EdgeAI Academy Summer School 2025

Autonomous Mobile Robots in Smart Manufacturing: A Real-World Approach to Edge AI and Computer Vision

Matthew Keating **(Dell Technologies)** - matthew.keating@dell.com

Ayaan Verghese **(Dell Technologies)** - ayaan.verghese@dell.com

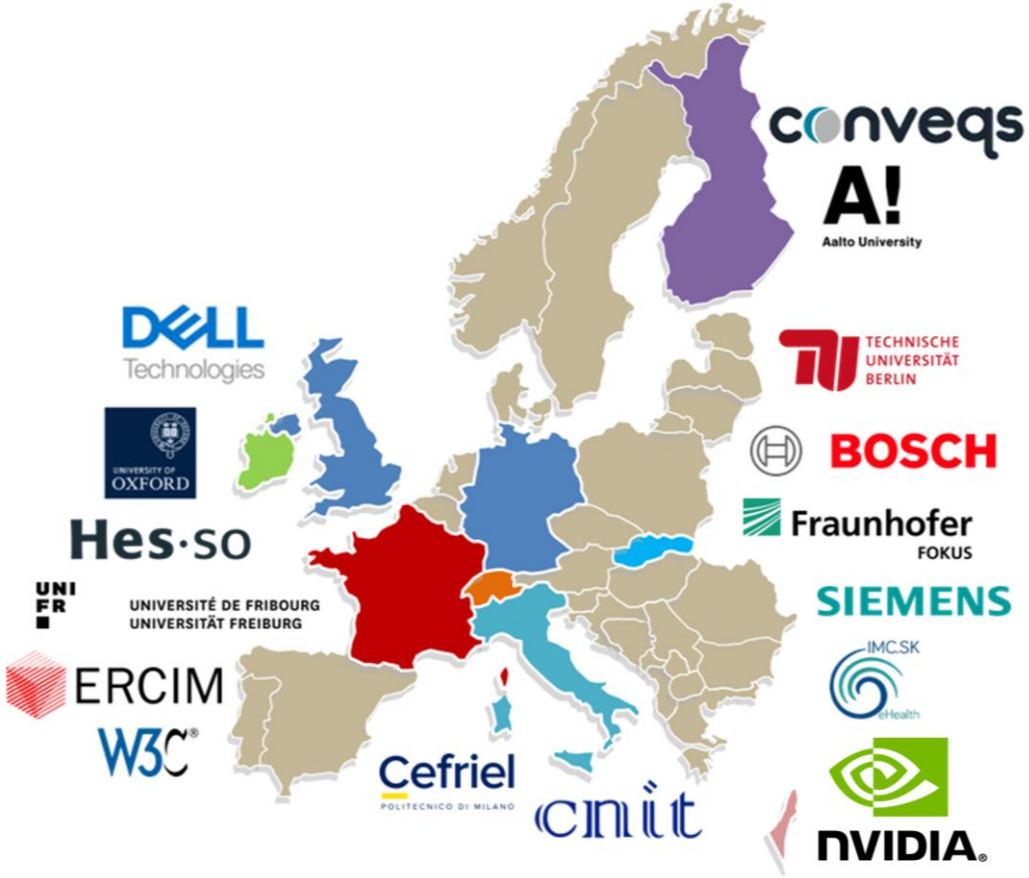
SMARTEDGE Project

Grant No. 101092908

https://www.smart-edge.eu/

SmartEdge Project Overview

- HORIZON-CL4-2022- DATA-01-03: Programming tools for decentralized intelligence and swarms
- Program: AI, Data, and Robotics
- Duration 36 Months
  - Start Date : 1st January 2023
  - End Date : 31st December 2025
- https://www.smart-edge.eu/

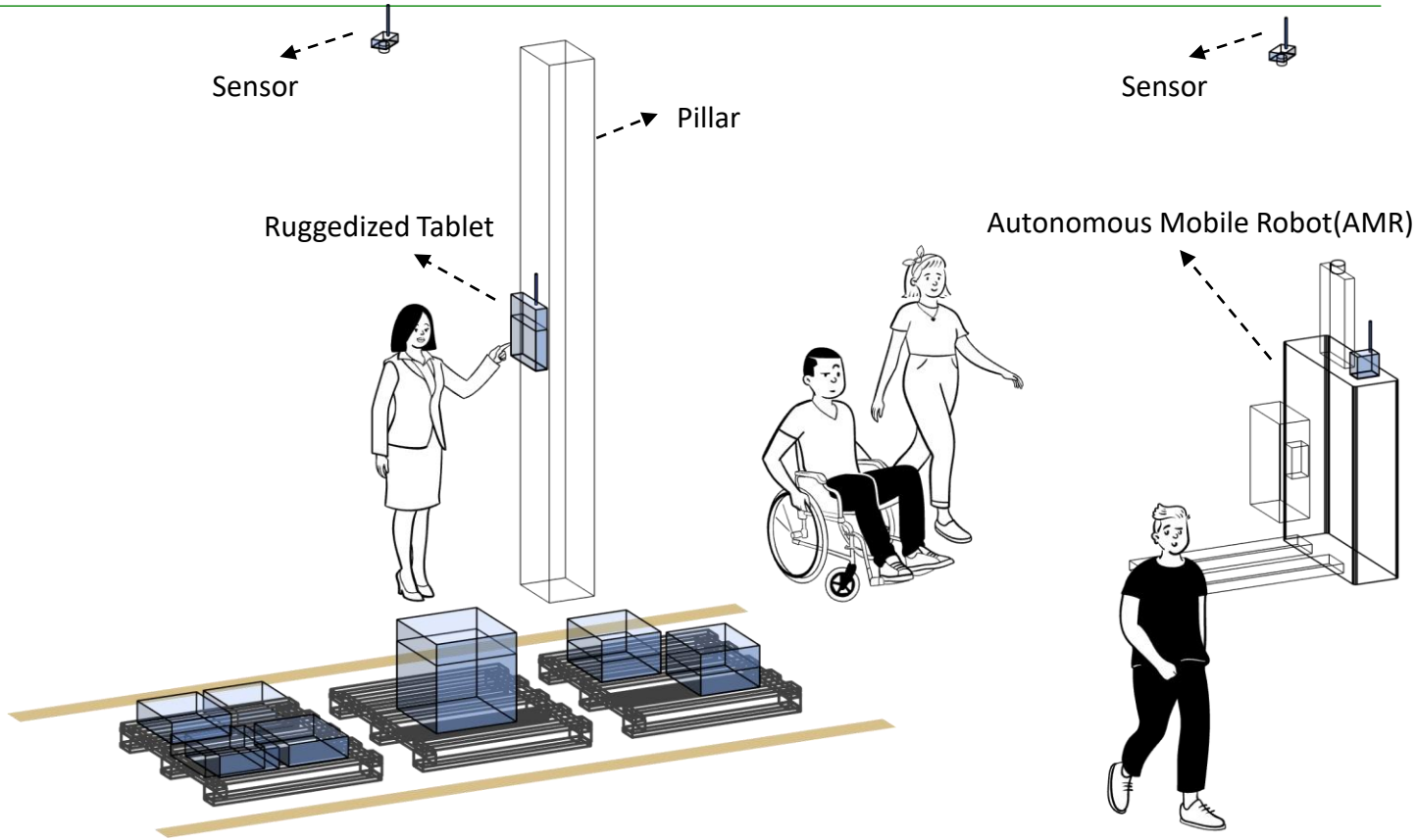


Solving a Real Manufacturing Problem with SmartEdge Swarms

Dell Ruggedized Tablet

PowerEdge XR4000z Server

5200 Edge Gateway



Challenges in Scene Understanding : Manufacturing Setting

- Lot of background noise in manufacturing scenes.
- Occlusions: Parts of the scene may be blocked by machinery, workers, or materials.
- Safety-critical environments that demand high model reliability.
- Objects in scene are custom to the factory- Lack of labelled data publicly available.

Pallets in a Manufacturing Setting

Half Euro Plastic Pallet

Half Euro Wood Pallet

Euro Wood Pallet

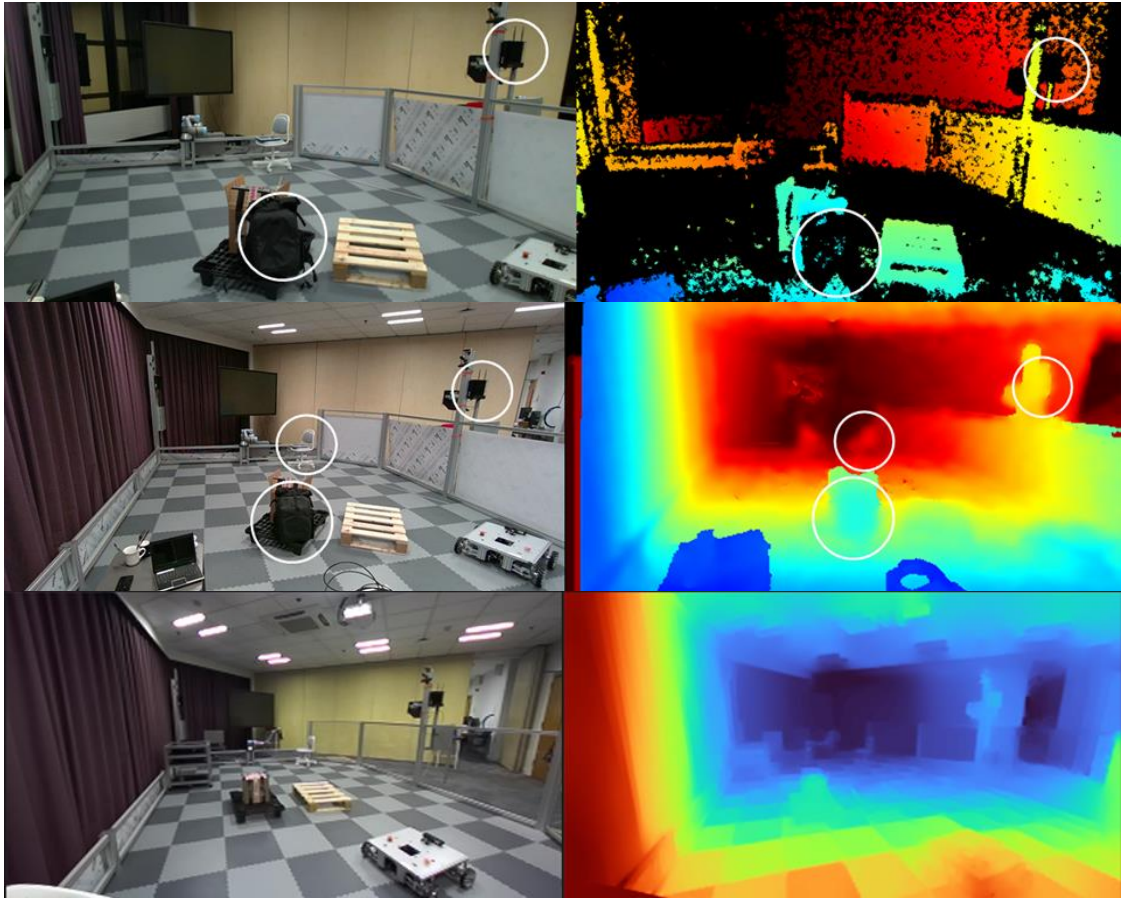
Half Euro Wood Pallet- Features Occluded by box

