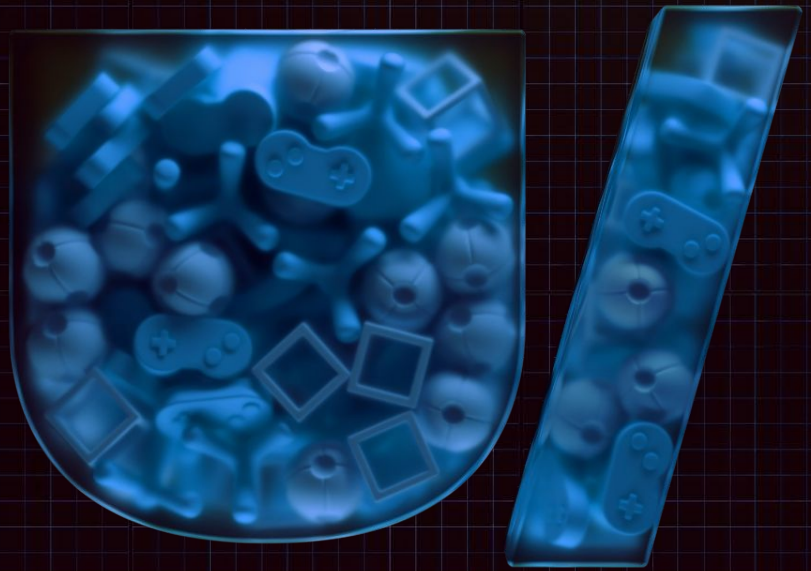
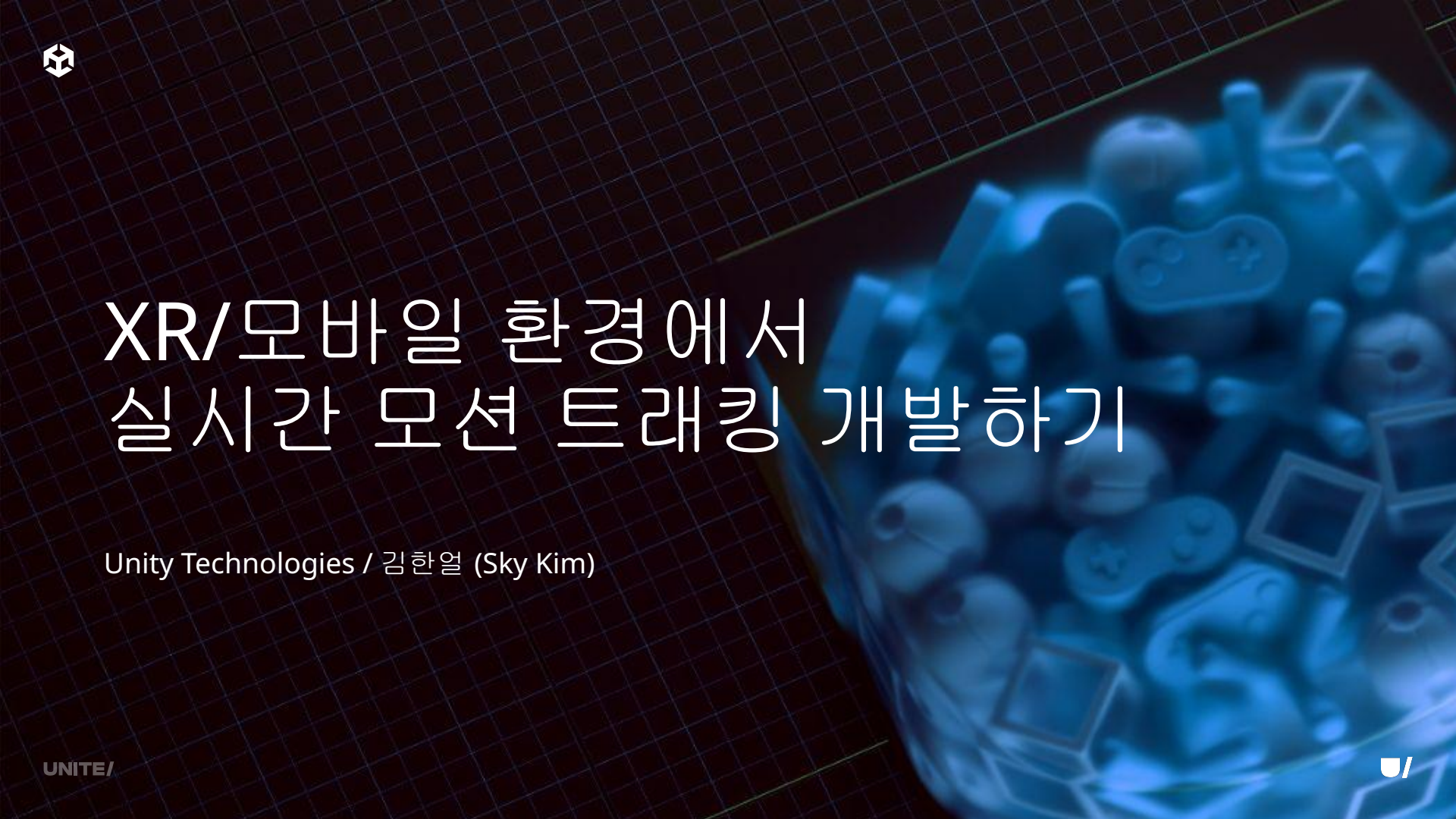
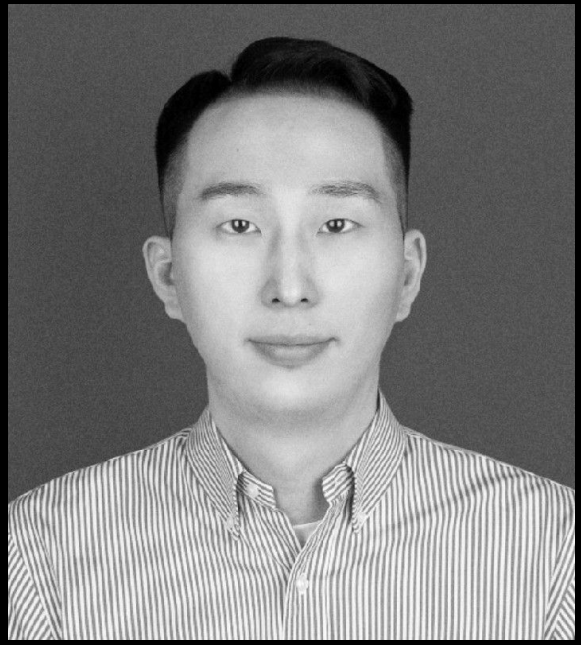




XR/모바일 환경에서 실시간 모션 트래킹 개발하기

Unity Technologies / 김한얼 (Sky Kim)

김한얼 (Sky Kim)

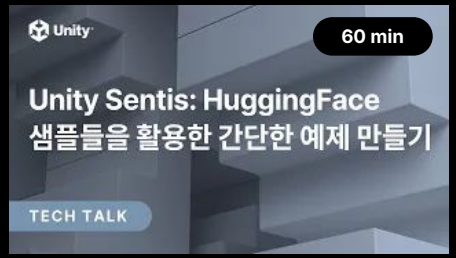


Senior Software Engineer, Unity
Engine Support Team in APAC
AI and Simulation Technical Support

