Live Perception for Mobile and Web

Google Research
MediaPipe

Google Research

MediaPipe

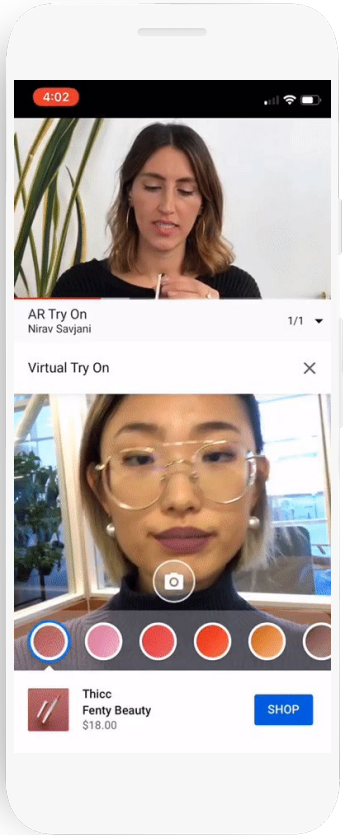
Live Perception: Viewfinder ML

Live Perception = ML on-device + in real-time + with low-latency

Applications: Virtual try-on, AR Effects, Smart framing, Device control, Smart cameras & mirrors

Benefits: Device-local, connection-free, Privacy-conscious

Immediate, Create-in-viewfinder, Enables actions/control

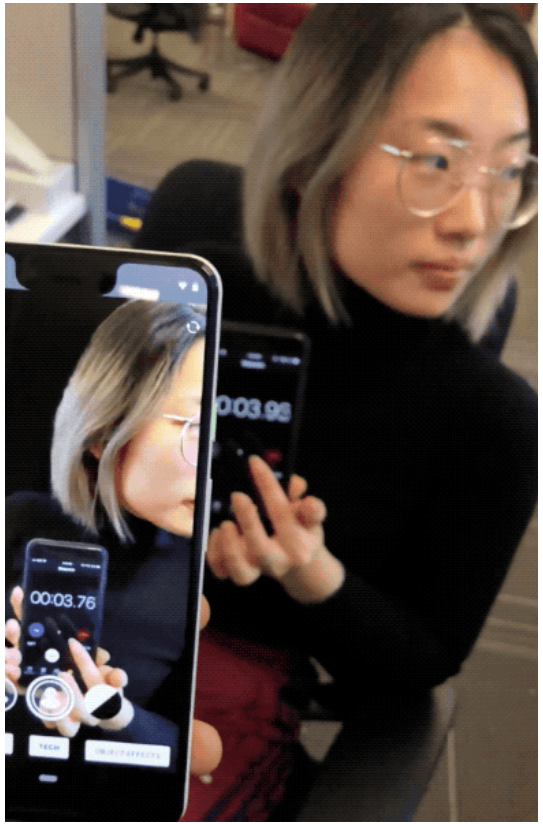


Google Research

MediaPipe

Live Perception: Technical challenges

- **Model constraints:** ML inference in < 20 ms
  - Low-capacity networks => domain specific
  - Follow 5 Core-Recipes for Live Perception
- **Critical cross-platform infrastructure:**
  - **MediaPipe** : ML solutions: Pipeline + GPU support + synchronization
  - **XNNPack** : CPU accelerated inference on mobile/web
  - **GPU delegate TF-Lite:** GPU-accelerated inference on mobile
- **Synchronization** : Delayed Viewfinder (~100-200 ms)



Real-world to viewfinder latency measured with 3 phones

Today's Talk

01 Proven ML solutions for Live Perception

- Face Meshes and Iris Tracking
- Hand skeletons and gestures
- Human Pose
- 2D Tracking for arbitrary objects
- 3D Object Detection

02 Infrastructure for Live Perception

- MediaPipe
- CPU and GPU-accelerated ML-inference

01 ML Solutions for Live Perception

Core-Recipes for Live Perception

1

ML Pipelines

ML pipelines with coupled, task-specific models vs. monolithic model

2

Reduce Augmentations

Coupled models => limited input domain => reduce augmentations => more capacity for accurate predictions

3

Reduce outputs

Reduce output layers: Regression over Heatmaps

4

Reduce Label noise

Use high-quality, consistent annotations => 30-50K GT samples are sufficient

5

Synth. Data for 3D

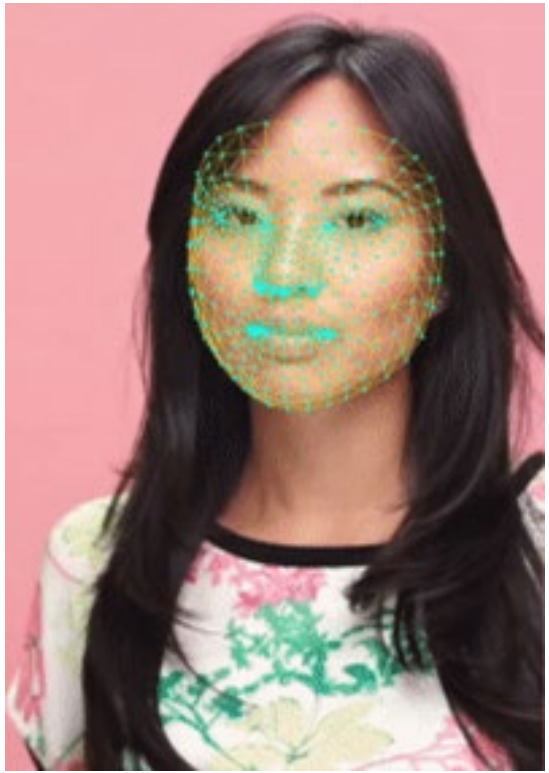
Unlock 3D via synth. data and multi-task training

Proven ML solutions

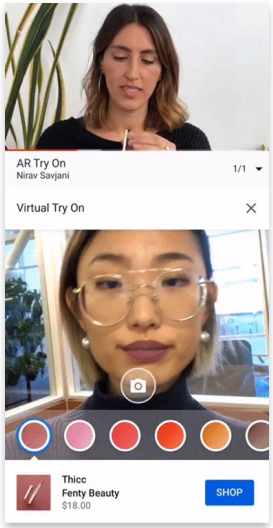
1. **Face Meshes and Iris Tracking: Powering Duo Effects, YouTube Stories, AR Ads** -
2. Hand skeletons and gestures
3. Human Pose
4. 2D Tracking: Powering Lens, Google Translate
5. 3D Object detection

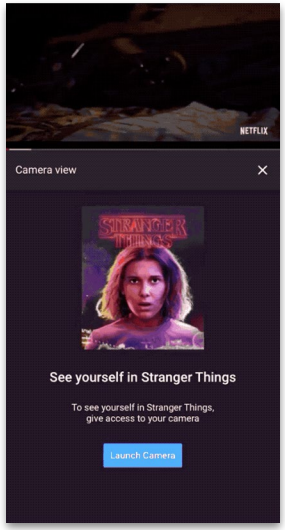
1) MediaPipe FaceMesh

- Powering YouTube Stories, AR-Ads, Duo Effects, MLKit Face Contour, ARCore
- Paper, AIblog, ModelCard, Web Page
- Difficult problem:
  - Predicting face geometry without use of dedicated depth sensor
  - Heatmap approaches don't scale to 480+ points
  - AR Makeup demands high-accuracy alignment
- Accelerated via TFLite GPU + XNNPack
- Available to devs in MediaPipe, TF.js, MLKit, ARCore



FaceMesh - Launches





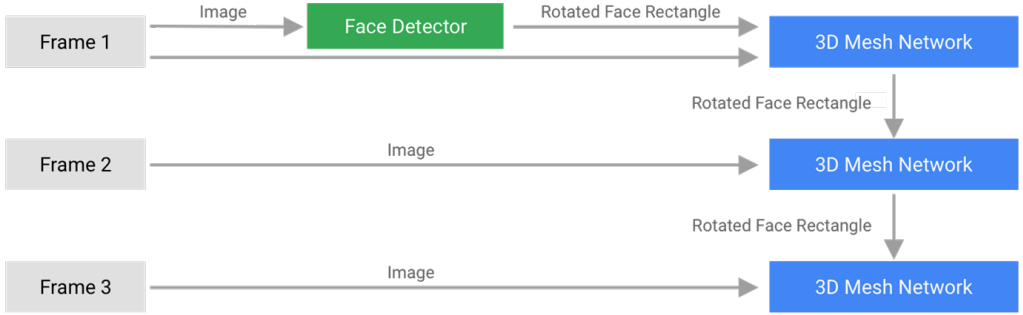
AR Ads

Duo

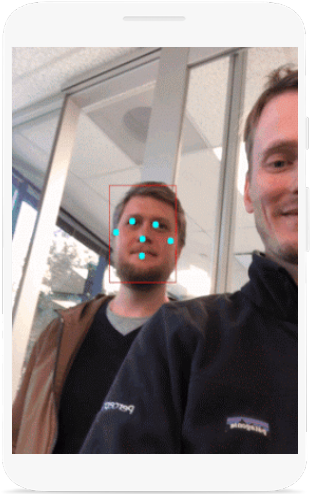
YT Stories

MediaPipe FaceMesh: ML Pipeline

- Two ML models:
  MediaPipe pipeline: **BlazeFace Detector** + **3D FaceMesh network**
  - **BlazeFace** returns oriented ROI
  - **3D Mesh network** predicts mesh from ROI crop
- **FaceMesh** model to re-use ROI from prev. frame
- **BlazeFace** only runs on 1st frame or when mesh indicates face miss



BlazeFace: Sub-millisecond Face Detection



- Predicts face rectangle + 6 landmarks
- Optimized for selfie uses / mobile use cases
- Leverages tflite GPU, XNNPack for tflite CPU and tf.js
- Paper, Web Page
- Powers all FaceMesh applications

Model	Average Precision	Inference Time, ms (iPhone XS)
MobileNetV2-SSD	97.95%	2.1
Ours	**98.61%**	**0.6**

Table 1. Frontal camera face detection performance

Device	MobileNetV2-SSD, ms	Ours, ms
Apple iPhone 7	4.2	**1.8**
Apple iPhone XS	2.1	**0.6**
Google Pixel 3	7.2	**3.4**
Huawei P20	21.3	**5.8**
Samsung Galaxy S9+ (SM-G965U1)	7.2	**3.7**

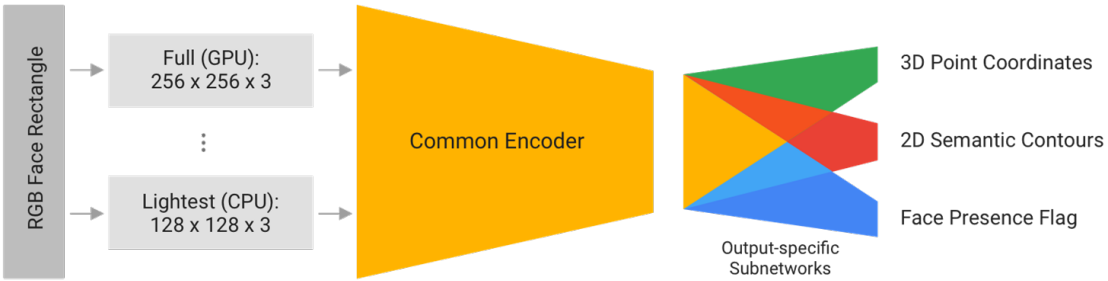
Table 2. Inference speed across several mobile devices

BlazeFace: Developer release

- Available to developers in MediaPipe and TF.js
- Dev response (Octi):
  "*My experience with BlazeFace has been really positive[..] **big leap in performance compared to Apple's face detector***."
- Model Card: ML Fairness evaluated across geo regions

