Machine Learning LeRobot avec SO-ARM101

Installation et prérequis

- Une carte graphique NVidia et une installation de Cuda ?
- Comparaison cartes graphiques (un peu dépassé car ne prend pas en compte les dernières génération type 5090) https://www.geeksforgeeks.org/machine-learning/choosing-the-right-gpu-for-your-machine-learning/
- Puissance de calcul dispo à l'Innov'Lab TPS https://www.innovlab-tps.net/calcul-et-stockage.html
  - Pour l'**entraînement** des modèles : Serveur de calcul Apollo 6500 doté de **4 GPU HGX A100, 80Go**
  - Pour l'**inférence** des modèles : Terminaux de calcul dotés GPU Nvidia **RTX 4090, 24Go**

Prérequis pour l'exécution d'un modèle d'IA :

- Config minimum : L'exécution semble résulter en un mouvement saccadé du robot sur une Quadro P620 (2 GB GDDR5).
- Config recommandée : 4-8 GB GDDR, testé avec RTX 2080 Super de 2019 (8 GB GDDR6)
- Il faut au moins 16 GB de RAM (CPU) sinon elle sature pendant l'enregistrement du dataset d'évaluation.

Prérequis pour l'entraînement d'un modèle d'IA :

- L'entrainement avec 100 épisodes et 100 000 steps a mis 12H sur une RTX 2080 Super de 2019
  - mais `batch_size=8` alors qu'il faut un minimum de 16, et dans l'idéal 64. Cela résulte sans doute en des mouvements saccadés et une mauvaise généralisation du modèle (variation de la position de la pile).
  - Il n'aboutit pas au bout de plus de 24H sur une Quadro P620 (via WSL2 et Docker)
- Config minimum pour ACT : `batch_size=16`
  - Tesla T4 (gratuit 5H / mois sur Google Colab) on a 15G de RAM qui permet donc un `batch_size=15`
  - https://github.com/huggingface/lerobot/issues/2213

- Config recommandée pour ACT : `batch_size=64`
  - avec A100 (12€ les 100 compute units, 70 = ~5H sur Google Colab) on a 40G de RAM ou 80G de RAM (extra RAM)
  - https://github.com/huggingface/lerobot/blob/main/docs/source/notebooks.mdx#training-act
  - https://colab.research.google.com/github/antonilo/real_world_robot_learning_sp25/blob/main/_tutorials/lerobot_tutorial/lerobot_tutorial.ipynb#scrollTo=ff7eafe4
  - avec un `batch_size=64` la RAM consommée était d'environ 50G, l'entraînement a mis 7H environ pour environ 80 compute units, soit environ 10€
  - https://huggingface.co/gautz/act_so101_pick_red_18650_drop_black_box_test2_100ep_64batch_60ksteps
  - https://wandb.ai/hentz-robotics/lerobot/runs/niiti0v3?nw=nwusergautz

Installation sous Linux

- Installer Miniconda pour Linux : l'environnement de développement Python

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
# Vérifier que la clé SHA256 de Miniconda3-latest-Linux-x86_64.sh ici :
https://repo.anaconda.com/miniconda/ correspond à :
sha256sum ~/Miniconda3-latest-Linux-x86_64.sh
bash ~/Miniconda3-latest-Linux-x86_64.sh
source ~/.bashrc
```

- Créer et activer l'environnement Conda

```
conda create -y -n lerobot python=3.10
conda activate lerobot
git clone https://github.com/huggingface/lerobot.git ~/lerobot
conda install ffmpeg=7.1.1 -c conda-forge
cd ~/lerobot && pip install -e ".[feetech]"
```

- A chaque ouverture de Terminal l'environnement python conda est activé, voir au bas du `~/.bashrc`
- Pour éviter les conflits, on propose d'avoir un fichier `~/.bashrc_conda` pour conda et un `~/.bashrc_ros` pour ros

Astuces pour activer/désactiver l'environnement conda sans passer par la modification du `~/.bashrc` :

- Ne pas activer conda au démarrage : `conda config --set auto_activate_base false`

- Ne pas configurer le shell pour initialiser conda au démarrage : `conda init --reverse $SHELL`

Installation Windows

- Le compte utilisateur doit avoir les droits pour créer des raccourcis (liens symboliques) dans les sous-dossiers de `C:\Users\$USER\lerobot\outputs\train\`
- Ils seront utilisés lors de l'entraînement pour créer un lien entre le dossier `last` et le dossier du dernier Checkpoint par ex. `100000`
  - Le plus sûr est de **travailler avec un compte administrateur**
  - Il faut peut-être aussi les droits dans le dossier `C:\Users\$USER\.cache\huggingface\lerobot\$HUGGINGFACE_USER`
- Installer Miniconda pour Windows : l'environnement de développement Python
- Ouvrir `Anaconda PowerShell Prompt`
- Créer et activer l'environnement Conda

