Project Lab Human Activity Understanding

Winter Semester 25/26

Battery Assembly

Group D:

- Jesus David Jimenez Sanchez
- Ioannis Papadongonas
- Lucía Balsa Picado

February 5th, 2026

Outline

- Introduction
  - Project setup
  - Pipeline

- Object Detection, 6DoF Pose Estimation and Scene Reconstruction
- Error Detection in Battery Insertion Order
- Error Detection in Screwing Order

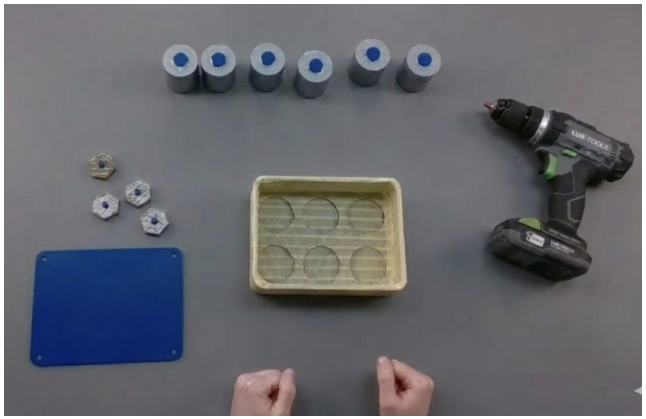
Introduction

- Production lines

- Use case - Assembly of a toy battery pack:

1. Inserting 6 battery cells
2. Covering case
3. Fixing 4 screws in a cross pattern

Fig 1: Battery pack assembly in BMW Group Plant Spartanburg



Project Setup

- Setup
  - Tabletop
  - x8 D435 Depth Sensors

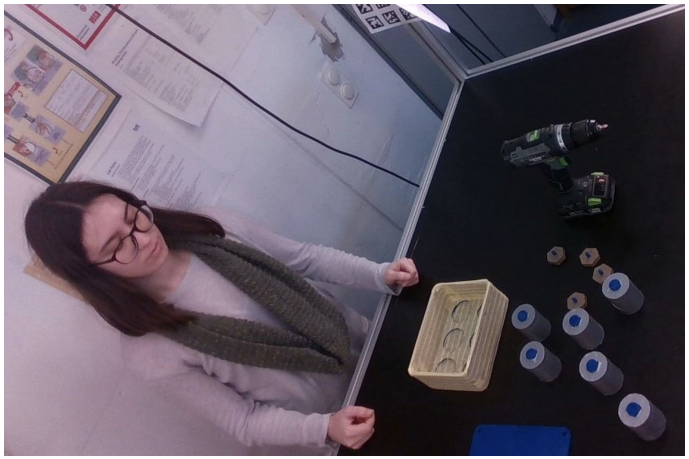


Fig A.1. Intel Realsense D435 Depth Camera

Project Pipeline

1. • Data Collection and Annotation
2. • Object Detection, 6D Pose Detection, Scene Reconstruction [Jesus]
3. • Error Detection in Battery Insertion Order [Lucía]
4. • Error Detection in Screwing Order [Ioannis]

Object Detection and 6DoF Pose Estimation Scene Reconstruction

Jesus David Jimenez

Object Detection YOLOv11 Model:

Goal:

Detect assembled objects to identify errors in the assembly process.

"You Only Look Once" computer vision model from Ultralytics (Sept 2024, PyTorch)

- Finetuned:

  YOLO11n-seg (2.9M parameters) for ultra-fast, edge, and IoT devices.

  YOLO11m-seg (20.1M parameters) for General-purpose desktop apps.

- 6 Object Classes: Person, Battery, Case, Case Cover, Screws, and E-ScrewDriver.

Fig 1.2: **Output** frame

Data Annotation

- SAM2 large with propagate function on Google Colab with A100 GPU.

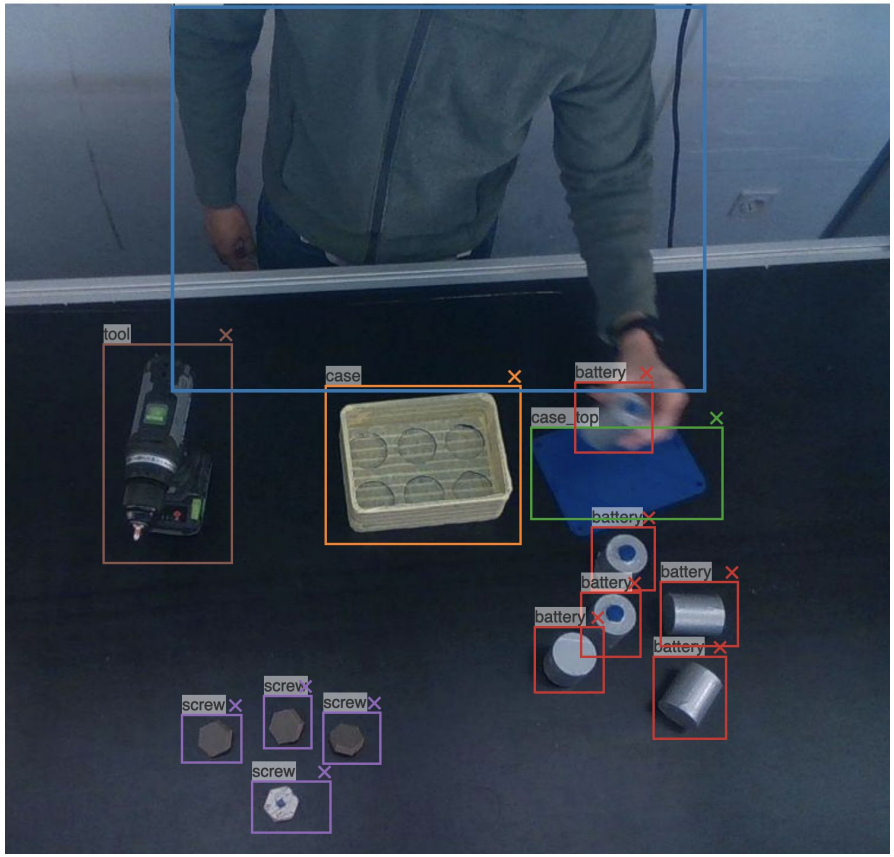


Fig 1 Data annotation with google colab

Data Annotation

Dataset	2100 images total; Split 90/10: 1890 (Training) / 210 (Validation).
Data Source	7/12 Recordings, 3/8 Cameras view, 100 frames each camera.