Sensor Testing in Lab

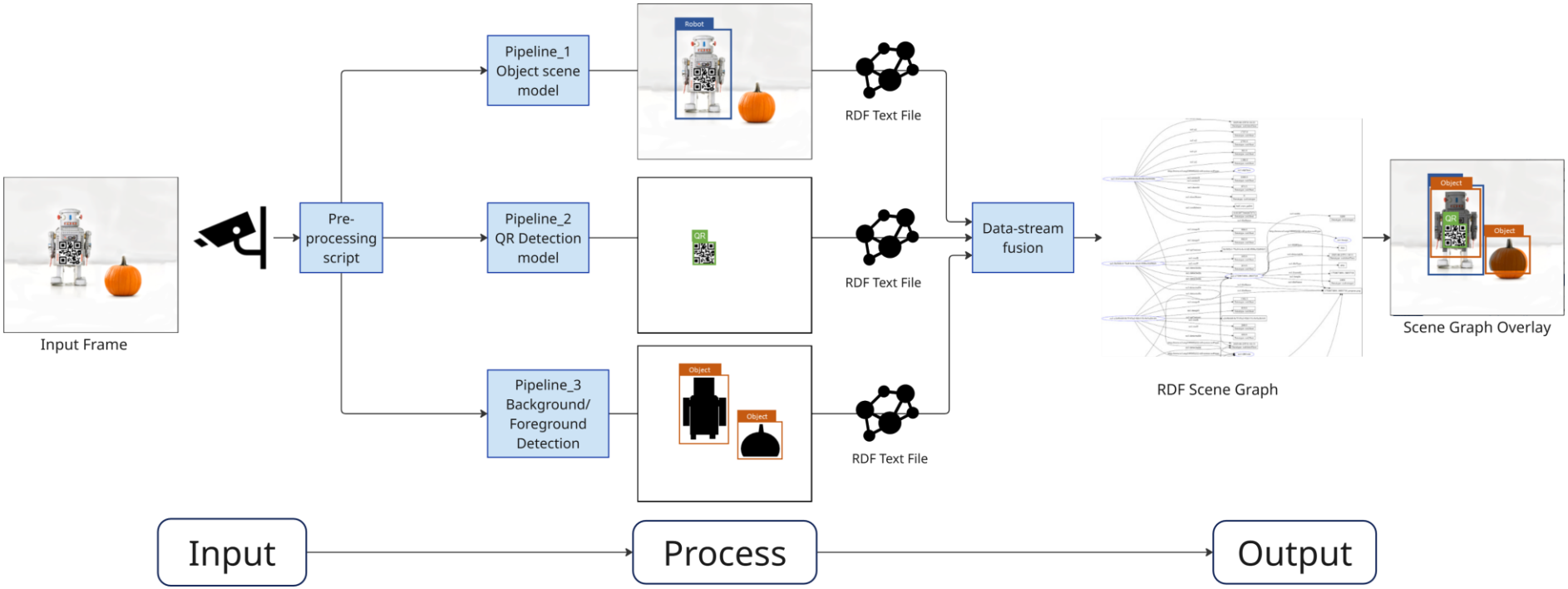
LiDAR

Stereophotogrammetry Camera (A)

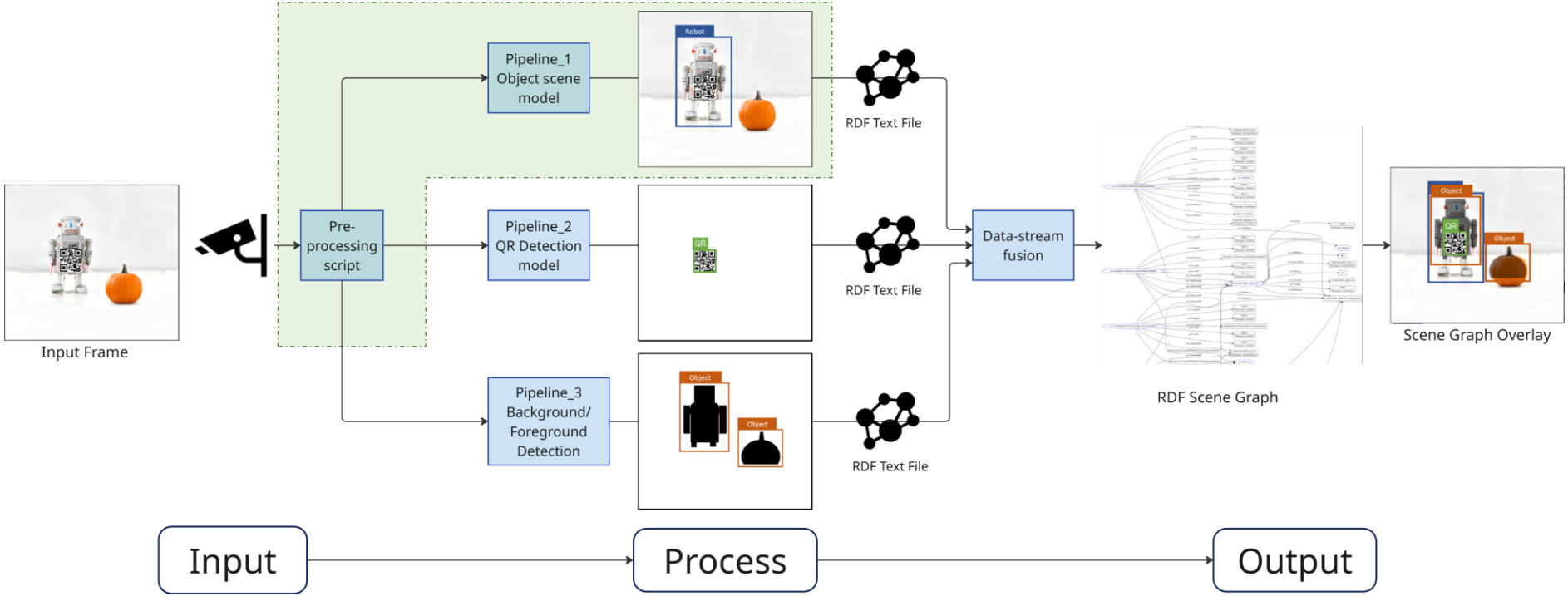
Stereophotogrammetry Camera (B)



Manufacturing Scene Understanding

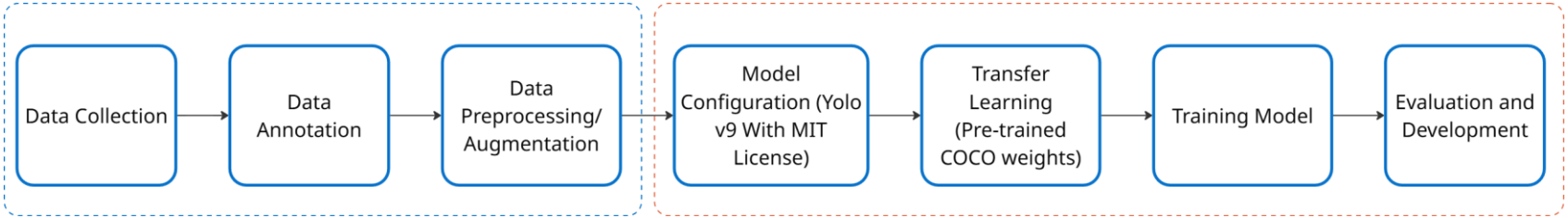


Pipeline 1: Object Scene Model



Training Custom Object Detection Models

- Aim: Identify different types of pallets as well as common objects that are found in a manufacturing scene.
- Process:



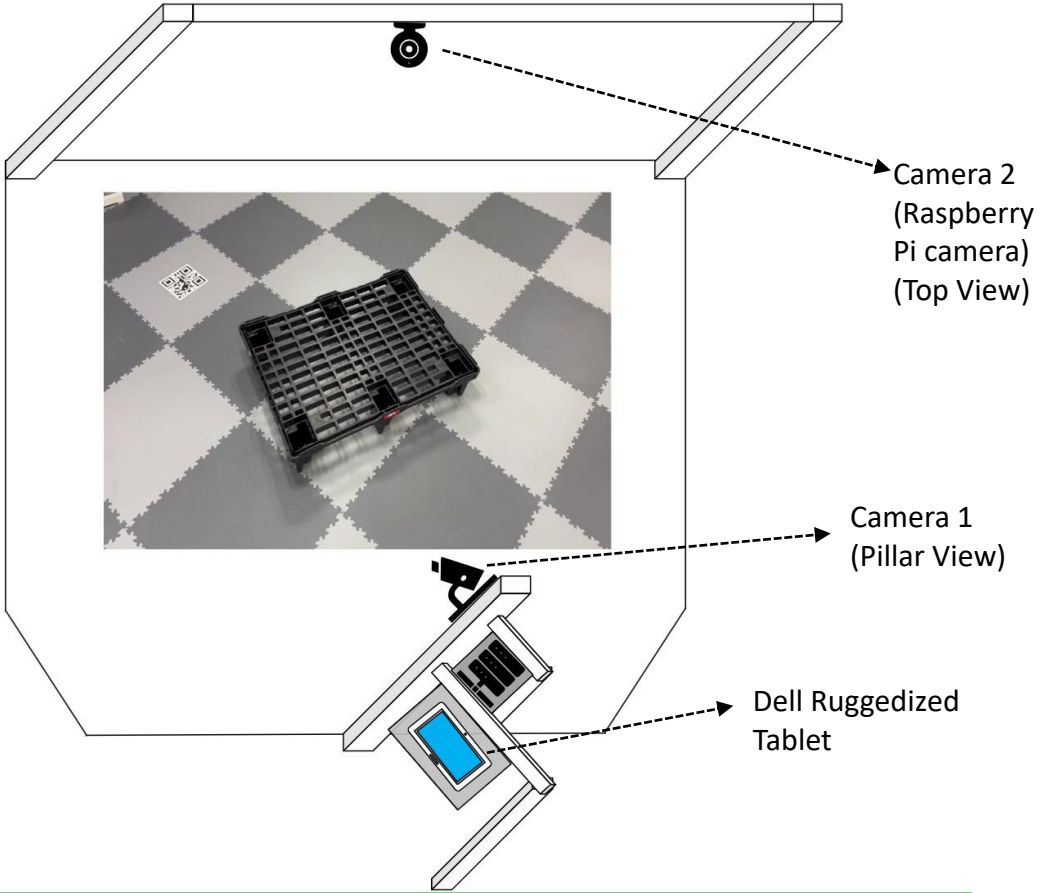
- 2 Datasets were used for this pipeline:
  - Created a dataset to detect a **custom Plastic Half Euro Pallet**:
    - ✓ Small dataset which consists of 850 images of half euro pallet in different backgrounds.
    - ✓ More images will be added later using 3D CAD models, Data augmentation to the training set.
  - LOCO: Logistics Object in Context Dataset
    - ✓ Consists of 5593 images that have been manually annotated, 152,421 total annotations.
    - ✓ 5 Classes: Forklift, Pallet Truck, Pallet, Small Load Carrier, Stillage

GitHub - MultimediaTechLab/YOLO: An MIT License of YOLOv9, YOLOv7, YOLO-RD

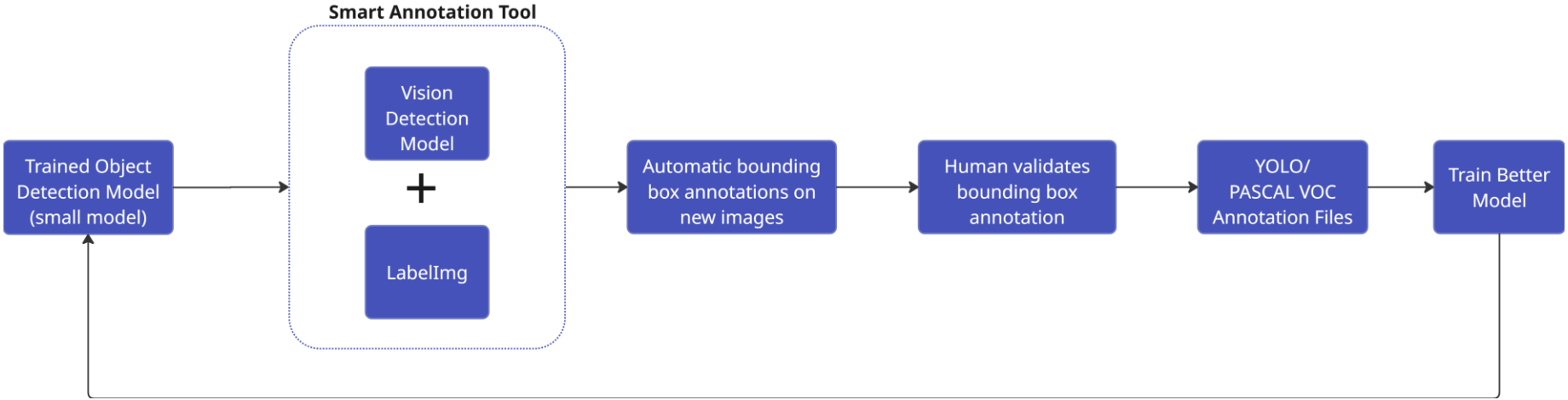
LOCO: Logistics Objects in Context ,Mayershofer, C., Holm, D.-M., Molter, B., Fottner, J. ,IEEE International Conference on Machine Learning and Applications (ICMLA) 2020

Multi View Inference with Yolo v9 – Custom pallet dataset

- Reduces Occlusion
- Improves Spatial Understanding
- Increases Robustness
- Boosts Detection Confidence
- Can be scaled to Multi-Camera Systems



Smart Annotation Tool



- Challenges during model data collection in manufacturing settings:
  - Most of the objects in a factory are custom objects.
  - Not many manufacturing datasets readily available to train object detection algorithms.
  - Class imbalance: Some objects appear more frequently than others.
  - As process' change in factories, model will need to be continuously updated.
  - **Most Time Consuming: Manual Annotations**

Citation: Tzutalin. LabelImg. Git code (2015). https://github.com/tzutalin/labelImg
GitHub - MultimediaTechLab/YOLO: An MIT License of YOLOv9, YOLOv7, YOLO-RD

A model to transform from Pixel Points to Physical Space

Problem

Autonomous Mobile Robots (AMRs) require precise real-world positioning information to interact with their environment.

Can we map image points to real-world points – without relying on high-cost/complexity systems like GPS or LiDAR

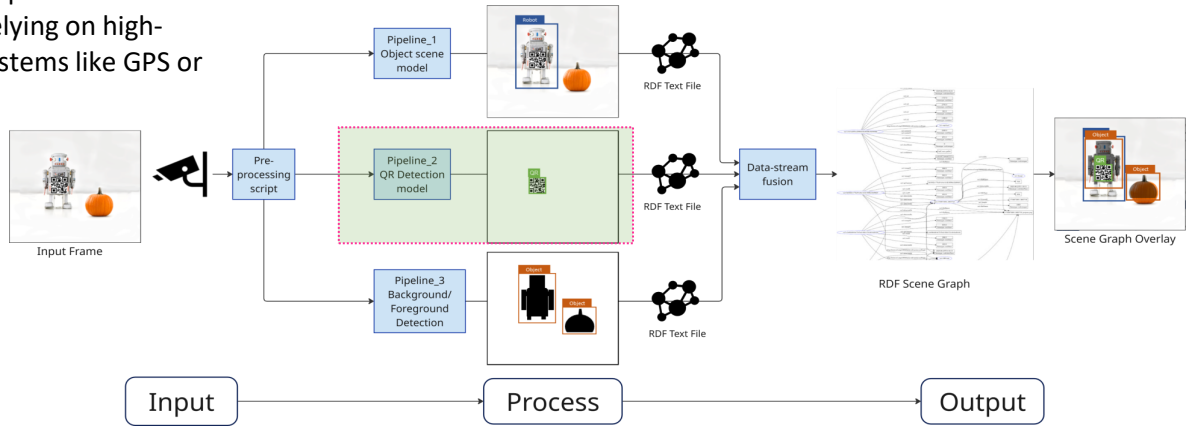
Use-case context - SmartEdge

Use QR Code's as 'anchors' to calibrate spatial relationships between 2D image points and 3D physical space

Expected challenges

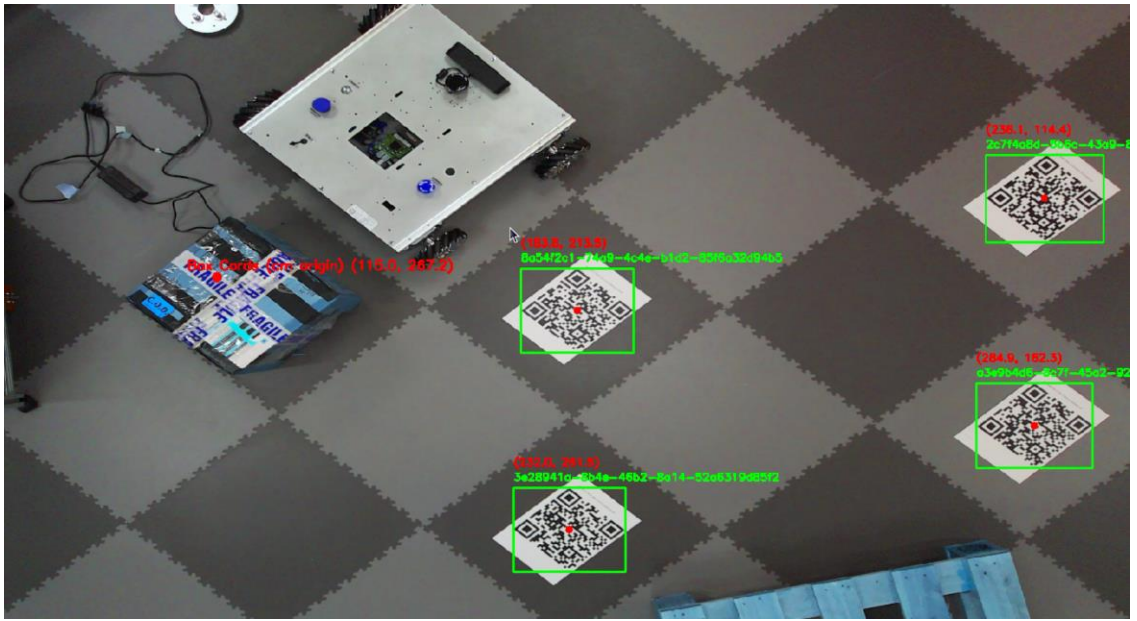
Planer assumption (2D to 2D) vs. real-world 3D / perspective distortions

Calibrating for the most efficient balance between cost, complexity, and compute cost