```
conda create -y -n lerobot python=3.10
conda activate lerobot
git clone https://github.com/huggingface/lerobot.git ~/lerobot
```

- Installer Pytorch pour la version de Cuda installée sur votre système (testé avec une version Cuda 127 installée et la version cu128 de Pytorch) et autres dépendances nécessaires

```
cd ~/lerobot
# pip install av poetry-core
conda install ffmpeg=7.1.1 -c conda-forge
pip3 install --pre torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu128
pip install -e ".[feetech]"
```

Débuguer l'installation si les scripts basculent sur le `cpu`

S'assurer que Pytorch a été installé, cela installe Cuda :

`pip3 install --pre torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu128`

Récupérer les infos système :

- `python lerobot/scripts/display_sys_info.py`
- `python -m torch.utils.collect_env`
- `python -c "import torch; print(torch.cuda.is_available())" && nvcc -V`

- cf. https://github.com/huggingface/lerobot/issues/928 > Here for additional information of my full installation.

Créer un compte HuggingFace et se login via un token

- Se créer un compte sur https://huggingface.co/settings/tokens
- Aller dans le menu > Access Tokens
- Récupérer un `TOKEN` avec des droits en écriture
- Ouvrir un Anaconda PowerShell lerobot
  `(lerobot) PS C:\Users\gauthier.hentz\github\lerobot>`
- Login avec le Token
  `huggingface-cli.exe login --token TOKEN`

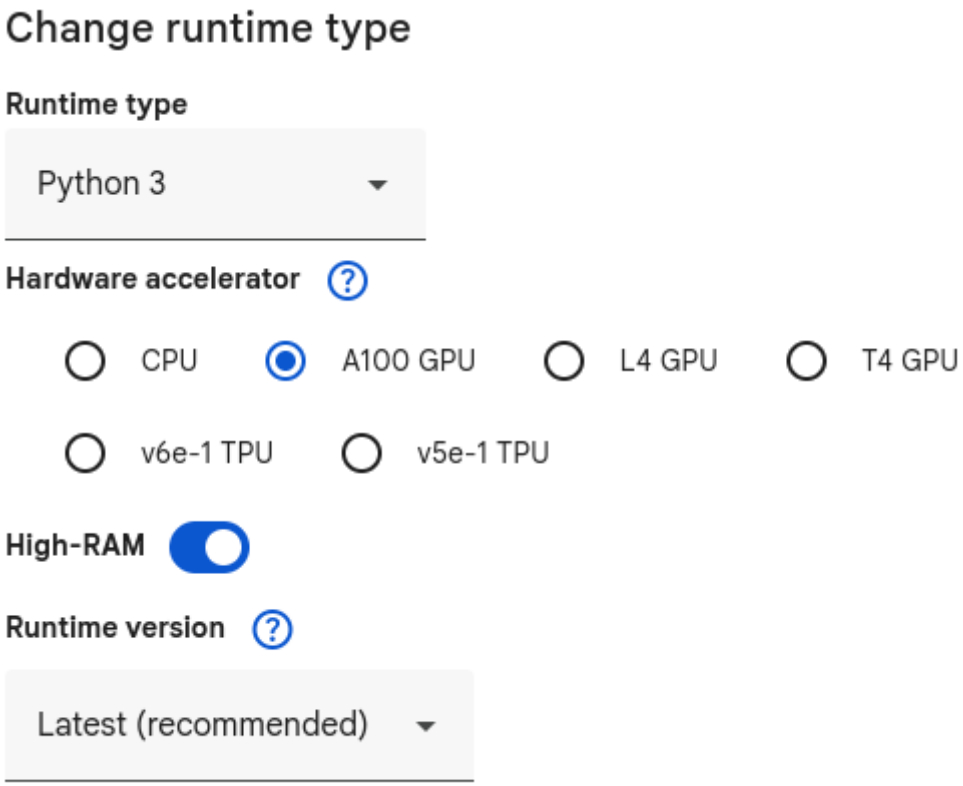
Vous pouvez maintenant uploader et downloaded des DataSets et Modèles vers le Hub HuggingFace pour collaborer avec des experts du Machine Learning.

Google Colab

Pour avoir accès à un GPU avec suffisamment de mémoire (16G recommandé et 64G dans l'idéal) on peut utiliser Google Colab

- Avec Tesla T4 (gratuit 5H / mois) on a 15G de RAM qui permet donc un `batch_size=15`
- Avec A100 (12€ les 70 crédits) on a 80G de RAM qui permet donc un `batch_size=64`

- Ouvrir le Notebook : https://colab.research.google.com/github/huggingface/notebooks/blob/main/lerobot/training-act.ipynb#scrollTo=NQUk3Y0WwYZ4
- Voir aussi https://colab.research.google.com/github/antonilo/real_world_robot_learning_sp25/blob/main/_tutorials/lerobot_tutorial/lerobot_tutorial.ipynb#scrollTo=ff7eafe4
- Change runtime type



- Install conda
- Install LeRobot
- Login avec un jeton d'API Weights & Biases
- Login avec un jeton d'API Hugging Face Hub
- Modifier la commande d'entrainement et l'exécuter :