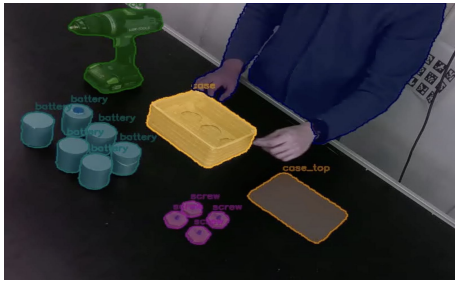


Fig 1.1: first view

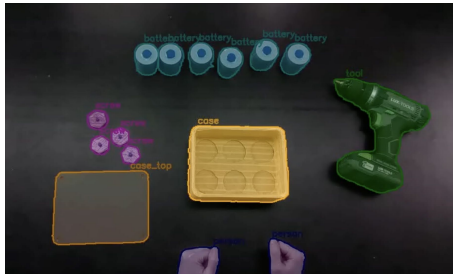


Fig 1.2: Second view

Fig 1.3 Third view

Finetuning

Learning Rate (Lr) From 0.01 to 0.0001

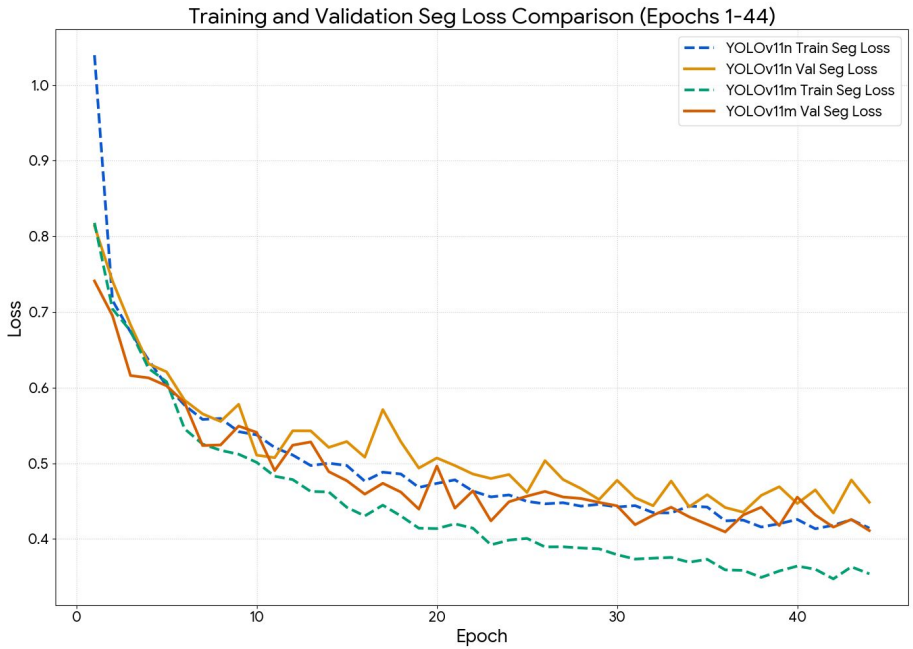


Fig 1.2 Training loss curve yolov1-seg nano and medium

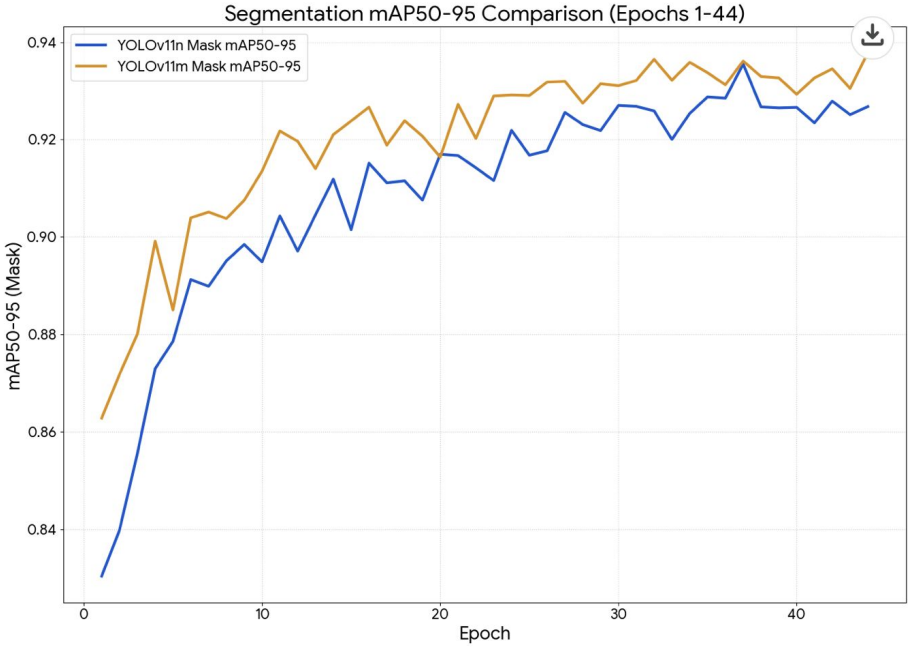


Fig 1.2 Training loss curve yolov1-seg nano and medium

Finetuning Visual Results

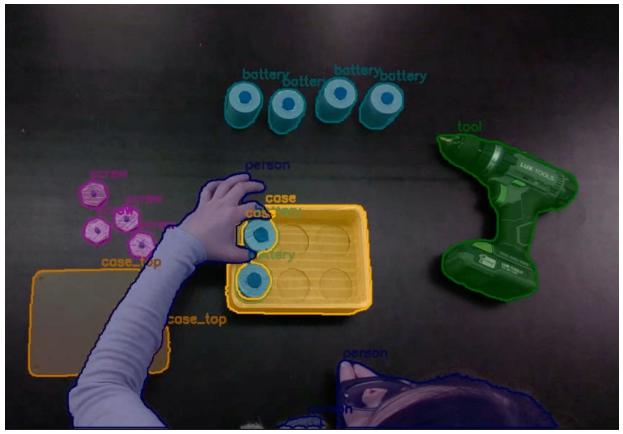


Fig 1.1: SAM2 Annotated Image

Fig 1.2: Finetuned yolov11n-seg inference

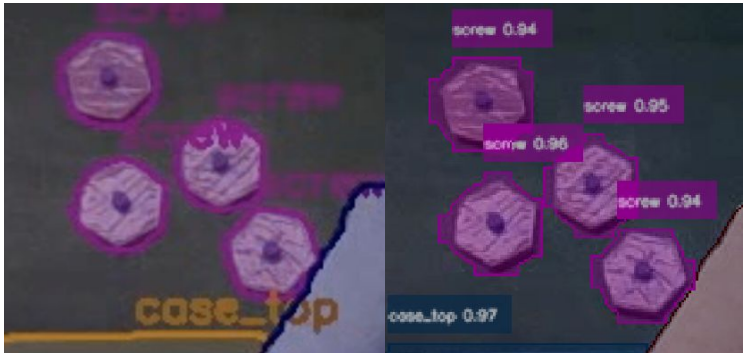


Fig 1.3: Screw segmentation comparison

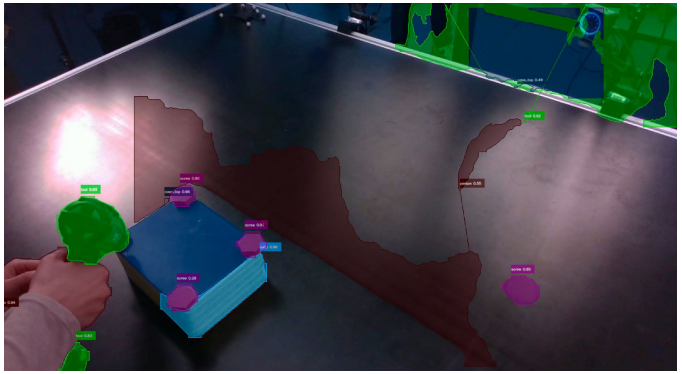


Fig 1.4 Bad detections on new camera view

6DoF Pose Estimation

Goal: Predict objects 3D-pose to verify correct screws fixing order.

Deep Object Pose - DOPE, detection and 6-DoF pose estimation of known objects from an RGB camera released by NVIDIA in 2018.

- Finetuned on Synthetic data

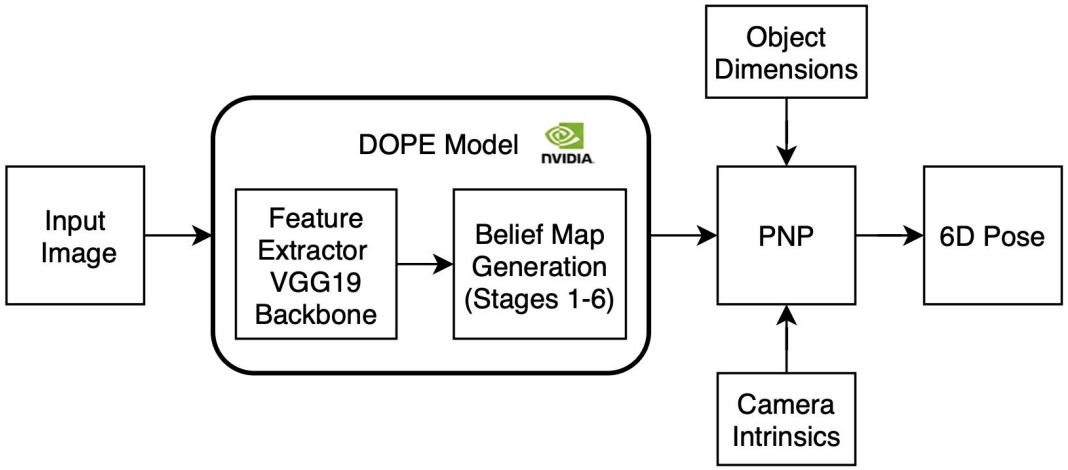


Fig 1.1 DOPE Architecture,

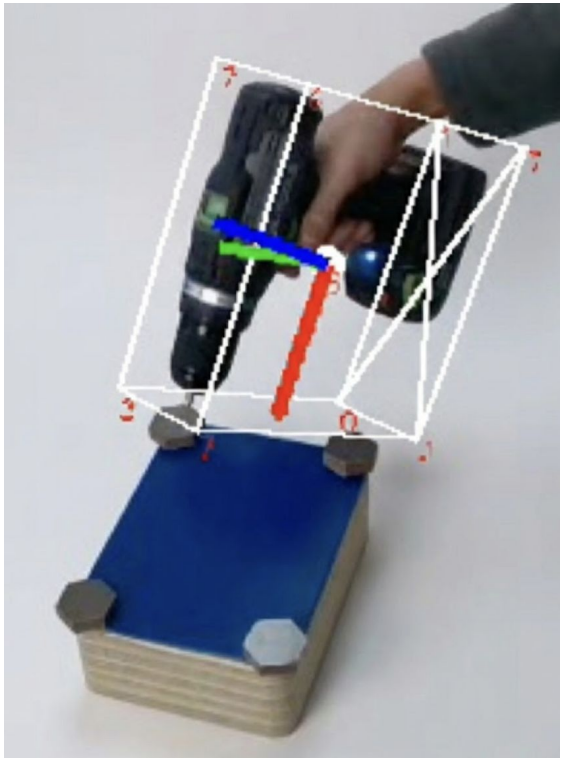


Fig 1.2 DOPE pose estimation

Synthetic data

- E-ScrewDriver: +7.000 images, Case: 2.000 images Battery: 1,500 images Screw: 1,500 images Finetuned on Synthetic data