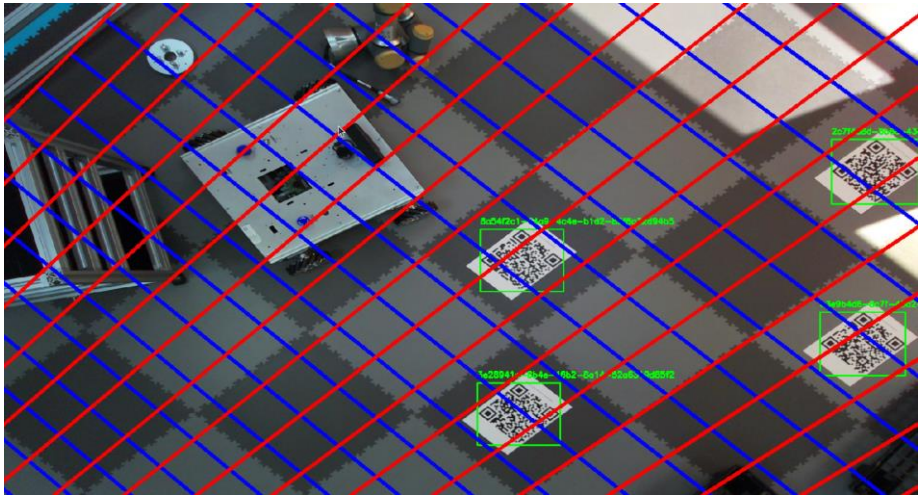
Designing a Cost-Effective Spatial Mapping System

- Objectives & Requirements
  - Create a lightweight, self-contained computer vision model for coordinate estimation
  - Modularity: Enable integration with other object detection models for holistic scene analysis.
  - Camera Sensor: Fixed or low-cost camera for input.
- Solution
  - QR Code Anchors: 4 QR codes placed at known positions on the manufacturing floor.
  - Homography (H) Matrix: Derive a transformation matrix to map image points to real-world coordinates.

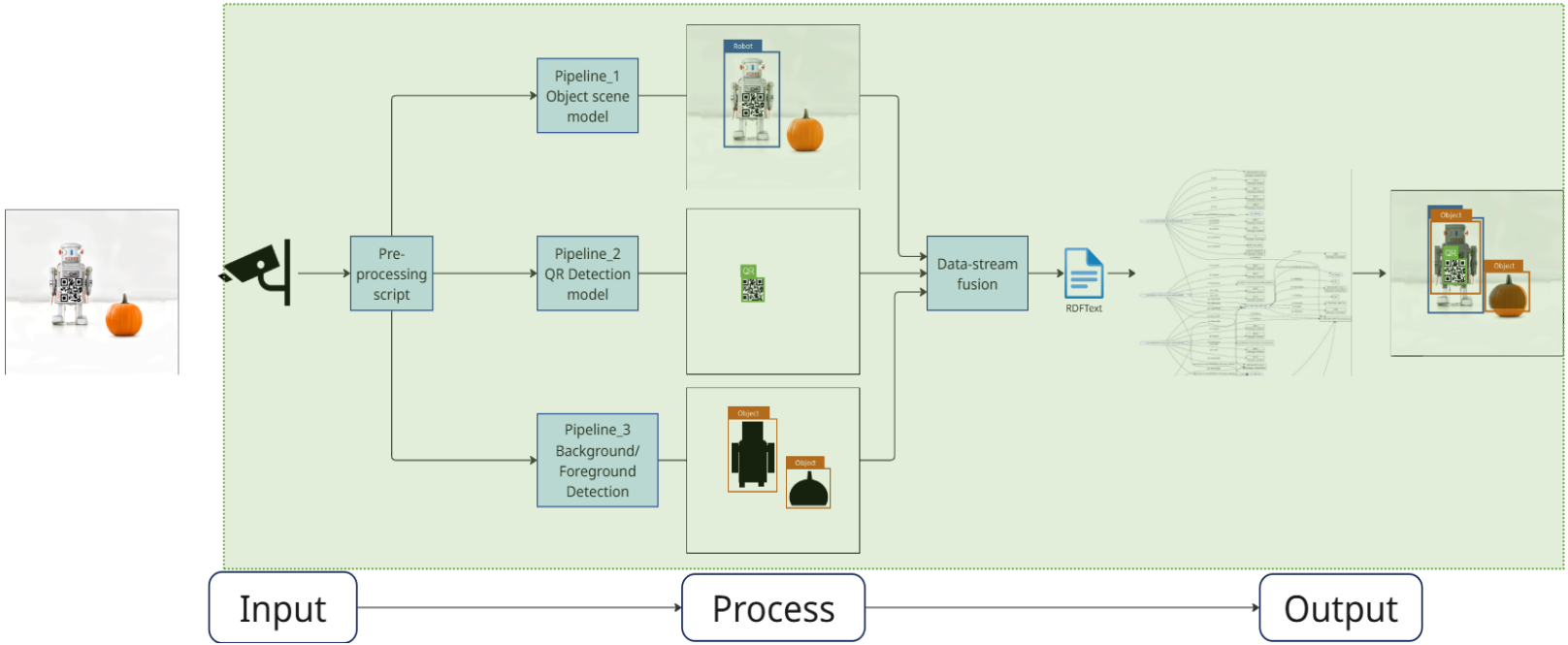


Technical Workflow: Calibration to Real-World Mapping

- Step 1: Camera Calibration
- Step 2: Homography (H) Matrix Generation
- Step 3: Apply H matrix to a given input point (X, Y) -> (rX, rY)
- Low-cost / High accuracy
  - ✓ Achieve reliable 2D / 3D point estimation accuracy using QR Codes and a single camera input
  - ✓ Distance calculation, object dimension extraction, robot / object localization
- Semantic Integration
  - ✓ Coordinates can be linked in our Manufacturing Scene Ontology
  - ✓ Enables spatial queries: “Objects within 1m of Robot3”, “Nearest charging station to Robot1”
- Findings (Pyzbar vs OpenCV for QR Code detection)
  - ❑ Pyzbar offers a more robust detection algorithms (ideal in dynamic environments)
  - ❑ OpenCV implements a native C library, enables a more efficient detection algo (no c-wrapper overhead)
- Limitations & Open Questions
  - ❑ Planer assumption: flat plane with camera mounted above
  - ❑ Scalability: Is it possible to come H matrices across multiple camera devices – requires synchronization



Modular Pipeline for Semantic Scene Understanding



Modular Pipeline for Semantic Scene Understanding

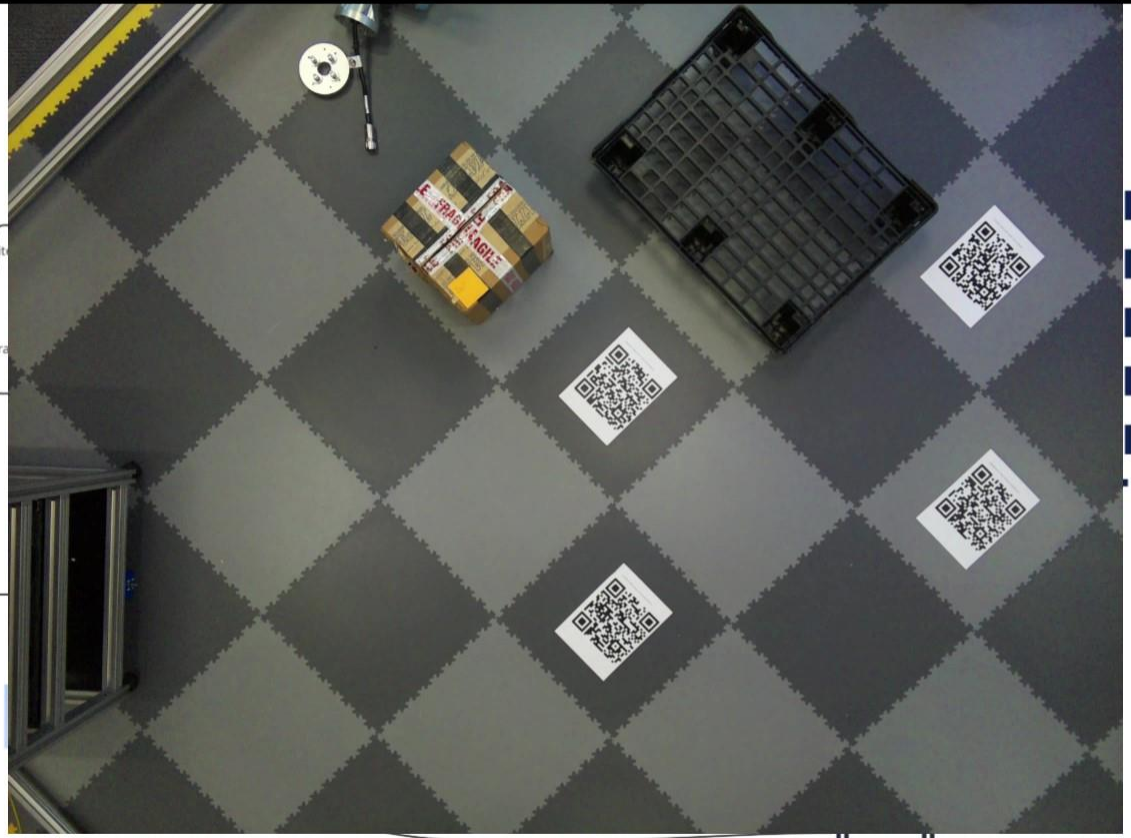
Problem - The coordinate estimation model is one of many in a broader scene understanding pipeline.