```
!cd lerobot && python src/lerobot/scripts/lerobot_train.py \
  --dataset.repo_id=gautz/so101_pick_red_18650_drop_black_box_test2_100ep \
  --policy.type=act \
  --output_dir=outputs/train/act_so101_pick_red_18650_drop_black_box_test2_100ep_64batch \
  --job_name=act_so101_pick_red_18650_drop_black_box_test2_100ep_64batch \
  --policy.device=cuda \
  --policy.push_to_hub=True \
  --policy.repo_id=gautz/act_so101_pick_red_18650_drop_black_box_test2_100ep_64batch \
  --batch_size=64 \
  --wandb.enable=true
```

- Surveiller l'éxecution depuis le runtime ou WanDB
- Ouvrir un Terminal et lancer la commande pour uploader le dernier Checkpoint du modèle :

```
huggingface-cli upload ${HF_USER}/act_so101_pick_red_18650_drop_black_box_test2_100ep_64batch \
```

```
/content/lerobot/outputs/train/act_so101_pick_red_18650_drop_black_box_test2_100ep_64batch/checkpoi
```

`nts/last/pretrained_model` https://api.wandb.ai/links/hentz-r

Astuces et Erreurs :

- Si vous utilisez Colab avec un compte gratuit, il faut utiliser l'extension Colab Auto Reconnect pour ne pas être déconnecté au boût de 1H. Il est probable que le runtime soit déconnecté par manque de crédit gratuit avant d'arriver à un checkpoint. Faire des CheckPoint plus souvent, ou passer à un compte payant.
- Si vous utilisez un compte payant, vous ne serez pas déconnecté mais il faut surveiller pour déconnecter le Runtime à la fin de l'entraînement (ou d'un Checkpoint)
- https://github.com/huggingface/lerobot/issues/1532

```
  18  cd /content/
  20  cd lerobot/
  24  python
  25              from huggingface_hub import HfApi
  26              hub_api = HfApi()
  27              hub_api.create_tag("gautz/so101_pick_red_18650_drop_black_box_test2_100ep",
tag="v2.1", repo_type="dataset"
  30  cd src/
  32  cd lerobot
  34  python -m lerobot.datasets.v30.convert_dataset_v21_to_v30 --repo-
id=gautz/so101_pick_red_18650_drop_black_box_test2_100ep
```

- https://stackoverflow.com/questions/78448832/colab-how-to-increase-the-num-workers-in-dataloader

Voir le rapport : https://api.wandb.ai/links/hentz-robotics/wdsr4k2r

Utilisation de LeRobot avec le bash Linux

- Activer l'environnement conda lerobot
  `cd ~/lerobot`
  `conda activate lerobot`

`lerobot-find-port`

A chaque connexion du robot : `sudo chmod 666 /dev/ttyACM1`

`lerobot-find-cameras opencv`

`lerobot-calibrate --robot.type=so101_follower --robot.port=/dev/ttyACM0 --robot.id=follower_arm_fan1`