→ Fun fact: Avid Hackathon participant
→ Contact: sky.kim@unity3d.com

지난 세션 소개

- → U/Day Seoul Game (2024)
  - — Unity Sentis 상세 기술 설명과 게임 콘텐츠 적용 튜토리얼
- → Monthly Talk (2024)
  - — Unity Sentis: Hugging Face 샘플들을 활용한 간단한 예제 만들기
- → U/Day Seoul Industry (2024)
  - — Unity Sentis: XR/모바일 기기의 온디바이스 AI 추론 및 최적화 방법
- → Unity 6와 시작하는 인공지능: 학습부터 시작하는 Sentis 사용기 (2024)
- → Unite Seoul 2025 (Today)
  - — XR/모바일 환경에서 실시간 모션 트래킹 개발하기



Agenda

→ Motion Tracking 기술의 발전
→ Mobile 환경에서의 Motion Tracking
→ XR 환경에서의 Motion Tracking
→ 나만의 데이터로 Gesture 인식 모델 만들기

Motion Tracking 기술의 발전

Motion Tracking 의 역사

2010년 이전: 마커 기반 광학식 모션 캡처, IMU 센서 모션캡처
→ Nintendo Wii, Xsens IMU

2010~2015년: RGB-D 센서 기반 Markerless 추적
→ Microsoft Kinect, Leap Motion

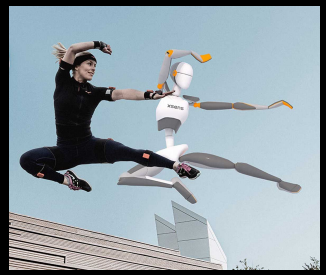
2015~2020: 딥러닝 기반 실시간 포즈 추정
→ ARKit(Apple), ARCore(Google), OpenPose(CMU)

2020~2025: AR/VR 시대, AI 센서 융합
→ Meta Quest, Apple Vision Pro, Mocopi(Sony), Move.ai 등

2025 이후: AI 글래스, XR 기반 Spatial Computing 대중화, Robotics 응용
→ Meta Orion

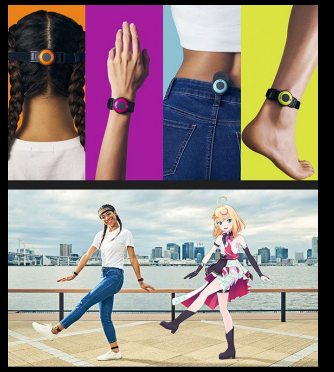
Nintendo Wii (2006)

OpenPose (2017)

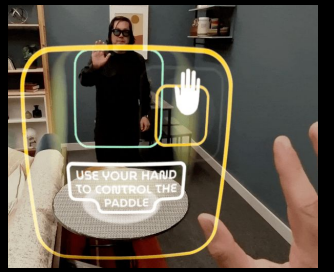


Xsens (2008)

Kinect (2011)



mocopi (2023)



Meta Orion (?)

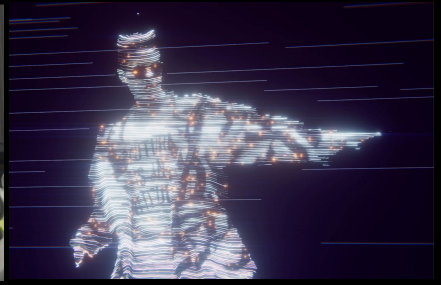
Motion Tracking 의 응용분야

엔터테인먼트(영화, 게임)
→ 영화 아바타, The Batman, 미디어 아트 분야에서 AI 모션캡처 활용
→ 게임 캐릭터 동작 생성

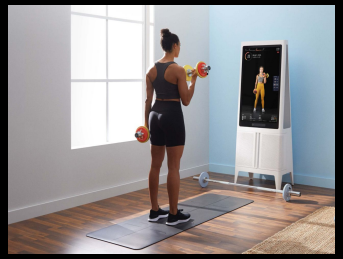
스포츠 및 의료 재활
→ 홈트레이닝용 스마트 피트니스 서비스(Tempo 등)
→ Kinect 활용 재활 운동 게임
→ AI 기반 걸음걸이 분석, 자세 교정 및 모니터링

메타버스 및 디지털 휴먼
→ VR 소셜 아바타 구현(VRChat), 버추얼 유튜버(Vtuber)
→ 아바타로 표정, 몸짓 실시간 반영

영화 아바타: 물의 길



Media Art (Keijiro)



Tempo

Neofect: Smart Glove

VR Chat

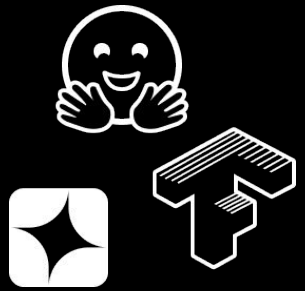
VTuber

Mobile 환경에서의 Motion Tracking

개발 환경

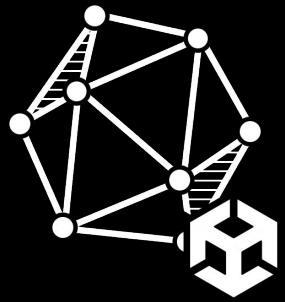
- → Unity 6000.0.40f1 (URP)
- → Sentis 2.1.2
- → 테스트 환경
  - — Samsung Galaxy S24+
  - — Meta Quest 3

AI 모델을 **Sentis** 에서 실행하는 과정



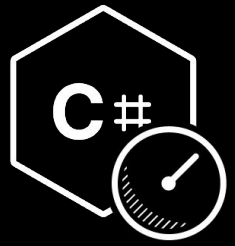
AI 모델 선택 후
ONNX 파일로 변환

→



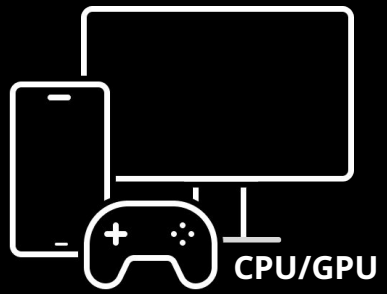
Unity로 임포트 및
모델 최적화

→



추론 코드작성

→



런타임 플랫폼에 배포
(Desktop, Console,
Mobile, WebGL,
WebGPU)

MediaPipe

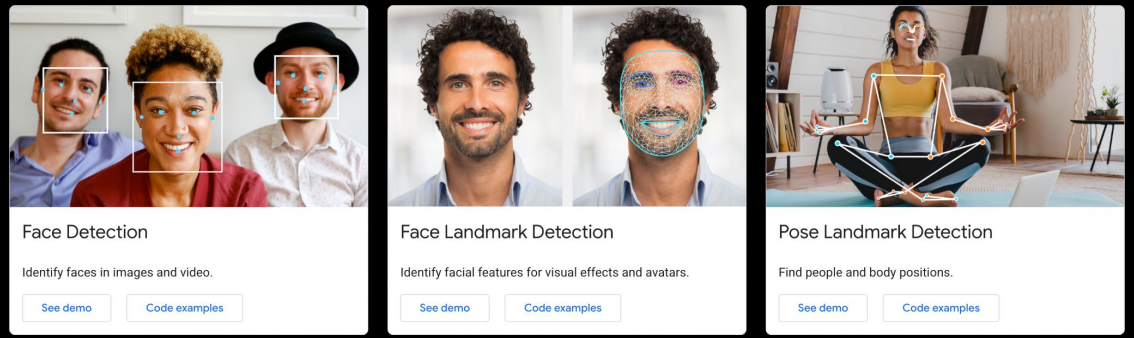
Google MediaPipe는 Vision, NLP, Audio 분야 등 다양한 Task에 대한 Pre-trained model을 제공 (Apache License 2.0)