- How do we unify outputs from diverse models into a coherent, actionable dataset?
- With Edge constraints in mind (high data volume, low latency required, limited compute available)

Solution – A modular pipeline framework leveraging semantic data structures

- Input layer & Pre-processing: Camera stream - split, tag, storage of incoming frames
- Processing (model) layer:
  - metadata extraction, object detection, geometric transformers
  - Each outputs it's respective feature data in RDF turtle format – with frame tag as a shared reference point
- Fusion Node: Aggregate model output into a single RDF Graph
- Design – decoupled nodes
  - Scalability – supports adding inputs, additional models, additional instances of a model
  - Compute efficiency: Activate models as needed - enable all detection capabilities for task-specific actions when the swarm is performing dynamic operations - ramp down high compute models while the swarm is static

1. Image frame extracted, tagged & stored



Raspberry Pi 4

Pre-processing script

Write

Write

Consumer / Client

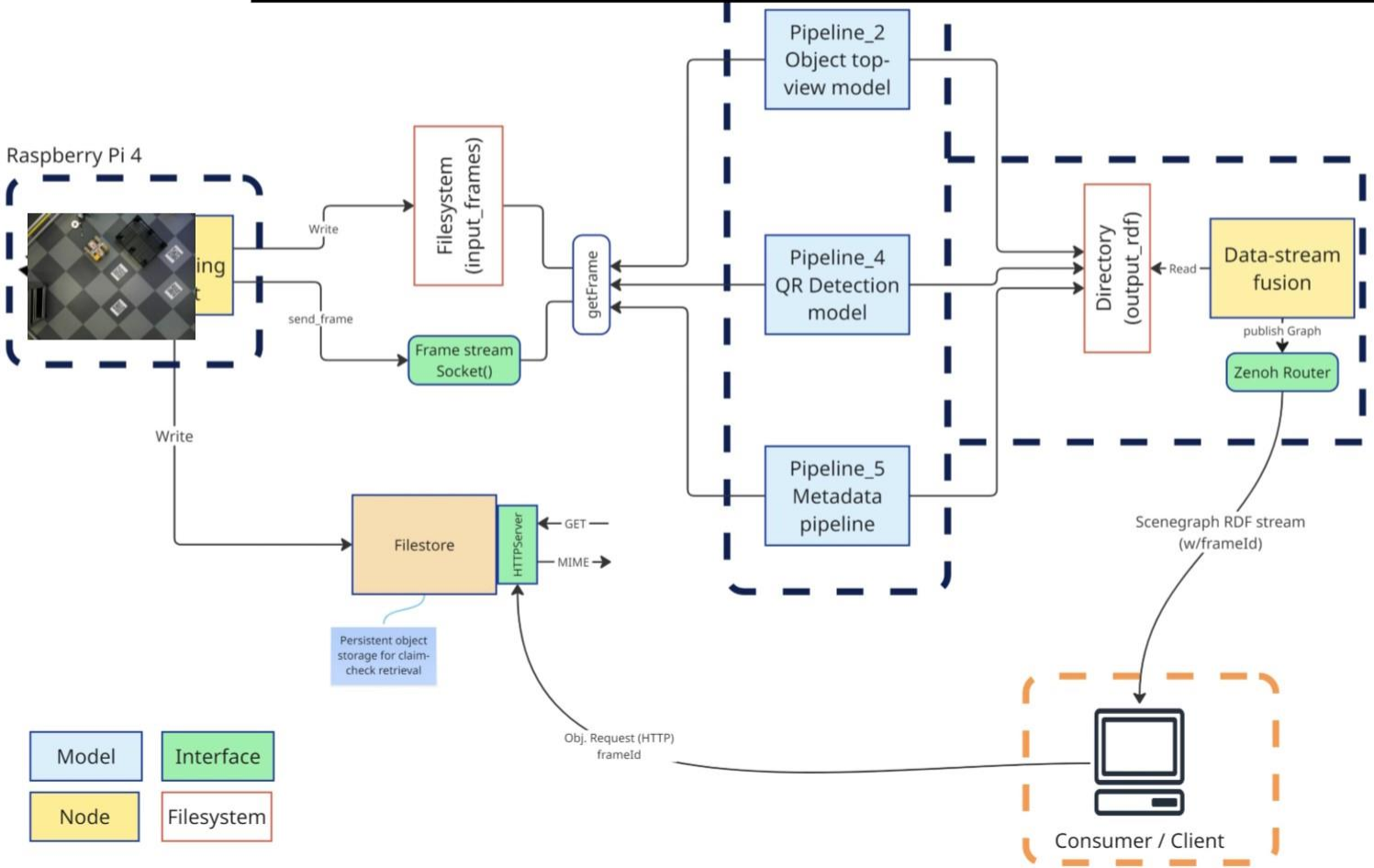
Model

Interface

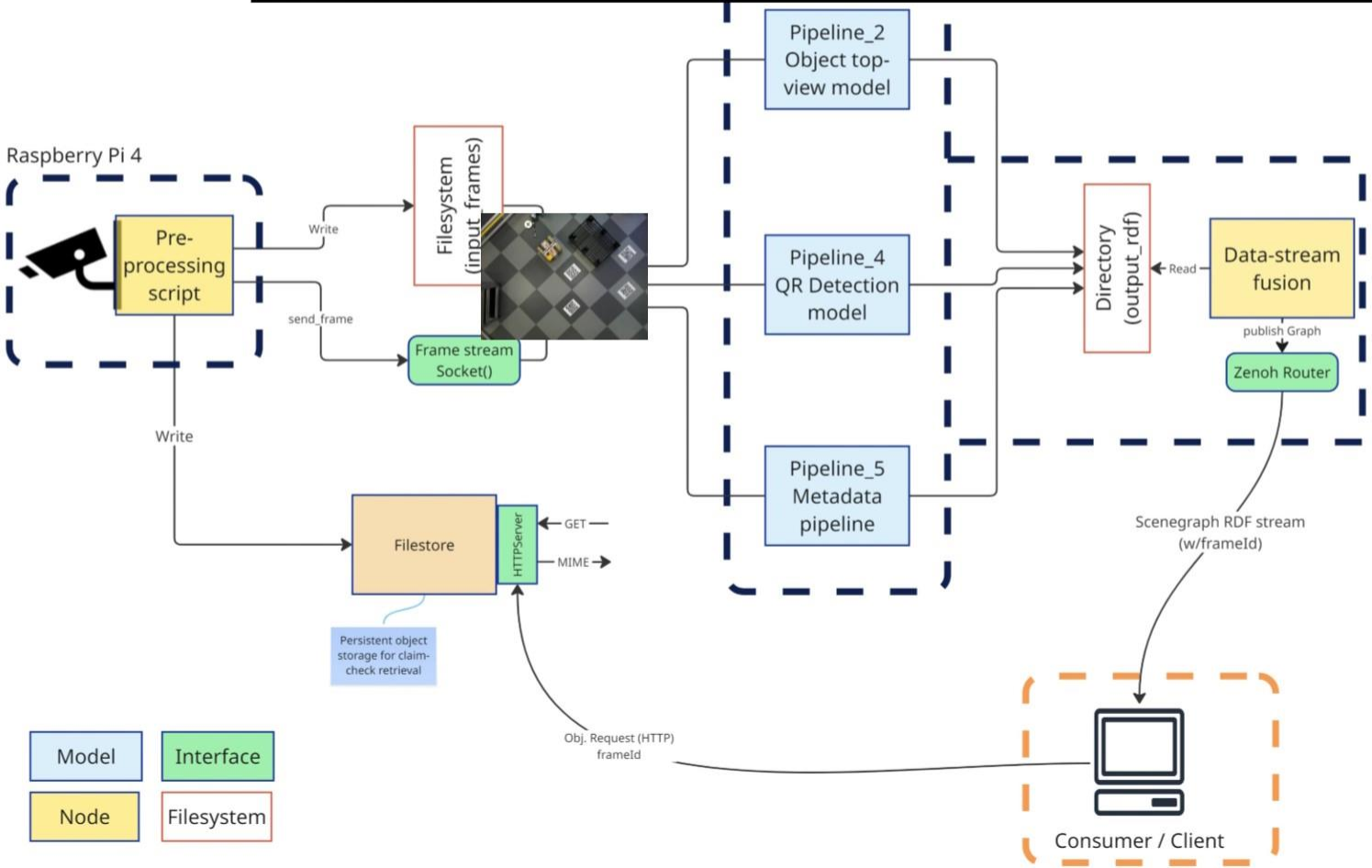
Node

Filesystem

1. Image frame extracted, tagged & stored

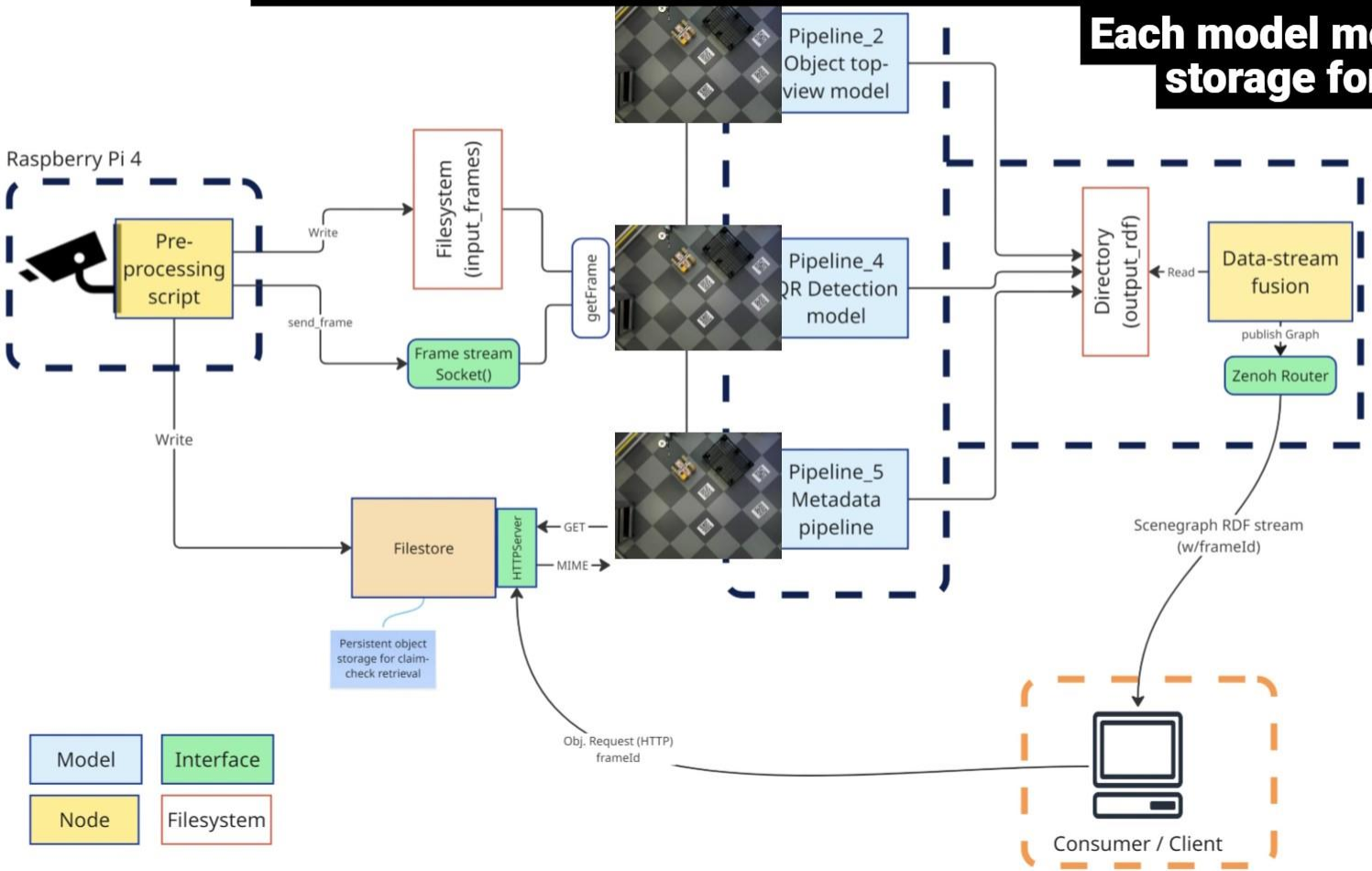


2. Tagged frame saved to shared storage



2. Tagged frame saved to shared storage

Each model monitors the shared storage for incoming frames



3. Model output respective features extracted

Output handler transforms feature data to individual model graphs

Raspberry Pi 4

Pre-processing script

Write

send_frame

Filesystem (input_frames)

Frame stream Socket()

getFrame

Write

Filestore

HTTPServer

GET

MIME

Persistent object storage for claim-check retrieval

Pipe Obje view

Pipe QR De m

Pipeline_5 Metadata pipeline

Directory (output_rdf)

Read

Data-stream fusion

publish Graph

Zenoh Router

RDFText

Scenegraph RDF stream (w/frameId)