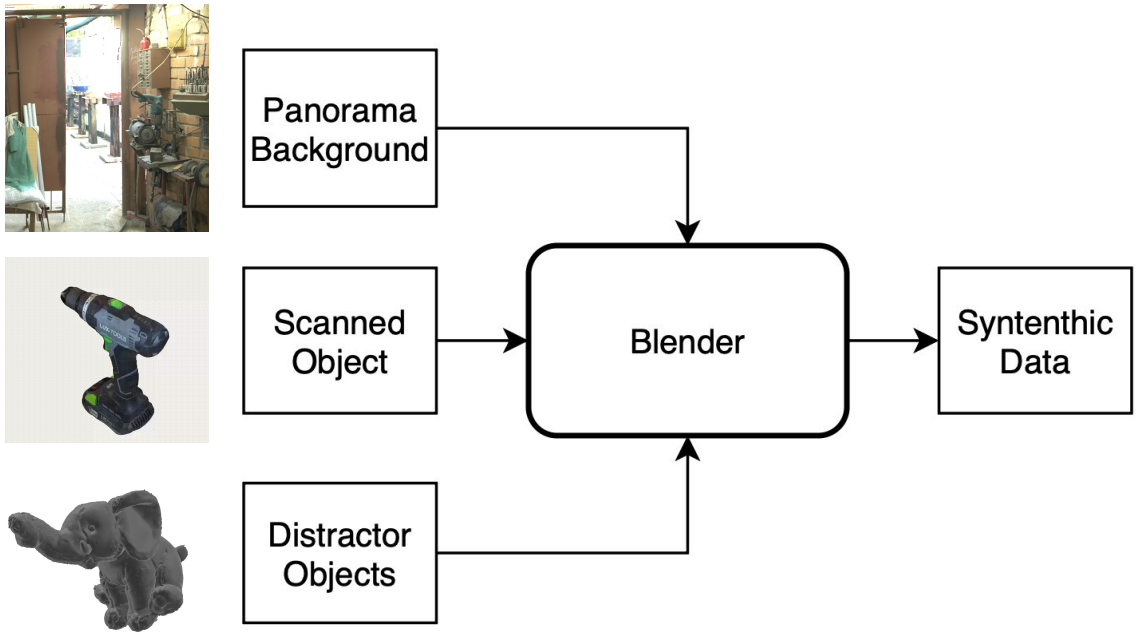


Fig 1.1 Synthetic Data Pipeline

Fig 1.2 Synthetic Data Example

Synthetic data

Fig 1.1 Synthetic Data Example Tool

Fig 1.2 Synthetic Data Example Case

Synthetic data

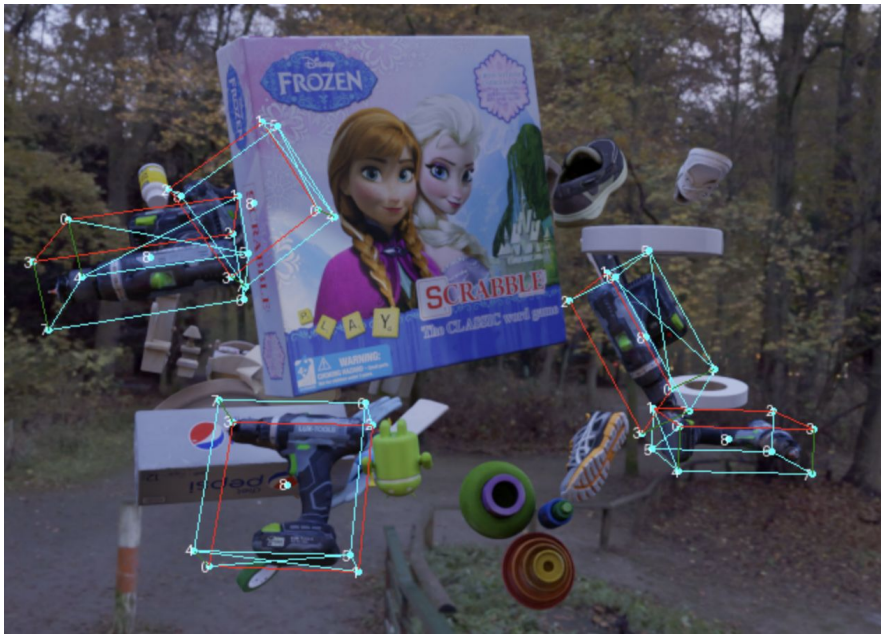


Fig 1.1 Synthetic Data Example Tool
with ground truel

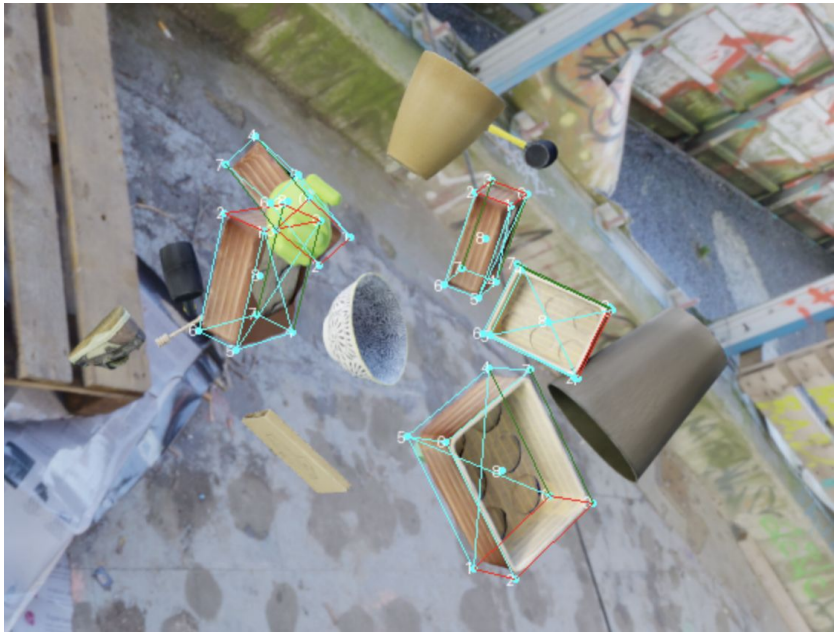


Fig 1.2 Synthetic Data Example Case
with ground true

DOPE Training

	Paper	Ours
Hardware	4 P100 (16GB) GPU	A100 (80GB)
Epochs	60	300
Images	60K Random + 60K Photo Realistic	7k Random
Batch Size	128	96
Optimizer	Adam	Adam
Learning Rate	0.0001	0.0001