→ **Vision examples**: Object Detection, Image Classification, Hand Landmark, Hand Gesture Recognition, Image Segmentation, Interactive Segmentation, Face Detection, Face Landmark Detection, Pose Landmark Detection, Image Embedding

→ **Text examples**: Text Classification, Text Embedding, Language Detection

→ **Audio examples**: Audio Classification

— https://ai.google.dev/edge/mediapipe/solutions/examples



MediaPipe 모델을 Sentis로 가져오기

→ MediaPipe 모델 중에서 적합한 Task 선택하여 .task 파일 다운로드하기 (Latest 클릭)

— https://ai.google.dev/edge/mediapipe/solutions/vision/face_landmarker

Model bundle	Input shape	Data type	Model Cards	Versions
FaceLandmarker	FaceDetector: 192 x 192 FaceMesh-V2: 256 x 256 Blendshape: 1 x 146 x 2	float 16	FaceDetector FaceMesh-V2 Blendshape	Latest

→ Task 파일 압축풀고, tflite 파일 얻기

— unzip face_landmarker.task

— face_detector.tflite, face_landmarks_detector.tflite, face_blendshapes.tflite 생성

→ Onnx 포맷으로 변경하기

— pip install tflite

— python -m tf2onnx.convert --opset 15 --tflite model.tflite --output model.onnx

→ 반드시 Model Card를 통해서 모델에 대해서 파악하기


Face Tracking
BlazeFace (Short Range) + FashMesh-V2 + Blendshape-V2

Face Tracking Overview

→ Face landmark detection guide

— https://ai.google.dev/edge/mediapipe/solutions/vision/face_landmarker

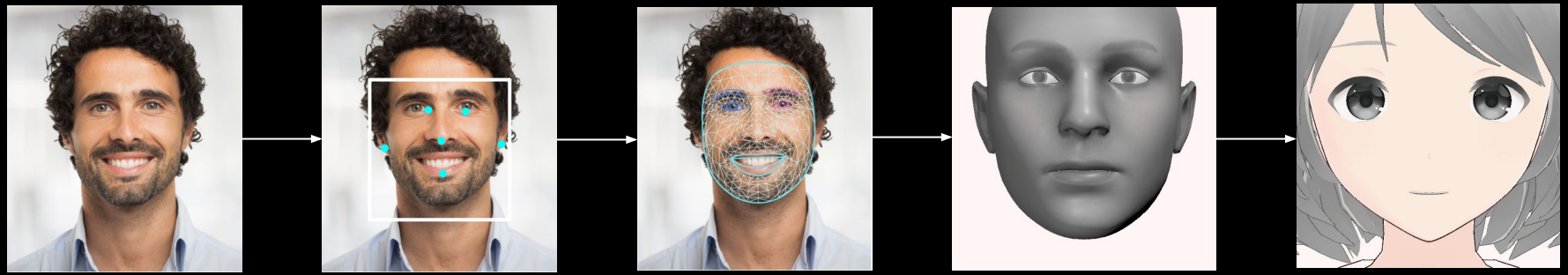


Image (RGB)

얼굴 영역 추출
(BlazeFace)

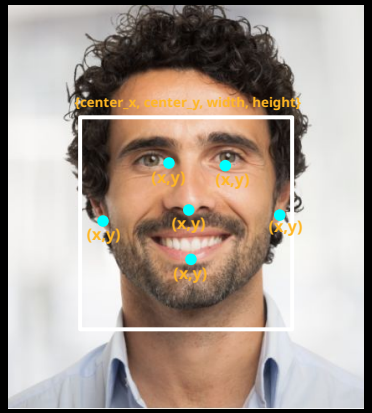
얼굴 랜드마크 추출
(FaceMesh)

Blendshape 추출
(Blendshape)

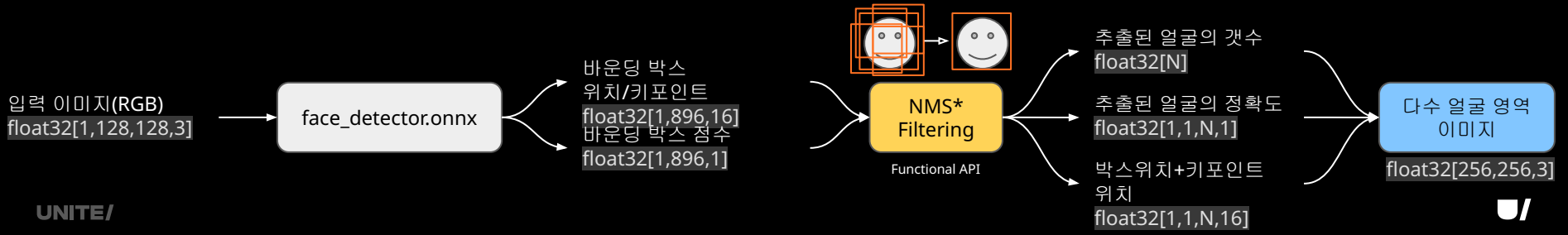
캐릭터 적용

MediaPipe BlazeFace (Short Range)

→ https://storage.googleapis.com/mediapipe-assets/MediaPipe BlazeFace Model Card (Short Range).pdf



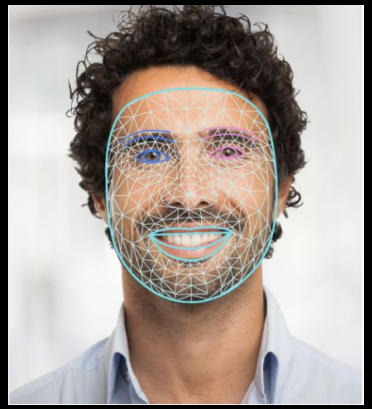
	설명
Details	스마트폰 전면 카메라 이미지에 최적화된 이미지에서 복수의 얼굴 경계박스와 키포인트를 추출
Architecture	SSD (Single Shot Multibox Detector) like with a custom encoder (CNN)
Model File	face_detector.onnx (418KB)



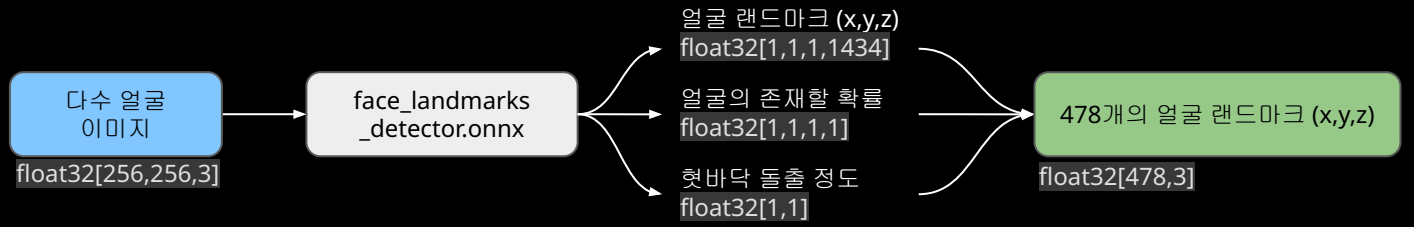
*NMS (Non-Maximum Suppression): 겹치는 여러개의 예측된 바운딩 박스 중에서 신뢰도가 가장 높은 하나의 결과만 남기고 나머지를 제거하여 중복

