Raspberry Pi 5 OS설치 (2025. 10. 03 기준)

https://downloads.raspberrypi.com/raspios_arm64/images/raspios_arm64-2024-11-19/

> 2024-11-19-raspios-bookworm-arm64.img.xz 다운로드

시스템

- Raspberry Pi OS: 2024-11-19 (Debian 12 Bookworm)
- Python: 3.11.x (기본 포함)
- 커널: 6.6.x

개발 도구

- VS Code: 최신 ARM64 버전
- OpenCV: 4.8.0 ~ 4.9.0
- TensorFlow: 2.15.0
- NumPy: 1.24.3

Raspberry Pi Imager > Use custom 선택 후 다운로드 받은 이미지 선택 후 진행

Raspberry Pi에 SD카드 삽입 이후 사용자, 비번, WiFi 설정

1. 시스템 업데이트 실행

```
sudo apt update          # 패키지 목록 업데이트
sudo apt upgrade –y      #시스템 업그레이드
                         #중간에 질문 나오면 기본값(Enter) 선택
```

2. 한글 설치

```
sudo apt install fonts-nanum fonts-nanum-coding fonts-nanum-extra –y
  # 나눔고딕, 나눔명조, 나눔코딩 폰트 설치

sudo apt install ibus ibus-hangul –y
  # 한글입력기 ibus방식으로 프레임워크와 한글엔진 설칠

im-config –n ibus
  # ibus를 기본입력기로 설정
  # ibus 설정 도구 실행 후 나오는 창 OK, YES, > ibus 선택 OK

sudo reboot
  #재부팅

IBus Preferences > Input Method > Add > Korean – Hangul 추가
  # ibus설정 eng없으면 추가
```

3. VS Code설치

VS Code ARM64 다운로드, 설치 :

```
sudo apt install software-properties-common apt-transport-https wget -y
wget -qO- https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor > packages.microsoft.gpg
sudo install -o root -g root -m 644 packages.microsoft.gpg /etc/apt/trusted.gpg.d/
sudo sh -c 'echo "deb [arch=arm64] https://packages.microsoft.com/repos/code stable main" > /etc/apt/sources.list.d/vscode.list'
sudo apt update
sudo apt install code –y
```

실행 :

```
code
```

VS Code 설정

Extensions > python install

Korean Language Pack 검색 > Korean Language Pack for Visual Studio Code install

4. OpenCV 설치

```
sudo apt install -y python3-opencv libopencv-dev
    # python3-opencv: Python용 OpenCV
    # libopencv-dev: C++ 개발용 라이브러리

sudo apt install -y python3-pil python3-numpy
    # Pillow(이미지 처리), NumPy(수치 계산) 설치
```

OpenCV 설치 확인

```
python3
import cv2
print('OpenCV version:', cv2.__version__)

# 출력 : OpenCV version: 4.6.0
# Bookworm의 APT 버전은 4.6.0
```

NumPy 확인

```
python3
import numpy as np
print('NumPy version:', np.__version__)
```

```
    # 출력 : NumPy version: 1.24.x
```

OpenCV 카메라 테스트

```
# test_camera.py
import cv2
# 카메라 열기
cap = cv2.VideoCapture(0)

if not cap.isOpened():
    print("카메라를 열 수 없습니다")
    exit()

print("카메라 테스트 시작 (q를 눌러 종료)")

while True:
    ret, frame = cap.read()
    if not ret:
        break

    cv2.imshow('Camera Test', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

OpenCV 기본 테스트

```
# 이미지 생성 테스트
import cv2
import numpy as np

# 빈 이미지 생성
img = np.zeros((400, 600, 3), dtype=np.uint8)

# 도형 그리기
cv2.rectangle(img, (50, 50), (550, 350), (0, 255, 0), 3)
cv2.circle(img, (300, 200), 80, (255, 0, 0), -1)
cv2.putText(img, "OpenCV Test", (180, 220),
            cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)

print("OpenCV 테스트 이미지 생성 완료")
```

```
print("창이 열립니다. 아무 키나 누르면 종료됩니다.")

cv2.imshow("OpenCV Test", img)
cv2.waitKey(0)
cv2.destroyAllWindows()

print("OpenCV 정상 작동!")
```

5. TensorFlow용 가상환경 설정

1) 가상환경 도구 설치

```
sudo apt install python3-pip python3-venv -y
 # pip와 venv 설치
```

2) Python 버전 확인