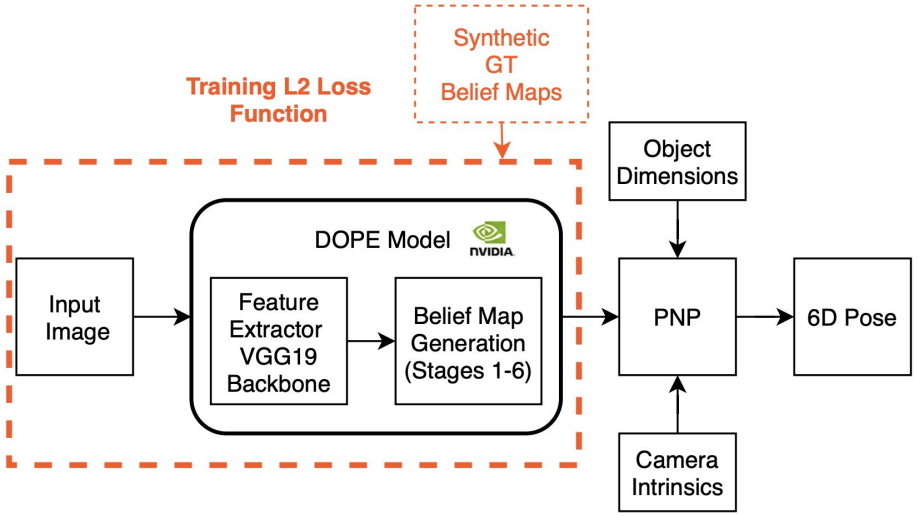


Fig 1.2 Synthetic Data Example Case

DOPE Belief Maps

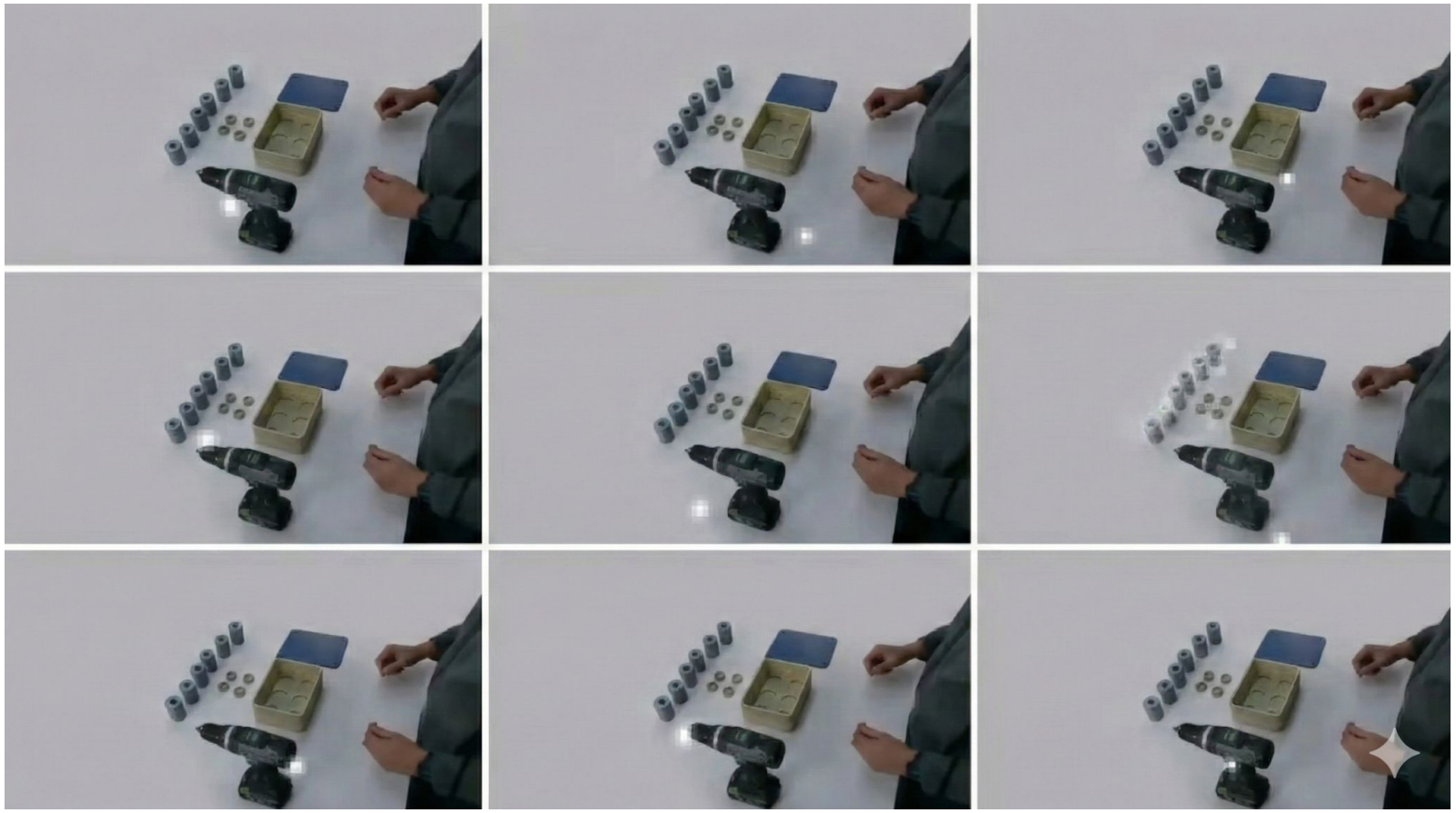


Fig 1 Tool Belief Maps before PnP *Gemini used to make the background brighter

DOPE Final Result

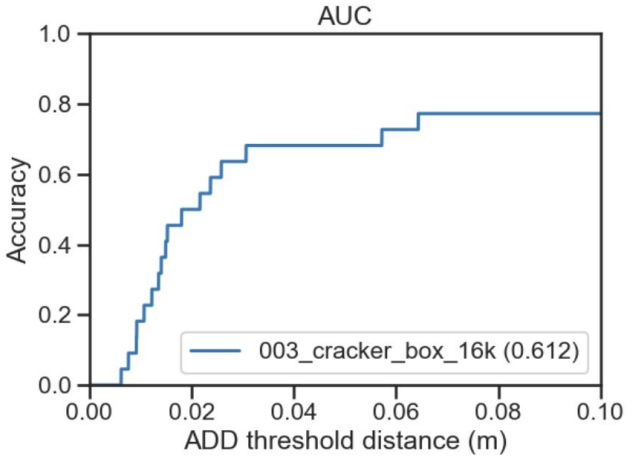


Fig 1.3 AUC metric repository
https://github.com/NVlabs/Deep_Object_Pose

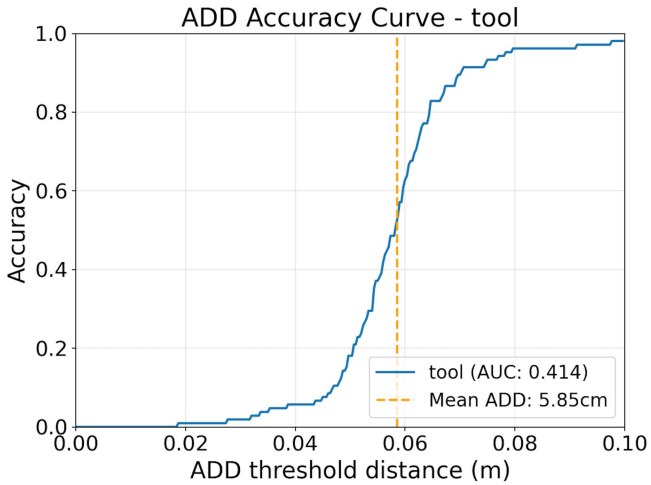


Fig 1.1 AUC metric TOOL, 105 detected objects

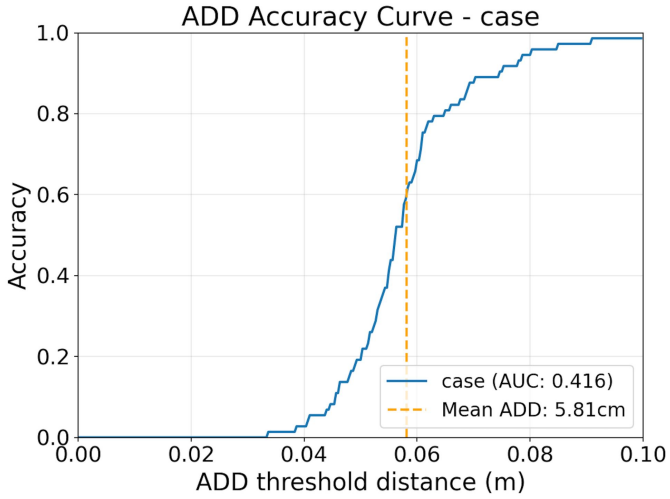


Fig 1.2 AUC metric CASE, 73 detected objects

DOPE Final Result

Limitations:

- Symmetric objects without distinct patterns
- Detection of small objects
- One model required per object

Fig 1.1 3d Scene in three.js

Fig 1.2 Dope case tool prediction

Scene Reconstruction

Goal: Use depth information for extra logic check

Visual Geometry Grounded Transformer
VGGT from Meta AI CVPR 2025

Feed forward neural network

- Input: n Images
- Architecture: DINOv2 for feature Extraction + L-Attention Blocks + feedward Layer
- Output: Camera parameters, depth maps and Point Cloud

Inference: 110 ms per 8 images

Hardware & Time: Training required 64 NVIDIA A100 GPUs running for approximately nine days

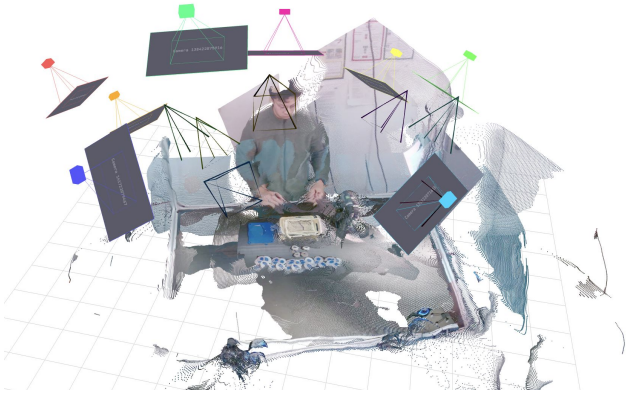


Fig 1.1 VGGT Prediction without Scaling in Three.js

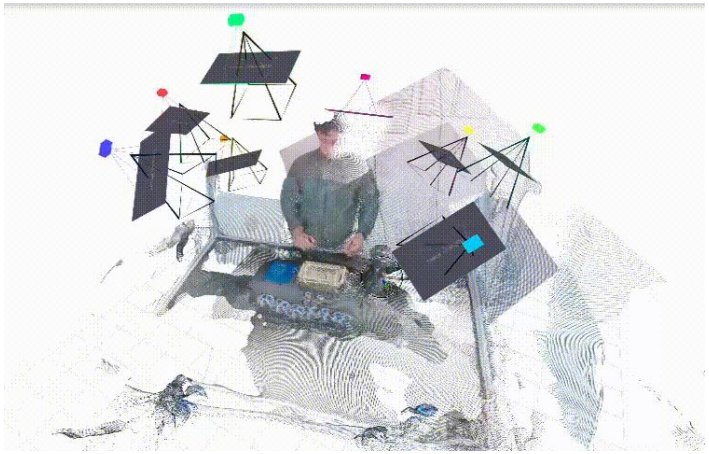


Fig 1.2 VGGT Prediction with 40% global Scaling in Three.js

Scene Reconstruction Metrics

Limitation: bad performance for extreme rotations, scaling factors necessary.