Obj. Request (HTTP) frameId

Consumer / Client

Model

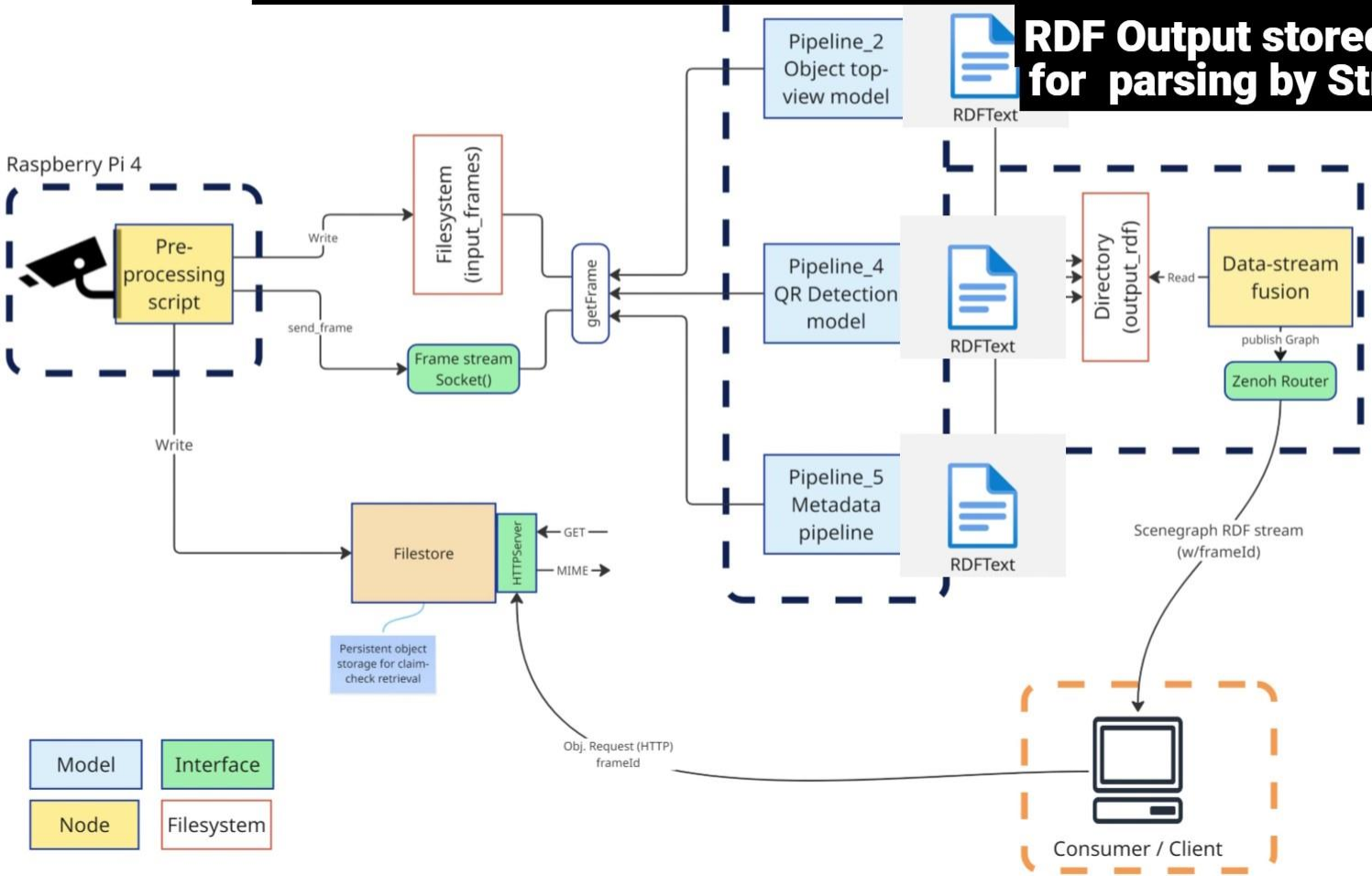
Interface

Node

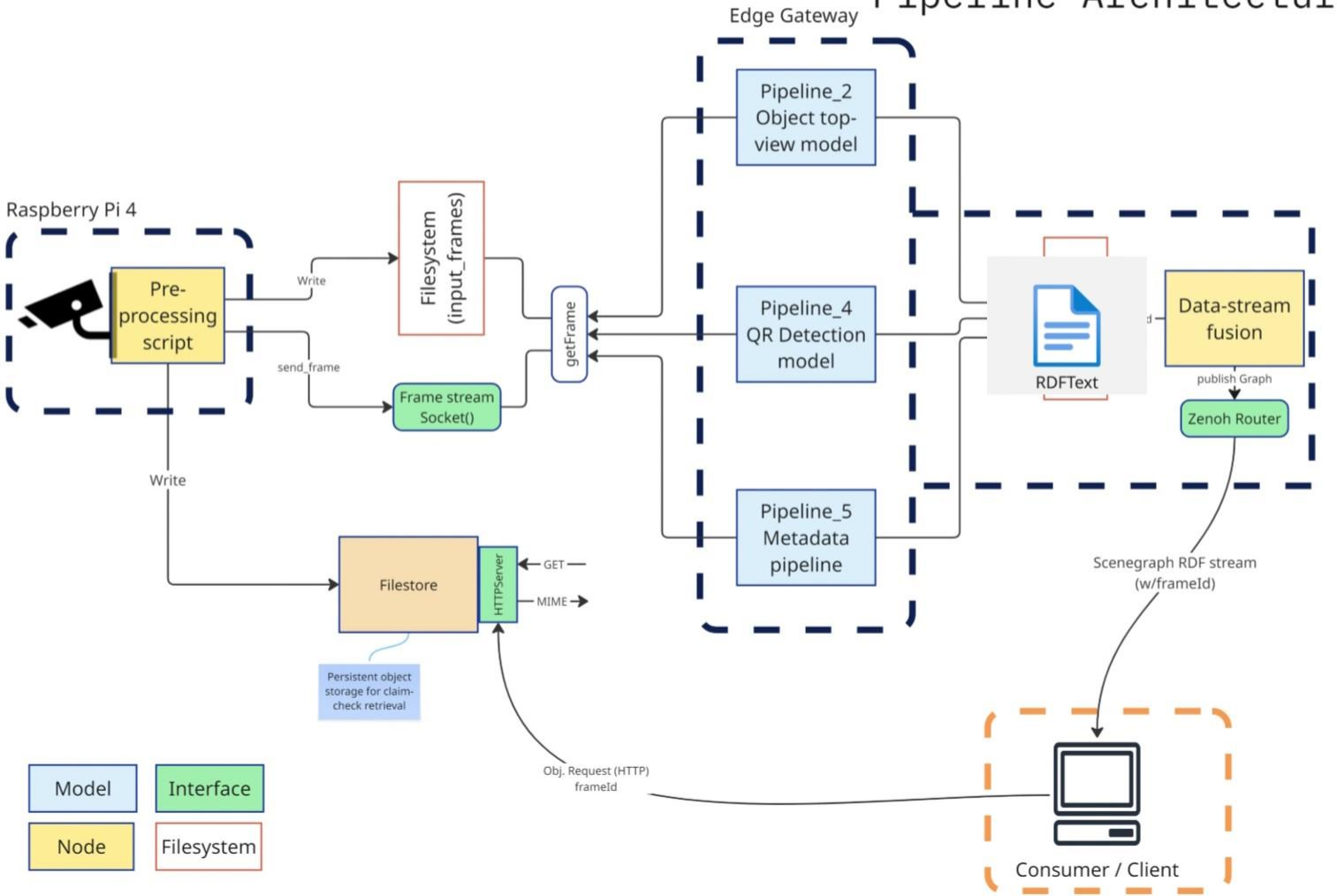
Filesystem

3. Model output respective features extracted

RDF Output stored in shard location for parsing by Stream Fusion Node

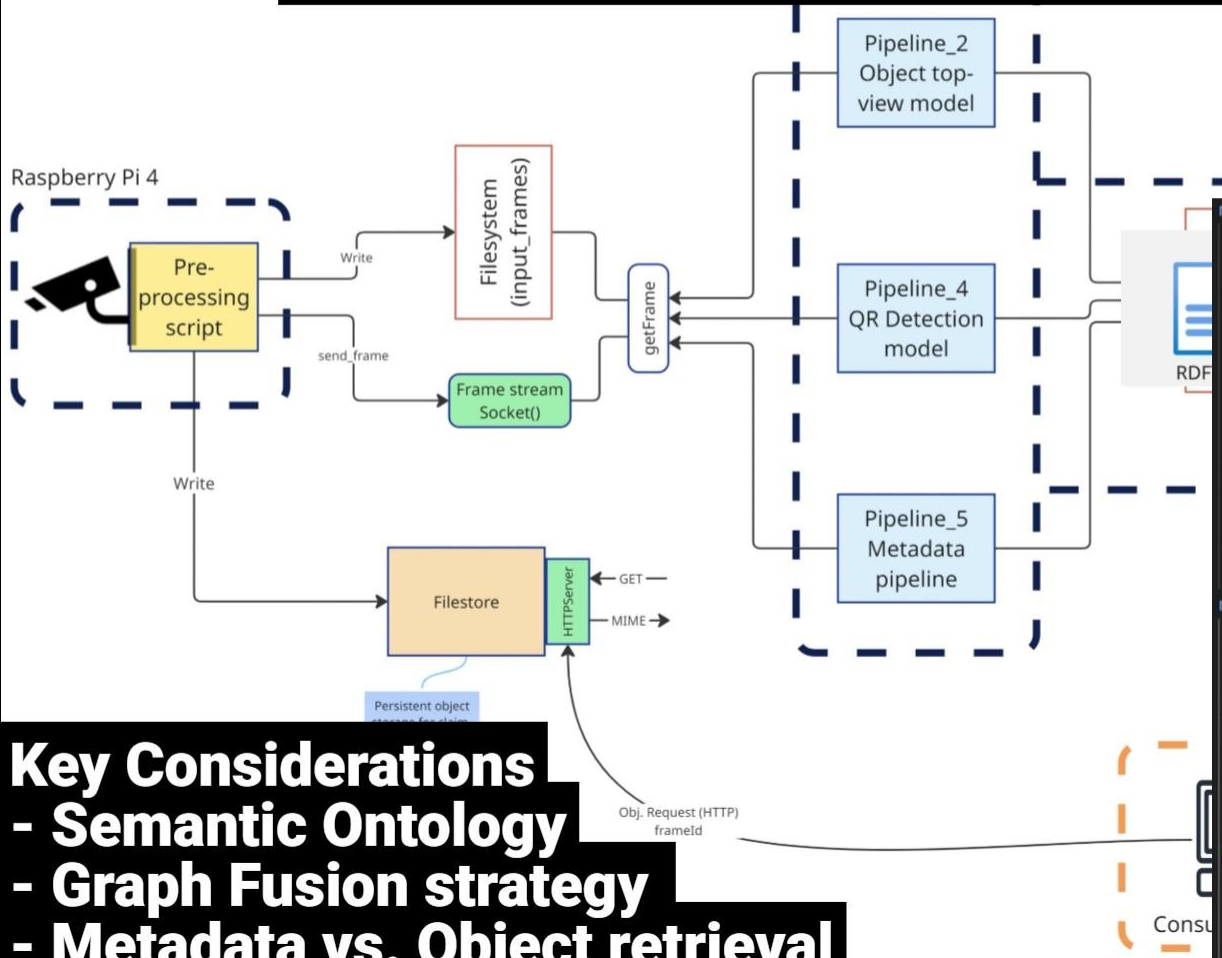


Pipeline Architecture - Design ref.



4. Stream Fusion node parses model graphs

A 'Scenegaph' is generated by dynamically combining each models feature graph



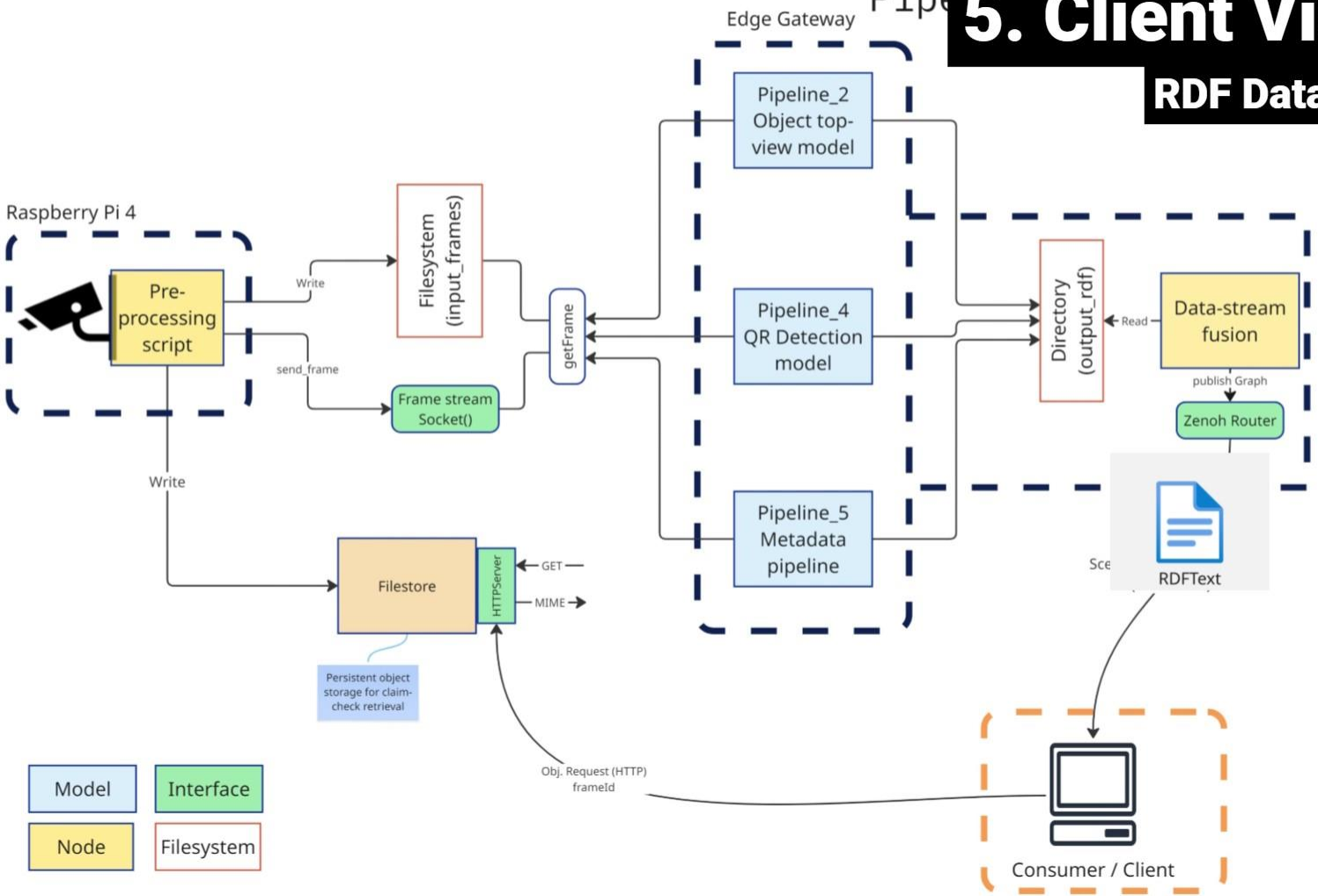

```
ns1:26416c64-8694-44b0-b31a-4fa6845a831a a ns1:objClass ;
    ns1:centerX "2219.0"^^xsd:float ;
    ns1:centerY "548.0"^^xsd:float ;
    ns1:classId "0" ;
    ns1:className "half_euro_pallet" ;
    ns1:confidence "0.858338475227356"^^xsd:float ;
    ns1:detectedAt "2025-07-01T12:22:55"^^xsd:dateTime ;
