

Objetivo

- La documentación va orientada a guiar paso a paso, para personas que no hallan tenido contacto con entornos de simulación sin uso de hardware real.

¿Qué son?

¿Qué es Isaac Sim?

NVIDIA Isaac Sim es un **simulador de robótica 3D** desarrollado por **NVIDIA**, usado para **diseñar, probar y entrenar robots en un entorno virtual realista**.

¿Qué es Isaac Lab?

Isaac Lab (NVIDIA Isaac Lab) es un **framework de investigación y desarrollo en robótica**, creado por **NVIDIA**, que se usa para **entrenar y evaluar robots**, especialmente con **aprendizaje por refuerzo (Reinforcement Learning)**, sobre **Isaac Sim**.

Entorno	¿Qué hace?
Isaac Sim	Mundo simulado
Isaac Lab	Laboratorio para el entrenamiento del robot dentro de Isaac Lab

Requisitos mínimos

- **OS:** Ubuntu 22.04 (Linux x64) o Windows 11 (x64)
- **RAM:** 32 GB mínimo o más.
- **GPU VRAM:** 16 GB o más.

Hardware/Software actual

- Se darán especificaciones actuales:
- **OS:** *Ubuntu 24.04.3 LTS x86_64*

- **GPU:** Geforce RTX 5090 de 32 GBs de Ram
- **RAM:** 62 GBs

Adjunto imagen que demuestra las especificaciones aneterioes

```
      .-/+oossssoo+/-.
      `:+ssssssssssssssss+:`
      -+ssssssssssssssssyyssss+-
      .ossssssssssssssssdMMMMyssso.
      /ssssssssssshdmmNNmmyNMMMMhssssss/
      +ssssssssshmydMMMMMMMMddddyssssss+
      /ssssssssshNMMMyhhyyyyhmNMMNhssssss/
      .ssssssssdMMMNhssssssssshNMMMdssssss.
      +ssssshhhyNMMMyssssssssssyNMMMyssssss+
      ossyNMMMNyMMhssssssssssshmmhssssssso
      ossyNMMMNyMMhssssssssssshmmhssssssso
      +ssssshhhyNMMMyssssssssssyNMMMyssssss+
      .ssssssssdMMMNhssssssssshNMMMdssssss.
      /ssssssssshNMMMyhhyyyyhdNMMNhssssss/
      +ssssssssdmydMMMMMMMMddddyssssss+
      /ssssssssshdmmNNNmyNMMMMhssssss/
      .ossssssssssssssssdMMMMyssso.
      -+ssssssssssssssssyyssss+-
      `:+ssssssssssssssss+:`
      .-/+oossssoo+/-.

iudc@iudc
-----
OS: Ubuntu 24.04.3 LTS x86_64
Host: X870 GAMING WIFI6 -CF-WCP
Kernel: 6.14.0-36-generic
Uptime: 26 mins
Packages: 2676 (dpkg), 26 (snap)
Shell: bash 5.2.21
Resolution: 1920x1080, 1920x1080
DE: GNOME 46.0
WM: Mutter
WM Theme: Adwaita
Theme: Yaru-dark [GTK2/3]
Icons: Yaru-dark [GTK2/3]
Terminal: gnome-terminal
CPU: AMD Ryzen 9 9950X3D (32) @ 5.75
GPU: AMD ATI 74:00.0 Device 13c0
GPU: NVIDIA 01:00.0 NVIDIA Corporati
Memory: 2938MiB / 61841MiB
```



```
-----+-----+-----+
| NVIDIA-SMI 580.95.05                Driver Version: 580.95.05          CUDA Version: 13.0     |
+-----+-----+-----+
| GPU   Name                               Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf              Pwr:Usage/Cap |      Memory-Usage | GPU-Util  Compute M. |
|                                           MIG M.         |
+-----+-----+-----+
|  0  NVIDIA GeForce RTX 5090                Off | 00000000:01:00.0 On  |          N/A         |
|  0%   47C   P8              42W / 575W | 504MiB / 32607MiB |      1%      Default |
|                                           N/A              |
+-----+-----+-----+

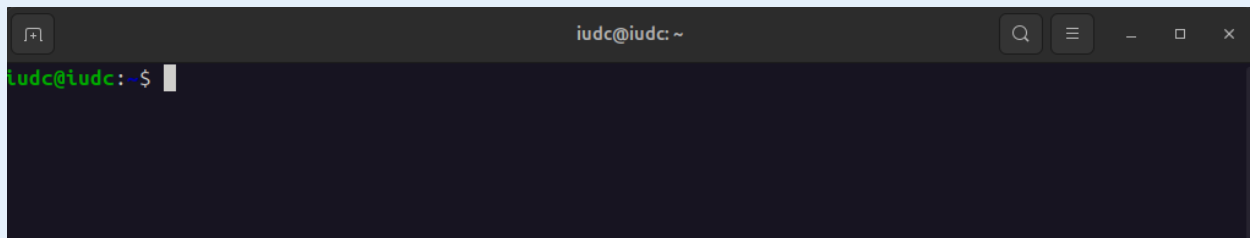
Processes:
+-----+-----+-----+
| GPU  GI  CI           PID  Type  Process name                        GPU Memory |
|   ID  ID  ID                   |                    Usage      |
+-----+-----+-----+
|  0   N/A N/A         2796   G   /usr/lib/xorg/Xorg                  123MiB |
|  0   N/A N/A         3333   G   /usr/bin/gnome-shell                48MiB |
|  0   N/A N/A         3483   G   ...bin/snapd-desktop-integration   12MiB |
|  0   N/A N/A         7066   G   /snap/obsidian/53/obsidian          37MiB |
|  0   N/A N/A         7207   G   .../7477/usr/lib/firefox/firefox    154MiB |
|  0   N/A N/A         9541   G   /usr/bin/gnome-control-center       18MiB |
+-----+-----+-----+
```

Instalación

Para la instalación se guiara paso a paso, instalación de entorno comando de ejecución, etc...