MediaPipe FaceMesh

→ https://storage.googleapis.com/mediapipe-assets/Model Card MediaPipe Face Mesh V2.pdf

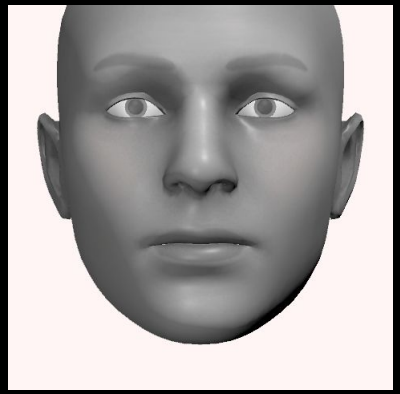


	설명
Details	스마트폰 전면 카메라의 단일 영상에서 478개의 3D 얼굴 랜드마크를 실시간으로 예측
Architecture	MobileNetV2-like with customized blocks for real-time (CNN)
Model File	face_landmarks_detector.onnx (4.9MB)

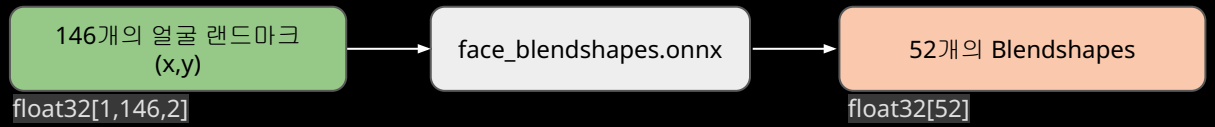


MediaPipe Blendshape V2

→ https://storage.googleapis.com/mediapipe-assets/Model Card Blendshape V2.pdf



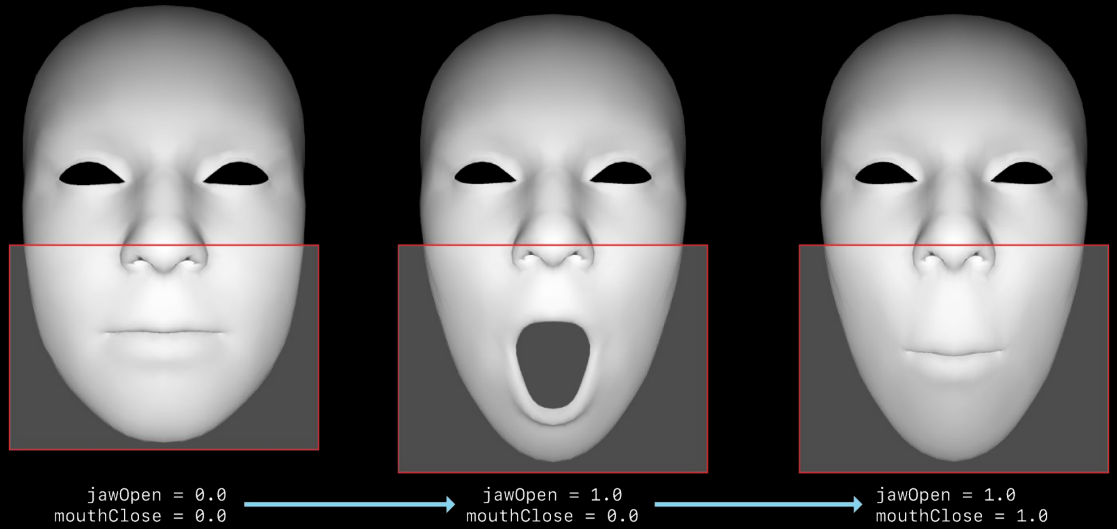
	설명
Details	FaceMesh의 146개 얼굴 랜드마크를 입력받아 ARKit 52개 blendshape를 실시간으로 예측
Architecture	MLP-Mixer (CNN)
Model File	face_blendshapes.onnx (1.9MB)



ARKit 52 Blendshapes

→ Apple ARKit 52 blendshapes

- Brow, Cheek, Eyes, Jaw, Mouth, Nose, …
- https://developer.apple.com/documentation/arkit



browDownLeft
browDownRight
browInnerUp
browOuterUpLeft
browOuterUpRight
cheekPuff
cheekSquintLeft
cheekSquintRight
eyeBlinkLeft
eyeBlinkRight
eyeLookDownLeft
eyeLookDownRight
eyeLookInLeft
eyeLookInRight
eyeLookOutLeft
eyeLookOutRight
…


Pose Tracking
BlazePose Pose Detector + BlazePose GHUM 3D + Retargeting

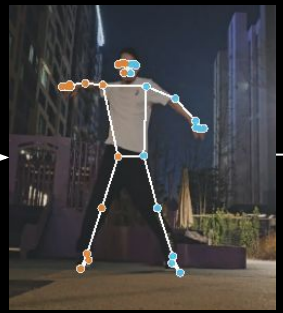
Pose Tracking Overview

→ Pose landmark detection guide

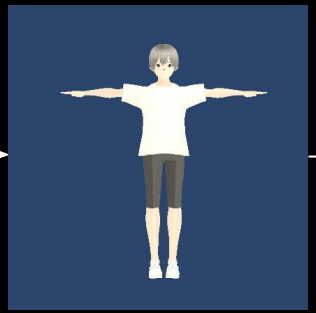
— https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker

Image (RGB)

바디 영역 추출
(Pose Detector)



바디 랜드마크 추출
(GHUM 3D)



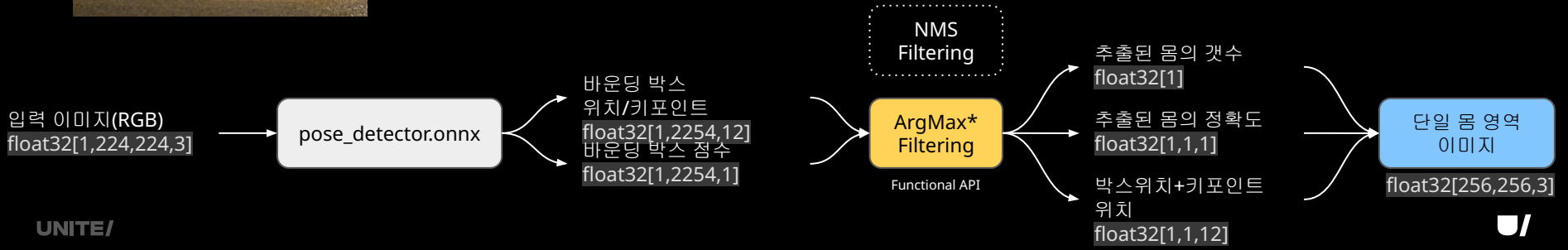
Retargeting

캐릭터 적용

MediaPipe BlazePose Pose Detector

https://storage.googleapis.com/mediapipe-assets/Model Card BlazePose GHUM 3D.pdf

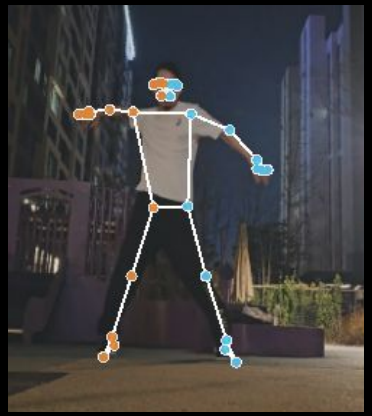
	설명
Details	모바일이나 웹캠 영상 이미지에서 몸 영역의 경계박스와 확률을 추출
Architecture	SSD (Single Shot Multibox Detector) like with a custom encoder (CNN)
Model File	pose_detector.onnx (15.1MB)