    ns1:detectedIn ns1:1751368970.0894926 ;
    ns1:fileName "1751368970.0894926_preproc.png" ;
    ns1:x1 "1681.0"^^xsd:float ;
    ns1:x2 "2757.0"^^xsd:float ;
    ns1:y1 "15.0"^^xsd:float ;
    ns1:y2 "1081.0"^^xsd:float .

ns1:2c7f4a8d-5b6c-43a9-85d4-7f3a2198b421 a ns1:QRCode ;
    ns1:centerX "1505.5"^^xsd:float ;
    ns1:centerY "465.5"^^xsd:float ;
    ns1:detectedAt "2025-07-01T12:22:52"^^xsd:dateTime ;
    ns1:detectedIn ns1:1751368970.0894926 ;
    ns1:fileName "1751368970.0894926_preproc.png" ;
    ns1:qrContent "2c7f4a8d-5b6c-43a9-85d4-7f3a2198b421" ;
    ns1:realX "217.0"^^xsd:float ;
    ns1:realY "115.0"^^xsd:float ;
    ns1:x1 "2736.0"^^xsd:float ;
    ns1:x2 "3011.0"^^xsd:float ;
    ns1:y1 "659.0"^^xsd:float ;
    ns1:y2 "931.0"^^xsd:float .
```

Key Considerations
- **Semantic Ontology**
- **Graph Fusion strategy**
- **Metadata vs. Object retrieval**

