# 1. DeepRacer-For-Dummies 시작하기

Docker compose를 활용하여 쉽고 빠르게 딥-레이서 로컬 학습환경을 구축한다.

<https://github.com/crr0004/deepracer>

더 많은 정보를 얻고 싶다면 위의 사이트를 참고하세요.

이 Dummy repo는 위 사이트를 참고하여 만들었습니다.

# 2. Youtube 간단설명

<https://www.youtube.com/watch?v=CFNcKmtVRSI>



# 3. 시작하기

## 1. 기본 요구 사항(아래에 기본 요구 사항들 설치방법 명시되어 있으니 걱정 말기.)

* Ubuntu 18.04, Nvidia GPU 사용, CUDA/CUDNN는 이미 설치되어 있어야 함.
* **Docker**, **Nvidia-Docker** runtime.
* **AWS cli** 이 설치되어 있어야 하며, AWS 계정이 .

계정 있음에 대한 자격증명이 (~/.aws/credentials)에 존재해야한다.

아마 AWS cli를 설치하면 /home에 위의 위치가 만들어지는 듯 하다.

* **vncviewer** 가 설치되어 있어야한다.

이것은 서버 컴퓨터를 사용하기 위해 사용하는 viewer이다.

보통은 우분투 컴퓨터와 연결하기 위해 vnc-Controler를 사용하지만, 여기서는 일단 viewer만 설치하면 되는 듯 하다.

Vnc 란? - <https://codingeg.tistory.com/18>

만약 위의 요구사항들을 이미 다 만족했다면, Initialization부분 즉 7부터 실행 작업을 하면 된다. 만약 요구사항을 만족하지 않는다면 아래의 과정을 차근차근 실행하면 됩니다.

# 4. 로컬 환경 설정 하기 – 요구사항 설치하기

윈도우에서 하려고 하면 못합니다. 우분투 설치해서 사용하세요. 우분투 듀얼 부팅으로 설치하는 방법은 아래와 같습니다.

## 윈도우&우분투 듀얼 부팅방법 알려주는 사이트

<https://medium.com/bigdatarepublic/dual-boot-windows-and-linux-aa281c3c01f9>

<https://win10faq.com/shrink-partition-windows-10/?source=post_page--------------------------->

junha - 듀얼 부팅 만드는 방법은 유투브에서 검색해 나오는 방법을 따라하는게 최고,

## the AWS CLI 설치하기

pip install -U awscli

그리고 아래의 과정을 해야 AWS 계정을 자격증명으로 사용할 수 있다.

<https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-configure.html>

위의의 사이트에서 Quickly configuring the AWS CLI만 하면 된다.

과정요약 = an IAM user key를 생성하고 그것을 다운로드해 내 컴퓨터에 등록한다.

## Docker-ce 설치하기.

(이 사이트 내용 정리한 것이 아래 내용<https://docs.docker.com/install/linux/docker-ce/ubuntu/> )

1.

sudo apt-get remove docker docker-engine docker.io containerd runc

2.

sudo apt-get update

3.

sudo apt-get install \

apt-transport-https \

ca-certificates \

curl \

gnupg-agent \

software-properties-common

4.

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

sudo apt-key fingerprint 0EBFCD88

5.

sudo add-apt-repository \

"deb [arch=amd64] https://download.docker.com/linux/ubuntu \

$(lsb\_release -cs) \

stable"

6.

sudo apt-get update

7.

sudo apt-get install docker-ce docker-ce-cli containerd.io

Docker가 잘 설치되었는지 확인한다.

sudo docker run hello-world

를 실행했을 때, Docker가 잘 실행되었다는 문구가 뜨면 성공한 것이다.

## Docker-compose 설치하기. (Docker-ce이후에 추가적으로 설치)

(from <https://docs.docker.com/compose/install/#install-compose> )

curl -L https://github.com/docker/compose/releases/download/1.24.1/docker-compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose

Docker-compose가 잘 설치되었는지 확인한다.

docker-compose --version

주의사항: 만약에 pip와 conda를 이용해서 docker-conpose를 설치했다면, 그 환경속에서 docker-compose –version을 실행하도록 하라. 또한 많은 OS들이 이미 docker-compose의 의존성과 상충하는 python 패키지를 이미 가지고 있을 수도 있다.

아래 2개의 과정을 통해, docker가 sudo없이 관리자 권한으로 작업을 수행할 수 있도록 만들어야한다.

(from <https://docs.docker.com/install/linux/linux-postinstall/> )

sudo groupadd docker

sudo usermod -aG docker $USER

아래의 작업을 통해서, 우분투를 재 실행하더라도, docker가 sudo없이 실행될 수 있도록 만들어주어라.

sudo systemctl enable docker

## nvidia-docker 설치하기

NVIDIA Container Toolkit을 사용하면 GPU를 이용해 가속 사용할수 있는 Docker Container를 빌드하고 실행할 수 있습니다.

