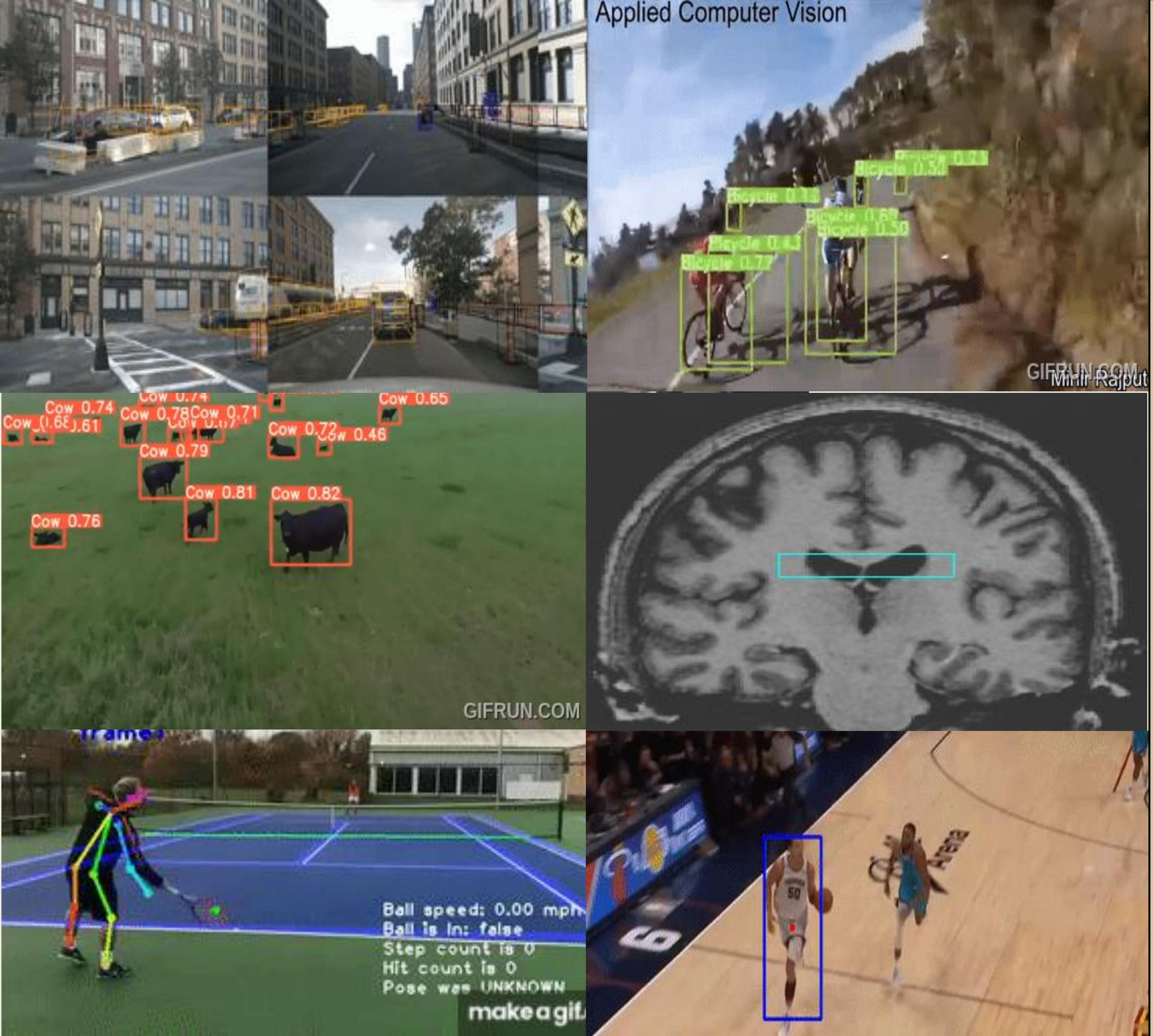
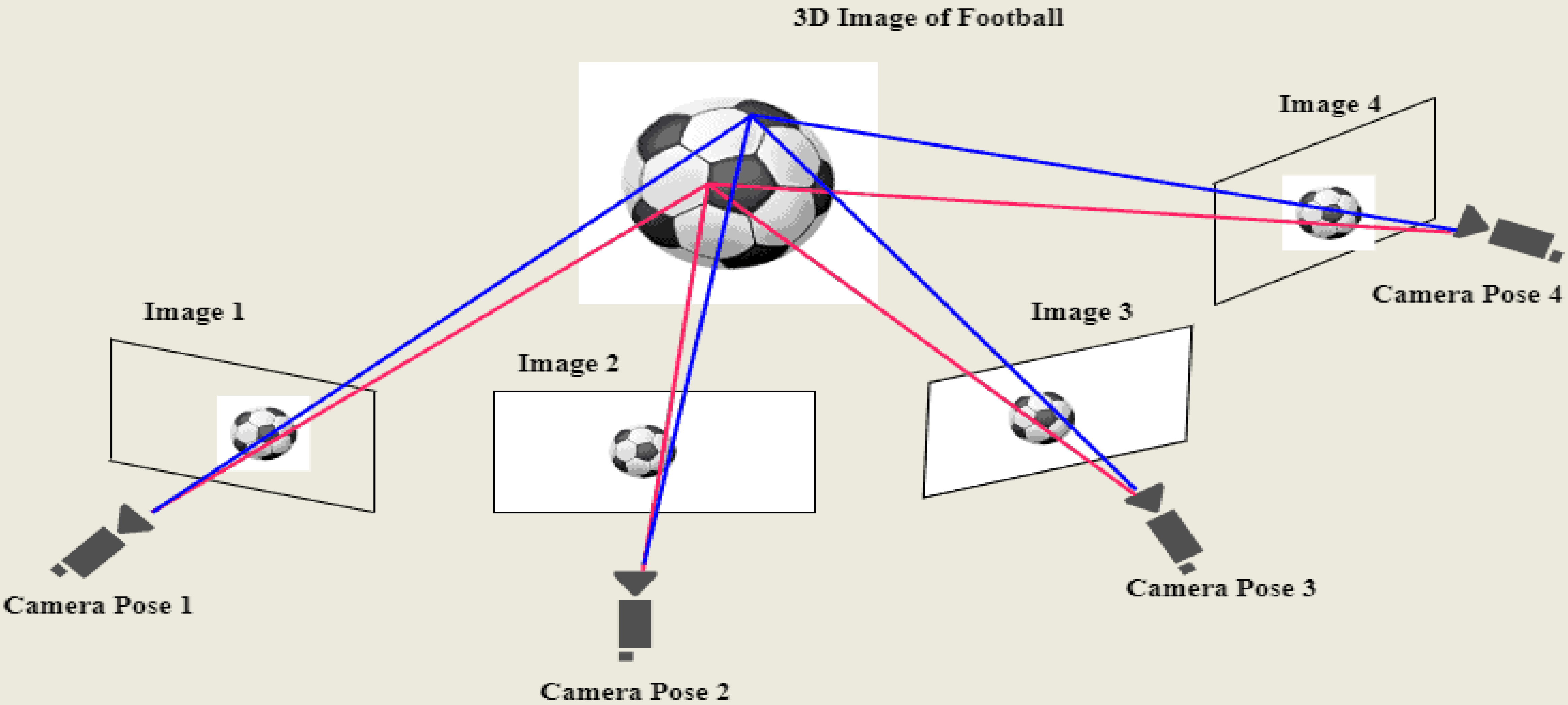
Multi-View Obscured Object Detection (MOOD)

