1. **Pruebas de abril 22**

Asegurarse de ejecutar esta prueba en primer lugar para poder hacer instalaciones en PC2

Prueba de conexión a internet con wifi en PC2

Objetivo:

Verificar la conexión WiFi del G1 de Unitree, asegurando que el robot puede acceder a una red inalámbrica y establecer una conexión a Internet.

Requerimientos Previos:

Tener el robot G1 encendido y conectado vía Ethernet para acceso inicial.

Contar con credenciales de acceso a la red WiFi.

Acceso SSH al PC2 del G1.

1 Acceso al PC2 del G1

Desde el computador, conectarse al PC2 del robot mediante SSH:

ssh unitree@192.168.123.164

Ingresar la contraseña cuando se solicite. 123

2 Configuración del WiFi en Modo STA

Ejecutar los siguientes comandos para habilitar y configurar la conexión WiFi:

- Desbloquear interfaces de red (si están bloqueadas):

sudo rfkill unblock all

- Habilitar la interfaz WiFi:

sudo ifconfig wlan0 up

- Encender la radio WiFi:

sudo nmcli radio wifi on

Conectar a la red WiFi especificada:

sudo nmcli device wifi connect "NOMBRE_DE_RED" password "CONTRASEÑA"

Verificar que la conexión está activa:

nmcli connection show --active

Si la conexión se estableció correctamente, el comando nmcli connection show --active debería mostrar la red WiFi conectada.

3 Verificación de Conectividad a Internet

Probar acceso a Internet con ping:

ping -c 4 8.8.8.8

Si hay respuesta, la conexión a Internet está funcionando.

Si no hay respuesta, seguir los pasos de resolución de problemas.

Verificar la ruta de conexión:

ip route

Se espera ver una salida similar a:

default via 192.168.X.1 dev wlan0

Donde 192.168.X.1 representa la dirección del gateway de la red.

4 Resolución de Problemas (Troubleshooting)

- La red WiFi no se muestra o no se conecta

Intentar conectar a una red diferente, como un hotspot desde un celular, para descartar problemas de permisos en la red de UniAndes.

Revisar el estado del WiFi con:

rfkill list

Si aparece Soft blocked: yes, significa que el WiFi está bloqueado por software. Para desbloquearlo:

sudo rfkill unblock wifi

Luego, activar nuevamente el WiFi:

sudo nmcli radio wifi on

- No hay acceso a Internet a pesar de estar conectado a la red

Para comprobar conexión se puede usar:

```
sofia@sofia-HP-Pavilion-Gaming-Laptop-15-dk0xxx:~$ nmcli connection show
NAME                UUID                                  TYPE      DEVICE
Milu 2              b358beef-91ac-4bdd-8a08-328cc964ad5c  wifi      wlo1
docker0             aa4c006a-5a68-4553-9e38-7fe96795d370  bridge    docker0
B2_Robot            eb5a5bdb-9c09-4db5-ad73-ddef5ca96bb5  wifi      --
eduroam             09b93350-aa03-4276-a672-9c57f7bc80f6  wifi      --
iPhone Milagros     66300446-42ee-4302-8580-766eaa415552  wifi      --
iPhone SMCV         2b9ea5d0-de8f-4dbe-82d8-57637c97b6b4  wifi      --
MARINI-5G           f2dc87cb-ff8e-426f-b3ab-fcfa424d1f5b  wifi      --
Profile 1           d7f6b0f3-736e-4667-a770-75698d96820d  ethernet  --
RoboticLab50        c61dc52e-7708-40f2-b749-f6dfa01e91fa  wifi      --
Rolo                a8fdc3b2-8ec3-4861-9bb8-c34a6d59de13  wifi      --
Said02-5G           35e53a2c-ff1a-435f-b591-2042fb799e04  wifi      --
SEMILLERO_ROBOTICA  b5d130b8-3800-4a30-9e62-144a19bc444f  wifi      --
unjiuh              66b2eca1-720f-441d-b37b-72599b0769a8  wifi      --
WiFi-UAO            5d04805b-221c-4396-91e2-2295195f74cf  wifi      --
sofia@sofia-HP-Pavilion-Gaming-Laptop-15-dk0xxx:~$
```

Comprobar si hay un firewall bloqueando la conexión:

sudo ufw disable

Reiniciar el servicio de red:

sudo systemctl restart NetworkManager

Verificar si hay una ruta de Internet configurada:

ip route

Si la ruta default no aparece, agregarla manualmente con:

sudo ip route add default via 192.168.X.1 dev wlan0

(Reemplazar 192.168.X.1 con la dirección del gateway de la red conectada.)