```
lerobot-calibrate --teleop.type=so101_leader --teleop.port=/dev/ttyACM1 --teleop.id=leader_arm_fan1
```

```
lerobot-teleoperate --robot.type=so101_follower --robot.port=/dev/ttyACM0 --robot.id=follower_arm_fan1 --teleop.type=so101_leader --teleop.port=/dev/ttyACM1 --teleop.id=leader_arm_fan1
```

```
lerobot-dataset-viz --repo-id gautz/so101_pick_red_18650_drop_black_box_test2_100ep --episode-index 0
```

```
lerobot-record --robot.type=so101_follower --robot.port=/dev/ttyACM0 --robot.id=follower_arm_fan1 --robot.cameras="{ top: {type: opencv, index_or_path: 4, width: 800, height: 600, fps: 20}, follower: {type: opencv, index_or_path: 2, width: 800, height: 600, fps: 20}}" --display_data=true --dataset.push_to_hub=False --dataset.repo_id=gautz/eval_act_so101_pick_red_18650_drop_black_box_test2_100ep_64batch_60ksteps --dataset.num_episodes=10 --dataset.single_task="Pick red 18650 battery place black box" --policy.path=gautz/act_so101_pick_red_18650_drop_black_box_test2_100ep_64batch_60ksteps
```

```
--resume=true
```

```
lerobot-train --dataset.repo_id=gautz/so101_pick_red_18650_drop_black_box_test2_100ep --policy.type=act --output_dir=outputs/train/act_so101_pick_red_18650_drop_black_box_test2_100ep_m620 --job_name=act_so101_pick_red_18650_drop_black_box_test2_100ep_m620 --policy.device=cuda --wandb.enable=false --policy.repo_id=gautz/act_so101_pick_red_18650_drop_black_box_test2_100ep_m620 --batch_size=2
```

nano lerobot/common/robot_devices/robots/configs.py

python lerobot/scripts/control_robot.py --robot.type=so101 --robot.cameras='{}' --control.type=teleoperate

```
lerobot-record --robot.type=so101_follower --robot.port=/dev/ttyACM0 --robot.id=follower_arm_fan1 --robot.cameras="{ top: {type: opencv, index_or_path: 0, width: 800, height: 600, fps: 20}, follower: {type: opencv, index_or_path: 2, width: 800, height: 600, fps: 20}}" --dataset.push_to_hub=False --dataset.repo_id=gautz/eval_act_so101_pick_battery_place_ifixit_test1_50ep_rtx507016_chkpt100kter --dataset.num_episodes=10 --dataset.single_task="Eval Pick battery place ifixit" --policy.path=outputs/train/act_so101_pick_battery_place_ifixit_test1_50ep_rtx507016/checkpoints/100000/pretrained_model --display_data=false
```

Server inference

```
pip install 'lerobot[feetech,async]'
```

```
python -m lerobot.async_inference.robot_client --server_address=127.0.0.1:8080 --robot.type=so101_follower --robot.port=/dev/ttyACM0 --robot.id=follower_arm_fan1 --robot.cameras="{ top: {type: opencv, index_or_path: 0, width: 800, height: 600, fps: 20}, follower: {type: opencv, index_or_path: 2, width: 800, height: 600, fps: 20}}" --task="Eval Pick battery place ifixit" --policy_type=act --pretrained_name_or_path=gautz/act_so101_pick_battery_place_ifixit_test1_50ep_rtx507016_chkpt100000 --policy_device=cuda --actions_per_chunk=50 --chunk_size_threshold=0.5 --aggregate_fn_name=weighted_average --debug_visualize_queue_size=True
```

Calibration robot et configuration caméras

`python -m lerobot.calibrate --teleop.type=so101_leader --teleop.port=/dev/ttyACM0 --teleop.id=leader_arm_fan1`

`python -m lerobot.calibrate --robot.type=so101_follower --robot.port=/dev/ttyUSB0 --robot.id=follower_arm_fan1`

Téléopération