Instalación de Conda

En una instancia de terminal, **CTRL + ALT + T** :

A terminal window with a dark background. The title bar shows 'iudc@iudc: ~'. The prompt is 'iudc@iudc:~\$' followed by a cursor.

Colocaremos el siguiente comando:

```
mkdir -p ~/miniconda3
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-
x86_64.sh -O ~/miniconda3/miniconda.sh
bash ~/miniconda3/miniconda.sh -b -u -p ~/miniconda3
rm ~/miniconda3/miniconda.sh
```

```
iudc@iudc-B650-EAGLE-AX:~$ mkdir -p ~/miniconda3
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh -O ~/miniconda3/miniconda.sh
bash ~/miniconda3/miniconda.sh -b -u -p ~/miniconda3
rm ~/miniconda3/miniconda.sh
--2025-12-15 08:38:55-- https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
Resolviendo repo.anaconda.com (repo.anaconda.com)... 104.16.191.158, 104.16.32.241, 2606:4700::6810:20f1, ...
Conectando con repo.anaconda.com (repo.anaconda.com)[104.16.191.158]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 157891003 (151M) [application/octet-stream]
Guardando como: '/home/iudc/miniconda3/miniconda.sh'

/home/iudc/miniconda3/miniconda.sh 100%[=====]

2025-12-15 08:39:09 (11,0 MB/s) - '/home/iudc/miniconda3/miniconda.sh' guardado [157891003/157891003]

PREFIX=/home/iudc/miniconda3
Unpacking bootstrapper...
Unpacking payload...

Installing base environment...

Preparing transaction: ...working... done
Executing transaction: ...working... done
installation finished.
```

Activamos **Conda** en la terminal después de la instalación:

```
source ~/miniconda3/bin/activate
```

```
iudc@iudc-B650-EAGLE-AX:~$ source ~/miniconda3/bin/activate
```

Ahora vamos a activar globalmente **Conda**

```
conda init --all
```

```
(base) iudc@iudc-B650-EAGLE-AX:~$ conda init --all
no change      /home/iudc/miniconda3/condabin/conda
no change      /home/iudc/miniconda3/bin/conda
no change      /home/iudc/miniconda3/bin/conda-env
no change      /home/iudc/miniconda3/bin/activate
no change      /home/iudc/miniconda3/bin/deactivate
no change      /home/iudc/miniconda3/etc/profile.d/conda.sh
no change      /home/iudc/miniconda3/etc/fish/conf.d/conda.fish
no change      /home/iudc/miniconda3/shell/condabin/Conda.psm1
no change      /home/iudc/miniconda3/shell/condabin/conda-hook.ps1
no change      /home/iudc/miniconda3/lib/python3.13/site-packages/xontrib/conda.xsh
no change      /home/iudc/miniconda3/etc/profile.d/conda.csh
modified       /home/iudc/.bashrc
modified       /home/iudc/.zshrc
modified       /home/iudc/.config/fish/config.fish
modified       /home/iudc/.xonshrc
modified       /home/iudc/.tcshrc

==> For changes to take effect, close and re-open your current shell. <==

(base) iudc@iudc-B650-EAGLE-AX:~$
```

Esto significa que cada vez que habremos una nueva terminal, nos aparecera lo siguiente:

```
(base) iudc@iudc:~$
```

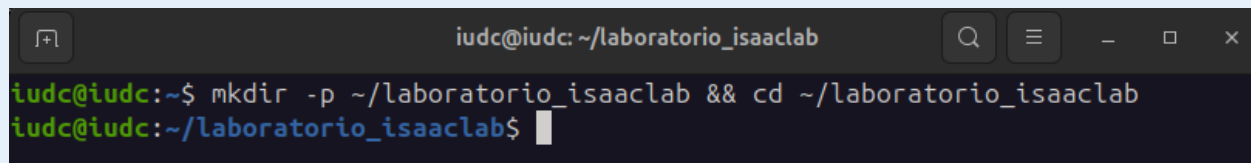
Instalación de Isaac Lab

Ahora para la instalación de `Isaac Lab`, nos podemos basar a partir de la documentación [Link Isaac Lab](#)

Ahora una instalación guiada

- Creamos una carpeta donde con clonaremos el

```
mkdir -p ~/laboratorio_isaaclab && cd ~/laboratorio_isaaclab
```



```
iudc@iudc: ~/laboratorio_isaaclab
iudc@iudc:~$ mkdir -p ~/laboratorio_isaaclab && cd ~/laboratorio_isaaclab
iudc@iudc:~/laboratorio_isaaclab$
```