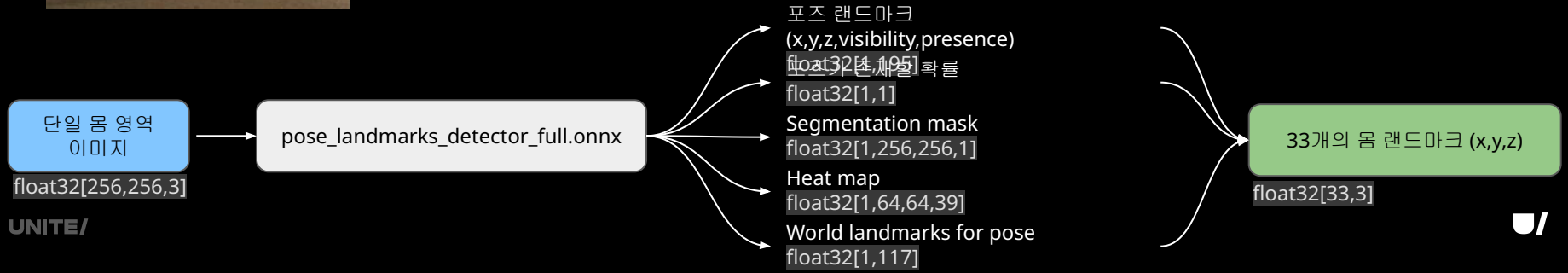
*ArgMax: 주어진 배열 또는 함수 내에서 최대값을 가지는 원소의 인덱스를 반환하는 연산

MediaPipe BlazePose GHUM 3D

https://storage.googleapis.com/mediapipe-assets/Model Card BlazePose GHUM 3D.pdf



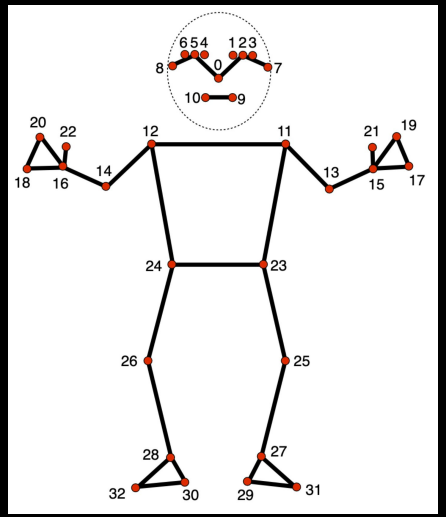
	설명
Details	단일 이미지를 통해 인물의 전체 신체 자세의 33개 3D 랜드마크를 실시간 추출
Architecture	MobileNetV2-like with customized blocks for real-time performance (CNN)
Model File	pose_landmarks_detector_full.onnx (12.8MB)



Retargeting and Inverse Kinematics (IK)

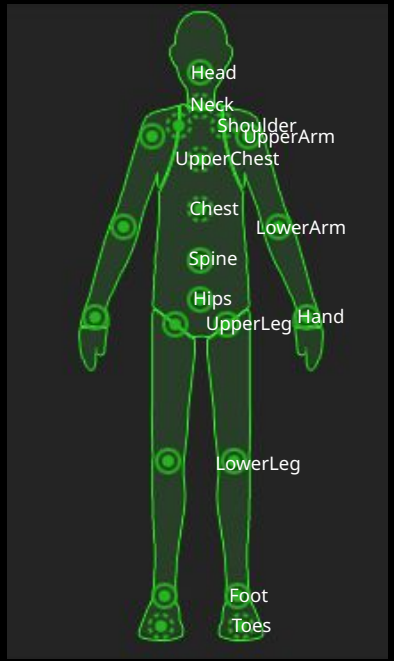
→ Retargeting from BlazePose 33 points 3D landmark to Unity Humanoid Bone

→ Calculate inverse kinematics for every bone



0 - nose
1 - left eye (inner)
2 - left eye
3 - left eye (outer)
4 - right eye (inner)
5 - right eye
6 - right eye (outer)
7 - left ear
8 - right ear
9 - mouth (left)
10 - mouth (right)
11 - left shoulder
12 - right shoulder
13 - left elbow
14 - right elbow
15 - left wrist
16 - right wrist
17 - left pinky
18 - right pinky
19 - left index
20 - right index
21 - left thumb
22 - right thumb
23 - left hip
24 - right hip
25 - left knee
26 - right knee
27 - left ankle
28 - right ankle
29 - left heel
30 - right heel
31 - left foot index
32 - right foot index

Head
LeftEye
RightEye
Neck
Spine
Hips
RightUpperArm
RightLowerArm
RightHand
RightThumbIntermediate
RightMiddleProximal
LeftUpperArm
LeftLowerArm
LeftHand
LeftThumbIntermediate
LeftMiddleProximal
RightUpperLeg
RightLowerLeg
RightFoot
RightToes
LeftUpperLeg
LeftLowerLeg
LeftFoot
LeftToes



Hand Tracking

Hand Detector + Hand landmarker + Gesture Embedder + Gesture Classifier

Hand Tracking Overview

→ Hand landmarks detection guide

— https://ai.google.dev/edge/mediapipe/solutions/vision/hand_landmarker

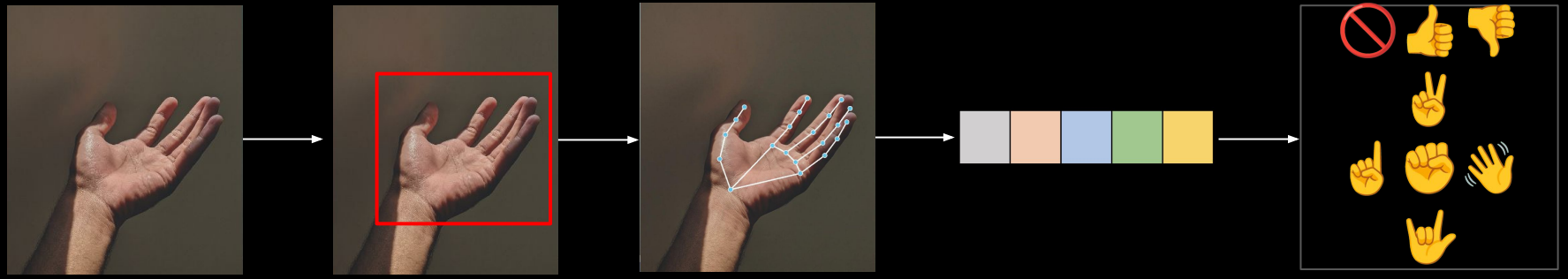


Image (RGB)