Robin Arun, Katie Hucker, Afraa Noureen, Jiayi Zhou

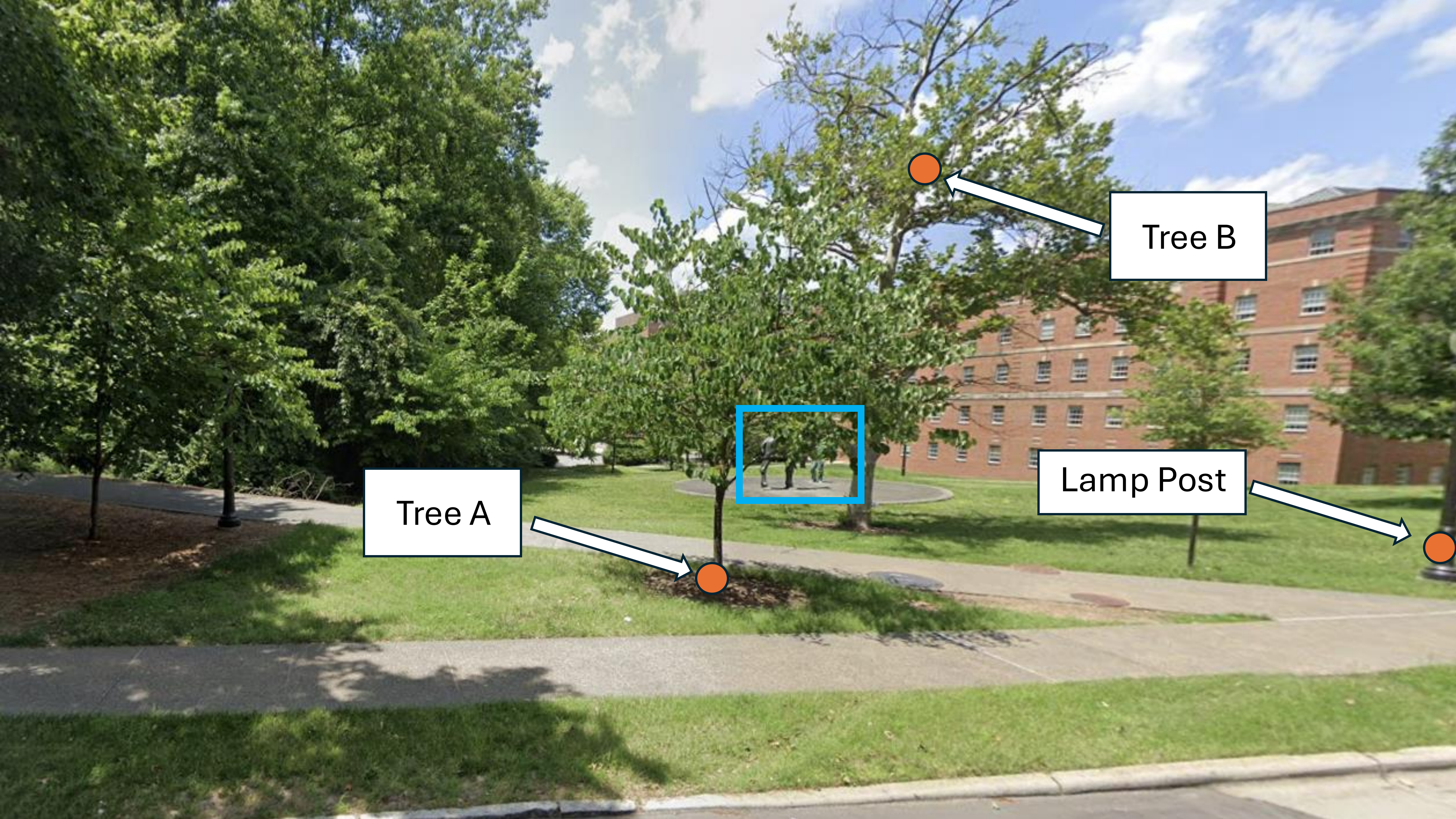


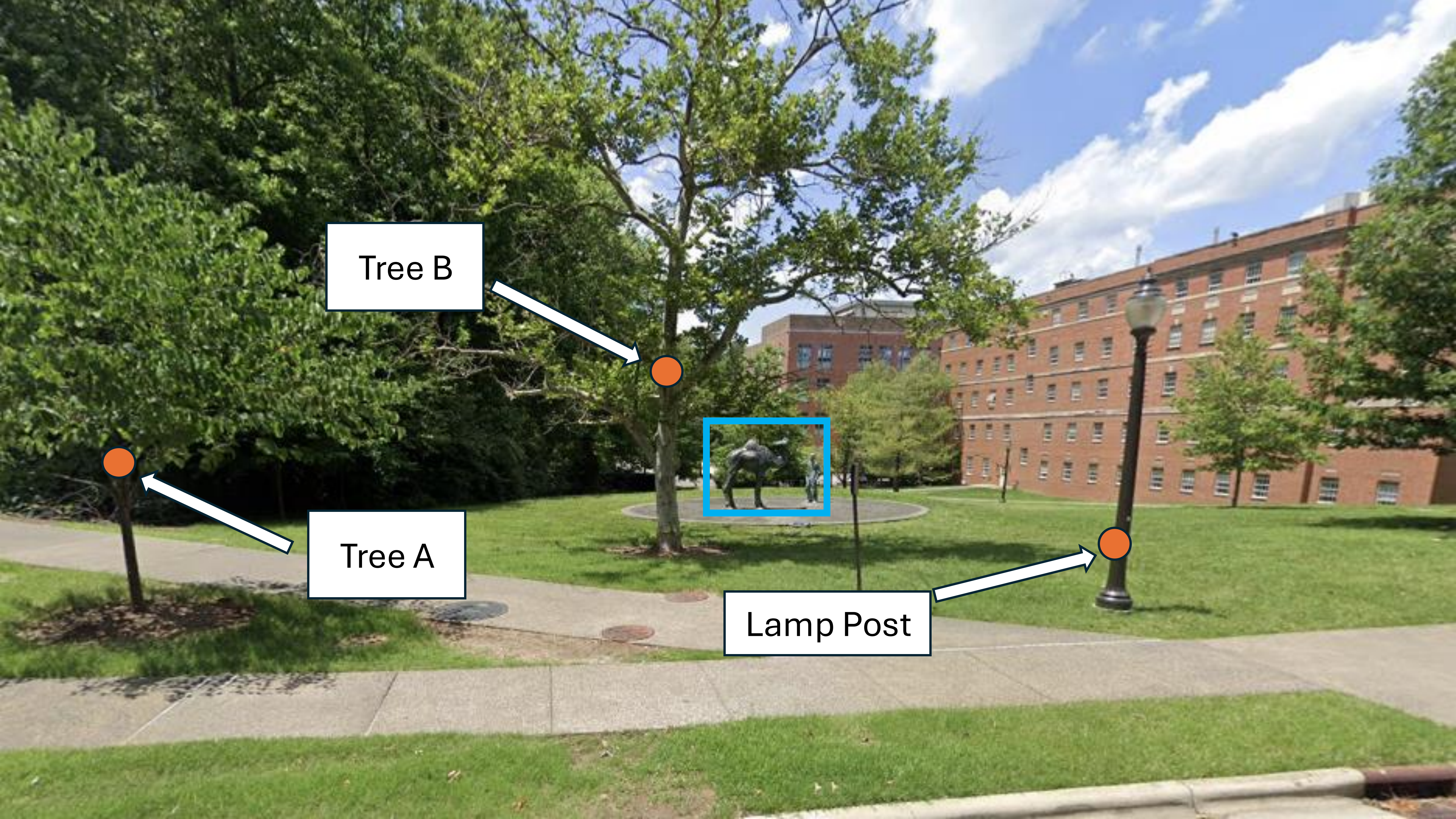
Introduction:

- Life in a 3D world
- Pedestrians, cars, animals.
- Movement, depth, visibility, light
- Pictures and video in a 2D plane.



Multiple images or *views* can create a 3D world

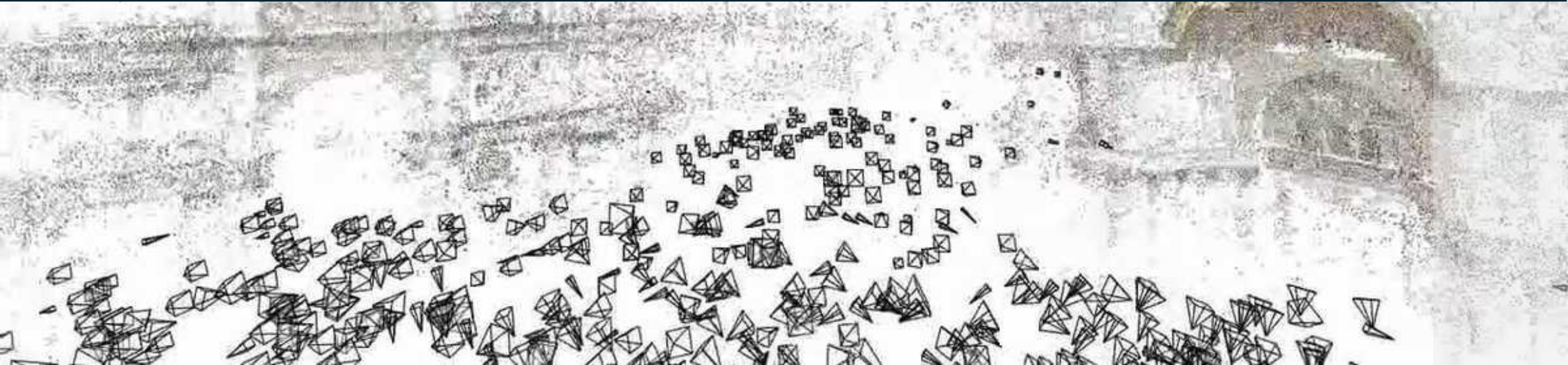

Tree B
Tree A
Lamp Post


Tree B
Tree A
Lamp Post

Definition

- Multi-view: multiple images of the **same** scene
- Obscured object: Targets within a scene are difficult to identify due to the angle of view and scene debris like trees, people, and buildings.
- Detection: Can we identify if the object is there, and what the object is? How confident are we?