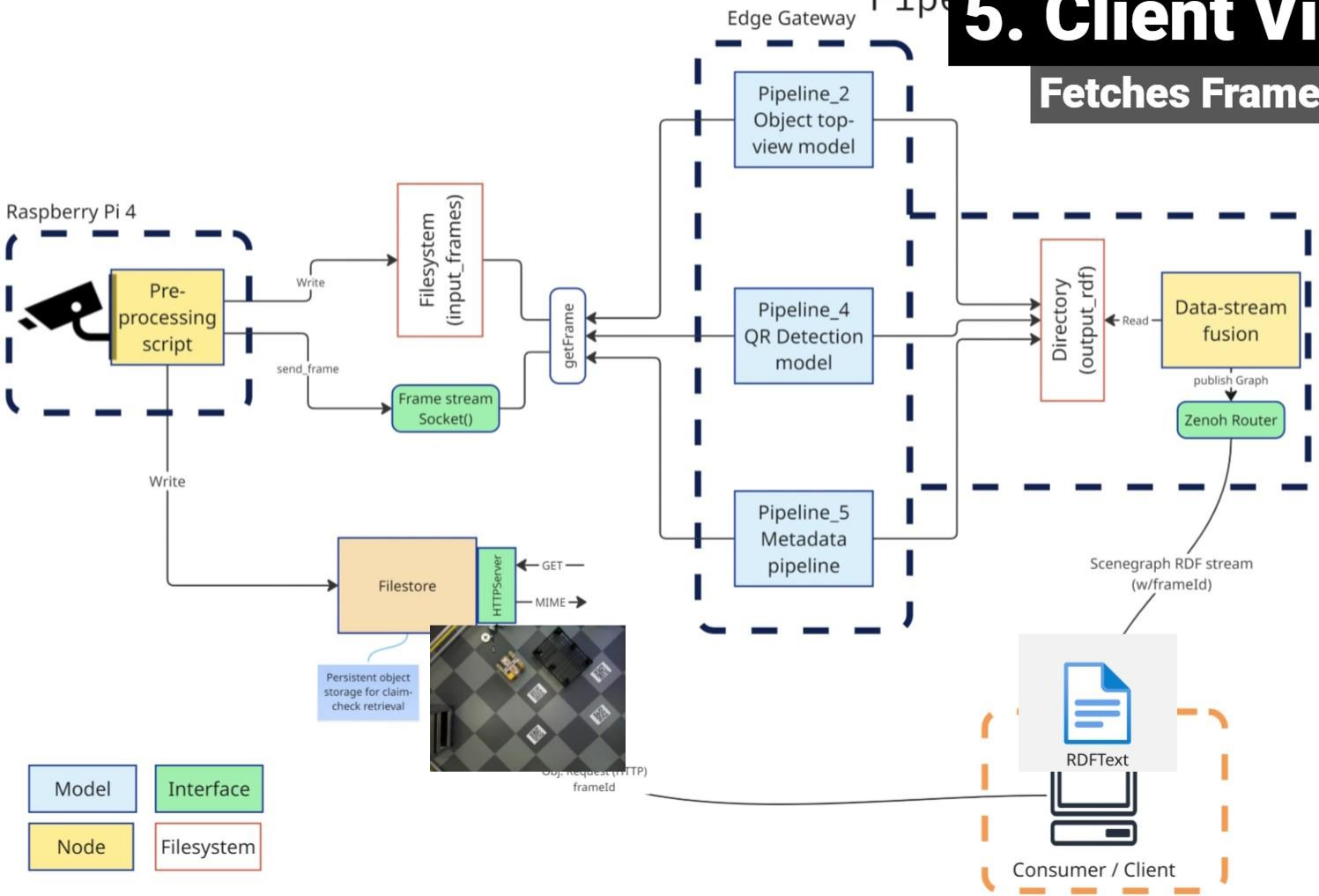
5. Client Visualization

RDF Data is published via a message queue



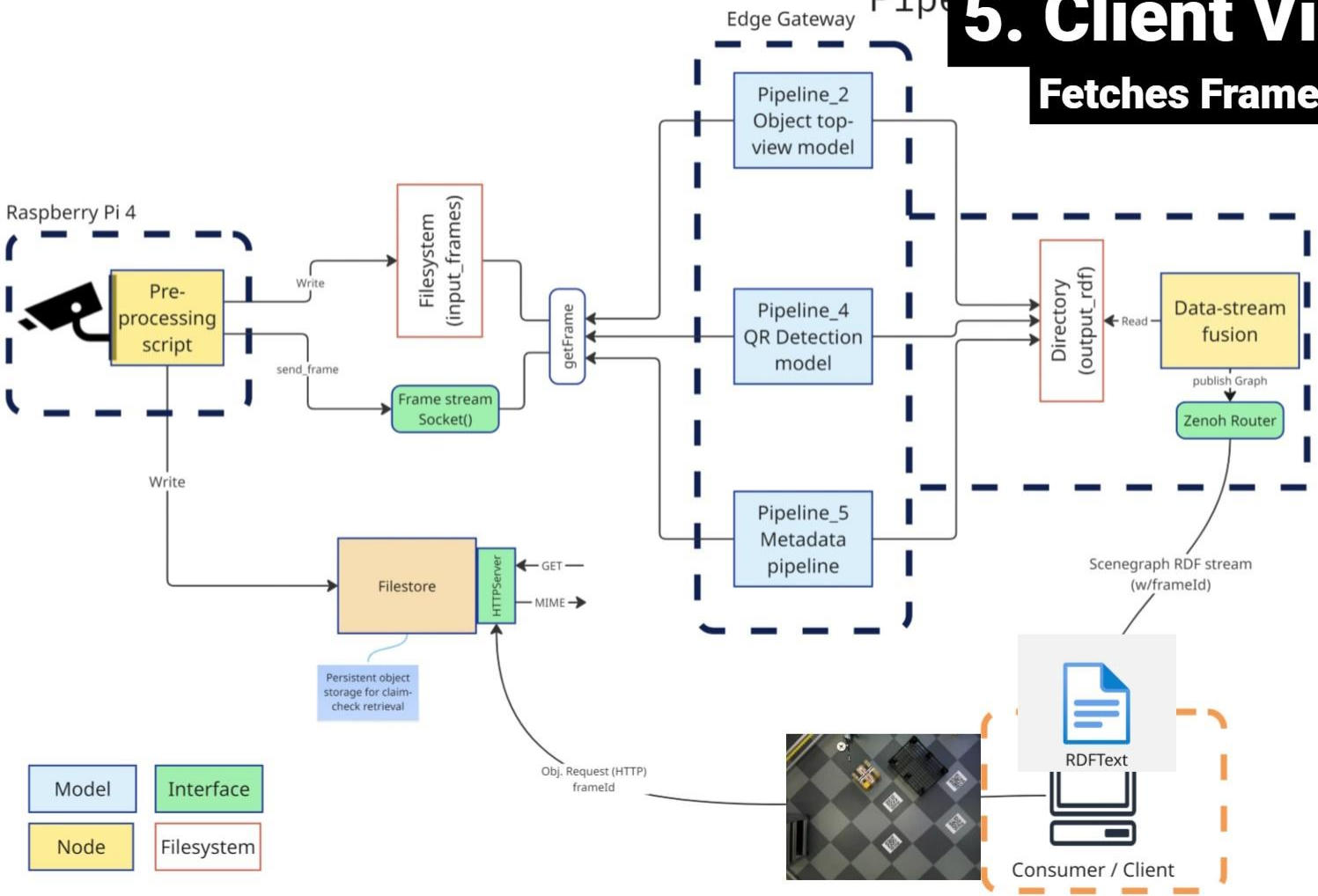
5. Client Visualization

Fetches Frame object with claim/check pattern



5. Client Visualization

Fetches Frame object with claim/check pattern



5. Client Visualization

Construct annotated image with all model features as final step

Edge Gateway

Raspberry Pi 4

Pre-processing script

Write

Model

Node

Filesystem

Consumer / Client

Semantic Scene Graphs: The Power of Structured Data

- Advantages of RDF/OWL:
  - Query-Ability: Use SPARQL to query relationships (e.g., “Find all AMRs within X a defective component”).
  - Semantic Inference - <QRCode1> detected in <Frame123> is linked to the <Robot6.location> in the graph – we infer QRCode1s location property
- Modular Model Synergy:
  - Ability to augmentation model outputs vs. running in isolation:
    - ✓ QR scanners (spatial anchors).
    - ✓ Object detectors (Pallet, Loader, or RobotMover).
    - ✓ Geometric transformers (distance/position calculations).

```
ns1:273e2a11-77e1-4983-8fbc-4d9306cb4d6c a ns1:objClass ;
    ns1:centerX "1147.5"^^xsd:float ;
    ns1:centerY "1116.0"^^xsd:float ;
    ns1:classId "0" ;
    ns1:className "half_euro_pallet" ;
    ns1:confidence "0.9201200008392334"^^xsd:float ;
    ns1:detectedAt "2025-07-01T20:56:36"^^xsd:dateTime ;
    ns1:detectedIn ns1:1751399776.9594297 ;
    ns1:fileName "1751399776.9594297_preproc.png" ;
    ns1:x1 "635.0"^^xsd:float ;
    ns1:x2 "1660.0"^^xsd:float ;
    ns1:y1 "612.0"^^xsd:float ;
    ns1:y2 "1620.0"^^xsd:float .

ns1:2c7f4a8d-5b6c-43a9-85d4-7f3a2198b421 a ns1:QRCode ;
    ns1:centerX "1506.5"^^xsd:float ;
    ns1:centerY "466.0"^^xsd:float ;
    ns1:detectedAt "2025-07-01T20:56:36"^^xsd:dateTime ;
    ns1:detectedIn ns1:1751399776.9594297 ;
    ns1:fileName "1751399776.9594297_preproc.png" ;
    ns1:qrContent "2c7f4a8d-5b6c-43a9-85d4-7f3a2198b421" ;
    ns1:realX "217.0"^^xsd:float ;
    ns1:realY "115.0"^^xsd:float ;
    ns1:x1 "2739.0"^^xsd:float ;
    ns1:x2 "3013.0"^^xsd:float ;
    ns1:y1 "660.0"^^xsd:float ;
    ns1:y2 "932.0"^^xsd:float .
```

Optimizing for the Edge: Lessons Learned

- Key challenges
  - Image storage: RPi4 Camera generates ~150mb for 20s of (unprocessed) capture
  - Frame rate / model synchronization issues
    - ✓ If the camera generates frames faster than models can process them, a backlog occurs.
    - ✓ Mitigation: Sample every nth frame - Spin up parallel model instances – Throttle input load
- Research Questions
  - How to manage time synchronization in multi-camera systems?
- Future Work
  - Graph fusion backlog Issue: Asynchronous model execution requires frame / scene batching method
  - Refinement of graph fusion strategy & scene ontology
  - Investigate model synchronization mechanisms
- **Semantic data structures are critical to enable this real-time, scalable edge AI system.**

This project is supported by the European Union's Horizon RIA research and innovation programme under grant agreement No. 101092908 (SMARTEDGE)

https://www.smart-edge.eu/