손 영역 추출
(Hands Detector)

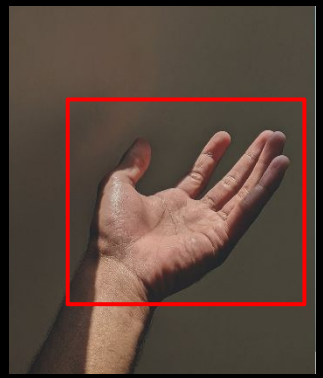
손 랜드마크 추출
(Hand Tracker)

손 임베딩 텐서 생성
(Gesture Embedding)

손 제스쳐 인식
(Gesture Classifier)

MediaPipe Hand Detector

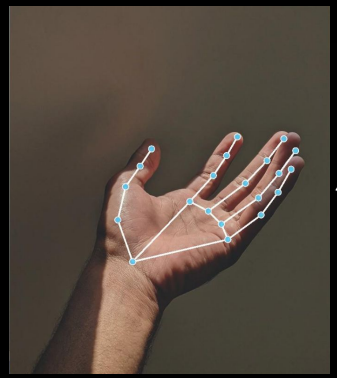
https://storage.googleapis.com/mediapipe-assets/Model Card Hand Tracking (Lite_Full) with Fairness Oct 2021.pdf

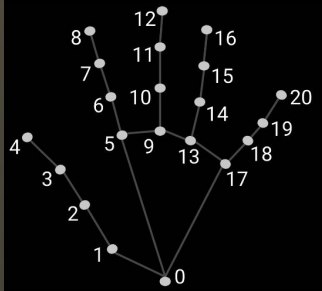


	설명
Details	모바일이나 웹캠 영상 이미지에서 손 영역의 경계박스와 확률을 추출
Architecture	SSD (Single Shot Multibox Detector) like with a custom encoder (CNN)
Model File	hand_detector.onnx (4.6MB)

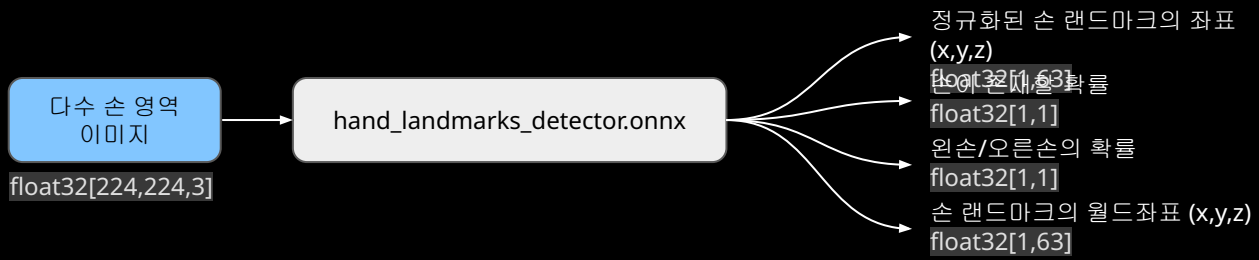
MediaPipe Hand Tracker

https://storage.googleapis.com/mediapipe-assets/Model Card Hand Tracking (Lite_Full) with Fairness Oct 2021.pdf





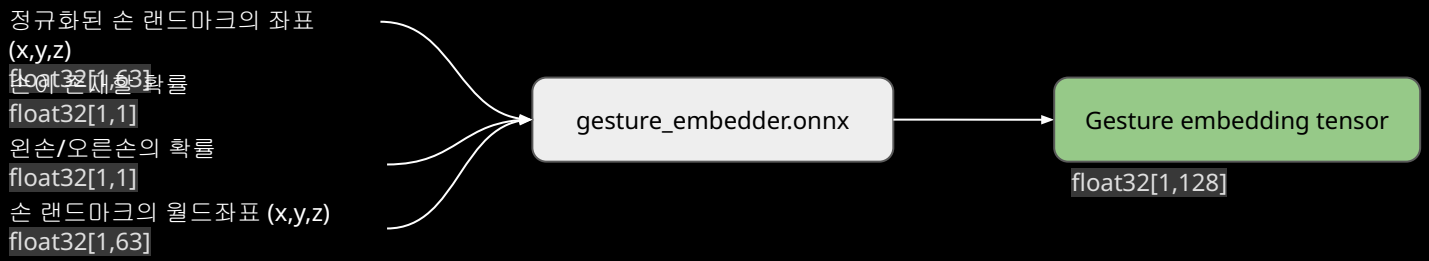
	설명
Details	단일 이미지를 통해 손의 21개 3D 랜드마크를 실시간 추출
Architecture	Regression model (CNN)
Model File	hand_landmarks_detector.onnx (10.9MB)



MediaPipe Gesture Embedding

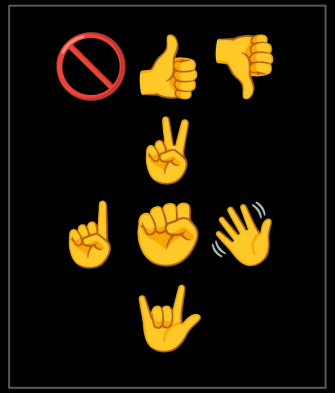
https://storage.googleapis.com/mediapipe-assets/gesture_recognizer/model_card_hand_gesture_classification_with_faireness_2022.pdf

	설명
Details	21개의 3차원 손의 랜드마크를 1x128 크기의 임베딩 텐서로 생성
Architecture	Fully Connected Neural Network with residual blocks
Model File	gesture_embedder.onnx (546KB)

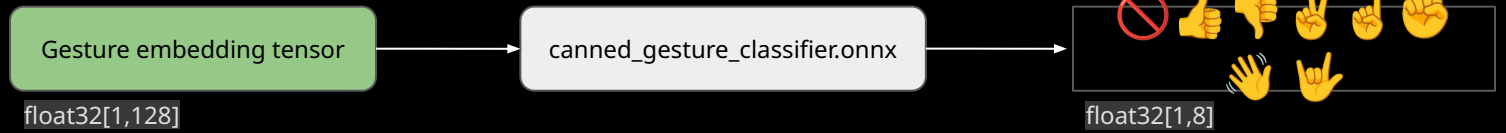


MediaPipe Gesture Classification

https://storage.googleapis.com/mediapipe-assets/gesture_recognizer/model_card_hand_gesture_classification_with_faireness_2022.pdf
https://ai.google.dev/edge/mediapipe/solutions/customization/gesture_recognizer



	설명
Details	128차원의 임베딩 텐서를 이용해서 8개 제스처의 확률을 계산
Architecture	Classification model (Fully Connected Neural Network)
Model File	canned_gesture_classifier.onnx (6KB)