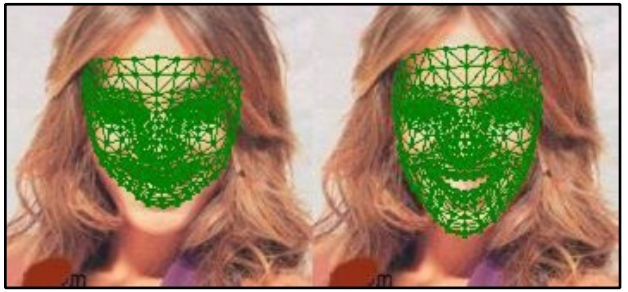
FaceMesh: Design Elements

- Predicts 486 3D landmarks from ROI cropped+rotated face detection
- Uses regression over heatmaps
- Tight detector coupling:
  - Limits cropping augmentation
  - Predicts face presence in ROI crop -> trigger detection
- Trained on 3 datasets: 2D contours + Real world mesh + synthetic mesh
- Paper

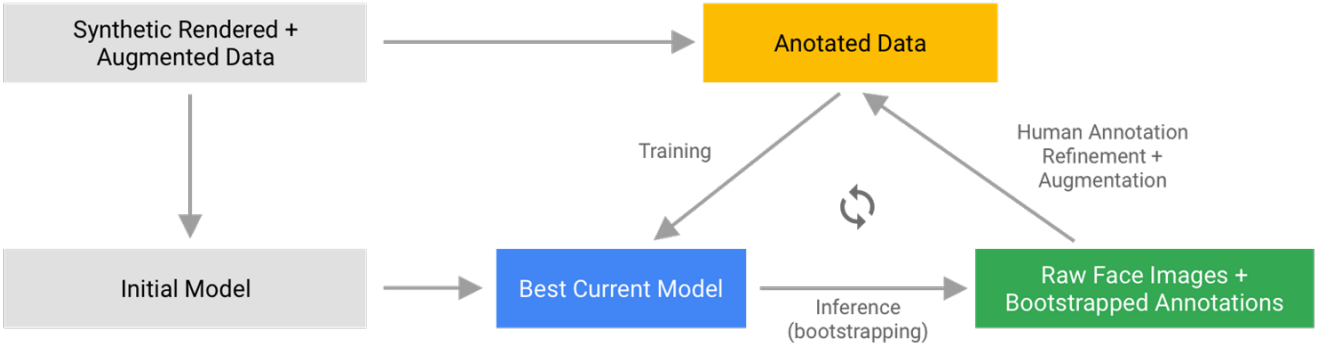


FaceMesh: Annotation challenge

- Annotating high-fidelity landmarks from scratch is hard!
- Bootstrap approach:
  a) Train initial model from 2D contours + synth. data
  b) Refine bootstrapped annotations w/ humans in the loop

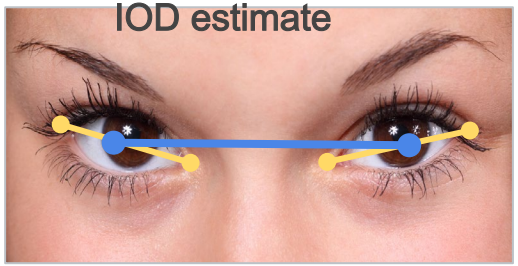


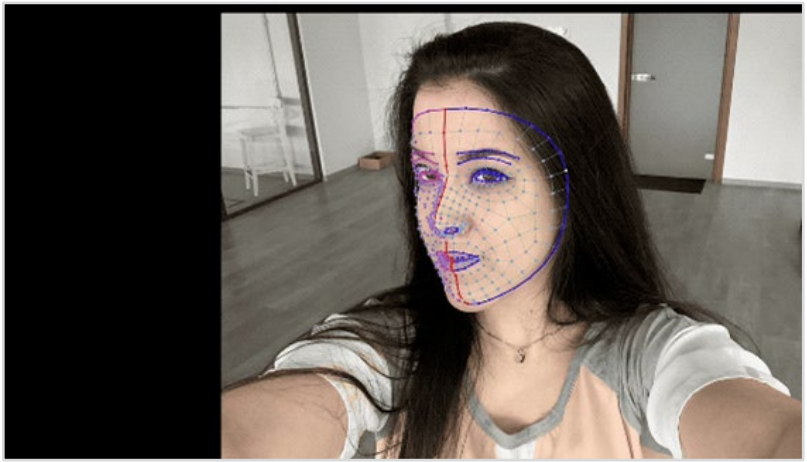
Synth data only 2D contours + synth



FaceMesh: Real world data

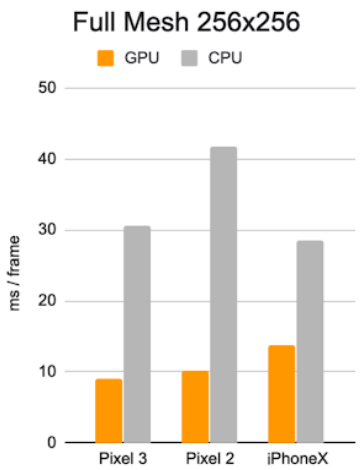
- Refinement of face contours via custom tool
- Build up dataset of 30K GT images
- Human annotation discrepancy:
  2.56% IOD - Interocular distance
- ~5 min per image

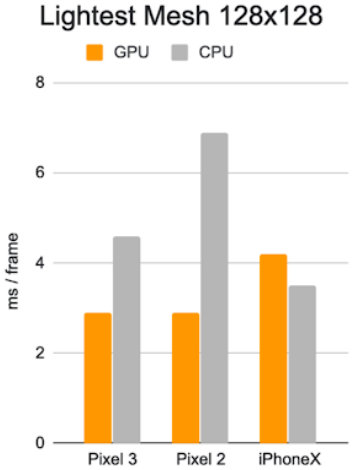




FaceMesh: Performance

- Two models: Full and lite
- Accuracy similar to humans (2.56%),
  - Ranging from 2.44% - 3.5% IOD across regions





ML Fairness

Test datasets have equal representation of 17 geographical regions, as defined by UN

We consider a model to be performing unfairly across representative subsets of the overall population **if the error range on them spans more than the human annotation discrepancy**

Human discrepancy: 2.56% IOD MAE

FaceMesh range 2.44% (N Africa) - 3.5% (N America)
Delta: 1.06 % << Human discrepancy

Results published in [MediaPipe FaceMesh Model Cards]

Evaluation Regions

Google Research

MediaPipe

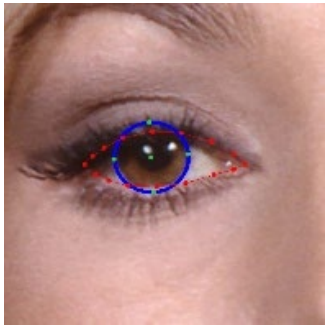
Face Mesh: AR Makeup

- Simulate realistic make-up via image filtering
- Paper
- Obtain accurate lip region from Face Mesh
- a) Divide light spectrum into several ranges,
  b) filter each range individually,
  c) apply virtual material
  d) merging and blend with original image.
- 3 Ranges: Mid-tones, Shadows, Highlights
- Highly realistic: 46% of AR images considered real by user study

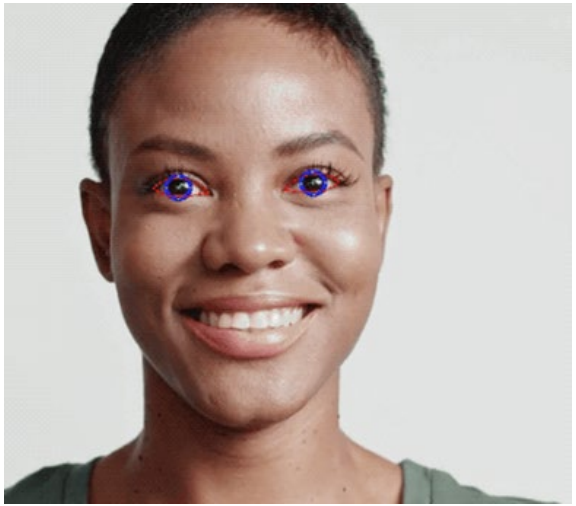
Actual \ Predicted	Real	AR
Real	62.38%	37.72%
AR	46.43%	53.57%

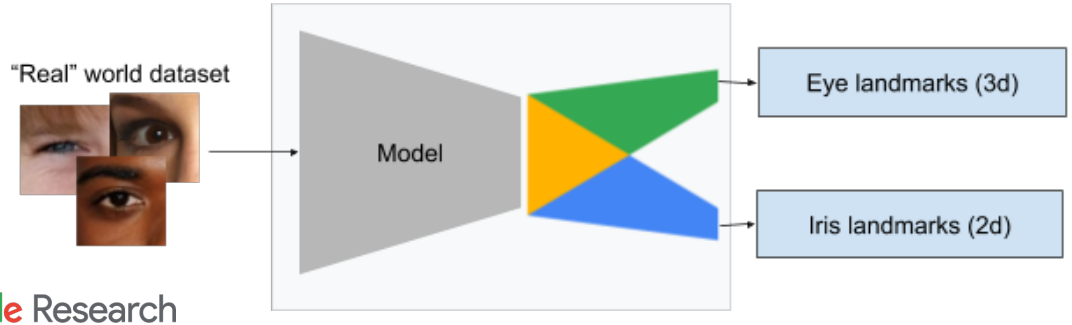
Table 1. Aggregated user survey results.

MediaPipe Iris



- Extends FaceMesh with 5 landmarks per eye
- Uses real world dataset of ~50K images
- Web Page
- Model card



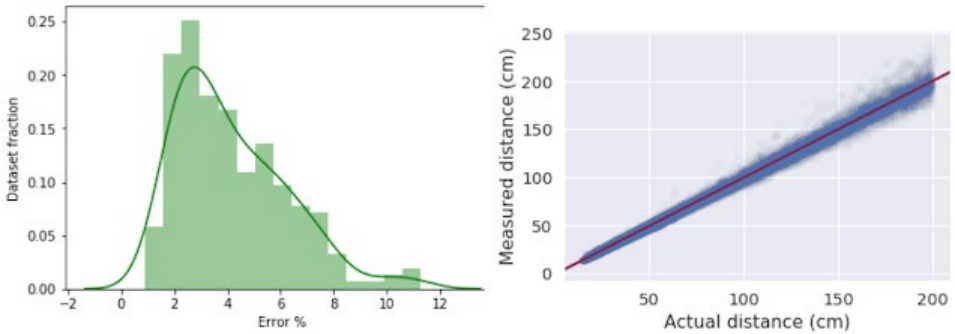


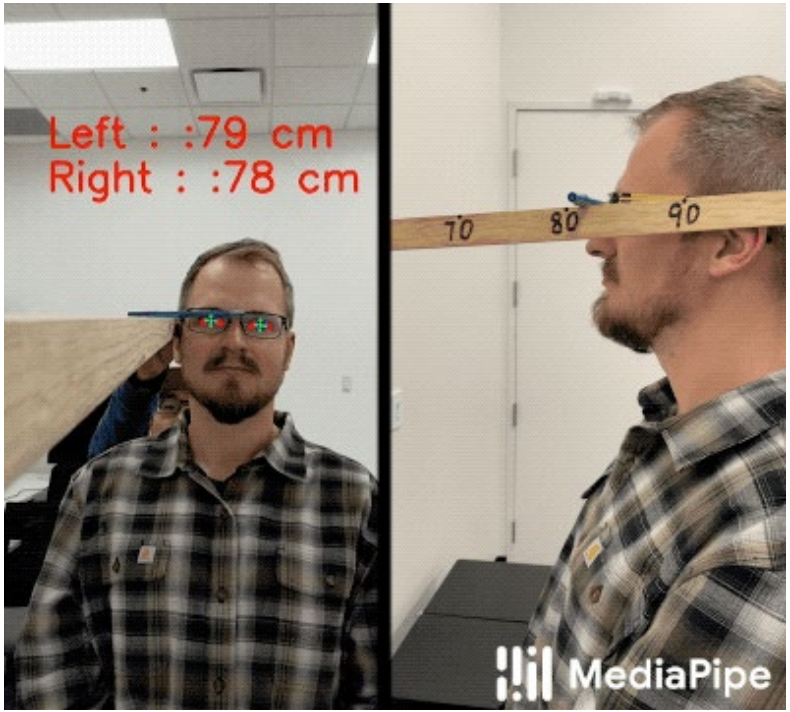
Google Research

MediaPipe

MediaPipe Iris : Single Image Depth

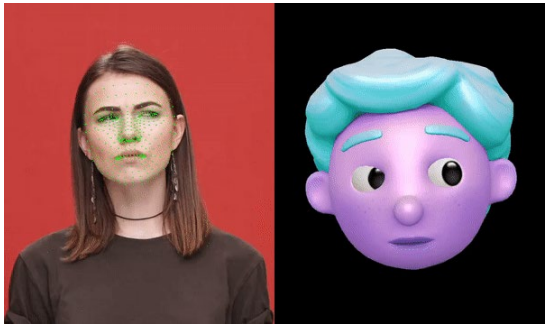
- Physical iris diameter roughly constant at 11.7±0.5 mm
- Assuming known focal (EXIF, Android/iOS APIs) can determine metric depth within avg. error of 5%