```
python -m lerobot.teleoperate \
    --robot.type=so101_follower \
    --robot.port=/dev/ttyUSB0 \
    --robot.id=follower_arm_fan1 \
    --robot.cameras="{ top: {type: opencv, index_or_path: 2, width: 640, height: 480, fps: 30},follower: {type: opencv, index_or_path: 4, width: 640, height: 480, fps: 30}}" \
    --teleop.type=so101_leader \
    --teleop.port=/dev/ttyACM0 \
    --teleop.id=leader_arm_fan1 \
    --display_data=true
```

Rejouer dataset en local :

```
python -m lerobot.replay \
    --robot.type=so101_follower \
    --robot.port=/dev/ttyUSB0 \
    --robot.id=follower_arm_fan1 \
    --dataset.repo_id=gautz/18650-test1-10ep \
    --dataset.episode=0 # choose the episode you want to replay
```

Machine Learning

Enregistrer dataset en local :

```
python -m lerobot.record \
    --robot.type=so101_follower \
    --robot.port=/dev/ttyUSB0 \
    --robot.id=follower_arm_fan1 \
    --robot.cameras="{ top: {type: opencv, index_or_path: 2, width: 640, height: 480, fps: 30},follower: {type: opencv, index_or_path: 4, width: 640, height: 480, fps: 30}}" \
```

```
    --teleop.type=so101_leader \
    --teleop.port=/dev/ttyACM0 \
    --teleop.id=leader_arm_fan1 \
    --display_data=true \
    --dataset.repo_id=gautz/18650-test1-10ep \
    --dataset.episode_time_s=10 \
    --dataset.reset_time_s=10 \
    --dataset.num_episodes=10 \
    --dataset.single_task="Pick red 18650 battery place black box" \
    --dataset.push_to_hub=False
```

Entrainer en local avec le CPU

```
python lerobot/scripts/train.py \
  --dataset.repo_id=gautz/18650-test1-10ep \
  --policy.type=act \
  --output_dir=outputs/train/act_so101_18650-test1-10ep \
  --job_name=act_so101_18650-test1-10ep \
  --policy.device=cpu # \
  --wandb.enable=false # true
```

Entrainer en local avec le GPU NVidia

```
python lerobot/scripts/train.py \
  --dataset.repo_id=gautz/18650-test1-10ep \
  --policy.type=act \
  --output_dir=outputs/train/act_so101_18650-test1-10ep \
  --job_name=act_so101_18650-test1-10ep \
  --policy.device=cuda \
  --wandb.enable=false
```

Enregistrer un dataset d'évaluation d'un modèle à un checkpoint donné :

```
python -m lerobot.record  \
--robot.type=so101_follower \
--robot.port=/dev/ttyUSB0 \
--robot.cameras="{ top: {type: opencv, index_or_path: 2, width: 800, height: 600, fps:
30},follower: {type: opencv, index_or_path: 4, width: 800, height: 600, fps: 30}}" \
--robot.id=follower_arm_fan1 \
--display_data=false \
```

```
--dataset.repo_id=gautz/eval_act_18650-test2-100ep \
--dataset.single_task="Pick red 18650 battery place black box" \
--policy.path=outputs/train/act_so101_18650-test2-100ep/checkpoints/last/pretrained_model \
--dataset.push_to_hub=False
```

Utilisation de LeRobot avec Anaconda Powershell Prompt

Calibration des robots

`python -m lerobot.setup_motors --robot.type=so101_follower --robot.port=COM13`
`python -m lerobot.setup_motors --robot.type=so101_leader --robot.port=COM14`

Machine Learning

Collecte de données :

Uploader les données vers le Hub HuggingFace :

https://huggingface.co/docs/lerobot/en/il_robots#dataset-upload

`huggingface-cli.exe upload gautz/so101_pick_red_18650_drop_black_box_test2_100ep ..\.cache\huggingface\lerobot\gautz\18650-test2-100ep\ --repo-type dataset`

Entrainer un modèle sur un Dataset (sur `IHA-QLIOVR-1`, compte admin) :

```
cd ..\gauthier.hentz\lerobot\
conda activate lerobot
cd .\lerobot\
conda create -y -n lerobot python=3.10
conda activate lerobot
conda install ffmpeg -c conda-forge
ffmpeg -encoders
conda install ffmpeg=7.1.1 -c conda-forge
pip install av poetry-core
```