Prueba 0: Instalación de SDK2 en Python en PC2

Objetivo

Asegurar que la SDK2 de Unitree en Python esté correctamente instalada en el PC2, permitiendo la ejecución de scripts de control del G1 de forma remota.

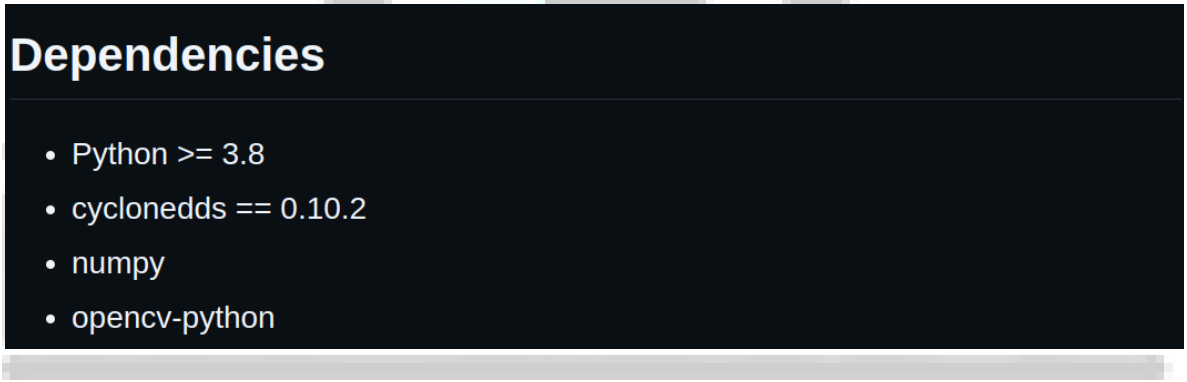
Requisitos previos

- Acceso a PC2 (con WiFi previamente configurado con el G1).
- Python ≥ 3.8 instalado.
- Git instalado.
- Acceso al repositorio de SDK2 de Unitree.

Pasos de instalación y verificación de SDK2 de python

Se va a interactuar con los diferentes códigos disponibles en python para el desarrollo, para eso primero se debe tener instalada la librería

Debemos tener estas dependencias:



- Se abre una terminal y se escribe:
  `pip install unitree_sdk2py`
- O desde la fuente:
  `cd ~`

  `sudo apt install python3-pip`

```
git clone https://github.com/unitreerobotics/unitree_sdk2_python.git

cd unitree_sdk2_python

pip3 install -e .
```

- Para validar la sdk se ejecuta en una terminal

```
cd
cd unitree_sdk2_pyhton/example/hello_world
python3 publisher.py
```

- Se abre otra terminal del contenedor y se ejecuta

```
cd
cd unitree_sdk2_pyhton/example/hello_world
python3 subscriber.py
```

- Se prueban los diferentes códigos es importante estar dentro de la carpeta donde se encuentran
- Para el ejemplo del tobillo es importante tener el robot colgado con la configuración de red realizada y en debug mode para prueba del movimiento de tobillos luego ejecutamos desde **~/unitree_sdk2_python/example/g1/low_level$**

```
python3 g1_low_level_example.py nombreInterfaz
```

- Para el ejemplo de la API de loco_client, el robot debe estar colgado pero en modo normal (en modo cero torque), luego desde **~/unitree_sdk2_python/example/g1/high_level** se ejecuta (PRECAUCIÓN EL CÓDIGO NO PRESENTA OPCIONES PARA DETENER DE EMERGENCIA)

```
python3 g1_loco_client_example.py nombreInterfaz
```

Van a aparecer las siguientes opciones:

```
option_list = [
    TestOption(name="damp", id=0),
    TestOption(name="stand_up", id=1),
    TestOption(name="sit", id=2),
    TestOption(name="move forward", id=3),
    TestOption(name="move lateral", id=4),
    TestOption(name="move rotate", id=5),
    TestOption(name="low stand", id=6),
    TestOption(name="high stand", id=7),
    TestOption(name="zero torque", id=8),
    TestOption(name="wave hand1", id=9), # wave hand without turning around
    TestOption(name="wave hand2", id=10), # wave hand and trun around
    TestOption(name="shake hand", id=11),
]
```

- Para el ejemplo de la API de loco_client con el ejemplo de wasd, el robot debe estar colgado pero en modo normal (en modo cero torque),

Prueba 1: Control Semiteledirigido del Brazo

Objetivo

Ajustar el uso de `arm_sdk` para controlar de forma manual las articulaciones del brazo del G1, permitiendo definir poses personalizadas para manipulación de objetos.