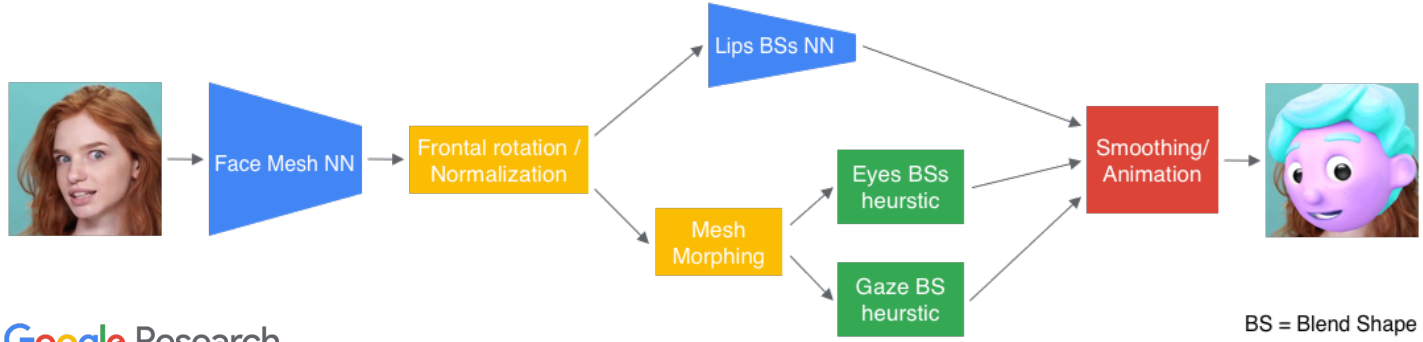




Puppets in Duo

- Using FaceMesh and Iris models
- Provides continuous semantics [0,1] via combination of heuristics and deep nets
- Drives Duo Puppets on Android / iOS





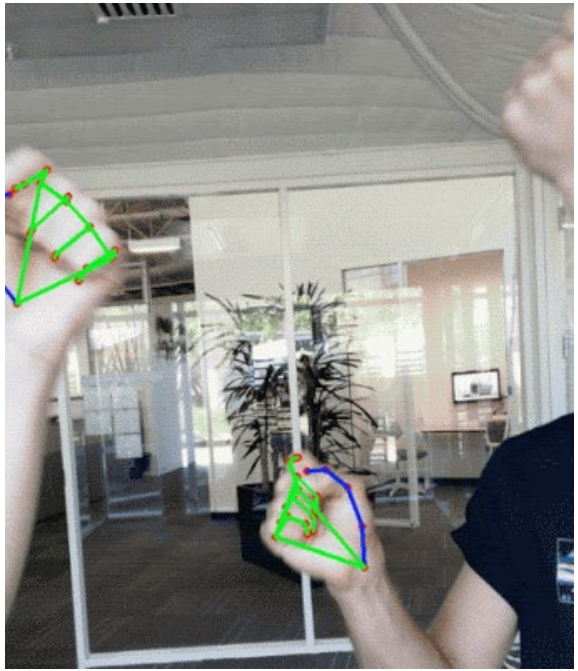
BS = Blend Shape

Proven ML solutions

1. Face Meshes and Iris Tracking: Powering Duo Effects, YouTube Stories, AR-Ads
2. **Hand skeletons and gestures**
3. Human Pose
4. 2D Tracking: Powering Lens, Google Translate
5. 3D Object detection

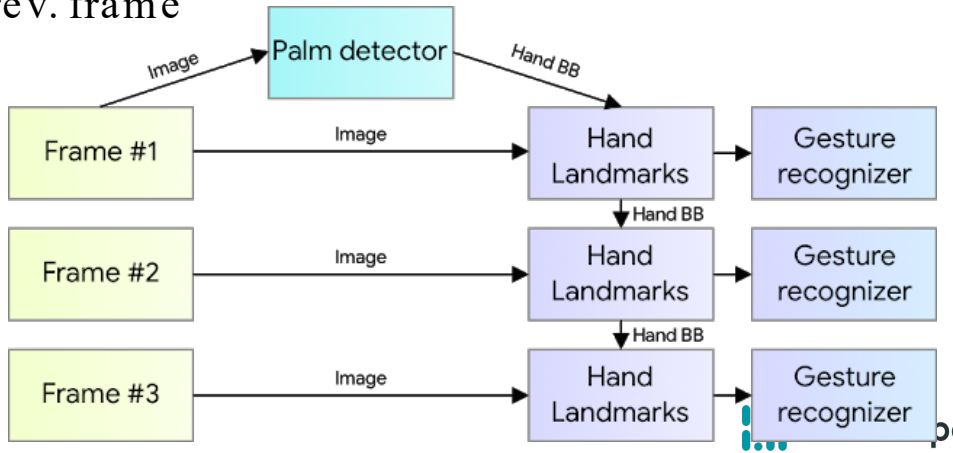
2) MediaPipe Hands

- Released in Mediapipe and TF.js, announced on AI blog
- Uniquely difficult ML problem:
  - Extensive scale of hand sizes: 20x-40x
  - Highly occluded (self-occlusions, e.g. fingers and multiple hands)
  - Myriad of poses and gestures
  - Lack of high contrast (vs. faces or people)
- Use cases:
  Gestures, Free Hand “Mouse” / “Manipulator”,
  AR Try-On (Rings, Watches), Sign language



MediaPipe Hand Pipeline

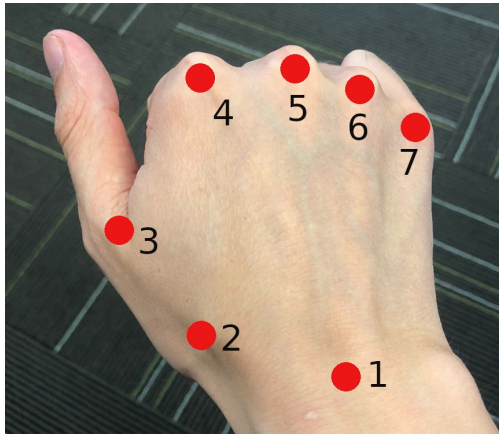
- Two ML models:
  MediaPipe pipeline: **Palm Detector** + **Hand Landmarks**
  - Palm detector returns oriented ROI
  - Landmark model predicts keypoints from ROI crop
- Landmark model to re-use ROI from prev. frame
- Palm detector is only run on 1st frame or when landmark indicates hand miss



BlazePalm Detector

Key Observation: Palm instead of Hand detection

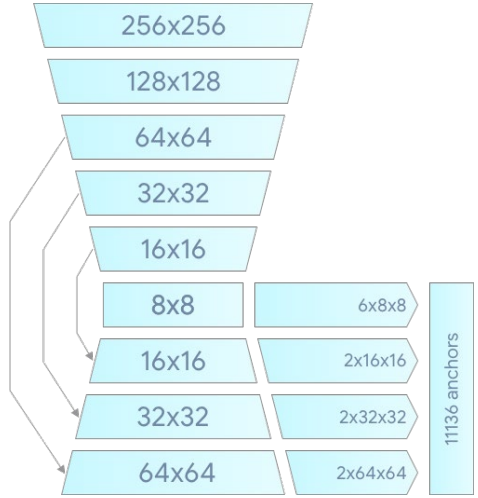
- Palms are rigid (no articulated fingers)
- Palms are small: NMS works better, as less occlusions
- Roughly square aspect ratio -> Anchor reduction
- Input: RGB frame
- Output: SSD-style anchors + 7 landmarks per palm
- 50K training images

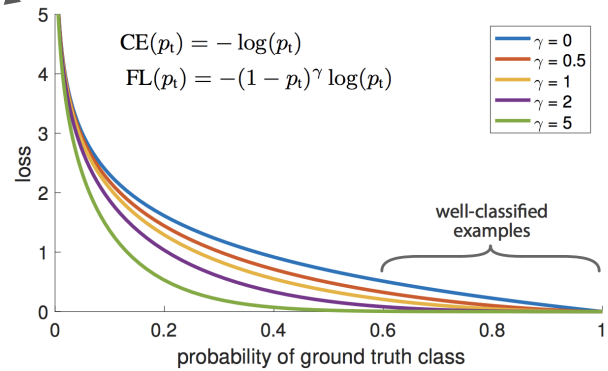


BlazePalm Detector

Design Elements:

- Added Decoder w/ skip connections:
  Feature pyramid network approach for small objects
- Focal Loss (RetinaNet) to address
  fg/bg imbalance (due to small palms)





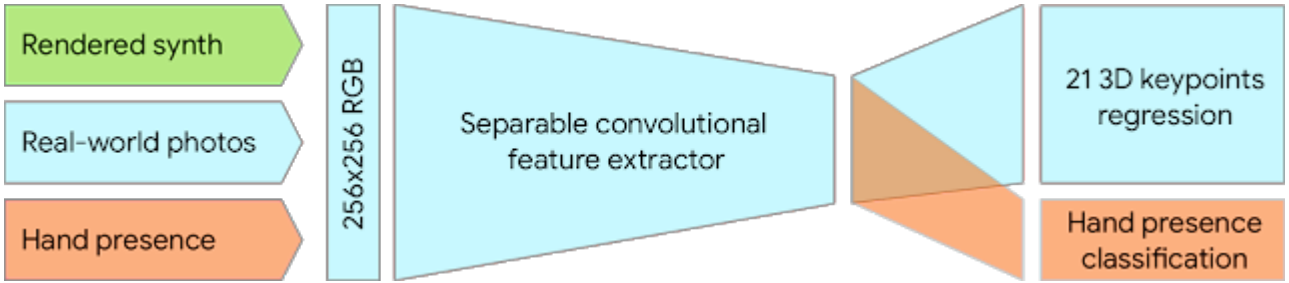
Model variation	Average Precision
No decoder + cross entropy loss	86.22
Decoder + cross entropy loss	94.07
Decoder + focal loss	**95.7**

Google Research

MediaPipe

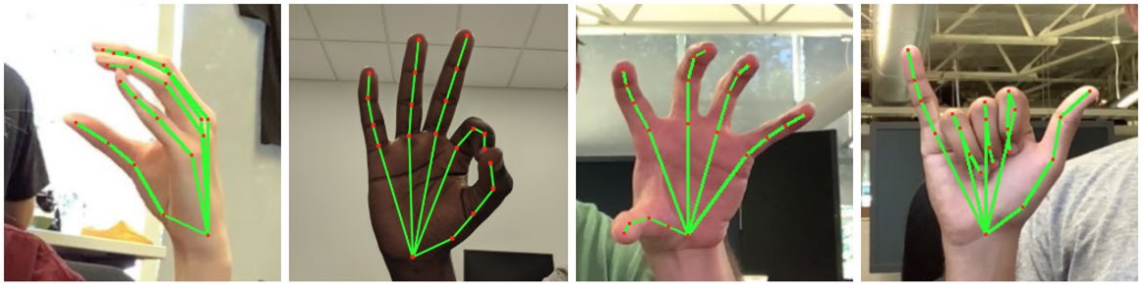
Hand Landmarks: Design Elements

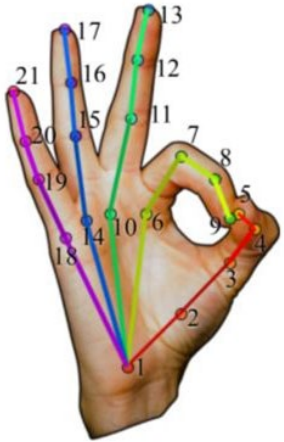
- Predicts 21 3D landmarks from ROI cropped+rotated palm detection
- Trained on 2 datasets: Real world + synthetic
- Predicts hand presence in ROI crop -> trigger detection