- Clonamos el entorno de `Isaac Lab`

```
git clone https://github.com/isaac-sim/IsaacLab.git
```

```
Tilix: iudc@iudc: ~/laboratorio_isaacclab
1: iudc@iudc: ~/laboratorio_isaacclab
(base) iudc@iudc:~/laboratorio_isaacclab$ git clone https://github.com/isaac-sim/IsaacLab.git
Clonando en 'IsaacLab'...
remote: Enumerating objects: 883170, done.
remote: Counting objects: 100% (710/710), done.
remote: Compressing objects: 100% (533/533), done.
remote: Total 883170 (delta 402), reused 177 (delta 177), pack-reused 882460 (from 3)
Recibiendo objetos: 100% (883170/883170), 692.51 MiB | 55.18 MiB/s, listo.
Resolviendo deltas: 100% (678241/678241), listo.
(base) iudc@iudc:~/laboratorio_isaacclab$
```

- Ingresamos al entorno clonado

```
cd IsaacLab/
```

```
Tilix: iudc@iudc: ~/laboratorio_isaacclab/IsaacLab
1: iudc@iudc: ~/laboratorio_isaacclab/IsaacLab
(base) iudc@iudc:~/laboratorio_isaacclab$ ls
IsaacLab
(base) iudc@iudc:~/laboratorio_isaacclab$ cd IsaacLab/
(base) iudc@iudc:~/laboratorio_isaacclab/IsaacLab$
```

- Creamos el entorno con Conda

```
conda create -n isaacclab python=3.11
conda activate isaacclab
```

```
Tilix: iudc@iudc: ~/laboratorio_isaacclab/IsaacLab
1: iudc@iudc: ~/laboratorio_isaacclab/IsaacLab
(base) iudc@iudc:~/laboratorio_isaacclab/IsaacLab$ conda create -n isaacclab python=3.11
conda activate isaacclab
Retrieving notices: done
WARNING: A conda environment already exists at '/home/iudc/miniconda3/envs/isaacclab'

Remove existing environment?
This will remove ALL directories contained within this specified prefix directory, including
any other conda environments.

(y/[n])?
```

- Confirmamos con `y` para continuar

```
readline      pkgs/main/linux-64::readline-8.3-hc2a1206_0
setuptools    pkgs/main/linux-64::setuptools-80.9.0-py311h06a4308_0
sqlite        pkgs/main/linux-64::sqlite-3.51.0-h2a70700_0
tk            pkgs/main/linux-64::tk-8.6.15-h54e0aa7_0
tzdata        pkgs/main/noarch::tzdata-2025b-h04d1e81_0
wheel         pkgs/main/linux-64::wheel-0.45.1-py311h06a4308_0
xorg-libx11   pkgs/main/linux-64::xorg-libx11-1.8.12-h9b100fa_1
xorg-libxau   pkgs/main/linux-64::xorg-libxau-1.0.12-h9b100fa_0
xorg-libxdmcp pkgs/main/linux-64::xorg-libxdmcp-1.1.5-h9b100fa_0
xorg-xorgproto pkgs/main/linux-64::xorg-xorgproto-2024.1-h5eee18b_1
xz            pkgs/main/linux-64::xz-5.6.4-h5eee18b_1
zlib          pkgs/main/linux-64::zlib-1.3.1-hb25bd0a_0

Proceed ([y]/n)? |
```

- Volvemos a confirmar con `y` , de esa manera finalizara la creación del entorno y automaticamente nos movera al entorno recién montado (`isaac lab`)

```
Downloading and Extracting Packages:

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#   $ conda activate isaac lab
#
# To deactivate an active environment, use
#
#   $ conda deactivate

(isaac lab) iudc@iudc:~/laboratorio_isaac lab/Isaac Lab$ |
```

- Ahora instalamos los paquetes de `Isaac Sim` y esperamos a que se instale la dependencia

```
pip install "isaacsim[all,extscache]==5.1.0" --extra-index-url
https://pypi.nvidia.com
```

- Ahora vamos a añadir las dependencias más importantes, esto es importante, ya que la siguiente dependencia va de la mano con la **GPU** , que es lo mas importante, la versión del `#CUDA` , como podemos conocer la versión del `#CUDA` , ingresando lo siguiente en la terminal

```
nvidia-smi
```

```
1: iudc@iudc: ~/laboratorio_isaacLab/IsaacLab
(isaacLab) iudc@iudc:~/laboratorio_isaacLab/IsaacLab$ nvidia-smi
Mon Dec 15 11:59:40 2025

+-----+
| NVIDIA-SMI 580.95.05                  Driver Version: 580.95.05          CUDA Version: 13.0     |
+-----+-----+
| GPU   Name                               Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf              Pwr:Usage/Cap |      Memory-Usage | GPU-Util  Compute M. |
|                               |                      | MIG M. |
+-----+-----+
|  0  NVIDIA GeForce RTX 5090               Off | 00000000:01:00.0  On |          N/A |
|  0%   44C    P8              35W / 575W | 288MiB / 32607MiB |    0%      Default |
|                               |                      | N/A |
+-----+-----+

+-----+
| Processes: |
| GPU   GI   CI          PID    Type   Process name                      GPU Memory |
|      ID   ID              |              |           | Usage |
+-----+-----+
|  0   N/A  N/A         2751     G   /usr/lib/xorg/Xorg                  119MiB |
|  0   N/A  N/A         3290     G   /usr/bin/gnome-shell                 32MiB |
|  0   N/A  N/A         3957     G   ...exec/xdg-desktop-portal-gnome     8MiB |
|  0   N/A  N/A         4581     G   /snap/obsidian/53/obsidian           34MiB |
|  0   N/A  N/A         4704     G   .../7477/usr/lib/firefox/firefox     19MiB |
+-----+-----+

(isaacLab) iudc@iudc:~/laboratorio_isaacLab/IsaacLab$ |

+-----+
| CUDA Version: 13.0 |
+-----+
```