Requisitos previos

- Acceso al script de control por `arm_sdk` (`RUTINA_BRAZOS.py`).
- PC2 con SDK2 funcional (ver prueba 0). Opcional

Procedimiento

Ejecutar script

```
python3 RUTINA_BRAZOS.py <interfaz_red>
```

1. **Mover el brazo a diferentes poses** utilizando las pautas de etapas asignadas.
2. Calibrar las posiciones para el sostenimiento adecuado de la caja sin choques en el entorno (ir moviendo el robot con el control remoto)

Prueba 2: Registro de Odometría + Visualización de Trayectoria del G1

Objetivo

Evaluar la consistencia de la odometría del robot G1 al recorrer una trayectoria en el entorno físico. Se busca:

- Observar en consola los valores de posición y orientación en tiempo real.
- Verificar que los desplazamientos del robot se reflejan correctamente en la odometría.
- **Guardar manualmente puntos clave** de la trayectoria para una futura secuencia de navegación automática.
- Usar una imagen de referencia para contrastar con la trayectoria recorrida.

Requisitos previos

- Conexión activa al robot G1 mediante **Ethernet**.
- SDK2 de Unitree en Python correctamente instalada.
- Robot encendido en **modo normal y main operation control**.
- Interfaz `loco_client` funcional.
- Suscripción correcta al canal `SportModeState_`.

- Script funcional: `g1_odometry.py`.

- Librería `unitree_sdk2py` instalada y accesible desde entorno Python.

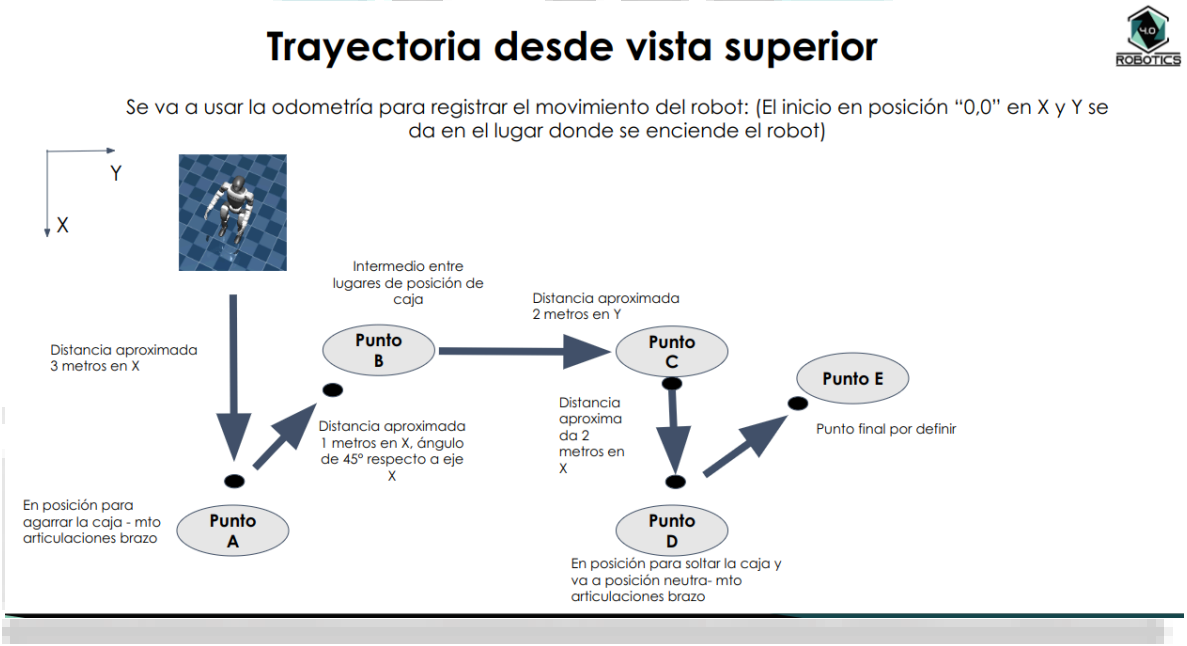
Procedimiento

1. **Preparar el entorno físico**:

    - Delimitar un área segura sin obstáculos.

    - Colocar **marcas visibles en el suelo** (punto de inicio + puntos de referencia).