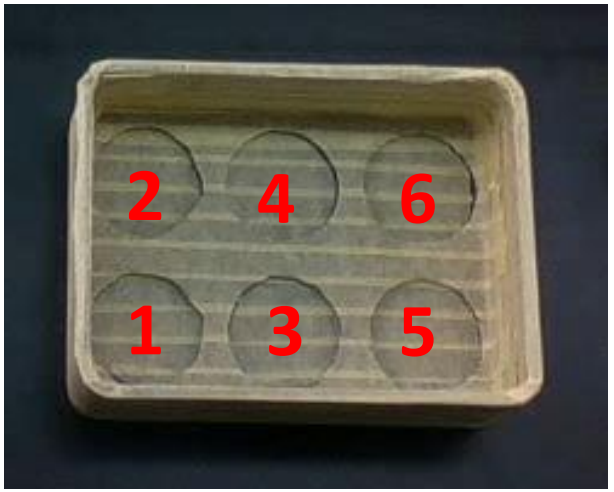
Parameter	Recommended Metric	Unit	Good Performance (Approx)
Rotation	Geodesic Angle	Degrees	<5
Translation	Euclidean Distance (L2)	Meters/Units	< 0.1 units (scale dependent)
Focal Length	Relative Error	Percent ($%$)	< 2-3%

Error Detection in Battery Insertion Order

Lucía Balsa Picado

Problem Definition

- Expected sequence:
  1 → 2 → … → 6
- Users may:
  - Break sequence
  - Correct themselves
- System must:
  - Detect error
  - Identify expected step
  - Work across != camera views



Sub-System Pipeline

Video

↓

YOLO Segmentation

↓

Frame-level geometry extraction

↓

Sequence modeling

- **FSM** (rule-based)
- **LSTM/Transformer** (learning-based)

↓

Error detection + user feedback

Frame-Level Data

Element	FSM pipeline	LSTM/Transformer pipeline
Case polygon	✔	✖
Case bbox	✔	✔
Case mask	✔	✖
Battery polygon	✔	✖
Battery centroid	✔	✔
Battery mask	✔	✖
Battery count	✔	✔

FSM-Based Error Detection

- Case → **6 fixed slots**
- Slot-level FSM: **EMPTY → ENTERING → INSERTED → EXITING**
- Order-level FSM: **1 → 2 → … → 6**

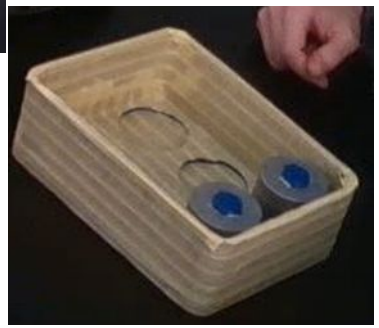
Derived quantity	**Source**
Slot masks	Case polygon
Battery–slot IoU	Battery mask × slot mask
Slot assignment	Highest Battery–slot IoU

FSM: Strengths & Limitations

```
[Frame 0100] Slot 1: ENTERING
[Frame 0101] Slot 1: INSERTED ✓
[Frame 0101] ✅ COMMITTED: Slot 1 | Order=[1]
[Frame 0165] Slot 3: ENTERING
[Frame 0166] Slot 3: INSERTED ✓
[Frame 0166] ⚠️ WRONG SLOT: 3 inserted (expected 2)
[Frame 0210] 🔙 Slot 3: REMOVED
[Frame 0210] ✅ Correction: Slot 3 cleared
[Frame 0277] Slot 2: ENTERING
[Frame 0278] Slot 2: INSERTED ✓
[Frame 0278] ✅ COMMITTED: Slot 2 | Order=[1, 2]
[Frame 0343] Slot 3: ENTERING
[Frame 0344] Slot 3: INSERTED ✓
[Frame 0344] ✅ COMMITTED: Slot 3 | Order=[1, 2, 3]
[Frame 0411] Slot 4: ENTERING
[Frame 0412] Slot 4: INSERTED ✓
[Frame 0412] ✅ COMMITTED: Slot 4 | Order=[1, 2, 3, 4]
[Frame 0491] Slot 5: ENTERING
[Frame 0492] Slot 5: INSERTED ✓
[Frame 0492] ✅ COMMITTED: Slot 5 | Order=[1, 2, 3, 4, 5]
[Frame 0587] Slot 6: ENTERING
[Frame 0588] Slot 6: INSERTED ✓
[Frame 0588] ✅ COMMITTED: Slot 6 | Order=[1, 2, 3, 4, 5, 6]

============================================================
FINAL INSERTION ORDER: [1, 2, 3, 4, 5, 6]
============================================================
✅ SUCCESS! All batteries inserted in correct order!
```

- **Pros**
  - High precision
  - Easy to debug
  - Expected next state
- **Cons**
  - Top camera view



LSTM-Based Error Detection

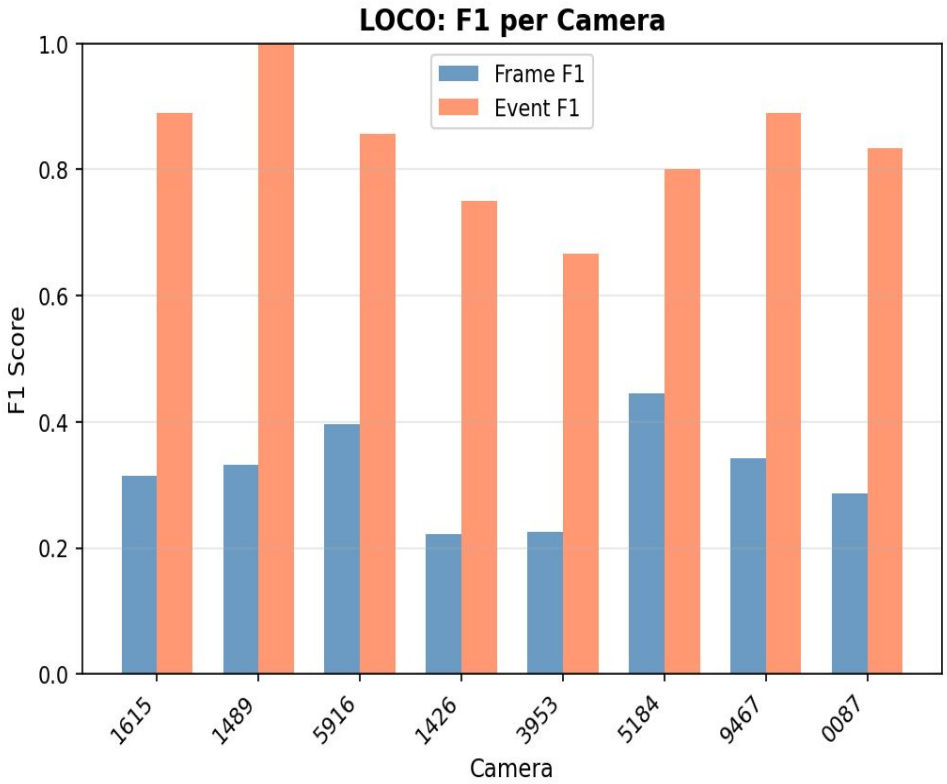
- **Input:** fixed-length vector
  - Number of batteries
  - Battery centroids
  - Pairwise centroid distances
  - Case bbox (center, size)
  - Centroid statistics (avg, std)
- **Architecture**
  - 2-layer LSTM
  - Hidden size: 128
  - Dropout: 0.3
  - Lr: 1e-4
  - Loss:
    - Masked BCE
    - Class imbalance weighting
- **Output**
  - Frame-level error probability → Thresholding

Evaluation Strategy: Leave-One-Camera-Out

- **LOCO cross-validation**
  - Train: 7 cameras
  - Test: 1 held-out camera

- **Metrics**
  - Frame-level Precision / Recall / F1
  - Event-level Precision / Recall / F1 ← Grouping consecutive error frames

LOCO Results



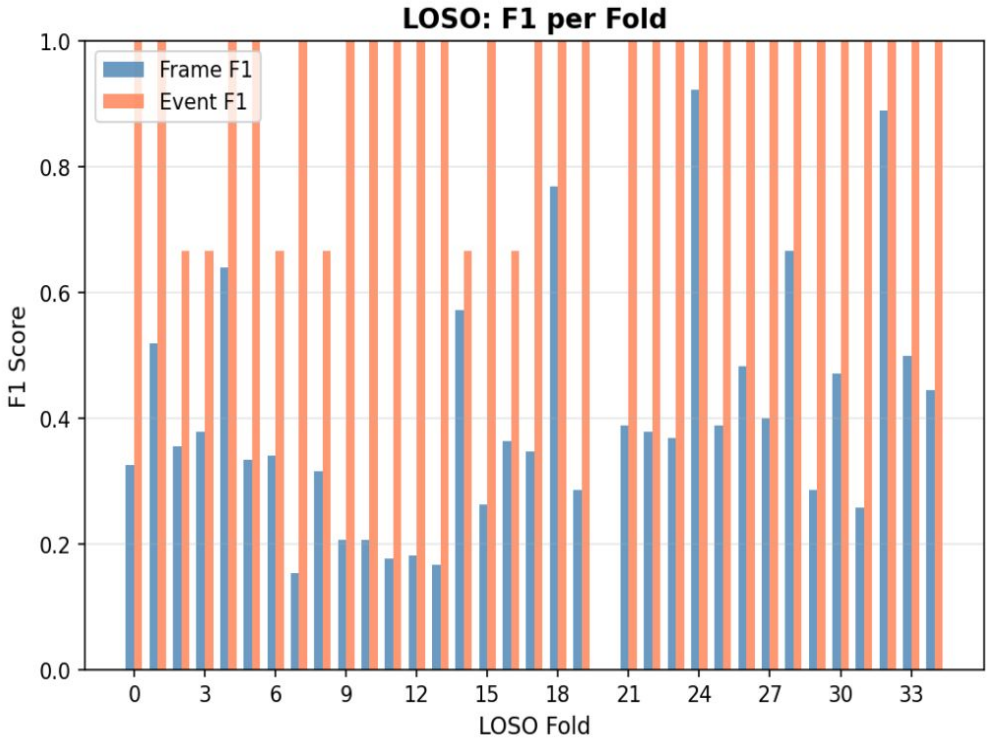

```
================================================
LOCO CROSS-VALIDATION RESULTS
================================================

📊 AGGREGATED FRAME METRICS (all cameras combined):
   Precision: 0.191
   Recall:    0.815
   F1:        0.310

📈 AVERAGE METRICS (macro-average across cameras):
   Frame: P=0.203 ± 0.059
          R=0.819 ± 0.126
          F1=0.320 ± 0.072

   Event: P=0.833 ± 0.142
          R=0.856 ± 0.113
          F1=0.836 ± 0.094
```