Hand Data Sources: GT annotations

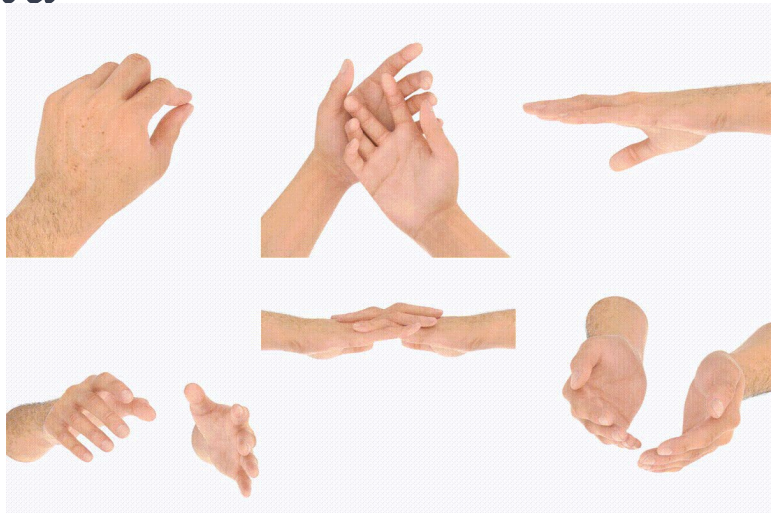
- 21 manually keypoints annotated on 100K images
- Human performance: 4% standard deviation w.r.t. palm height

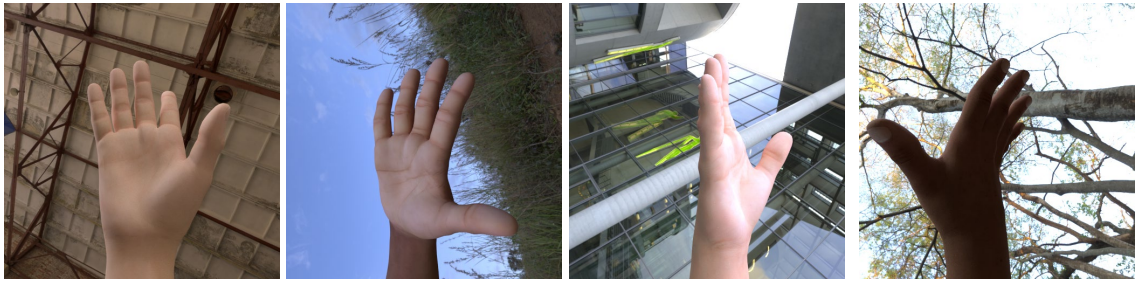




Hand Data Sources: Synth Data

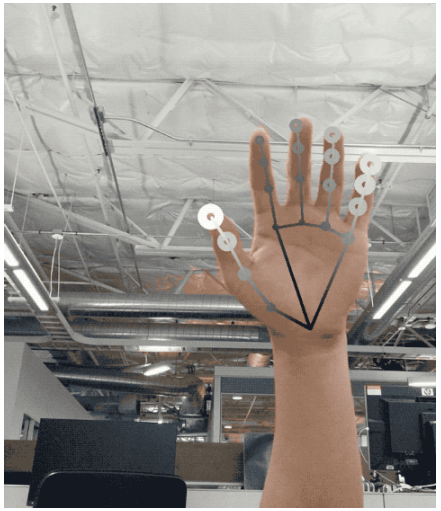
- 100K renders, 120+ gestures
- Rendered in blender w/ HDRi lighting
- Mapped to same 21 keypoints





Hand Landmarks: Results

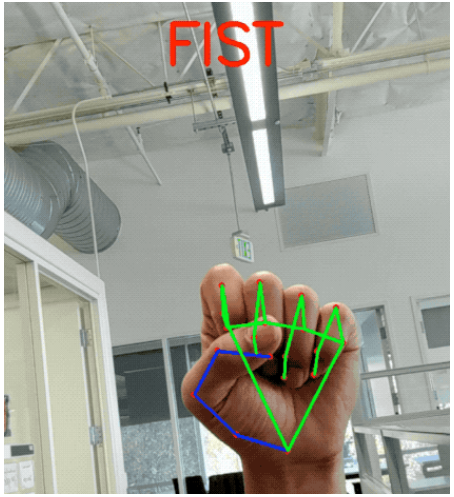
- Using synth. + real-world provides sizable quality boost
- Synth. Data unlocks 3D prediction
  - Train z-coord on 3D only
  - Mask out gradient on real world data



Dataset	Mean regression error normalized by palm size
Only real-world	16.1 %
Only rendered synthetic	25.7 %
Mixed real-world + synthetic	**13.4 %**

MediaPipe Hands: Gestures

- Derived from Hand Landmarks (heuristics or ML model)
- Advantage: Scales to hundred of gestures -
- no separate network for each gesture
- Model card

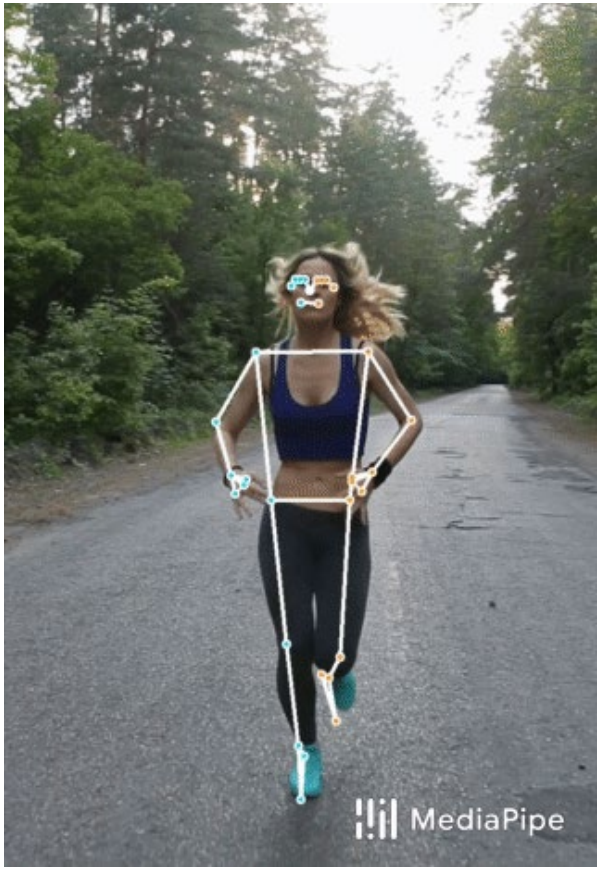


Proven ML solutions

1. Face Meshes and Iris Tracking: Powering Duo Effects, YouTube Stories, AR-Ads
2. Hand skeletons and gestures
3. **Human Pose**
4. 2D Tracking: Powering Lens, Google Translate
5. 3D Object detection

3) MediaPipe BlazePose

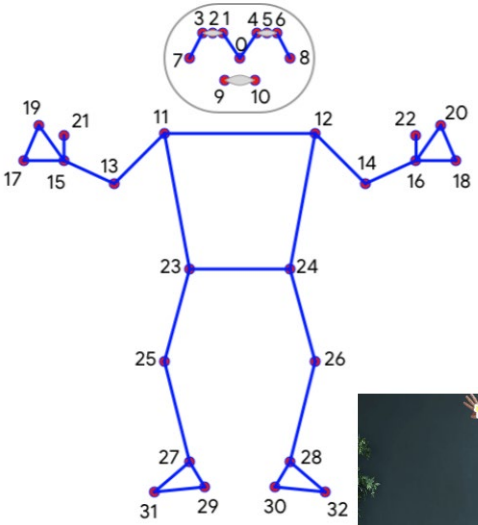
- Released in Mediapipe, announced on AI blog
- **Goal:** Beyond SOTA quality at real-time speeds
- Challenges:
  - Default COCO 17 keypoint topology lacks crucial hand and foot location
  - Existing models not accurate for use cases: Sign Language, Fitness / Yoga, Dancing
- Properties:
  - Real-time, mobile-first, web-friendly
  - Focus on tracking *few* people, *accurately*, *limited scale* (up to 6m)



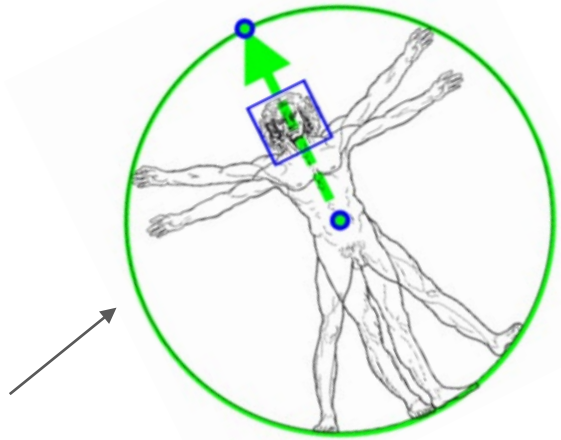
BlazePose Topology

New 33 keypoint architecture

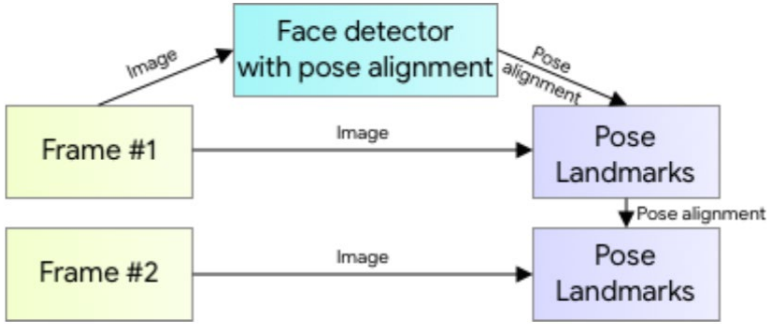
- Localizes hands and feet (scale + rotation)
- Superset of:
  - Coco Pose
  - BlazeFace
  - BlazePalm



BlazePose Pipeline



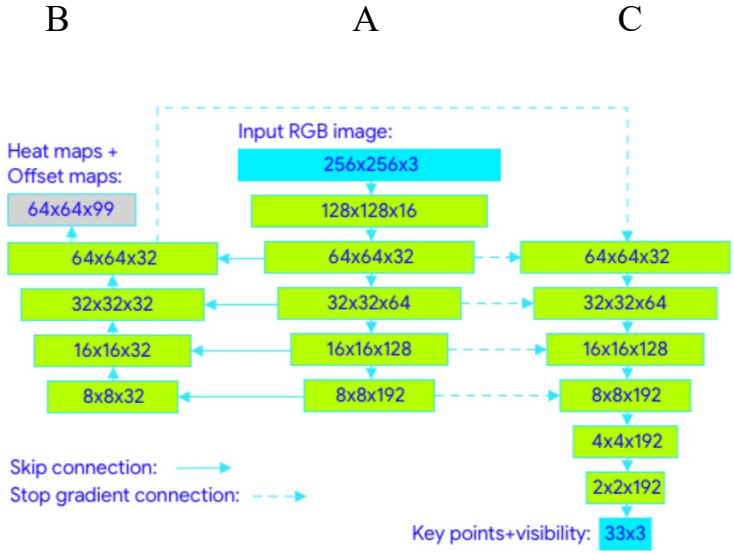
- **MediaPipe** pipeline:
  Extended **BlazeFace Detector** + **Pose Landmarks**
  - Extended BlazeFace detector:
    Returns oriented person ROI via two virtual keypoints
    (center and scale / radius of ROI)
  - Landmark model predicts from ROI crop
    - 33 body keypoints w/ visibility
    - 2 virtual landmark for crop of next frame