- Con base a la verificación del `#CUDA` modificaremos el siguiente comando de instalacion para la `#Dependencia` , modificando la parte `/cu128` a `/cu130`

```
pip install -U torch==2.9.0 torchvision==0.24.0 --index-url
https://download.pytorch.org/whl/cu130
```

- Para versiones menores de `#CUDA`

```
pip install -U torch==2.7.0 torchvision==0.22.0 --index-url
https://download.pytorch.org/whl/cu128
```

- Ahora vamos a verificar la instalación, ingresamos en la terminal el siguiente comando:

```
isaacsim isaacsim.exp.full.kit
```

⚠ Atención

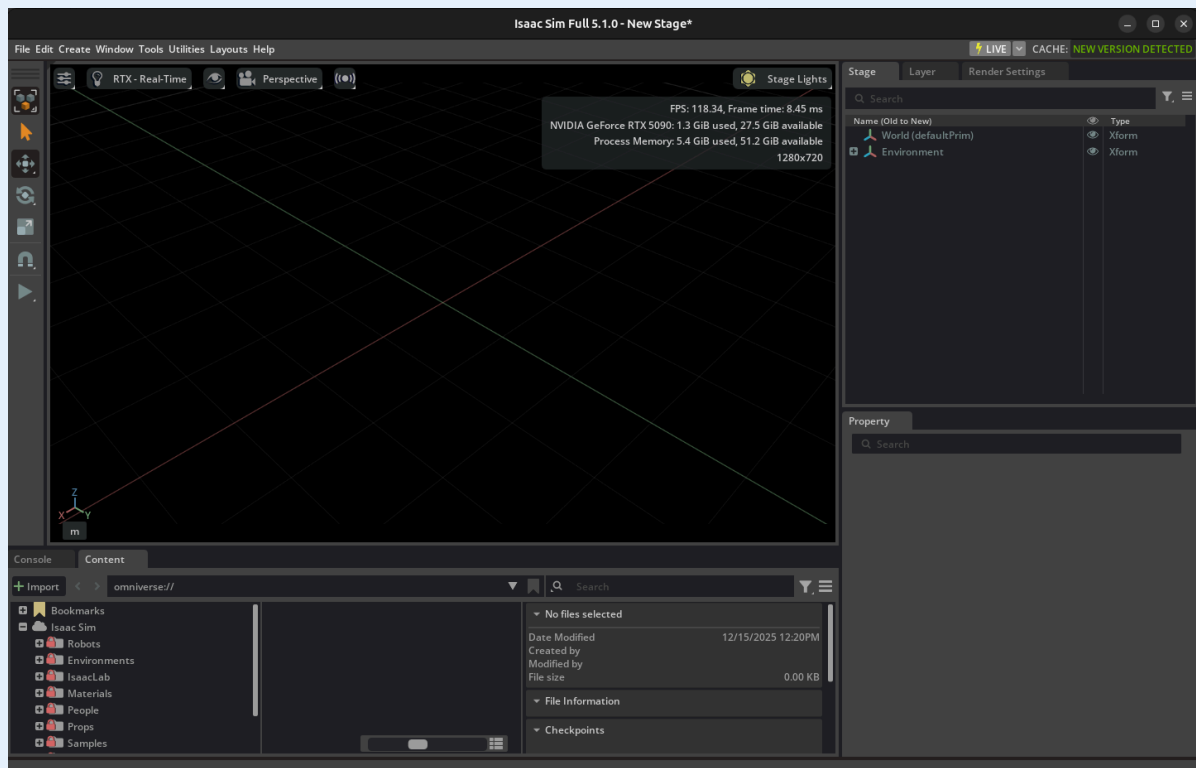
Nos pedira una confirmación de intalación de un kit, el cuál confirmaremos ingresando `yes`


```
Tilix: iudc@iudc: ~/laboratorio_isaacLab/IsaacLab
1:iudc@iudc: ~/laboratorio_isaacLab/IsaacLab
(isaacLab) iudc@iudc:~/laboratorio_isaacLab/IsaacLab$ isaacsim

By installing or using Omniverse Kit, I agree to the terms of NVIDIA OMNIVERSE LICENSE AGREEMENT (EULA)
in https://docs.omniverse.nvidia.com/platform/latest/common/NVIDIA_Omniverse_License_Agreement.html

Do you accept the EULA? (Yes/No): yes|
```

- Esperamos, normalmente nos estara pidiendo que forcemos la salida ya que es muy pesado el software, aun asi teniendo lo minimo una **GPU 5090**, daremos continuamente a `Esperar/Wait` , hasta el punto de que no nos pida *forzar la salida*.