Leave-One-Sequence-Out (LOSO) Results



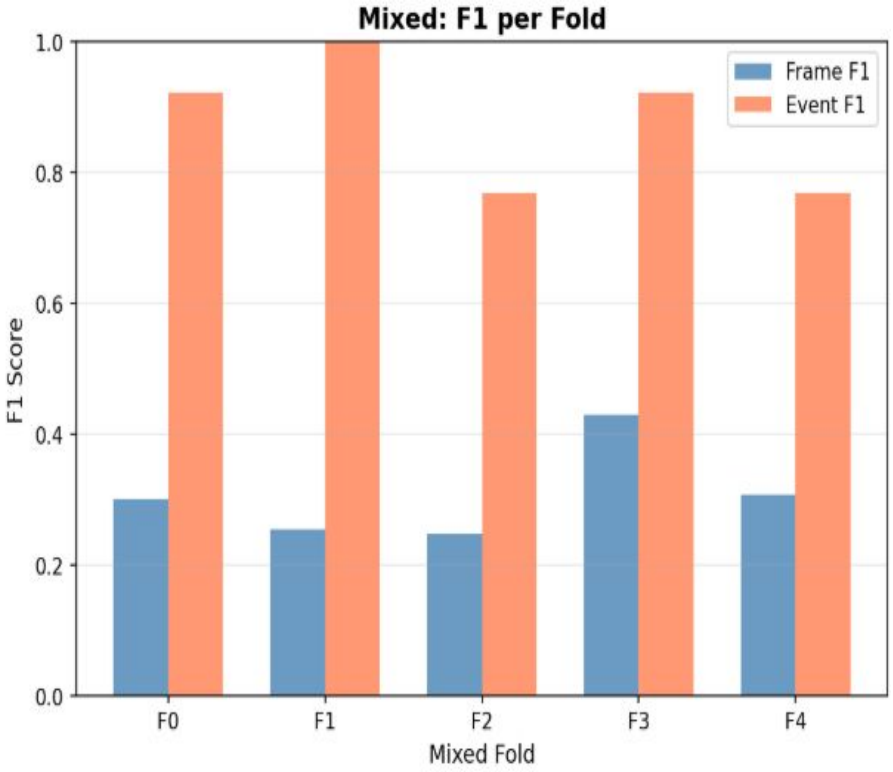

```
============================================================
LOSO CROSS-VALIDATION RESULTS
============================================================

📊 AGGREGATED FRAME METRICS (all sequences combined)
   Precision: 0.219
   Recall:    0.931
   F1:        0.355

📈 AVERAGE METRICS (macro-average across folds):
   Frame: P=0.275 ± 0.201
          R=0.932 ± 0.176
          F1=0.393 ± 0.199

   Event: P=0.886 ± 0.242
          R=0.971 ± 0.167
          F1=0.914 ± 0.201
```

Mixed Approach Results




```
============================================================
MIXED CROSS-VALIDATION RESULTS
============================================================

📊 AGGREGATED FRAME METRICS (all sequences combined):
   Precision: 0.190
   Recall:    0.772
   F1:        0.304

📈 AVERAGE METRICS (macro-average across folds):
   Frame: P=0.196 ± 0.053
          R=0.788 ± 0.136
          F1=0.308 ± 0.065

   Event: P=0.933 ± 0.082
          R=0.829 ± 0.107
          F1=0.877 ± 0.092
```

Transformer-Based Error Detection

- **Architecture**
  - Model dim: 64
  - Attention heads: 8
  - Encoder layers: 3
  - Feed-forward hidden dim: 256
  - Dropout: 0.1
  - Lr: 1e-4

```
================================================
LOCO CROSS-VALIDATION RESULTS
================================================

📊 AGGREGATED FRAME METRICS (all cameras combined):
   Precision: 0.175
   Recall:    0.815
   F1:        0.288

📈 AVERAGE METRICS (macro-average across cameras):
   Frame: P=0.173 ± 0.030
          R=0.810 ± 0.215
          F1=0.283 ± 0.051

   Event: P=0.762 ± 0.172
          R=0.863 ± 0.193
          F1=0.806 ± 0.173
```

Final LSTM Model

```
Global (all cameras combined):
  Frame:  P=0.229 R=0.688 F1=0.344
  Event:  P=1.000 R=0.750 F1=0.857

Macro Average (across 8 cameras):
  Frame:  P=0.206±0.157 R=0.688±0.428 F1=0.307±0.220
  Event:  P=0.750±0.433 R=0.750±0.433 F1=0.750±0.433
```

```
Camera 138422075916:
  Total frames: 88
  Match: ✓ (Prediction overlaps ground truth)

  Metric                    PREDICTED                    GROUND TRUTH
  ------------------------------------------------------------------------
  Events detected           1                            1
  Array indices             [(14, 22)]                   [(18, 21)]
  Original frame_idx        [(210, 330)]                 [(270, 315)]
  Normalized time [0,1]     [('0.159', '0.261')]         [('0.205', '0.250')]
```

Windowed LSTM Model

- Window: past + current frames
- Size: 15
- Stride: 8

```
Global (all cameras combined):
  Frame:  P=0.167 R=0.812 F1=0.277
  Event:  P=0.700 R=0.875 F1=0.778

Macro Average (across 8 cameras):
  Frame:  P=0.179±0.102 R=0.812±0.348 F1=0.286±0.149
  Event:  P=0.750±0.354 R=0.875±0.331 F1=0.792±0.331
```

- Inter-camera range variability → Intersection
- **Pro:**
  - Camera generalization
- **Con:**
  - Latency

Deployed Models

- **LSTM**
  - Multi-camera intersection

- **FSM**
  - Sequence tracking
  - User guidance

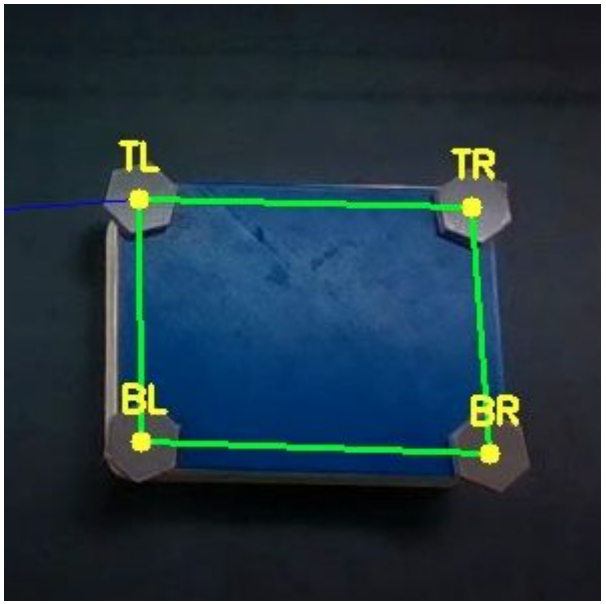
Error Detection in Screwing Order

Ioannis
Papadongonas

Problem Definition

- Expected sequence:
  BL -> TR -> BR -> TL

- System must:
  - Detect if the user is following the correct cross sequence logic

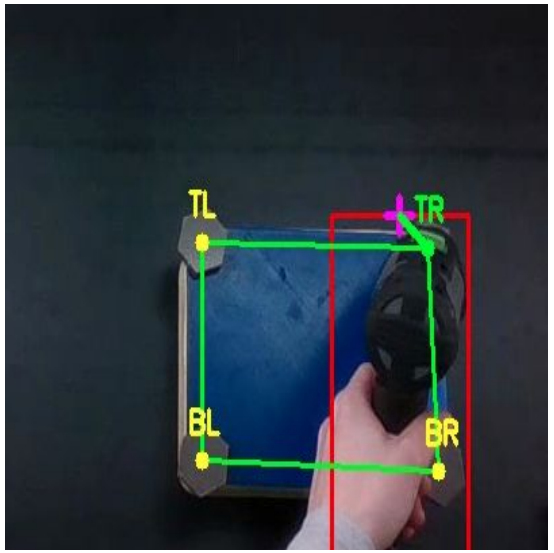


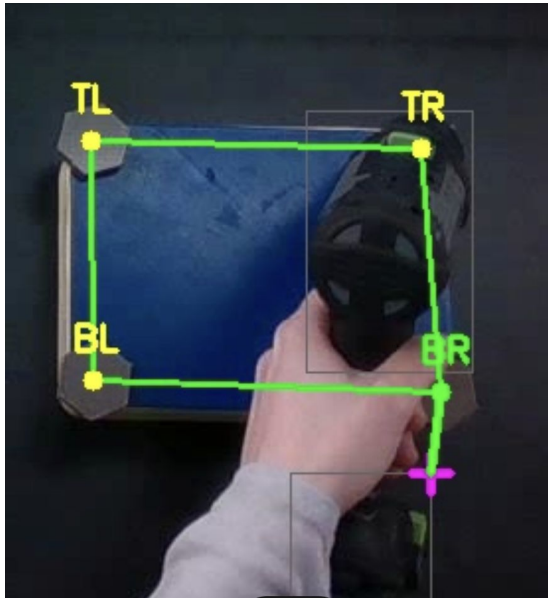
First Approach

YOLO:
Detects different classes and merges batteries, case and box in one "super box"

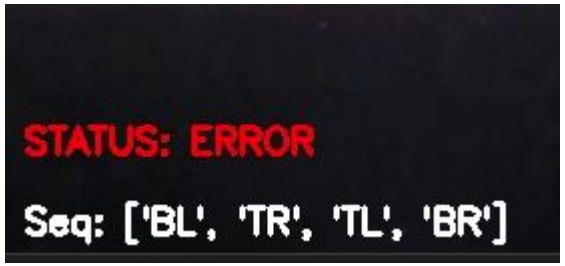
Away-Vector Logic:
Calculates a vector from the Person's Centroid to the Tool's Center, active drill tip as the point on the tool bounding box furthest from the operator.