BlazePose: Landmarks

Unique architecture:

- Regression network with heatmap supervision
- Network flow: A -> B -> C
- Training procedure:
  - Train A + B with heatmaps (from GT annotation)
  - Cut Heatmap head
  - Train C (A, B constant) for regression head



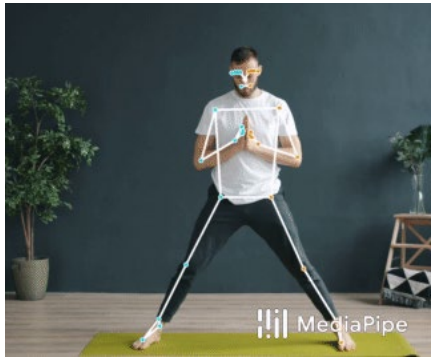
Model	FPS	AR Dataset, PCK@0.2	Yoga Dataset, PCK@0.2
OpenPose (body only)	0.4[1]	**87.76**	83.37
BlazePose Full	10[2]	80.6	**83.5**
BlazePose Lite	**31**[2]	72.8	72.9

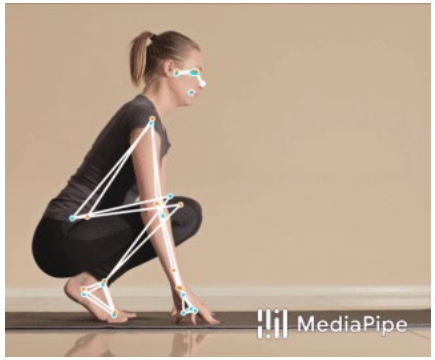
Table 1. BlazePose vs OpenPose

BlazePose: Applications

- Powering new MLKit Pose API
- Real-time performance on mobile CPU/GPU

	Pixel 3 CPU via XNNPACK, FPS	Pixel 3 GPU, FPS
BlazePose lite	44	112
BlazePose full	18	69





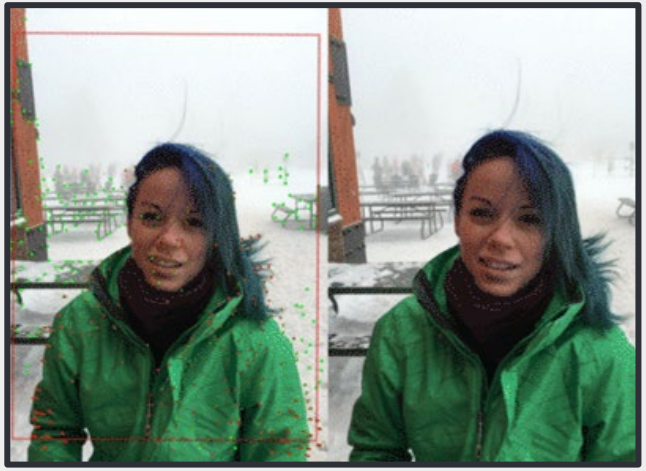
Proven ML solutions

1. Face Meshes and Iris Tracking: Powering Duo Effects, YouTube Stories, AR-Ads
2. Hand skeletons and gestures
3. Human Pose
4. **2D Tracking : Powering Lens, Google Translate**
5. 3D Object detection

4) MediaPipe BoxTracking

- **General purpose tracking** : Tracks any target
- 2 Stage process:

1. Motion Analysis from Sparse Optical Flow, fg/bg classification => tracking metadata
2. Tracking metadata vs. appearance
   => Track multiple objects with minimal increase in latency

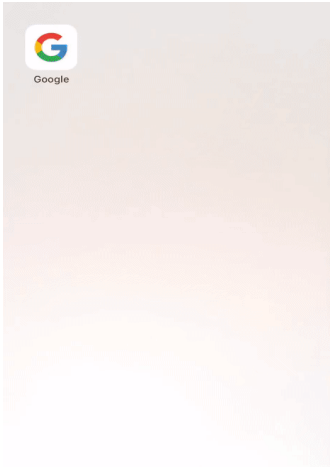


Motion Analysis

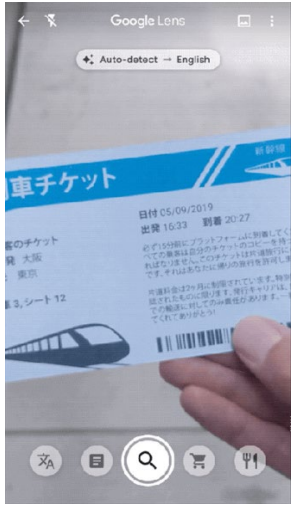
Initial launch Motion Stills ‘18

BoxTracking - Launches

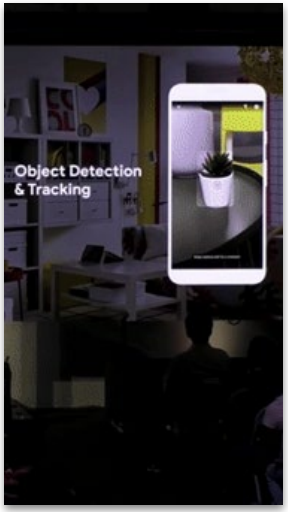
May 2018
Lens Gleaming



Dec 2018
Lens text highlighting



I/O 2019
Lens translate



I/O 2019
MLKit

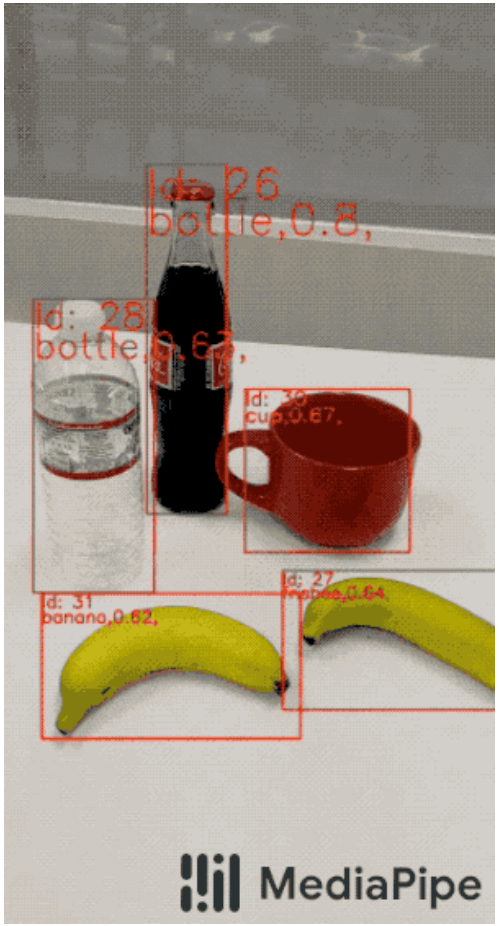
July 2019
Google translate

Google Research

MediaPipe

MediaPipe BoxTracking

- Real-time on-device tracking and object detection
- Available to Developers in MediaPipe
- [Web Page]



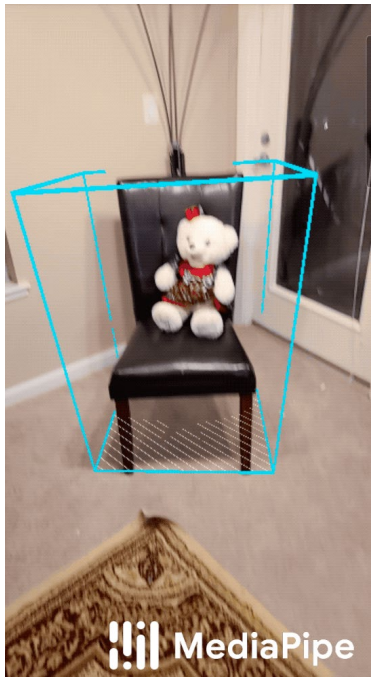
Proven ML solutions

1. Face Meshes and Iris Tracking: Powering Duo Effects, YouTube Stories, AR-Ads
2. Hand skeletons and gestures
3. Human Pose
4. 2D Tracking: Powering Lens, Google Translate
5. **3D Object detection**

MediaPipe Objectron: 3D Object detection

Beyond 2D Object Detection:

- 3D bounding box detection and tracking on mobile
- Category level recognition (e.g. shoes, furniture) vs. instances/templates (particular product)
- 9DOF task: Detects 6 DOF pose (rotation + translation) and 3 DOF dimension
- Released in MediaPipe
- [AI blog](), [Model Card]()



26fps on mobile
(Samsung S20)

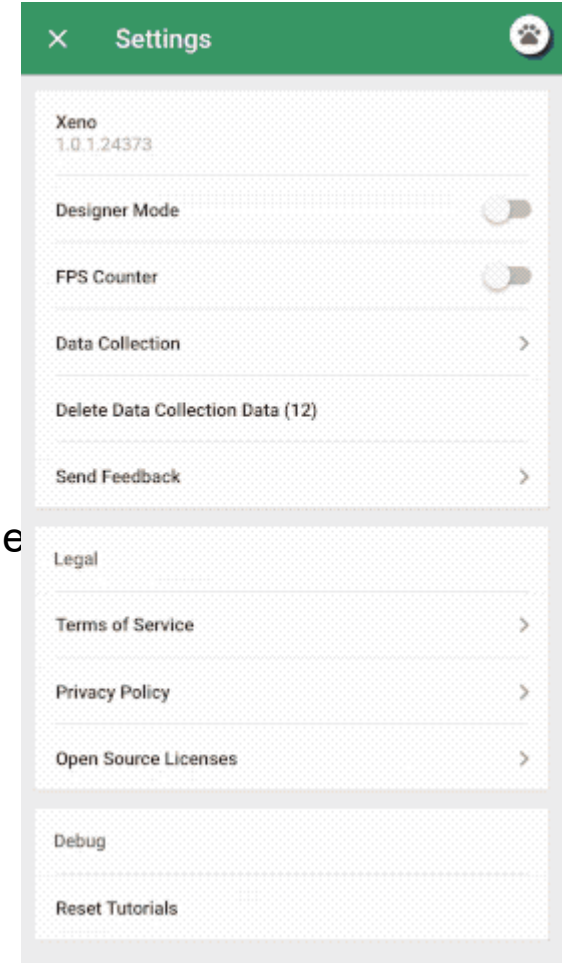
Objectron: Data collection

Challenge : How to annotate world metrically?

Observation : AR Sessions are sparse form of self-driving scenario

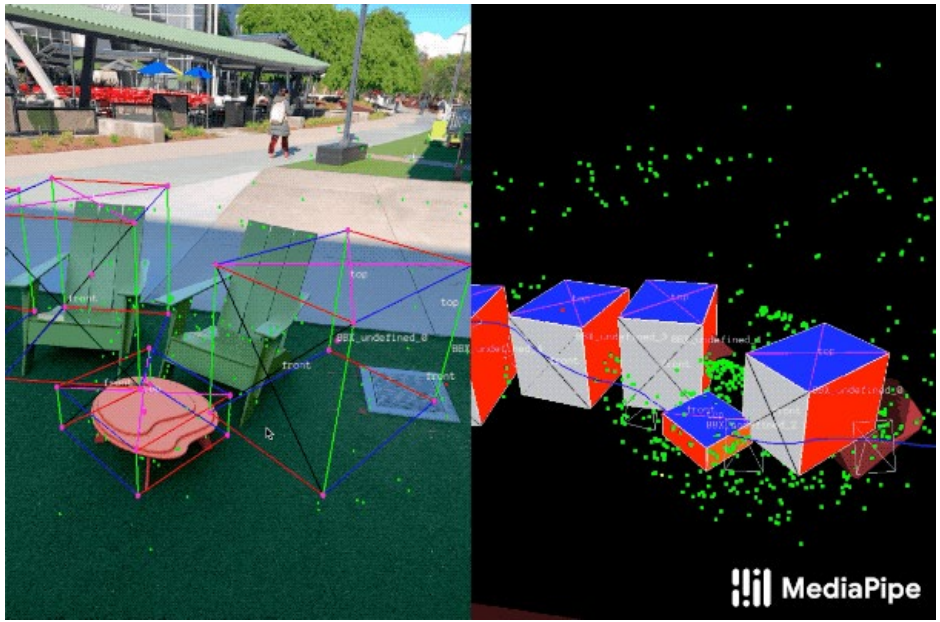
ARSession: ARCore/ARKit capture 3D point clouds, plane geometry, and camera poses

Data: 16k videos / 5M images collected across 9 countries.