<https://github.com/NVIDIA/nvidia-docker> - 이 방법을 따라서 설치할 것.

현재 설치되어 있는 nvidia docker관련 내용을 확인하고 싶으면 이것 사용.

sudo apt list --installed | grep nvidia

Nvidia 드라이버를 새로 설치하기 위해, 기존의 nvidia 드라이버를 삭제합니다. nvidia-docker를 설치하고 그 다음에 nvidia-드라이버를 다시 설치할 예정입니다.)

sudo apt-get purge nvidia\*

nvidia-docker runtime을 설치합니다.

**(from**[**https://github.com/NVIDIA/nvidia-docker/wiki/Installation-(version-2.0)**](https://github.com/NVIDIA/nvidia-docker/wiki/Installation-(version-2.0))**)**

distribution=$(. /etc/os-release;echo $ID$VERSION\_ID)

curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add -

curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sudo tee /etc/apt/sources.list.d/nvidia-docker.list

sudo apt-get update

sudo apt-get install nvidia-docker2

sudo pkill -SIGHUP dockerd

적절한 Nvidia Driver를 다시 설치 합니다.

<https://www.nvidia.com/Download/index.aspx?lang=en-us>

이 사이트를 사용해 당신의 GPU에 맞는 Nvida Driver를 설치하세요.

(junha) – 아래의 3줄 명령어는 GPU driver를 설치하는 예시입니다.

- 이 사이트를 이용해 Nvidia- driver를 설치하세요.

(아래의 사이트 중 하나 선택하여 드라이버 설치 – 위의 사이트 추천.)

- <https://hiseon.me/linux/ubuntu/install_nvidia_driver/>

- <https://codechacha.com/ko/install-nvidia-driver-ubuntu/>

sudo add-apt-repository ppa:graphics-drivers

sudo apt-get update

sudo apt install nvidia-driver-410 && sudo reboot

드라이버가 잘 설치되어 있는지 확인합니다.

아래의 명령을 처서 나의 GPU에 맞는 driver가 정상적으로 설치되어 있는지 확인하세요.

nvidia-smi

nvcc --version

## 로컬 컴퓨터에 VNC-viewer설치하기

아래의 사이트를 사용해 간단하게 설치할 수 있습니다.

<https://www.techspot.com/downloads/5760-vnc-viewer.html>

(junha) – 위의 사이트가 안되면 이 사이트 사용할 것.

- <https://www.realvnc.com/en/connect/download/viewer/linux/>

## Installing the Nvidia deep learning libraries (CUDA/CUDNN) for GPU hardware:

CUDA & CUDNN 를 설치하기 위해 아래의 사이트를 사용하세요.

: <https://medium.com/@zhanwenchen/install-cuda-and-cudnn-for-tensorflow-gpu-on-ubuntu-79306e4ac04e>

(junha) – 이 사이트를 사용하는게 편합니다. (한글 사이트)

- <https://hiseon.me/linux/ubuntu/cuda-install/>

- <https://greedywyatt.tistory.com/106>

아나콘다를 사용해서 CUDA/CUDNN을 설치할 수도 있습니다. 많은 사용자들이 아나콘다를 사용하는게 편하다고 합니다. 그래서 conda를 사용해 cuda/cudnn을 설치하는 방법을 아래에 간단히 기입해 놓습니다.

Downloading Anaconda

sudo apt-get update -y && sudo apt-get upgrade -y

cd /tmp/

sudo wget https://repo.anaconda.com/archive/Anaconda3-2019.03-Linux-x86\_64.sh

Installing Anaconda

bash Anaconda3-2019.03-Linux-x86\_64.sh

"yes" for using the default directory location

“yes” for running conda init

Activating Anaconda

source ~/.bashrc

Verifying the conda package manager works

conda list

Installing CUDA/CUDNN

conda install cudnn==7.3.1 && conda install -c fragcolor cuda10.0

# 5. 모든 요구사항이 만족되어 있다면 여기부터 실행 - Initialization

## 7. ./init.sh 실행하기.

위의 요구사항들을 모두 설치했다면, "./init.sh".을 실행합니다.

이것으로 deepracer를 로컬로 트레이닝 할 수 있는 모든 환경을 설정합니다.

