A-2A), creemos que la capacidad de alimentar hasta 5A permitió otorgar alimentación durante picos de corriente.

Compatibilidad

Raspberry Pi 5 acepta como mínimo la versión de Ubuntu 24.04 LTS, la cual no es compatible con ROS Humble Hawksbill que requiere Ubuntu 22.04 LTS. El proyecto terminó siendo implementado conteniendo Ubuntu 22 dentro de Ubuntu 24 con la herramienta Docker. Ya que el proyecto con la rama *jazzy* no dio resultado.

El problema de dockerizar la solución es que no solo agrega una carga de procesamiento, sino que no se cuenta con un ambiente de escritorio para visualizar las simulaciones.

Modulo de Camara

No es un problema necesariamente de la Raspberry Pi 5, sino que es el manejo de periféricos de Ubuntu, este no es capaz de reconocer la cámara por más que se importen funcionalidades de Raspberry Pi OS. Mucha suerte si lo intenta.

Como levantar el proyecto

Firmware Nano

El firmware instalado en el Arduino Nano fue modificado para utilizar los encoders impresos por la cátedra. Este firmware está contemplado para ser utilizado con la rama *humble*, pruebas con la rama *jazzy* dieron resultados negativos, puede que la solución para levantar el proyecto con la rama *jazzy* sea factiblemente arreglar compatibilidades en el firmware del Arduino.

Acceso a la Raspberry

La SBC está configurada para conectarse a las redes privadas de los integrantes del curso, por lo que se deberá acceder al escritorio del dispositivo utilizando un monitor y teclado, **Usuario: andino; Contraseña: robotica2024**.

Se recomienda configurar la red wifi utilizando un celular como access point, para aportar movilidad y fácil acceso a la ip del dispositivo. Aun así, en caso de no disponer de esta tecnología, aplicaciones móviles como [Fing](#) o similares permiten escanear la red de forma rápida.

Una vez configurada la red, se puede acceder mediante SSH por la terminal de cualquier pc conectada a la misma red.

```
$ ssh andino@IP.DEL.ANDINO
```

Se pedirá la contraseña la cual es la indicada anteriormente.

Una vez dentro de la raspberry, se debe acceder al contenedor docker.

```
$ sudo docker start andino #Inicializa el contenedor  
$ sudo docker exec -it andino bash #Se accede por terminal
```

Se accede así al contenedor docker en donde se debe acceder al directorio del proyecto y levantar el robot

```
$ cd ~/andino_ws #puede que sea cd ~/ws  
$ source install/setup.bash
```

Con esto ya se puede proceder con el uso del robot.

Mejoras

Nuestro equipo de trabajo no fue capaz de levantar el proyecto con la rama *jazzy*, ni hacer uso de la cámara, ni la navegación con Nav2.

Actualizar el firmware del nano puede que sea clave para utilizar el proyecto con esta rama, se debe modificar ([/andino/andino_firmware](#)) para que utilice los encoders impresos, si se desea puede basarse el código con nuestro firmware:

github.com/Petrigh/andino/andino_firmware

La cámara hasta (Diciembre 2024) tiene problemas de compatibilidad con arquitecturas arm64, lo cual imposibilitó el uso de la misma. Si en un futuro se actualizan paquetes, puede que instalarlo resuelva el problema.

La navegación no fue utilizada porque la solución está dentro de un contenedor docker, pasar la solución sin contenerla con la rama *jazzy* podrá desbloquear esta funcionalidad.