```
pip3 install --pre torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu128
pip install -e .
python lerobot/scripts/train.py --dataset.repo_id=gautz/18650-test2-100ep --policy.type=act --
output_dir=outputs/train/act_so101_18650-test2-100ep --job_name=act_so101_18650-test2-100ep --
policy.device=cuda --wandb.enable=false
```

Enregistrer un dataset d'évaluation d'un modèle à un checkpoint donné :

```
python -m lerobot.record  --robot.type=so101_follower --robot.port=/dev/ttyUSB0 --robot.cameras="{
top: {type: opencv, index_or_path: 2, width: 800, height: 600, fps: 30},follower: {type: opencv,
index_or_path: 4, width: 800, height: 600, fps: 30}}" --robot.id=follower_arm_fan1 --
display_data=false --dataset.repo_id=gautz/eval_act_18650-test2-100ep --dataset.single_task="Pick
red 18650 battery place black box" --policy.path=outputs/train/act_so101_18650-test2-
100ep/checkpoints/last/pretrained_model --dataset.push_to_hub=False
```

Uploader un modèle vers HuggingFace

https://github.com/huggingface/lerobot/?tab=readme-ov-file#add-a-pretrained-policy

- Rechercher le dossier du Checkpoint voulu, par ex. `C:\Users\User\github\lerobot\outputs\train\act_so101_18650-test2-100ep\checkpoints\100000`
- Définir le User HuggingFace (qui doit être Login via un Token) `gautz` et le Repo `act_so101_pick_red_18650_drop_black_box_test2_100ep_100000` , puis uploader le Checkpoint : `huggingface-cli.exe upload gautz/act_so101_pick_red_18650_drop_black_box_test2_100ep_100000 .\outputs\train\act_so101_18650-test2-100ep\checkpoints\100000`

Historique du Terminal complet

```
wget "https://repo.anaconda.com/miniconda/Miniconda3-latest-Windows-x86_64.exe" -outfile
".\Downloads\Miniconda3-latest-Windows-x86_64.exe"
cd .\github\lerobot\
conda create -y -n lerobot python=3.10
conda activate lerobot
conda install ffmpeg -c conda-forge
conda install ffmpeg=7.1.1 -c conda-forge
wsl --shutdown
pip install -e .
pip install -e ".[feetech]" # or "[dynamixel]" for example
python lerobot/find_port.py
python lerobot/find_cameras.py opencv
python -m lerobot.setup_motors --robot.type=so101_follower --robot.port=COM13`
```

```
python -m lerobot.record  --robot.type=so101_follower --robot.port=COM13 --robot.cameras="{ top: {type: opencv, index_or_path: 2, width: 800, height: 600, fps: 30},follower: {type: opencv, index_or_path: 4, width: 800, height: 600, fps: 30}}" --robot.id=follower_arm_fan1 --display_data=false --dataset.repo_id=gautz/eval_act_18650-test2-100ep --dataset.single_task="Pick red 18650 battery place black box" --policy.path=outputs/train/act_so101_18650-test2-100ep/checkpoints/last/pretrained_model --dataset.push_to_hub=False
python -m lerobot.calibrate     --teleop.type=so101_leader  --teleop.port=COM14 --teleop.id=leader_arm_fan1`
python -m lerobot.calibrate     --robot.type=so101_follower  --robot.port=COM13 --robot.id=follower_arm_fan1
python -m lerobot.teleoperate   --robot.type=so101_follower   --robot.port=COM13   --robot.id=follower_arm_fan1   --robot.cameras="{ top: {type: opencv, index_or_path: 1, width: 640, height: 480, fps: 30},follower: {type: opencv, index_or_path: 2, width: 640, height: 480, fps: 30}}"   --teleop.type=so101_leader   --teleop.port=COM14   --teleop.id=leader_arm_fan1 --display_data=true
python lerobot/find_cameras.py opencv
python -m lerobot.teleoperate   --robot.type=so101_follower   --robot.port=COM13   --robot.id=follower_arm_fan1   --robot.cameras="{ top: {type: opencv, index_or_path: 1, width: 800, height: 600, fps: 30},follower: {type: opencv, index_or_path: 2, width: 800, height: 600, fps: 30}}"   --teleop.type=so101_leader   --teleop.port=COM14   --teleop.id=leader_arm_fan1 --display_data=true
pip install -e .
python lerobot/find_cameras.py opencv
python -m lerobot.teleoperate   --robot.type=so101_follower   --robot.port=COM13   --robot.id=follower_arm_fan1   --robot.cameras="{ top: {type: opencv, index_or_path: 1, width: 800, height: 600, fps: 30},follower: {type: opencv, index_or_path: 0, width: 800, height: 600, fps: 30}}"   --teleop.type=so101_leader   --teleop.port=COM14   --teleop.id=leader_arm_fan1 --display_data=true
python -m lerobot.record  --robot.type=so101_follower --robot.port=COM13 --robot.cameras="{ top: {type: opencv, index_or_path: 1, width: 800, height: 600, fps: 30},follower: {type: opencv, index_or_path: 0, width: 800, height: 600, fps: 30}}" --robot.id=follower_arm_fan1 --display_data=false --dataset.repo_id=gautz/eval_act_18650-test2-100ep --dataset.single_task="Pick red 18650 battery place black box" --policy.path=outputs/train/act_so101_18650-test2-100ep/checkpoints/last/pretrained_model --dataset.push_to_hub=False
python lerobot/scripts/display_sys_info.py
python -m torch.utils.collect_env
python -c "import torch; print(torch.cuda.is_available())" && nvcc -V
```