Geometric Tracking :
Problem: Hand blocks one screw, tracker fails,
Solution: Affine Transformations, if we have 3 screws, we calculate the 4th
Result: The green box never distorts or vanishes





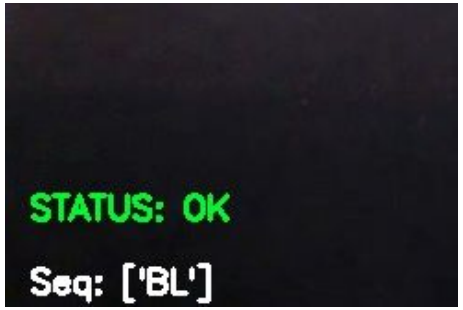
First Approach



Deterministic State Machine (FSM)
Input: Coordinates of the Tool Tip and Screw Centers.
Filter: Requires the tool tip to remain within a target's Strict_Hit_Radius (75px) for N frames to distinguish an intentional action from accidental motion

Strict Sequence Enforcement Validation:
System refuses to log a step as correct unless it matches the exact required order (e.g., BL → TR).
Lingering Handler: If the user rests on a previously completed screw, the system ignores it rather than flagging a false positive error.
Strict Sequencing: If the operator skips a screw or makes a mistake the system perceives it as wrong.

Real-Time Feedback Loop
Text Feedback: Translates technical state into human instructions.



Final Approach

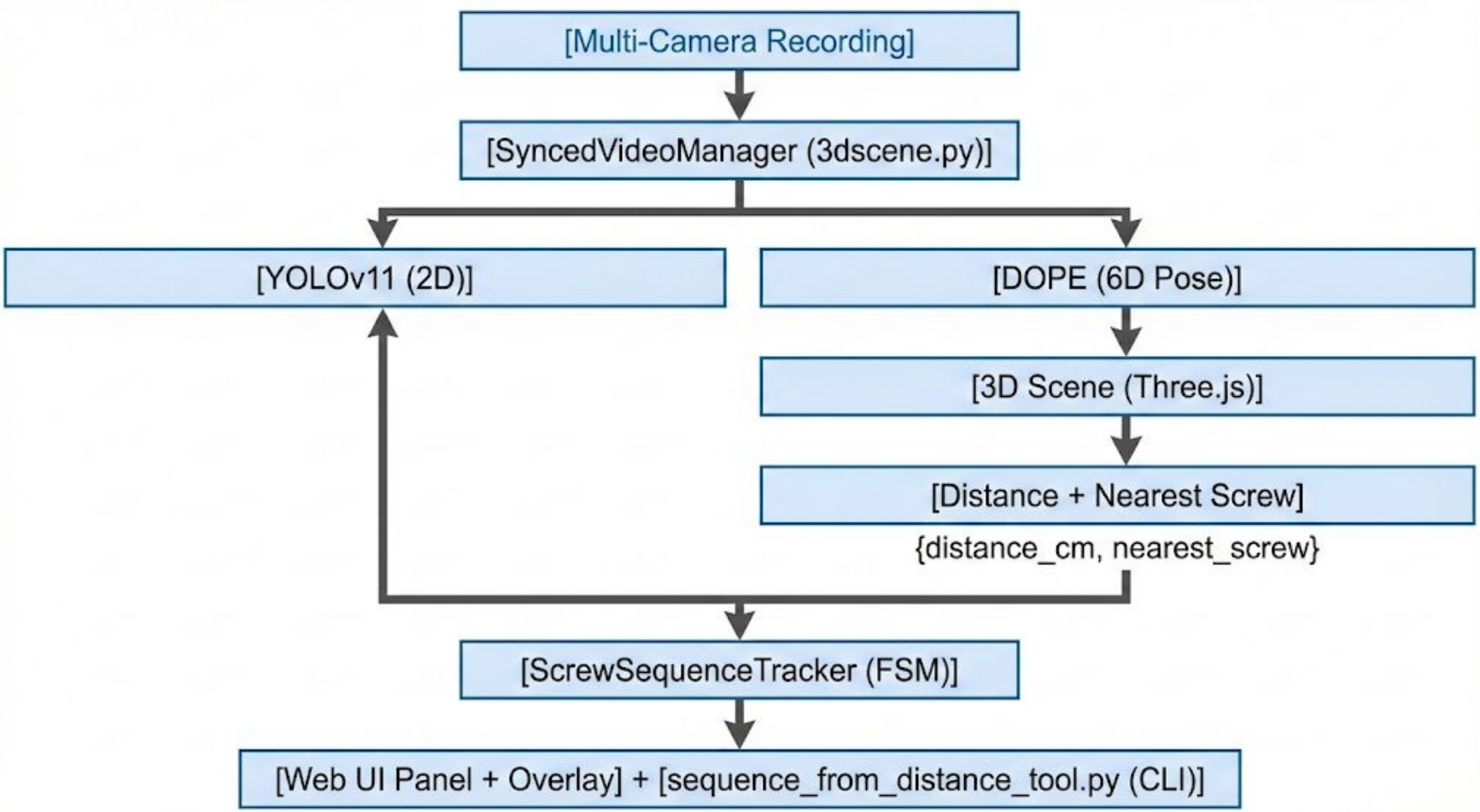
DOPE: estimates tool and cases poses (transparent bounding boxes)

YOLO: secondary role and for visualization purposes

FSM: tracks and determines correctness of screw usage

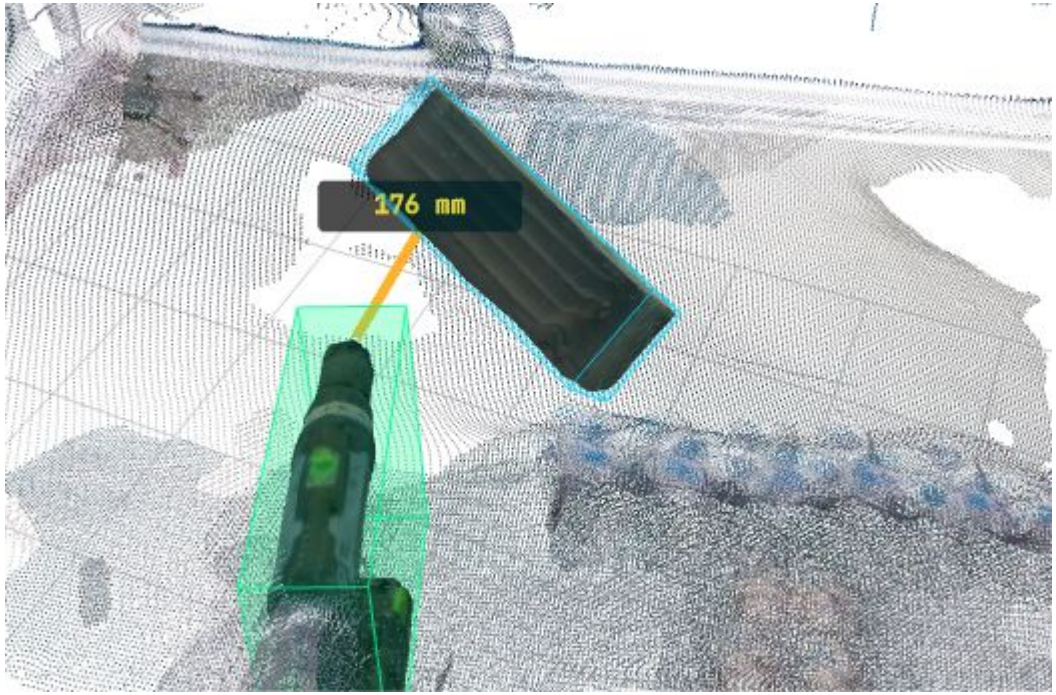
GUI: frontend places the models in 3d and offers individual camera view and screws sequence tracking feedback

Pipeline


[Multi-Camera Recording]
[SyncedVideoManager (3dscene.py)]
[YOLOv11 (2D)]
[DOPE (6D Pose)]
[3D Scene (Three.js)]
[Distance + Nearest Screw]
{distance_cm, nearest_screw}
[ScrewSequenceTracker (FSM)]
[Web UI Panel + Overlay] + [sequence_from_distance_tool.py (CLI)]