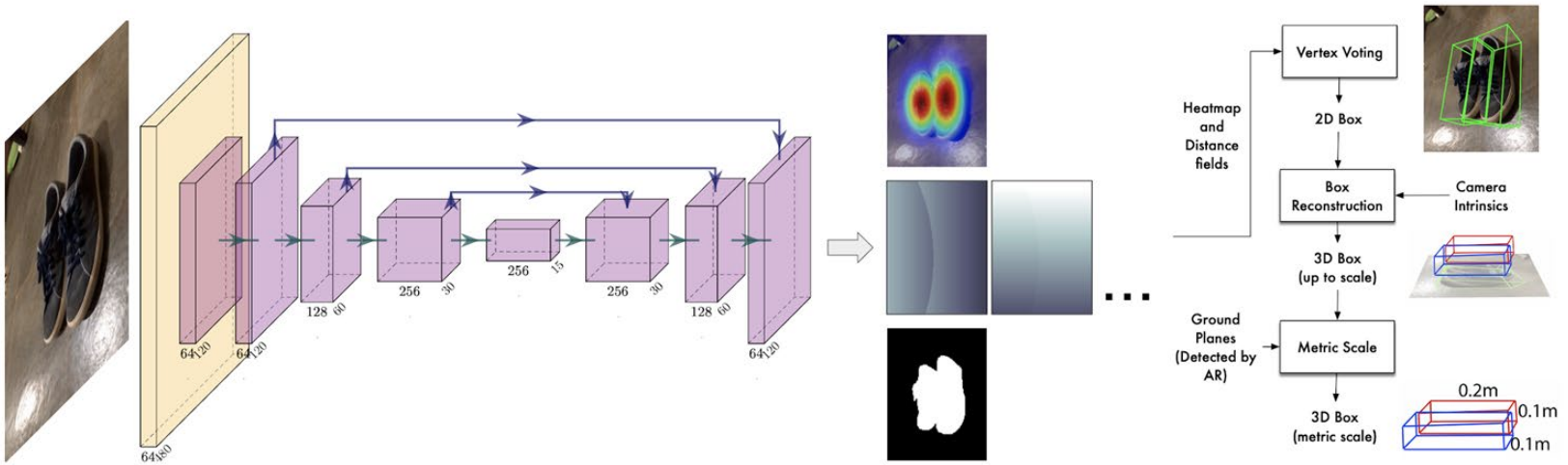
Objectron: Data annotation

1. Record video with ARSession data tracking world and rich 3D geometry data
2. **Efficient:** Annotate **single frame** in 3D => cuboids for every frame in the video
3. **Annotation advantage:**
   7 min per video => **1.5s per frame !!**



Objectron: Multi - Task network

- Single stage anchor free 3D object detection
- Multi-head: centroid heatmap, distance field, segmentation mask
- Output 3D bounding box up to scale (EPnP)



Google Research

MediaPipe



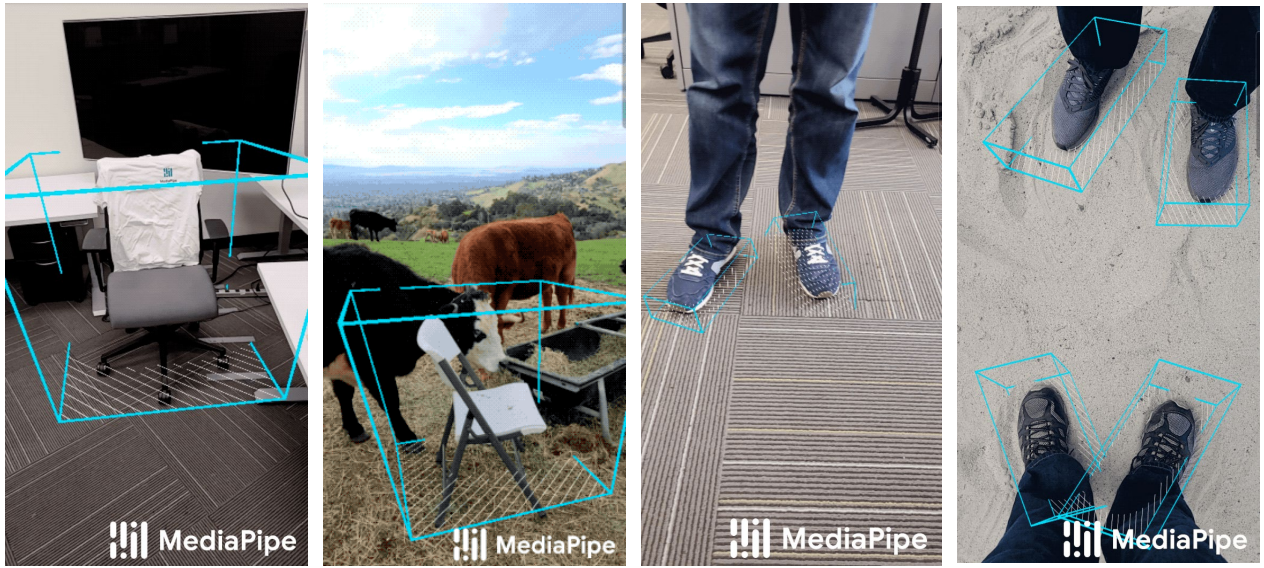
totes
totes
totes





Objectron on mobile

Inference time: 31 ms / frame on Pixel 3 (via TFLite GPU)



02 Infrastructure for Live Perception

Enabling Infrastructure

1. **MediaPipe: Cross -platform ML solutions made simple**
2. Accelerated on-device ML Inference
   a. XNNPack
   b. GPU backend in TF-Lite

MediaPipe: Cross-platform ML solutions made simple

- Build world-class ML solutions and applications for mobile, edge, cloud and the web.
- Widely used at Google in Research & Products to build ML solutions

End-to-end acceleration

Fast ML inference and accelerated pre/post-processing leveraging CPU/GPU

Build once, deploy anywhere

Unified solutions work across Android, iOS, desktop/server, IoT and web

Ready-to-use solutions

Cutting-edge ML solutions demonstrating full power of the framework

Free and open source

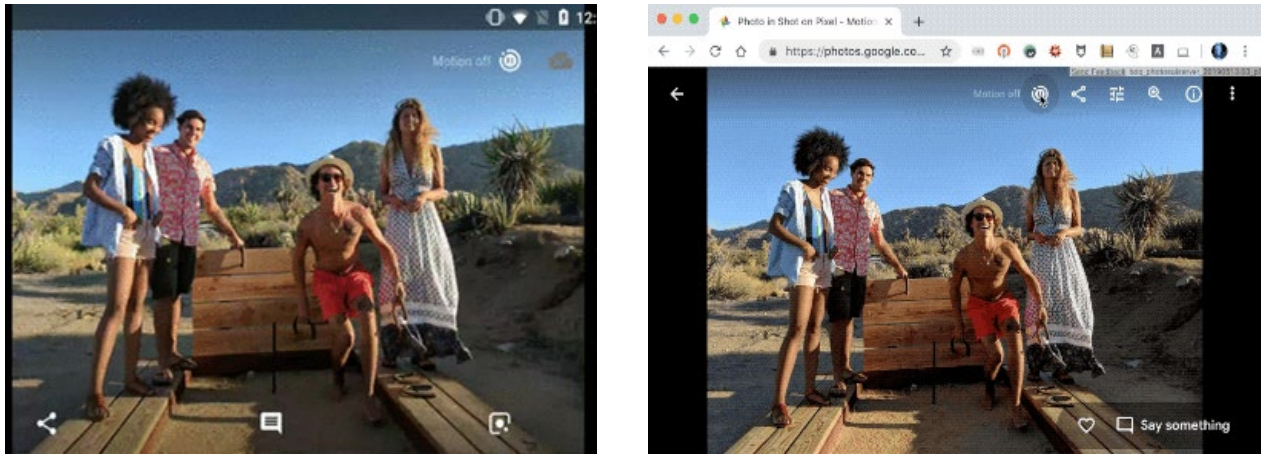
Framework and solutions both under Apache 2.0, fully extensible and customizable

MediaPipe: E2E Acceleration

- Deep integration with TensorFlow Lite on mobile/IoT for full HW acceleration (CPU, GPU, Edge TPU)
- Example: on Samsung S9
  - Face mesh on **GPU 2x+** speed v.s. CPU
  - Hair segmentation on **GPU 6x+** speed v.s. CPU
- GPU accelerated pre/post-processing with multi-context support, enabling end-to-end accelerated pipelines

MediaPipe: Build Once, Deploy Anywhere

- ML solutions easily compiled onto Android, iOS, desktop/server, Python, IoT and web



Google Motion Photos on Android and Desktop Chrome

MediaPipe: Ready-to-use solutions

	Android	iOS	Desktop	Python	Web
Face Detection	✅	✅	✅		✅
Face Mesh	✅	✅	✅		
Iris NEW	✅	✅	✅		✅
Hands	✅	✅	✅		✅
Pose NEW	✅	✅	✅	✅	✅
Hair Segmentation	✅		✅		✅
Object Detection	✅	✅	✅		
Box Tracking	✅	✅	✅		
Objectron	✅				
KNIFT	✅				
AutoFlip			✅		

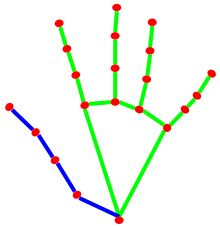
MediaPipe: Free and Open Source

- Permissive Apache 2.0 license
- Build and distribute your own ML solutions
- Use and distribute ready-to-use ML solutions
- Customize ready-to-use ML solutions: plug in your own model, modify parameters, extend with new functionalities ...