Right
Victory
0.8893496


Multi-Person Tracking
Yolo-Pose

Multi-Person Tracking Overview

→ Ultralytics Yolo Pose Estimation

— https://github.com/ultralytics/ultralytics/blob/main/docs/en/tasks/detect.md

BlazePose 사용시

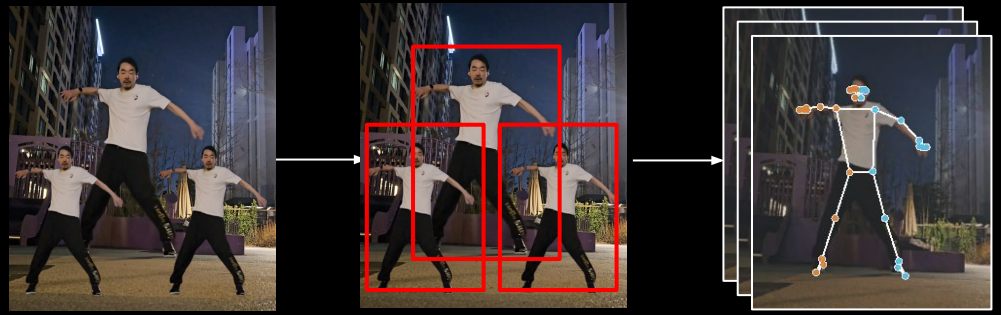


Image (RGB)

바디 영역 추출
(Pose Detector)

바디 3D 랜드마크 추출
(GHUM 3D)

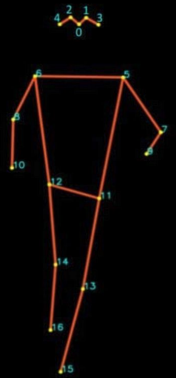
Yolo-Pose 사용시

Image (RGB)

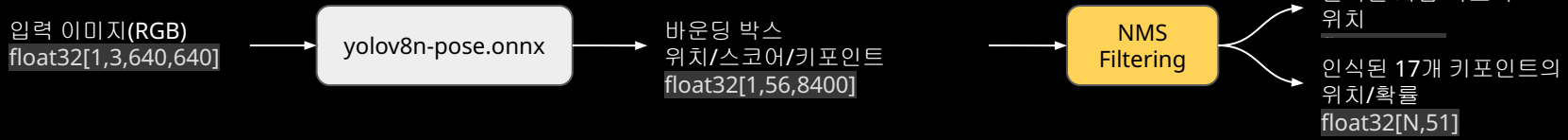
바디 2D 랜드마크 추출
(Yolo-pose)

Yolov8n-Pose

https://huggingface.co/Ultralytics/YOLOv8



	설명
Details	사람 전신 영역에 대한 바운딩 박스와 17개의 2D Pose Landmark를 실시간 동시 추출
Architecture	Backbone, Neck, Head (CNN기반)
Model File	yolov8n-pose.onnx (13.5MB)



3 people

Mobile 환경에서 Sentis 최적화 꿀팁 3대장

모델과 환경에 맞는 Backend Type 설정하기

→ CPU: Burst
→ GPUCompute: Compute shader
→ GPUPixel: Pixel shader

Async/Await으로 Main Thread Blocking 방지하기

→ ReadbackAndClone()
→ await ReadbackAndCloneAsync()

ScheduleIterable로 Layer를 여러 개로 나눠서 실행하기

→ ScheduleIterable
→ Schedule.MoveNext()

→ https://docs.unity3d.com/Packages/com.unity.sentis@2.1/api/Unity.Sentis.BackendType.html?q=backend
→ https://docs.unity3d.com/Packages/com.unity.sentis@2.1/manual/read-output-async.html
→ https://docs.unity3d.com/Packages/com.unity.sentis@2.1/manual/split-inference-over-multiple-frames.html

XR 환경에서의 Motion Tracking

Meta Quest 3: Body Tracking

→ Face Tracking: 카메라 기반 (Quest Pro만 지원), 음성 기반 (Quest 3/3S/Pro 지원)

→ Body Tracking: 3-Point vs. IOBT (Inside-Out Body Tracking) with Generative Legs (Quest 3/3S/Pro 지원)

→ Eye Tracking: 카메라 기반 (Quest Pro만 지원)

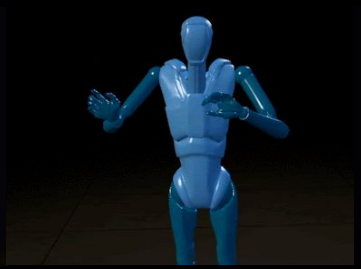
→ Hand Tracking: XR Hands (Unity Package)

- **Movement SDK**: https://github.com/oculus-samples/Unity-Movement
- **XR Hands**: https://docs.unity3d.com/Packages/com.unity.xr.hands@1.5/manual/index.html

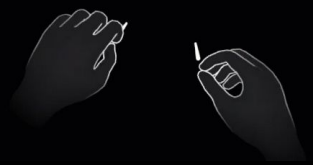
Face Tracking

3-Point vs. IOBT

Body Tracking (Upper body)



Body Tracking (Full body)



Hand Tracking

(Demo) Full body Tracking Demo

→ Meta Quest 3 + Movement SDK

→ Import Mixamo Character (https://www.mixamo.com)

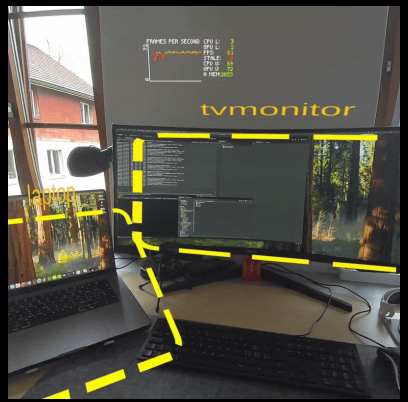
→ GameObject > Movement Samples > Body Tracking > Animation Rigging Retargeting (fullbody)

(my nephew's posture)

Meta Quest 3: Passthrough Camera API

→ Meta Quest의 카메라 접근 권한 허용

- Meta Quest 3/3s, HorizonOS: v74 이상
- Frame Rate: 30fps / Image latency: 40-60ms
- Available resolutions per eye: 320x240, 640x480, 800x600, 1280x960
- Sentis를 이용한 이미지 처리 적용 가능*
- https://github.com/oculus-samples/Unity-PassthroughCameraApiSamples

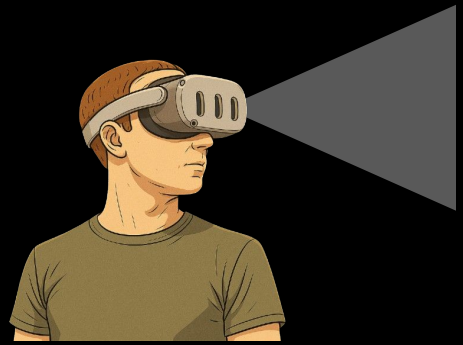


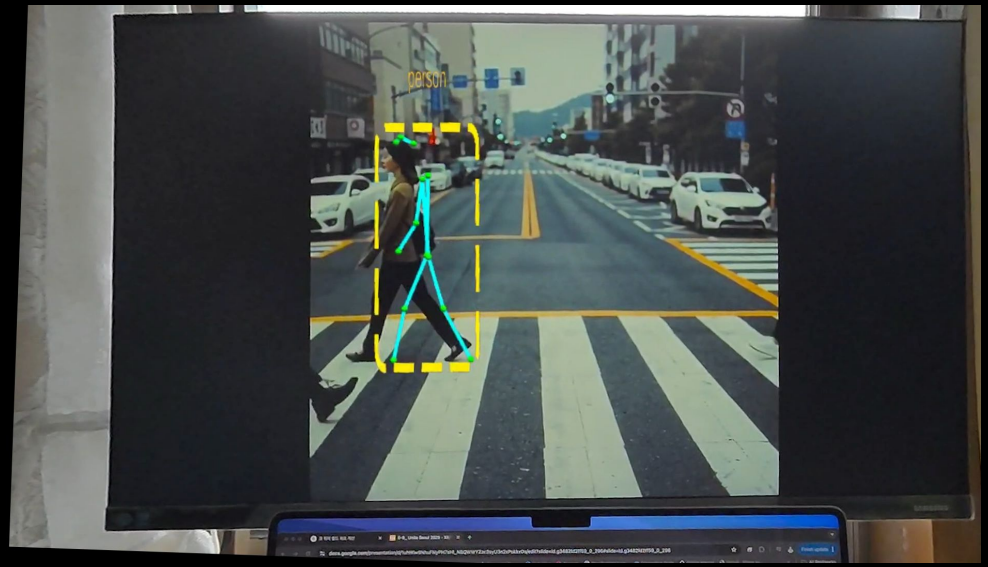
(Demo) Passthrough Human-pose Tracking Demo