Instalación de las herramientas de `#Isaac-Lab`

- Instalación de dependencias usando `apt`

```
sudo apt install cmake build-essential
```

- Continuando en la terminal, ubicado en `~/laboratorio_isaacLab/IsaacLab` con el entorno de `#Conda` activado:

```
1:iudc@iudc: ~/laboratorio_isaacLab/IsaacLab
(isaacLab) iudc@iudc:~/laboratorio_isaacLab/IsaacLab$ |
```


- Ahora vamos a correr la instalación con el siguiente comando y esperamos a que finalice, puede tomar:

```
./isaaclab.sh -i
```

Nos instalara las extensiones de `#Isaac-Lab` y para la estructura de entrenamiento por refuerzo.

Con esto finalizamos la instalación de `#Isaac-Lab`

Comando de ejecución

- En esta sección se realizara demostración de los comandos que se puede usar para, el entorno, tanto entrenamiento y visualización.

Entornos

- El siguiente comando nos permitirá visualizar los entornos activos para su ejecución:

```
./isaaclab.sh -p scripts/environments/list_envs.py
```

```

+-----+
| 1 | Isaac-Repose-Cube-Allegro-Direct-v0 | isaacclab_tasks.direct.allegro_hand.allegro_hand_env_cfg:AllegroHandEnvCfg |
| 2 | Isaac-Ant-Direct-v0 | isaacclab_tasks.direct.ant.ant_env:AntEnvCfg |
| 3 | Isaac-Velocity-Flat-Anymal-C-Direct-v0 | isaacclab_tasks.direct.anymal_c.anymal_c_env_cfg:AnymalCFlatEnvCfg |
| 4 | Isaac-Velocity-Rough-Anymal-C-Direct-v0 | isaacclab_tasks.direct.anymal_c.anymal_c_env_cfg:AnymalCRoughEnvCfg |
| 5 | Isaac-AutoMate-Assembly-Direct-v0 | isaacclab_tasks.direct.automate.assembly_env:AssemblyEnvCfg |
| 6 | Isaac-AutoMate-Disassembly-Direct-v0 | isaacclab_tasks.direct.automate.disassembly_env:DisassemblyEnvCfg |
| 7 | Isaac-Cart-Double-Pendulum-Direct-v0 | isaacclab_tasks.direct.cart_double_pendulum.cart_double_pendulum_env:CartDoublePe |
| 8 | Isaac-Cartpole-Direct-v0 | isaacclab_tasks.direct.cartpole.cartpole_env:CartpoleEnvCfg |
| 9 | Isaac-Cartpole-RGB-Camera-Direct-v0 | isaacclab_tasks.direct.cartpole.cartpole_camera_env:CartpoleRGBCameraEnvCfg |
| 10 | Isaac-Cartpole-Depth-Camera-Direct-v0 | isaacclab_tasks.direct.cartpole.cartpole_camera_env:CartpoleDepthCameraEnvCfg |
| 11 | Isaac-Cartpole-Showcase-Box-Box-Direct-v0 | isaacclab_tasks.direct.cartpole_showcase.cartpole.cartpole_env_cfg:BoxBoxEnvCfg |
| 12 | Isaac-Cartpole-Showcase-Box-Discrete-Direct-v0 | isaacclab_tasks.direct.cartpole_showcase.cartpole.cartpole_env_cfg:BoxDiscreteEnv |
| 13 | Isaac-Cartpole-Showcase-Box-MultiDiscrete-Direct-v0 | isaacclab_tasks.direct.cartpole_showcase.cartpole.cartpole_env_cfg:BoxMultiDiscre |
| 14 | Isaac-Cartpole-Showcase-Discrete-Box-Direct-v0 | isaacclab_tasks.direct.cartpole_showcase.cartpole.cartpole_env_cfg:DiscreteBoxEnv |
| 15 | Isaac-Cartpole-Showcase-Discrete-Discrete-Direct-v0 | isaacclab_tasks.direct.cartpole_showcase.cartpole.cartpole_env_cfg:DiscreteDiscre |
| 16 | Isaac-Cartpole-Showcase-Discrete-MultiDiscrete-Direct-v0 | isaacclab_tasks.direct.cartpole_showcase.cartpole.cartpole_env_cfg:DiscreteMultiD |
| 17 | Isaac-Cartpole-Showcase-MultiDiscrete-Box-Direct-v0 | isaacclab_tasks.direct.cartpole_showcase.cartpole.cartpole_env_cfg:MultiDiscreteB |
| 18 | Isaac-Cartpole-Showcase-MultiDiscrete-Discrete-Direct-v0 | isaacclab_tasks.direct.cartpole_showcase.cartpole.cartpole_env_cfg:MultiDiscreteD |
| 19 | Isaac-Cartpole-Showcase-MultiDiscrete-MultiDiscrete-Direct-v0 |

```

Son 152 entornos activos en total.

```

| 152 | Isaac-Navigation-Flat-Anymal-C-Play-v0 | isaacclab_tasks.manager_based.navigation.config.anymal_c.navigation_env_cfg:NavigationE
+-----+