**init.sh** 를 run함으로써 아래의 작업들을 일렬로 자동으로 수행해줍니다.

1. 여기서 git내용을 다운 - <https://github.com/crr0004/deepracer.git>
2. 폴더 만들고 폴더로 이동 - mkdir -p ~/.sagemaker && cp config.yaml ~/.sagemaker
3. rl\_deepracer\_coach\_robomaker.py 의 내용을 "crr0004/sagemaker-rl-tensorflow:nvidia” 위치에 올려 hyperparameter를 수정합니다.
4. Instance\_type또한 로컬 GPU로 변경해줍니다.
5. Reward.py내용과 model-metadata파일을 Minio버킷에 복사

the local deepracer training을 시작하거나 멈추고 싶다면,

아래의 scripts를 사용하세요. 사용할 수 있는 Script에 대한 설명은 아래에 있습니다.

### 각 .sh파일에 대한 간단한 설명 (deepracer-for-dummies/scripts/)

* training(deepracer-for-dummies/scripts/training)
  + start.sh
    - docker compose를 사용하여, 전체 training을 시작합니다.
    - sagemaker를 이용한 학습의 결과를 모니터링 할 수 있도록 GUI를 위한 터미널을 열어줍니다.
    - 자동으로 vncviewer를 열어서 Gazabo를 통행 진행과정을 볼 수 있습니다.
  + stop.sh
    - 위에서 시작한 환경(학습)을 중지합니다.
    - Sagemaker 컨테이너에서 시작된 교육 컨테이너를 자동으로 찾아 중지합니다.
  + upload-snapshot.sh
    - 지금까지 학습한 결과를 snapshot으로 AWS에 저장합니다. 어떤 체크 포인트도 없다면 최신 snapshot을 검색합니다. (uploads a specific snapshot to S3 in AWS. If no checkpoint is provided, it attempts to retrieve the latest snapshot)
  + set-last-run-to-pretrained.sh
    - 마지막 학습 저장을 위해 rl-deepracer-sagemaker 을 rl-deepracer-pretrained 로 변경합니다. 그래서 새로운 학습 실행의 시작점을 다시 만들 수 있도록 합니다. (renames the last training run directory from ***rl-deepracer-sagemaker*** to ***rl-deepracer-pretrained*** so that you can use it as a starting point for a new training run.)
  + delete-last-run.sh
    - 모든 시스템 파일을 삭제합니다. 이것을 했을 때 발생하는 문제는 책임지지 않습니다. (WARNING: this script deletes files on your system. I take no responsibility for any resulting actions by running this script. Please look at what the script is doing before running it so that you understand)
    - 모든 스냅 샷 및 로그 파일을 포함하여 마지막 학습 내용을 삭제합니다. 이 명령을 실행하려면 sudo가 필요합니다. (deletes the last training run including all of the snapshots and log files. You will need sudo to run this command.)
* Evaluation(deepracer-for-dummies/scripts/evaluation)
  + start.sh
    - docker conpose를 사용하여 평가를 위한 전체 환경을 시작합니다.
    - 또한 sagemaker training 디렉토리에서 로그 출력을 모니터 할 수있는 터미널 창을 엽니 다.
    - 또한 vncviewer가 자동으로 열리므로 Gazebo에서 진행되는 교육을 볼 수 있습니다.
  + stop.sh
    - 전체 환경을 막습니다
    - Sagemaker 컨테이너에서 시작된 컨테이너를 자동으로 찾아 중지합니다.
* log-analysis (deepracer-for-dummies/scripts/log-analysis)
  + start.sh
    - AWS와 Ptak에서 확장된 로그 파일들을 분석하기 위해서 jupyer를 Nvidai-Docker의 컨텡너로 실행합니다. (starts a container with Nvidia-Docker running jupyter labs with the log analysis notebooks which were originally provided by AWS and then extended by Tomasz Ptak)
    - robomaker의 로그는 컨테이너에 자동으로 로그파일이동이 됩니다. (the logs from robomaker are automatically mounted in the container so you don't have to move any files around)
    - 컨테이너로 이동하려면 시작시의 로그 출력값을 보세요. 토큰 쿼리 매개 변수를 포함한 URL을 가져 와서 localhost : 8888 의 brower에 붙여 넣어야 합니다. (in order to get to the container, look at the log output from when it starts. You need to grab the URL including the token query parameter and then paste it into the brower at **localhost:8888**.)
  + stop.sh
    - 로그 분석 컨테이너를 중지합니다

### 하이퍼파라미터 수정

Learning에 사용되는 hyperparameter의 값은 아래의 파일에서 수정하면 됩니다.

./overrides/rl\_deepracer\_coach\_robomaker.py.

### Action 공간 설정 & Reward Function 수정하기

deepracer-for-dummies/docker/volumes/minio/bucket/custom\_files directory

여기에서 Action space와 Reward Function을 수정할 수 있습니다.

### Track 선택하기

deepracer-for-dummies/docker

여기에 있는 .env 파일의 환경변수(environment variable)를 통해서 track 선택이 됩니다.