    - Mantener una imagen impresa o digital de la **trayectoria esperada** (flechas, curva, esquinas, etc.). La versión inicial es:



Ejecutar el script:

```
python3 g1_odometry.py <interfaz_red>
```

- Reemplazar `<interfaz_red>` por la interfaz de red conectada al robot (ej. `enp0s31f6` o `eth0`).

2. **Mover el robot manualmente con el control** siguiendo la ruta física marcada.
3. **Observar en consola** la odometría cada 500 ciclos:
   - Posición actual: `x, y, z`
   - Orientación: `roll, pitch, yaw`
   - Velocidad lineal y angular
4. **Registrar manualmente** las coordenadas relevantes del trayecto (inicio, giros, destino) para su uso en navegación autónoma.
5. **Comparar la trayectoria real con la esperada** mediante la imagen de referencia.

Visualización de trayectoria (opcional)

Si se desea graficar la trayectoria en tiempo real o posterior:

- Modificar el script para almacenar `x_values`, `y_values`.

Usar `matplotlib` para graficar:

```
import matplotlib.pyplot as plt

plt.plot(x_values, y_values)

plt.xlabel("x [m]")

plt.ylabel("y [m]")

plt.title("Trayectoria registrada por odometría")

plt.grid(True)

plt.show()
```

Verificación de éxito

- Los valores impresos en consola muestran **coherencia** con los desplazamientos reales del robot.
- La trayectoria es **continua, sin saltos ni errores de signo**.
- Los **puntos clave quedan identificados** para su uso posterior en automatización.
- Se visualiza correctamente la trayectoria aproximada en gráfico (si aplica).

Troubleshooting

Problema	Posible causa	Acción recomendada
Valores de odometría en 0	`SportModeState_` no se actualiza	Verificar modo del robot y conexión por Ethernet
Datos congelados	`first_update` no se activa	Confirmar suscripción a `rt/lowstate`
Desfase en orientación (`yaw`)	Deriva natural por giro manual sin realineamiento	Tomar puntos de referencia física al girar
No imprime odometría	`counter_` puede estar muy alto para pruebas cortas	Reducir frecuencia de impresión (ej. cada 100)

Prueba 3: Navegación Autónoma por Odometría con Múltiples Objetivos

Objetivo:

Verificar que el robot G1 de Unitree pueda desplazarse de manera autónoma hacia una secuencia de posiciones predeterminadas utilizando exclusivamente su odometría interna y el script `g1_autonomousV1.py`.

Procedimiento:

1. **Conexión y Verificación Inicial:**
    - Asegúrate de tener conexión Ethernet estable entre el PC y el robot G1.

- Enciende el robot en modo normal (R1 + X).
- Verifica que el entorno Python esté correctamente configurado con la SDK2.

2. **Ejecución del Script:**
   - Abre una terminal en el entorno configurado y navega hasta el directorio del script.

Ejecuta:

```
python3 g1_autonomousV1.py
```

- Ingresa la interfaz de red correspondiente (ej. `eth0`).
- Indica **"no"** cuando se pregunte si deseas cargar puntos desde archivo.

Cuando se soliciten los objetivos manuales, ingresa los siguientes puntos utilizados en la Prueba 2:
Ejemplo:

```
0.5 0.0 0.0
1.0 0.5 1.57
1.0 1.0 3.14
fin
```

3. **Observación del Comportamiento:**
   - El robot debe:
     - Realizar una rotación previa para orientar su cuerpo hacia el objetivo.

- Avanzar hacia el punto con control proporcional adaptativo.
- Realinear su orientación una vez llegado al objetivo.

- Repetir esta secuencia para cada punto ingresado.

4. **Finalización:**
   - Confirmar que el robot se detenga automáticamente al finalizar todos los objetivos o si se interrumpe manualmente.
   - Evaluar si llegó a los puntos dentro del margen aceptable.

Criterios de Éxito:

Criterio	Umbral
Tolerancia de posición final	≤ 10 cm
Tolerancia de orientación final	≤ 0.3 rad
Tiempo máximo por objetivo	30 segundos
Respuesta ante interrupción	Robot se detiene de forma segura
Estabilidad del movimiento	No se presentan oscilaciones o caídas
Retroceso preventivo	Ejecutado si se detecta orientación opuesta

Notas Técnicas:

- Se utiliza control proporcional adaptativo con reducción de ganancia en errores pequeños (tipo "P con freno").
- El movimiento lateral es limitado (máximo ±0.2 m/s) y el retroceso permitido es leve (-0.15 m/s máx).
- El sistema normaliza automáticamente los valores de yaw para evitar discontinuidades.
- El script permite cargar objetivos por archivo `.txt` en pruebas futuras.