```

⚠ Importante

Al visualizar los entornos, podemos ver que para cada uno, hay dos entornos, uno para entrenar y otro para visualizar, para proximos ejemplos nos enfocaremos en el del **Unitree-G1**

```

| 86 | Isaac-Velocity-Rough-G1-v0 | isaacclab_tasks.manager_based.locomotion.velocity.config.g1.rough_env_cfg:G1RoughEnvCfg |
| 87 | Isaac-Velocity-Rough-G1-Play-v0 | isaacclab_tasks.manager_based.locomotion.velocity.config.g1.rough_env_cfg:G1RoughEnvCfg |
| 88 | Isaac-Velocity-Flat-G1-v0 | isaacclab_tasks.manager_based.locomotion.velocity.config.g1.flat_env_cfg:G1FlatEnvCfg |
| 89 | Isaac-Velocity-Flat-G1-Play-v0 | isaacclab_tasks.manager_based.locomotion.velocity.config.g1.flat_env_cfg:G1FlatEnvCfg |

```

Ejecución de entrenamiento

- Para la ejecución necesitamos seleccionar un entorno de la anterior imagen, `Flat` que es terreno plano, y `Rough` es terreno random.
- Comando de ejecución para el entrenamiento:

```
./isaacclab.sh -p scripts/reinforcement_learning/rsl_rl/train.py --  
task Isaac-Velocity-Flat-G1-v0 --headless
```

Importante

El `flag/bandera` que es `--headless`, nos permite entrenar sin visualización, solo ver en la terminal iteraciones

```
#####  
Learning iteration 1/1500  
  
Computation: 144353 steps/s (collection: 0.644s, learning 0.037s)  
Mean action noise std: 1.00  
Mean value_function loss: 3.6480  
Mean surrogate loss: -0.0029  
Mean entropy loss: 52.5727  
Mean reward: -6.27  
Mean episode length: 47.02  
Episode_Reward/track_lin_vel_xy_exp: 0.0063  
Episode_Reward/track_ang_vel_z_exp: 0.0023  
Episode_Reward/lin_vel_z_l2: -0.0022  
Episode_Reward/ang_vel_xy_l2: -0.0543  
Episode_Reward/dof_torques_l2: -0.0022  
Episode_Reward/dof_acc_l2: -0.0049  
Episode_Reward/action_rate_l2: -0.0138  
Episode_Reward/feet_air_time: 0.0002  
Episode_Reward/flat_orientation_l2: -0.0129  
Episode_Reward/dof_pos_limits: -0.0003  
Episode_Reward/termination_penalty: -0.1956  
Episode_Reward/feet_slide: -0.0040  
Episode_Reward/joint_deviation_hip: -0.0026  
Episode_Reward/joint_deviation_arms: -0.0029  
Episode_Reward/joint_deviation_fingers: -0.0018  
Episode_Reward/joint_deviation_torso: -0.0008  
Metrics/base_velocity/error_vel_xy: 0.0859  
Metrics/base_velocity/error_vel_yaw: 0.4277  
Episode_Termination/time_out: 0.0270  
Episode_Termination/base_contact: 0.4604  
-----  
Total timesteps: 196608  
Iteration time: 0.68s  
Time elapsed: 00:00:02  
ETA: 00:29:23
```

¿Porqué?

- Al contar con el hardware mínimo no podemos poder visualizar el entorno con la funcionalidad completa, entonces solo podemos esperar a que finalice el entrenamiento. Al usar el `#Hardware` mínimo, el porcentaje de uso de la *GPU* es mayor al `75%`, se sobre calienta.

0	NVIDIA	GeForce RTX 5090	Off	00000000:01:00.0	On	N/A
30%	58C	P1	220W / 575W	5843MiB / 32607MiB	79%	Default
						N/A
Processes:						
GPU	GI ID	CI ID	PID	Type	Process name	GPU Memory Usage
0	N/A	N/A	2751	G	/usr/lib/xorg/Xorg	246MiB
0	N/A	N/A	3290	G	/usr/bin/gnome-shell	43MiB
0	N/A	N/A	3957	G	...exec/xdg-desktop-portal-gnome	8MiB
0	N/A	N/A	4581	G	/snap/obsidian/53/obsidian	38MiB
0	N/A	N/A	4704	G	.../7477/usr/lib/firefox/firefox	367MiB
0	N/A	N/A	8275	G	/proc/self/exe	55MiB
0	N/A	N/A	16383	G	/usr/bin/gnome-text-editor	27MiB
0	N/A	N/A	19621	C+G	...nda3/envs/isaacrab/bin/python	4920MiB