MediaPipe Example: Hand Tracking

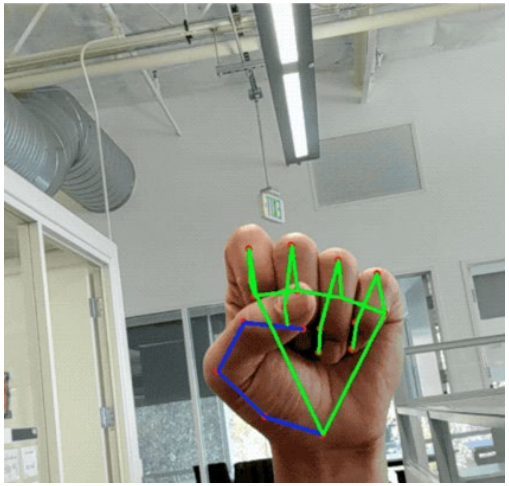
Image

Model



Landmarks

There's more to it



Simple ML pipeline

Video input, e.g., from live camera viewfinder

Geometric transformation to resize and rotate etc

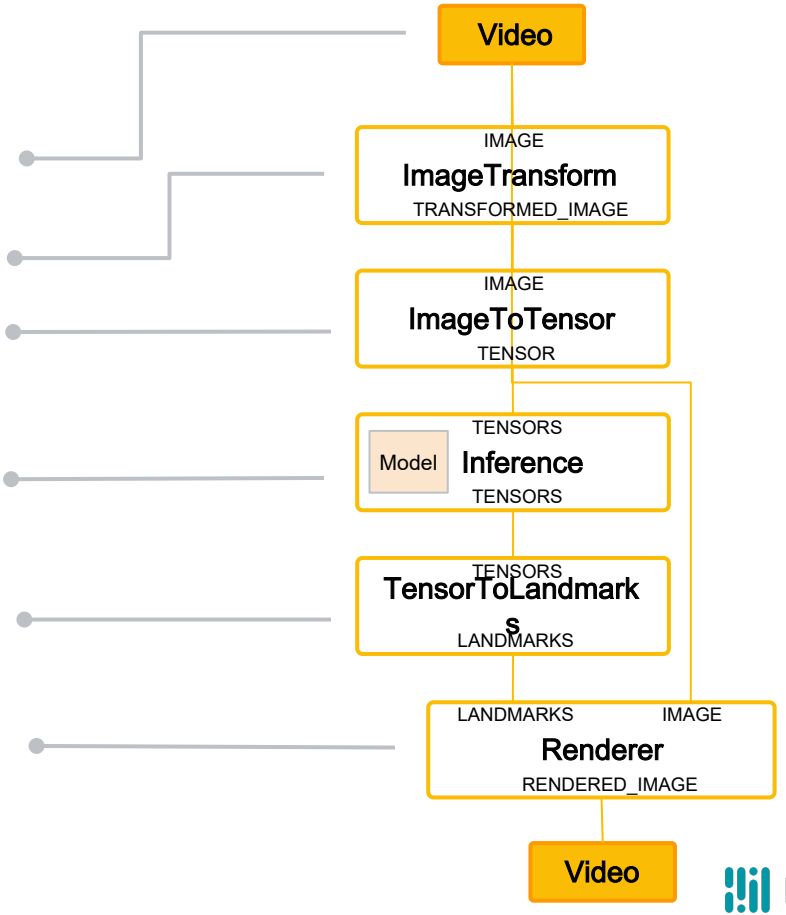
Converts images to tensors

For a given ML model, runs ML inference

Decodes tensors into high -level metadata

Renders landmarks onto associated images

Video output, e.g., display



Google Research

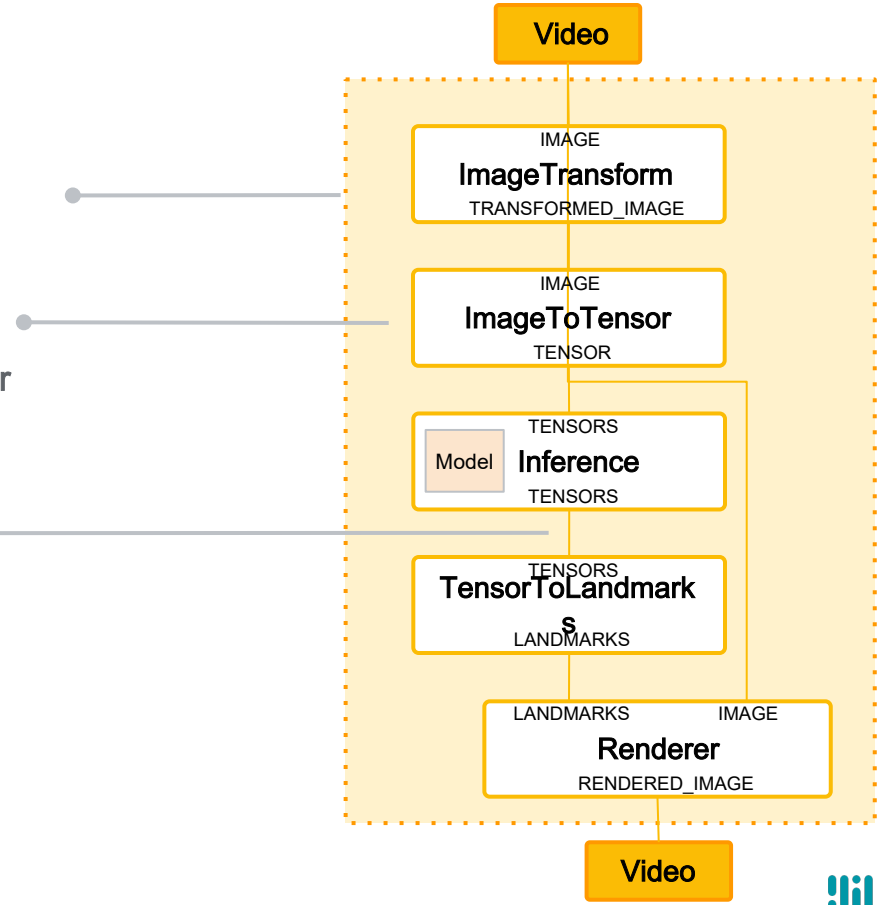
MediaPipe

MediaPipe pipeline

A MediaPipe **Graph** represents a **perception pipeline**

Each node in the pipeline is a MediaPipe **Calculator**

Two nodes can be connected by a **Stream**, which carries a sequence of **Packets** with ascending timestamps



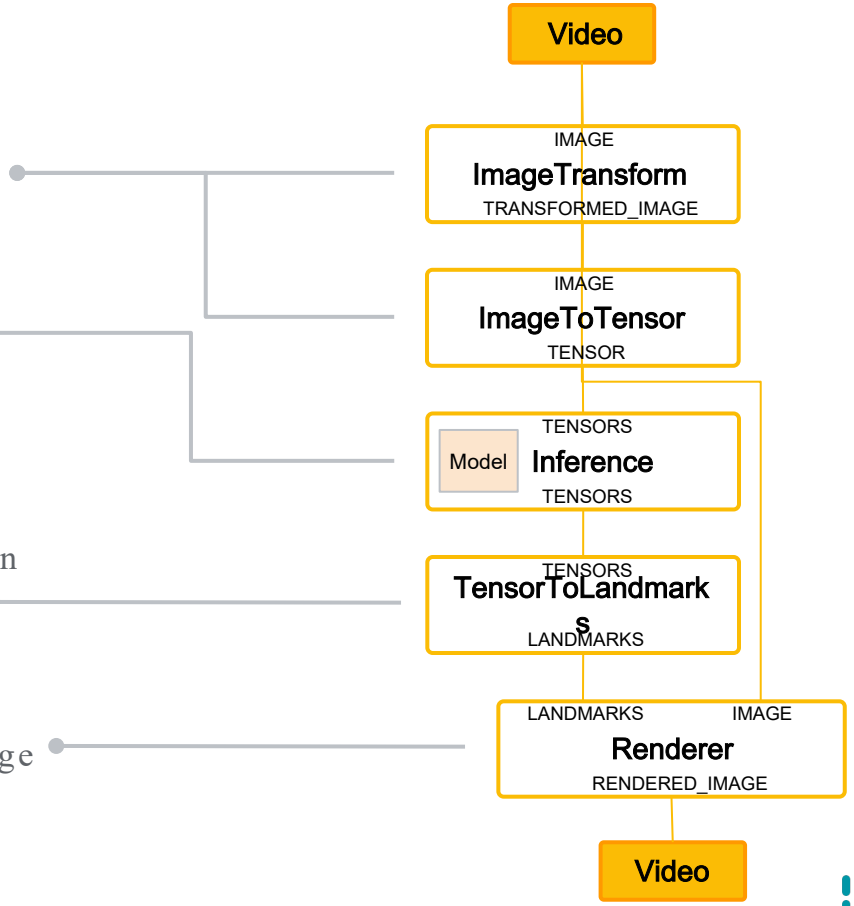
Built-in Calculators

Family of **image and media processing** calculators

Native integration with TensorFlow and TF Lite for **ML inference**

Family of **ML post-processing** calculators for common ML tasks, e.g., detection, segmentation and classification

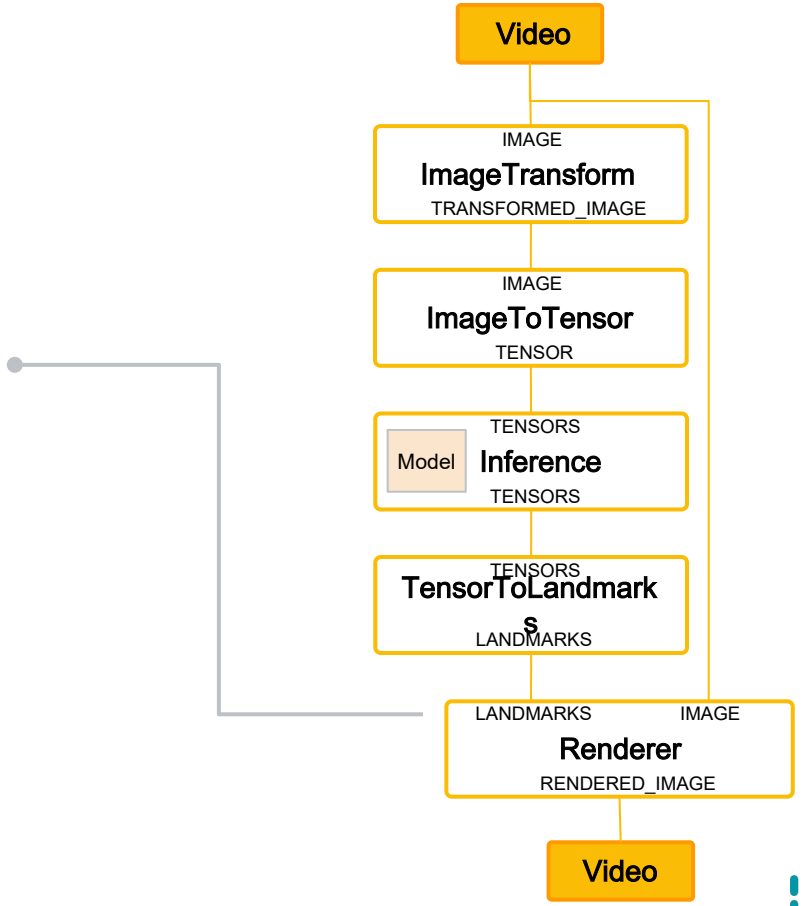
Family of **utility** calculators for, e.g., image annotation, flow control



Synchronization

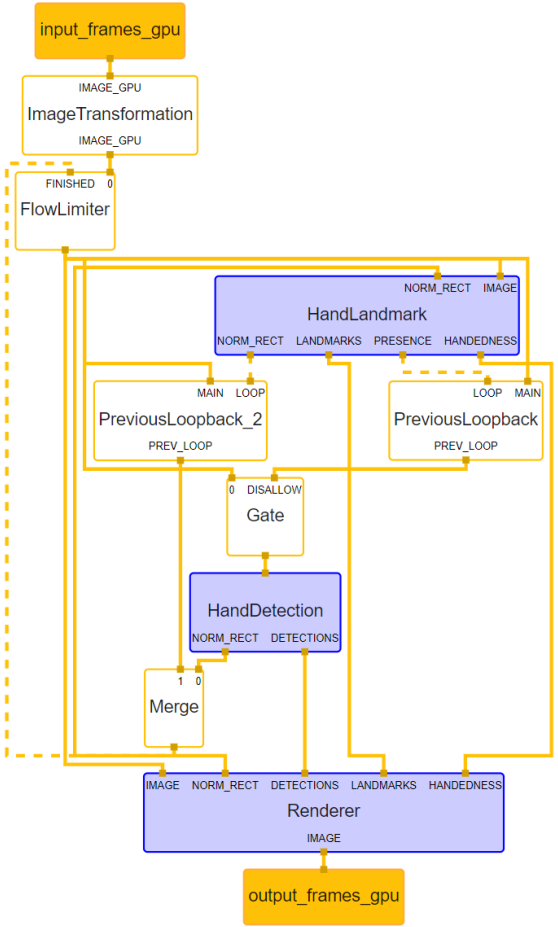
Synchronization handled by MediaPipe framework to align time-series data

- By default all inputs to a calculator are synchronized
- Here: Framework buffers input for time-aligned rendering hiding ML latency



Hand Pipeline

- All GPU E2E pipeline from input, over ML inference to render
- Two models BlazePalm detector and Hand landmarks working together *across* frames
- Details fairly involved, check out: viz.mediaipe.dev



Enabling Infrastructure

1. MediaPipe: Cross-platform ML solutions made simple
2. **Accelerated on -device ML Inference**
   a. XNNPack
   b. GPU backend in TF-Lite