```
python -c "import torch; print(torch.cuda.is_available())"
pip3 install --pre torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu128
python -m lerobot.record  --robot.type=so101_follower --robot.port=COM13 --robot.cameras="{
top: {type: opencv, index_or_path: 1, width: 800, height: 600, fps: 30},follower: {type:
opencv, index_or_path: 0, width: 800, height: 600, fps: 30}}" --robot.id=follower_arm_fan1 --
display_data=false --dataset.repo_id=gautz/eval_act_18650-test2-100ep --
dataset.single_task="Pick red 18650 battery place black box" --
policy.path=outputs/train/act_so101_18650-test2-100ep/checkpoints/last/pretrained_model --
dataset.push_to_hub=False --wandb.enable=false
python -m lerobot.record  --robot.type=so101_follower --robot.port=COM13 --robot.cameras="{
top: {type: opencv, index_or_path: 1, width: 800, height: 600, fps: 30},follower: {type:
opencv, index_or_path: 0, width: 800, height: 600, fps: 30}}" --robot.id=follower_arm_fan1 --
display_data=false --dataset.repo_id=gautz/eval_act_18650-test2-100ep --
dataset.single_task="Pick red 18650 battery place black box" --
policy.path=outputs/train/act_so101_18650-test2-100ep/checkpoints/last/pretrained_model --
dataset.push_to_hub=False     --dataset.episode_time_s=10  --dataset.reset_time_s=10    --
dataset.num_episodes=10
exit
cd .\github\lerobot\
conda activate lerobot
python -m lerobot.record  --robot.type=so101_follower --robot.port=COM13 --robot.cameras="{
top: {type: opencv, index_or_path: 1, width: 800, height: 600, fps: 30},follower: {type:
opencv, index_or_path: 0, width: 800, height: 600, fps: 30}}" --robot.id=follower_arm_fan1 --
display_data=false --dataset.repo_id=gautz/eval_act_18650-test2-100ep --
dataset.single_task="Pick red 18650 battery place black box" --
policy.path=outputs/train/act_so101_18650-test2-100ep/checkpoints/last/pretrained_model --
dataset.push_to_hub=False     --dataset.episode_time_s=15  --dataset.reset_time_s=10    --
dataset.num_episodes=10
python -m lerobot.teleoperate   --robot.type=so101_follower   --robot.port=COM13   --
robot.id=follower_arm_fan1   --robot.cameras="{ top: {type: opencv, index_or_path: 1, width:
800, height: 600, fps: 30},follower: {type: opencv, index_or_path: 0, width: 800, height: 600,
fps: 30}}"   --teleop.type=so101_leader   --teleop.port=COM14   --teleop.id=leader_arm_fan1
--display_data=true
exit
cd .\github\lerobot\
conda activate lerobot
python -m lerobot.teleoperate   --robot.type=so101_follower   --robot.port=COM13   --
robot.id=follower_arm_fan1   --robot.cameras="{ top: {type: opencv, index_or_path: 1, width:
```