→ Meta Quest 3 + Passthrough Camera API

→ Yolov8n-pose: Multi-person tracking model (Sentis)

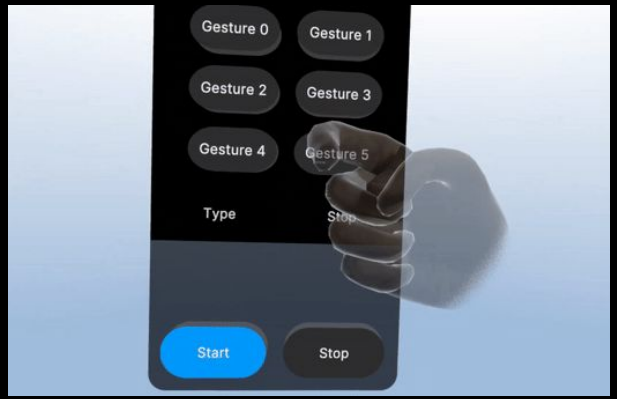
→ Bounding Box + Pose Tracking

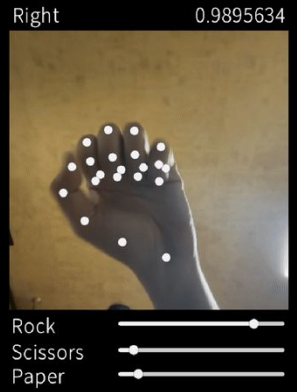




나만의 데이터로 Gesture 인식 모델 만들기

Gesture Classification Overview





Meta Quest에서 수어 인식하기

XR Hands의 keypoint를 이용해서 커스텀 수어 인식 만들기 (MLP)

https://github.com/skykim/XR-Gesture-Detection

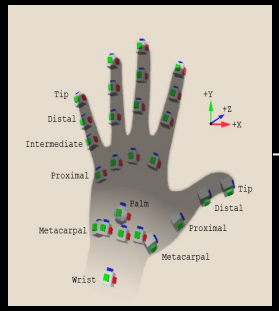
Mobile 장비에서 손 제스처 인식하기

Blaze Hand의 keypoint를 이용해서 묵/찌/빠 인식 만들기 (MLP)

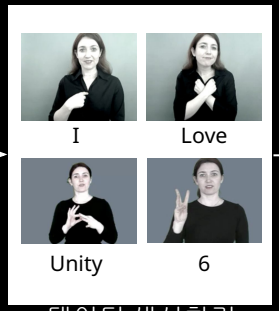
https://github.com/skykim/202407_TechTalk_SentisDemo

Meta Quest 에서 수어 인식하기

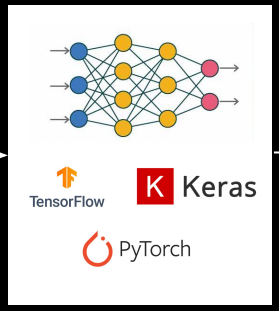
→ XR Hands의 keypoint를 이용해서 커스텀 수어 인식 만들기 (MLP)
→ Tutorial: https://github.com/skykim/XR-Gesture-Detection



데이터 형태 선정하기
(XR Hands Keypoints)



데이터 생성하기
(데이터셋 활용 또는 직접 캡쳐)

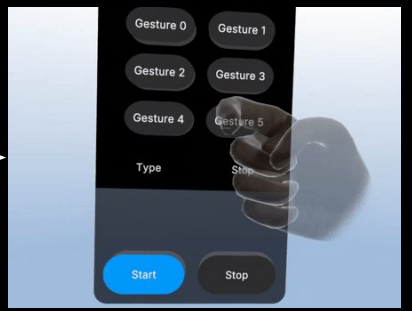


MLP 모델 학습하기
(Tensorflow, Keras, Pytorch)

tf2onnx

torch.onnx

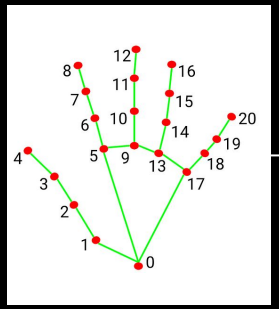
ONNX로 변환하기



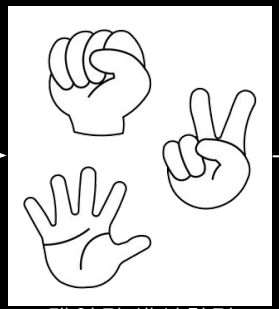
Sentis에서 실행하기

Mobile 장비에서 손 제스처 인식하기

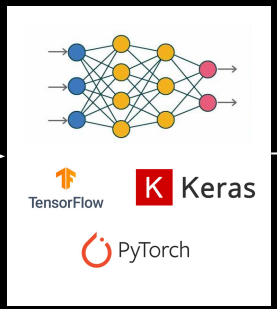
→ Blaze Hand의 keypoint를 이용해서 묵/찌/빠 인식 만들기 (MLP)
→ Tutorial: https://github.com/skykim/202407_TechTalk_SentisDemo



데이터 형태 선정하기
(BlazeHands Keypoints)



데이터 생성하기
(데이터셋 활용 또는 직접 캡쳐)

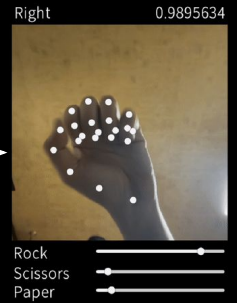


MLP 모델 학습하기
(Tensorflow, Keras, Pytorch)

tf2onnx

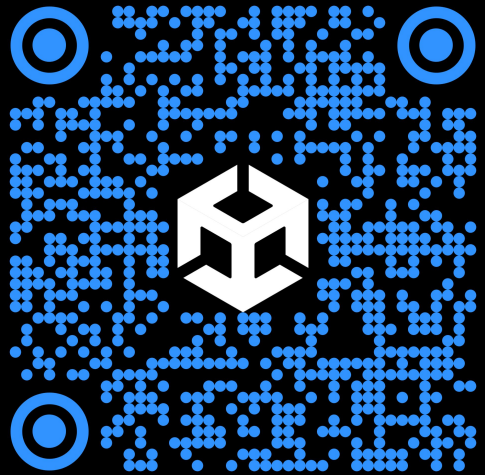
torch.onnx

ONNX로 변환하기



Sentis에서
실행하기

Sample Code



https://github.com/skykim/uniteseoul2025-motiontracking

Thank you