On-device Inference: Challenges

- Mobile CPUs Are Less Powerful than Server CPUs
  - On-device inference = adding a significant compute-intensive task to CPU
- Limited Memory and memory bandwidth
  - Model size matters
- Additional Costs
  - Increased energy consumption = shorter battery life
  - Increased thermal profile = throttling = slower computation
- Two solutions:
  - CPU acceleration via XNNPack
  - GPU acceleration via GPU delegate for TF-Lite

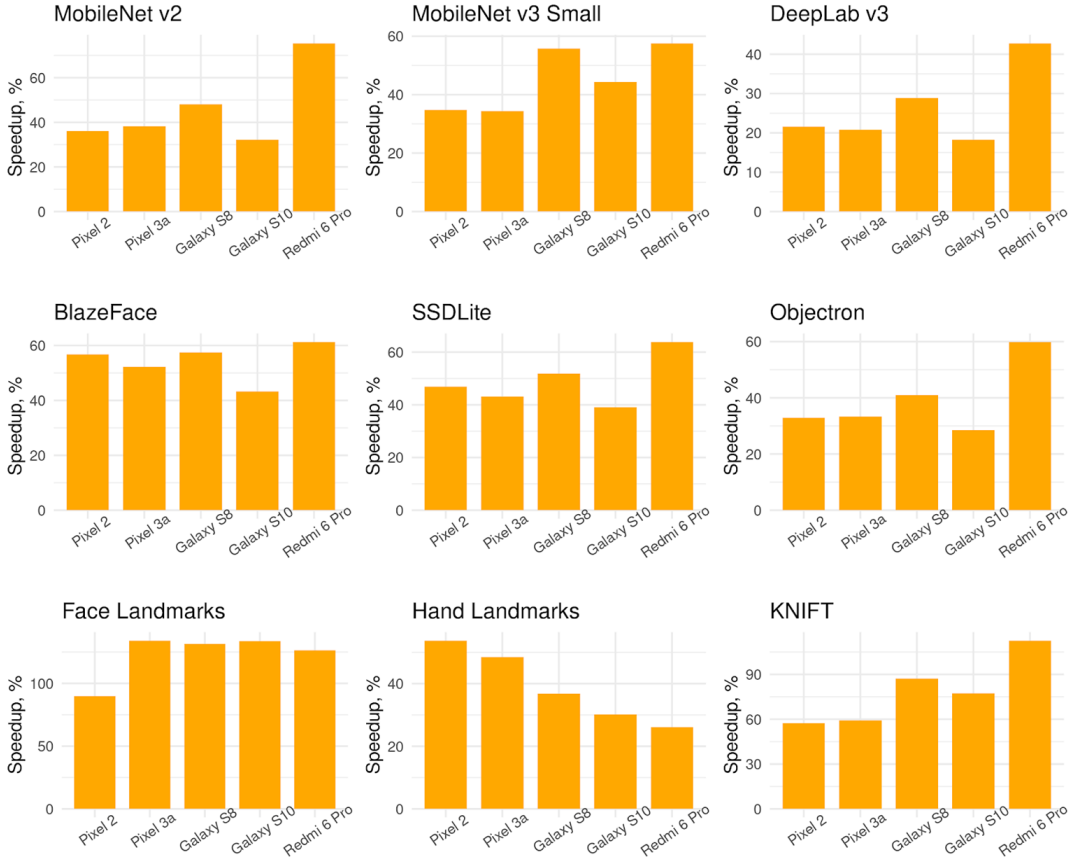
XNNPack: Optimized CPU ops

- Neural network **operator** library
  - Providing highly optimized implementations for floating-point neural network operators
  - Operators are optimized for ARM NEON / x86-64
  - Critical ops (conv, depthwise conv, deconv, FC), are further tuned in assembly for commonly-used ARM cores in mobile phones
  - Operator fusion: Detects common combinations (conv + padding) and fuses into single accelerated op
- Released July, 2020, blog
- Github page

XNNPack: Speedups

Provides 30-100% speedup over former TFLite CPU implementation

Opt-in for TFLite, default in future release

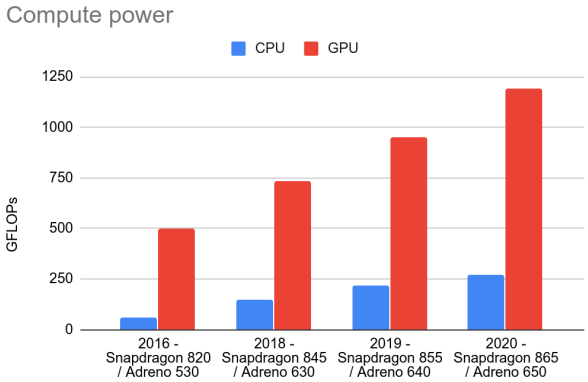


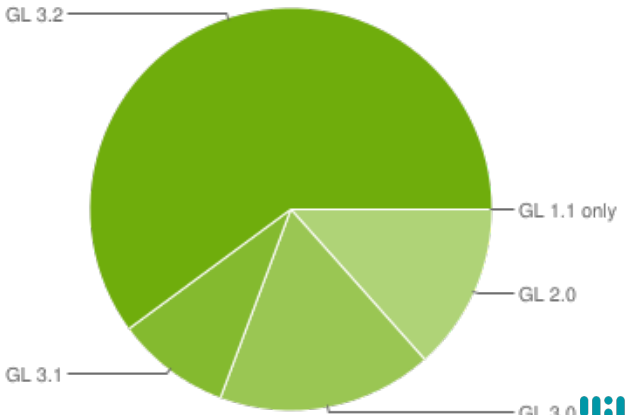
Google Research

MediaPipe

GPU backend in TF-Lite

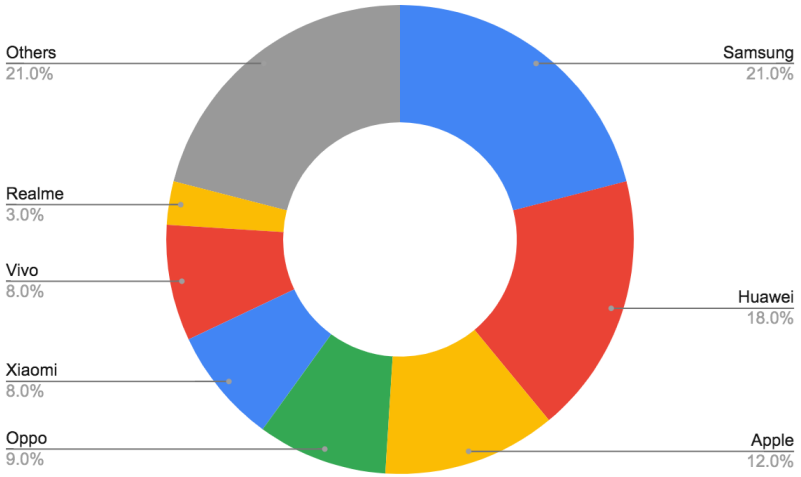
- GPU: Ubiquitous accelerator on mobile
  - 5x+ more compute power vs. CPU
  - Majority of Android phones support ES 3.1+ Compute (July'20)
- Web page, Blog





NPU as alternative?

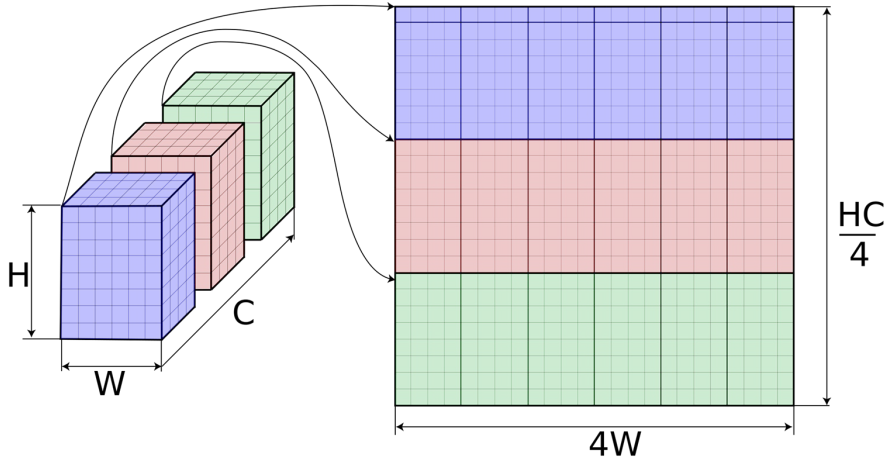
- NPU landscape still fragmented, per-device solutions
  a. Limited to vendor-specific apps, e.g. camera apps
  b. Limited to premium devices
  c. No unifying SDK available



Market Share by Phone Manufacturer '19 Q3

GPU acceleration: Memory layout

- GPU-Friendly Tensor Representation:
  Split channels into multiples of 4
- Example: [H, W, C] = [8, 6, 12]



GPU acceleration: Optimized Shaders

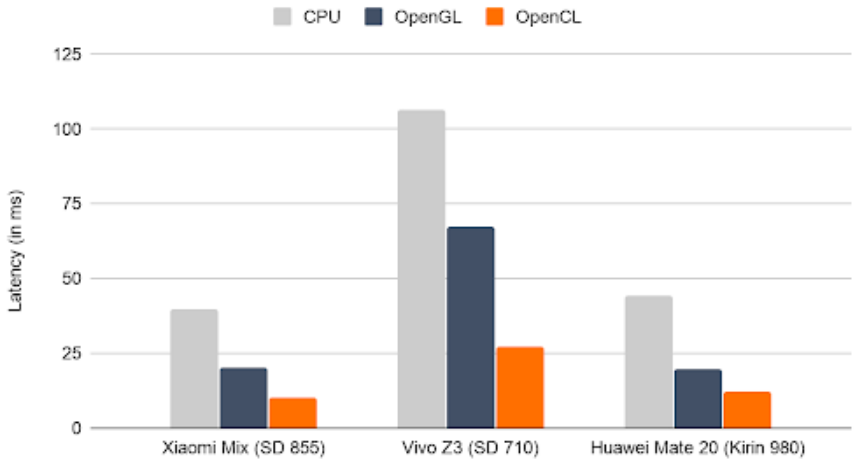
Optimization ingredients:

1. Reduce Number of Shader Programs
   - Fuse element-wise ops with more complex ops, e.g. CONV_2D
1. Reduce Memory I/O
   - Inline small tensors into the shader program
1. Create Shader Program Specializations
   - Parameter-specific specializations, e.g. 1x1 CONV_2D
   - HW-specific specializations, e.g. Adreno, Mali

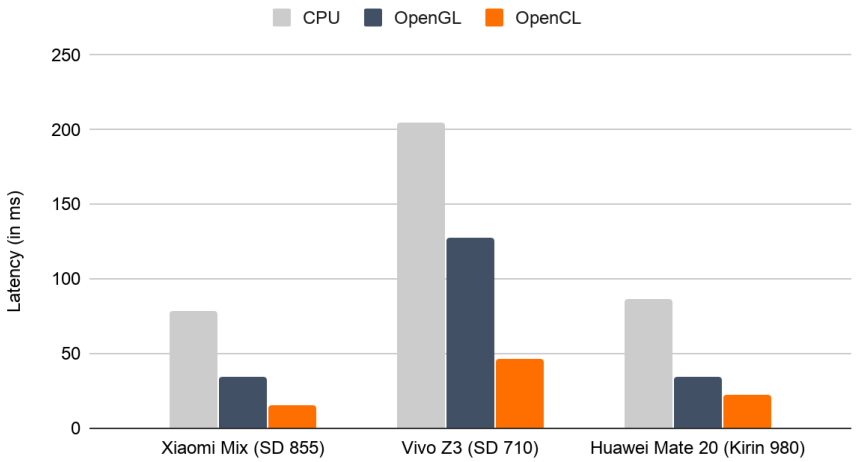
GPU acceleration: Performance

OpenGL acceleration: ~2x faster than CPU

OpenCL acceleration: ~4x faster than CPU



MNasNet 1.3



MobileNet v3

Google Research

MediaPipe

03 Questions?