```
python3 —version
 # 출력 : Python 3.11.x
```

3) TensorFlow 전용 가상환경 생성

```
python3 -m venv env
 # 홈 디렉토리에 env라는 가상환경 생성
```

4) 가상환경 활성화와 가상환경 안에서 TensorFlow Lite 설치

```
source ~/env/bin/activate
 # 확인 : (env) pi@raspberrypi:~ $

pip install —upgrade pip
 # 최신 pip으로 업그레이드

pip install numpy
 # TensorFlow 호환 NumPy 설치

pip install tflite-runtime
 # 경량화된 TensorFlow 런타임 설치

 TensorFlow Lite 테스트


python3
try:
    import tflite_runtime.interpreter as tflite
    print("☑ TensorFlow Lite 설치 성공!")
    print("TFLite Runtime 사용 가능")
except ImportError as e:
    print(f"❌ TFLite 설치 실패: {e}")

 #출력:
```

☑ TensorFlow Lite 설치 성공!
TFLite Runtime 사용 가능

설치된 패키지 확인

```
pip list
  # 패지키 리스트 출력 예 :
        numpy                1.26.x
        tflite-runtime       2.x.x
        matplotlib           3.x.x
        pillow               10.x.x
        ..............
```

5) 가상환경 종료

```
deactivate #가상환경 비활성화
```

TensorFlow Lite 테스트

```
#TensorFlow_test.py

import sys
print(f"Python 경로: {sys.executable}")
print(f"Python 버전: {sys.version}")
print()

try:
    import tflite_runtime.interpreter as tflite
    print("☑ TensorFlow Lite 로드 성공!")
except ImportError as e:
    print(f"❌ TensorFlow Lite 로드 실패: {e}")
    print("₩n가상환경이 활성화되었는지 확인하세요:")
    print("  source ~/cos/env/bin/activate")
    sys.exit(1)

try:
    import numpy as np
    print(f"☑ NumPy 버전: {np.__version__}")
except ImportError:
    print("❌ NumPy 로드 실패")

print("₩n🎉 모든 라이브러리 정상 작동!")
```

예시 구조

```
    home/cos/env/
```

home/cos/code/TensorFlow_test.py

실행 :

```
source ~/env/bin/activate   # 가상환경 활성화
cd ~/cos/code               # 코드있는 경로 진입
python TensorFlow_test.py   # 실행 명령
```

6. Face Recognition 설치 (시스템에 설치)

1) 시스템 준비

```
# 시스템 업데이트
        sudo apt update
        sudo apt upgrade -y

# 필수 의존성 설치
        sudo apt install -y cmake build-essential
        sudo apt install -y libopenblas-dev liblapack-dev
        sudo apt install -y libx11-dev libgtk-3-dev
        sudo apt install -y python3-dev python3-pip

# 사용 가능한 메모리 확인
        free -h
            # 8GB 모델: 문제없음 ☑
            # 4GB 모델: 스왑 메모리 증설 권장

# 스왑 메모리 증설 (4GB 모델만)
        sudo dphys-swapfile swapoff
        sudo nano /etc/dphys-swapfile

# CONF_SWAPSIZE=2048 로 변경
        sudo dphys-swapfile setup
        sudo dphys-swapfile swapon
```

2) dlib 설치

```
# --break-system-packages는 Bookworm에서 필요
        pip3 install dlib —break-system-packages
# 다른 터미널에서
        htop    # CPU 100% 사용 확인
# dlib 컴파일 실패 시 의존성 재설치
        sudo apt install --reinstall build-essential cmake python3-dev
        pip3 install dlib --break-system-packages --no-cache-dir
```

3) face_recognition 설치

pip3 install face-recognition --break-system-packages

4) 설치확인

```
~ python3
import cv2
import face_recognition
import dlib
print(f"☑ OpenCV: {cv2.__version__}")
print(f"☑ dlib: {dlib.__version__}")
print(f"☑ face_recognition: 설치 완료")
print("₩n모든 라이브러리 정상 설치!")
```

**** 통합 테스트 스크립트 ****

```
/home/cos/
├── code/
|    └── face_recognition_full_test.py  ← 여기에 파일
├── env/                                  # 가상환경
└── install_face_recognition.sh           # 설치 스크립트
```

** .sh 실행 (설치)

~/install_face_recognition.sh

→ dlib, face_recognition 설치됨

** .py 실행 (사용)

python3 ~/code/face_test.py

→ 설치된 것을 사용해서 얼굴 인식