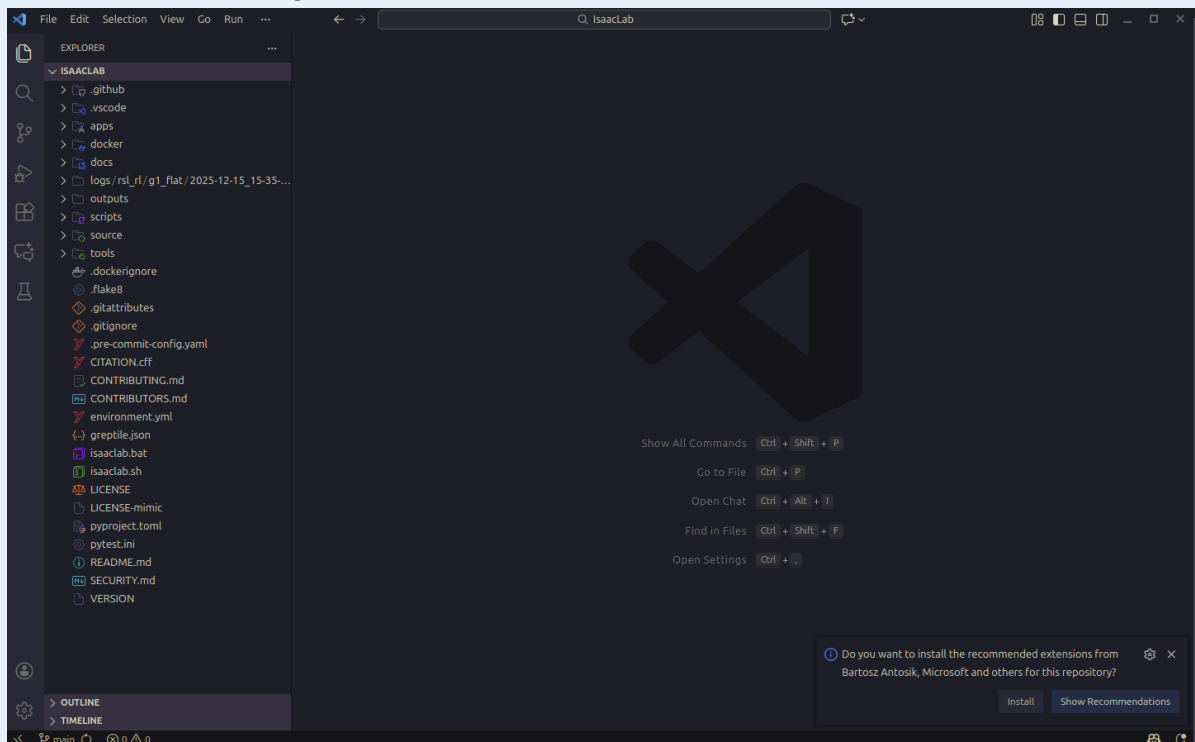
Visualización del entrenamiento

Al ya finalizar el entrenamiento se nos generaran archivos donde estaran puntos de guardado, parametros del entorno usado, para ello usar **VSCode**

- Si ya cuentas con **VSCode** , usa el siguiente comando en la siguiente ubicación **~/laboratorio_isaacrab/IsaacLab**

```
2: iudc@iudc: ~/laboratorio_isaacrab/IsaacLab
(isaacrab) iudc@iudc: ~/laboratorio_isaacrab/IsaacLab$ code .|
```

- Visualizaremos lo siguiente



- Al ejecutar el anterior entrenamiento, se nos generaron las siguientes carpetas

```
> logs
> outputs
```

- Dentro de la carpeta `logs`, contamos con los siguientes directorios y archivos

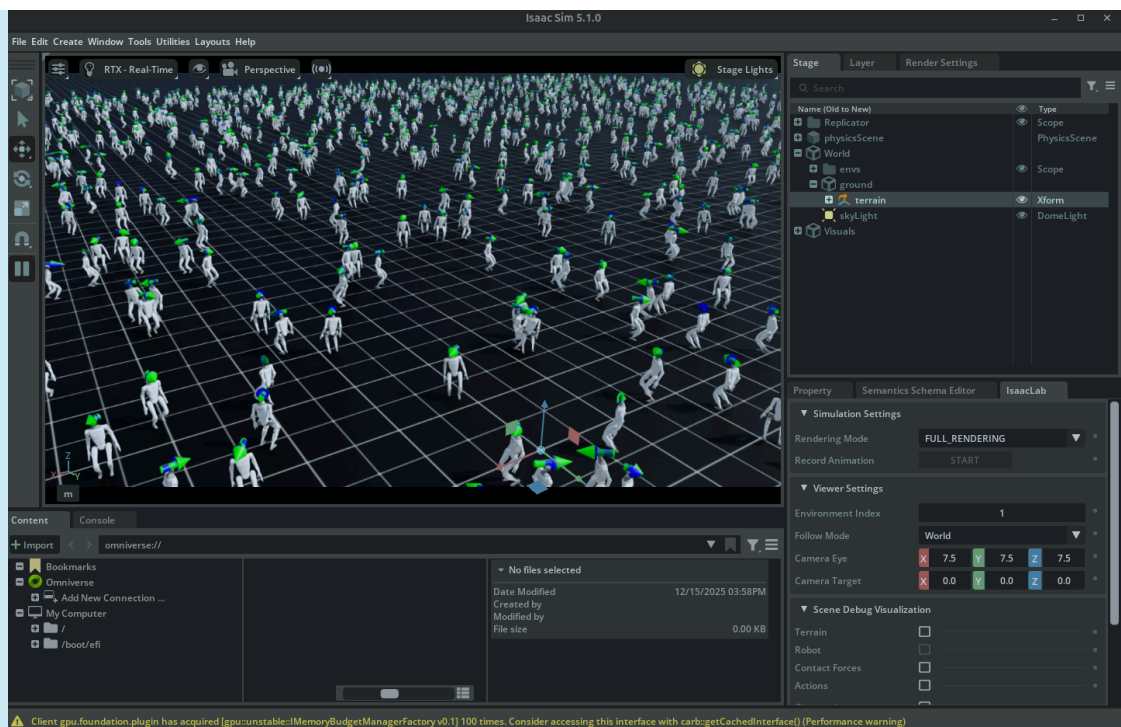
```
logs/rsl_rl/g1_flat/2025-12-15_15-35-54
├── git
│   └── IsaacLab.diff
├── params
│   └── events.out.tfevents.1765830964.iudc.19621.0
├── model_0.pt
├── model_50.pt
├── model_100.pt
├── model_150.pt
├── model_200.pt
├── model_250.pt
├── model_300.pt
├── model_350.pt
└── model_400.pt
```