```
800, height: 600, fps: 30},follower: {type: opencv, index_or_path: 0, width: 800, height: 600,
fps: 30}}"   --teleop.type=so101_leader   --teleop.port=COM14   --teleop.id=leader_arm_fan1
--display_data=true
python lerobot/find_port.py
python -m lerobot.teleoperate   --robot.type=so101_follower   --robot.port=COM15   --
robot.id=follower_arm_fan1   --robot.cameras="{ top: {type: opencv, index_or_path: 1, width:
800, height: 600, fps: 30},follower: {type: opencv, index_or_path: 0, width: 800, height: 600,
fps: 30}}"   --teleop.type=so101_leader   --teleop.port=COM14   --teleop.id=leader_arm_fan1
--display_data=true
python lerobot/find_cameras.py opencv
python -m lerobot.teleoperate   --robot.type=so101_follower   --robot.port=COM15   --
robot.id=follower_arm_fan1   --robot.cameras="{ top: {type: opencv, index_or_path: 0, width:
800, height: 600, fps: 30},follower: {type: opencv, index_or_path: 2, width: 800, height: 600,
fps: 30}}"   --teleop.type=so101_leader   --teleop.port=COM14   --teleop.id=leader_arm_fan1
--display_data=true
python -m lerobot.record  --robot.type=so101_follower --robot.port=COM15 --robot.cameras="{
top: {type: opencv, index_or_path: 1, width: 800, height: 600, fps: 30},follower: {type:
opencv, index_or_path: 0, width: 800, height: 600, fps: 30}}" --robot.id=follower_arm_fan1 --
display_data=false --dataset.repo_id=gautz/eval_act_18650-test2-100ep --
dataset.single_task="Pick red 18650 battery place black box" --
policy.path=outputs/train/act_so101_18650-test2-100ep/checkpoints/last/pretrained_model --
dataset.push_to_hub=False     --dataset.episode_time_s=15  --dataset.reset_time_s=10    --
dataset.num_episodes=10
python -m lerobot.record  --robot.type=so101_follower --robot.port=COM15 --robot.cameras="{
top: {type: opencv, index_or_path: 0, width: 800, height: 600, fps: 30},follower: {type:
opencv, index_or_path: 2, width: 800, height: 600, fps: 30}}" --robot.id=follower_arm_fan1 --
display_data=false --dataset.repo_id=gautz/eval_act_18650-test2-100ep --
dataset.single_task="Pick red 18650 battery place black box" --
policy.path=outputs/train/act_so101_18650-test2-100ep/checkpoints/last/pretrained_model --
dataset.push_to_hub=False     --dataset.episode_time_s=15  --dataset.reset_time_s=10    --
dataset.num_episodes=10
python -m lerobot.teleoperate   --robot.type=so101_follower   --robot.port=COM15   --
robot.id=follower_arm_fan1   --robot.cameras="{ top: {type: opencv, index_or_path: 0, width:
800, height: 600, fps: 30},follower: {type: opencv, index_or_path: 2, width: 800, height: 600,
fps: 30}}"   --teleop.type=so101_leader   --teleop.port=COM14   --teleop.id=leader_arm_fan1
--display_data=true
cd .\github\lerobot\
cd ..
cd .\.cache\
```

```
cd .\huggingface\ls
cd .\huggingface\lerobot\calibration\cd ..
cd .\huggingface\lerobot\
cd .\gautz\
cd ..
cd .\github\lerobot\
cd .\outputs\train\
cd .\act_so101_18650-test2-100ep\
cd .\checkpoints\
```

Migration dataset v2.1 to v3

https://github.com/huggingface/lerobot/blob/main/docs/source/lerobot-dataset-v3.mdx#migrate-v21--v30

Sources

https://wiki.seeedstudio.com/lerobot_so100m/

https://huggingface.co/spaces/lerobot/visualize_dataset?path=%2Fsebastiandavidlee%2Fbimanual-so101-handover-plushie%2Fepisode_40

https://huggingface.co/blog/sherryxychen/train-act-on-so-101

https://github.com/sherrychen1120/so101_bench

https://www.linkedin.com/posts/sherryxychen_robotics-machinelearning-ai-ugcPost-7378831270207954944-WsI5?utm_source=share&utm_medium=member_desktop&rcm=ACoAAA1icaQBkYXRoM1Oslzrh8psuL0MmHpaT_4

Revision #50
Created 8 April 2025 20:29:23 by admin_idf
Updated 8 May 2026 17:17:40 by admin_idf