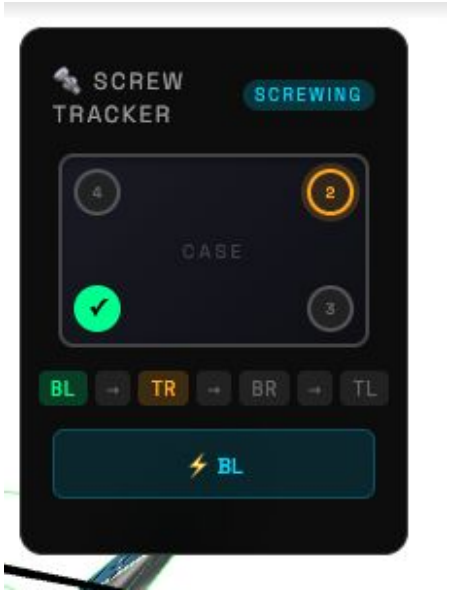
Final Approach - Distance and Nearest Screw

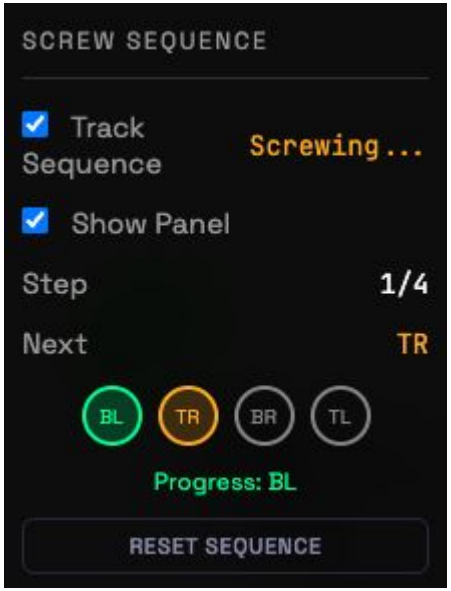
- **Detection:** 4 top vertices of the case bounding box
- **Computation:** distance from tool centroid to each corner
- **Selection:** nearest corner -> nearest screw
- **Notification:** frontend and tracking mechanism

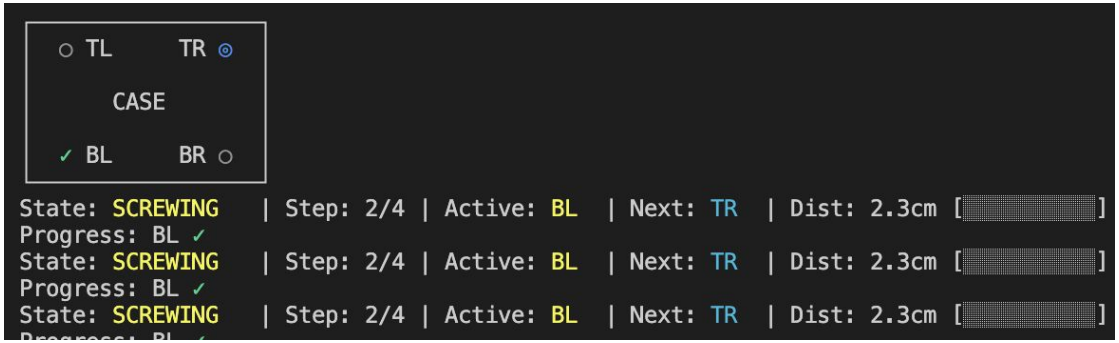


Final Approach - Screw Sequence Tracker

- Screw Sequence: BL -> TR -> BR -> TL
- Sequence Status stores expected vs actual sequence.
- 3D Driven State Machine (distance_cm, nearest_screw):
  - If distance <= threshold -> Screwing state
  - If Screwing state frames > frames to complete -> complete screw
  - Correctness is checked against defined sequence

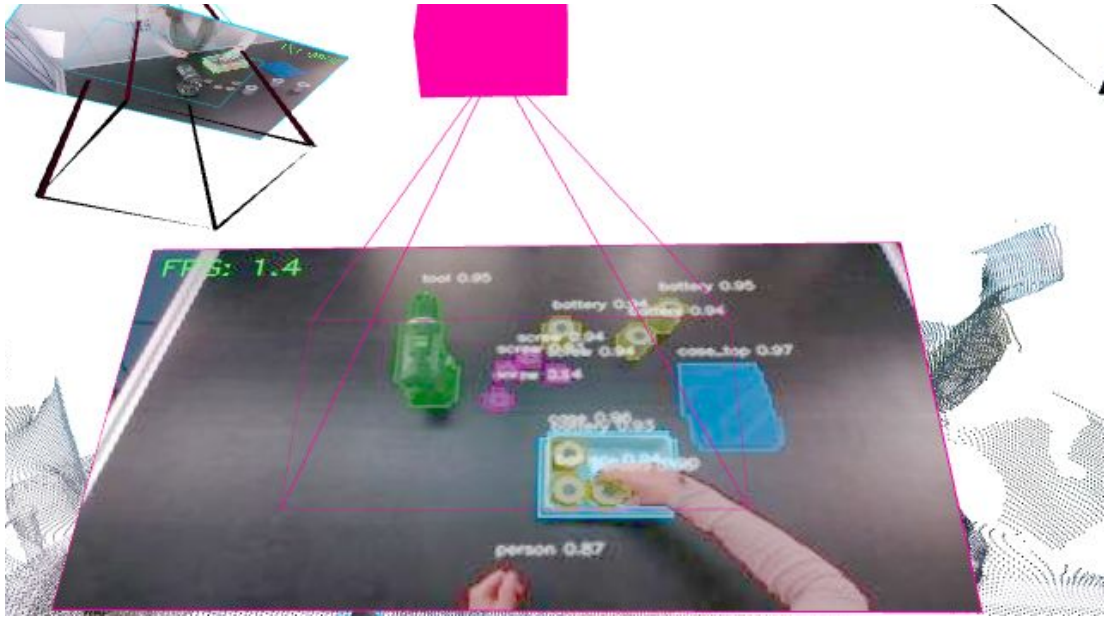






Final Approach - Rolo of YOLO

- YOLO runs on one camera to detect screws + tool in 2D
- Classifies screen corners and stabilize pixel positions
- Overlays drawn on camera frames in UI for better visualization and detection of the objects
- 3D distance is primary source for deciding when a screw is completed, due to YOLO inconsistencies (tool / hand covering super box)



Final Approach - Efficiency (DOPE_STOP_AT_ STAGE = 1, dope_inference_interval = 5)

Recording	Status	Sequence	Correct	Detected Correct
recording_1	SUPPORTED	BL -> TR -> BR -> TL	Yes	Yes
recording_2	SUPPORTED	BL -> TR -> BR -> TL	Yes	Yes
recording_3 (default)	SUPPORTED	BL -> TR -> BR -> TL	Yes	Yes
recording_4	SUPPORTED	BL -> TR -> BR -> TL	Yes	Yes, needs DOPE higher settings (DOPE_STOP_AT_STAGE =3, dope_inference_interval = 3)
recording_5	SUPPORTED	BL -> TR -> BR -> TL	Yes	Yes
recording_6	SUPPORTED	TL -> BL -> BR -> (BL) -> TR	No	Yes
recording_7	SUPPORTED	BL -> TR -> TL -> BR	No	No, needs better performance settings (< 1 fps)
recording_8	SUPPORTED	BL -> TR -> BR -> TL	Yes	No, TL time is less than threshold so its not detected as done
recording_9	SUPPORTED	BL -> BR -> TR -> (BR) -> TL	No	Yes, needs DOPE higher settings (DOPE_STOP_AT_STAGE =3, dope_inference_interval = 3)
recording_10	SUPPORTED	BL -> BR -> TR -> (BR) -> TL	No	Yes, needs DOPE higher settings (DOPE_STOP_AT_STAGE =3, dope_inference_interval = 3)
recording_11	SUPPORTED	BL -> TR -> BR -> TL	Yes	Yes
recording_12	SUPPORTED	BL -> TR -> BR -> TL	Yes	Yes

Failed Attempts

- **Static Region of Interest Hardcoding Method:**
  Initially defined fixed pixel coordinates for the screws (e.g., x=200, y=300), assuming the box would never move.
  Why it Failed: The moment the camera angle changed or the operator bumped into the box, the system broke. It lacked invariance to translation and rotation.
- **Naive "Top-Bias" Tip Detection Method:**
  Assumed the drill tip was always the highest point of the tool's bounding box.
  Why it Failed: This only worked when the operator stood at the bottom of the screen. When the tool entered from the top or side (e.g., inverted views), the system mistakenly identified the battery pack as the drill tip.
- **Global Video Rotation Method:**
  Attempted to automatically rotate the entire video feed based on the tool's entry point to force a "standard upright" view.
  Why it Failed: This confused the pre-trained YOLO model, which expects upright objects. It introduced jitter and made the coordinate system difficult to map back to the real world.

Demo & Conclusion

Questions?

References

- https://www.ultralytics.com/
- https://github.com/NVlabs/Deep_Object_Pose
- https://vgg-t.github.io/
- https://docs.blender.org/api/current/info_quickstart.html
- https://docs.pytorch.org/docs/stable/index.html