- Lo mas importante son los archivos con la extensión `.pt`, esto nos permitira visualizar el entrenamiento.

🔥 Ejecución

- Para la ejecución usaremos el siguiente comando:

```
./isaacclab.sh -p scripts/reinforcement_learning/rsl_rl/play.py
--task Isaac-Velocity-Flat-G1-v0
```



Visualización por checkpoint

```
./isaacsim.sh -p scripts/reinforcement_learning/rsl_rl/play.py
--task Isaac-Velocity-Flat-G1-v0 --checkpoint
logs/rsl_rl/g1_flat/2025-12-15_15-35-54/model_1499.pt
# Comando de ejemplo
```

Con la ruta absoluta

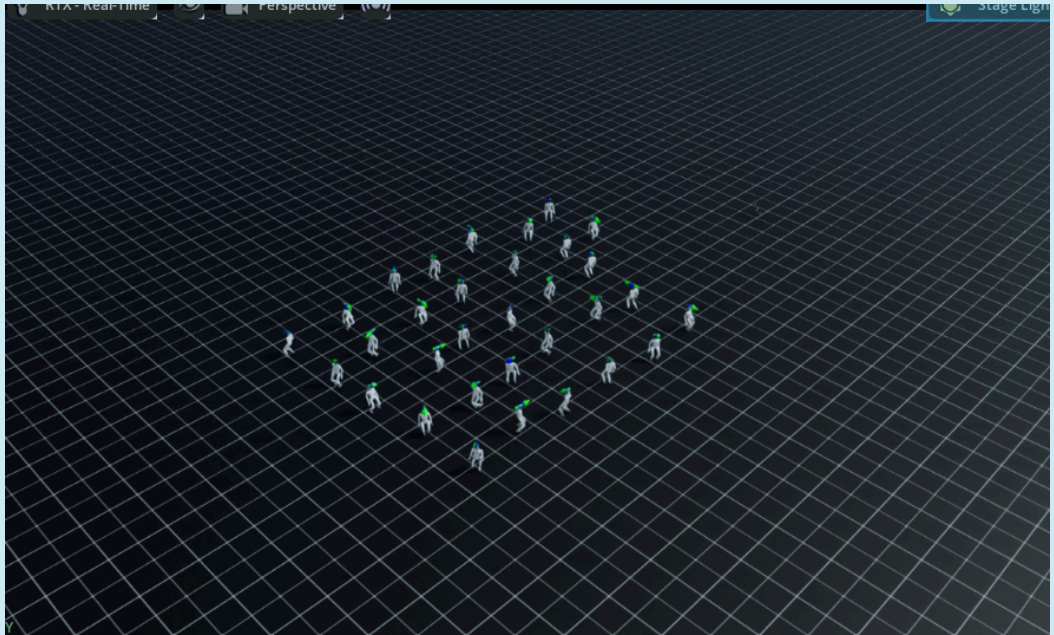
```
./isaacsim.sh -p scripts/reinforcement_learning/rsl_rl/play.py
--task Isaac-Velocity-Flat-G1-v0 --checkpoint
/home/iudc/laboratorio_isaacsim/IsaacLab/logs/rsl_rl/g1_flat/2
025-12-15_15-35-54/model_1499.pt
```

Visualización con un número N de robots

- Usar el siguiente comando

```
./isaacsim.sh -p scripts/reinforcement_learning/rsl_rl/play.py
--task Isaac-Velocity-Flat-G1-v0 --num_envs 32 --checkpoint
logs/rsl_rl/g1_flat/2025-12-15_15-35-54/model_1499.pt
```


- Esto nos permite visualizar con mayor fluidez



Cosas importantes

1. Al realizar un entrenamiento nos genera archivos con la extension `.pt` , esos archivos los podemos usar para integrar al robot en la vida real y en simulación, pero lo complicados es la parte de la configuración. Ya que puede variar para el robot, pero el robot `Unitree-G1` , en la vida real ya tiene configurado estabilizador y modos de caminata, ya sea integración en simulación, necesita tener en cuenta, ya que la politica se configura para un numero especifico de grados de libertad (12,23,27,29...) y no todos funcionan tal cuál para cada robot.
2. Todo el entorno anterior funciona unicamente con hardware de `#NVidia` .
3. El hardware usado para el desarrollo de la documentación es lo minimo para poder ejecutar `#Isaac-Lab` con `#Isaac-sim` .