Our Problem



What is the problem?

- We **sacrifice** scene contextualization and poor object visibility from single 2D images and videos, *unless we have one good look.*
- A model which **understands the 3D world** using co-registered images can provide stronger detection capabilities, *using **all the looks.***

Why does this model matter?

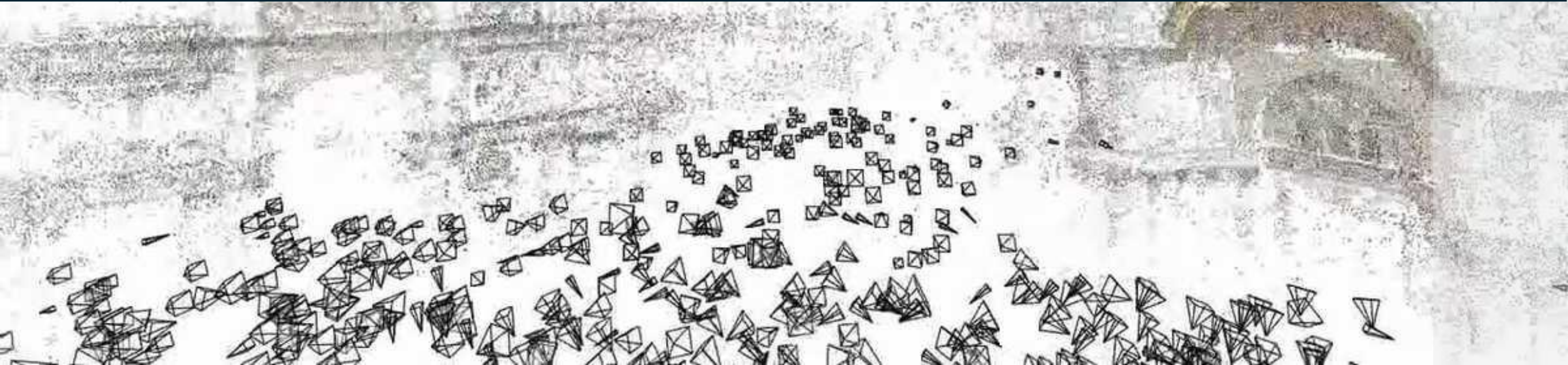
- Hidden Objects are a **risk.**
- **High-Stakes** impact
- Traditional methods are computationally expensive.
- A 3D interpretation of the world without a 3D reconstruction

Goal: Develop a model which can…

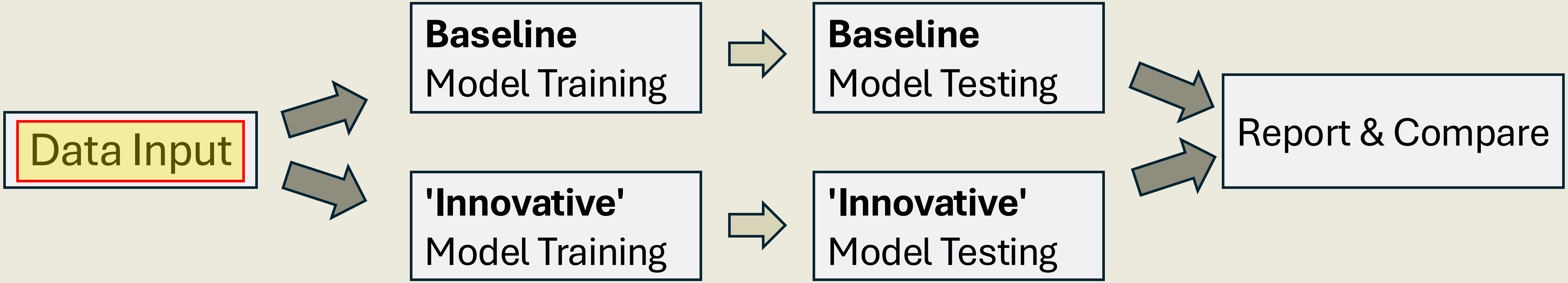
1. **input** multiple pictures from the same scene
2. **use** how those pictures relate to one another within the scene
3. **detect** and classify the obscured object of interest *correctly*
4. **interact** in a useable tool

1. How much can partially obscured object detection improve using multi-views?	2. How do different levels of obscurement with multiple views affect detection accuracy?
3. What number of scene views do we need for high detection performance?	4. What features of our objects contribute to detected boxes?

How can we answer these questions?



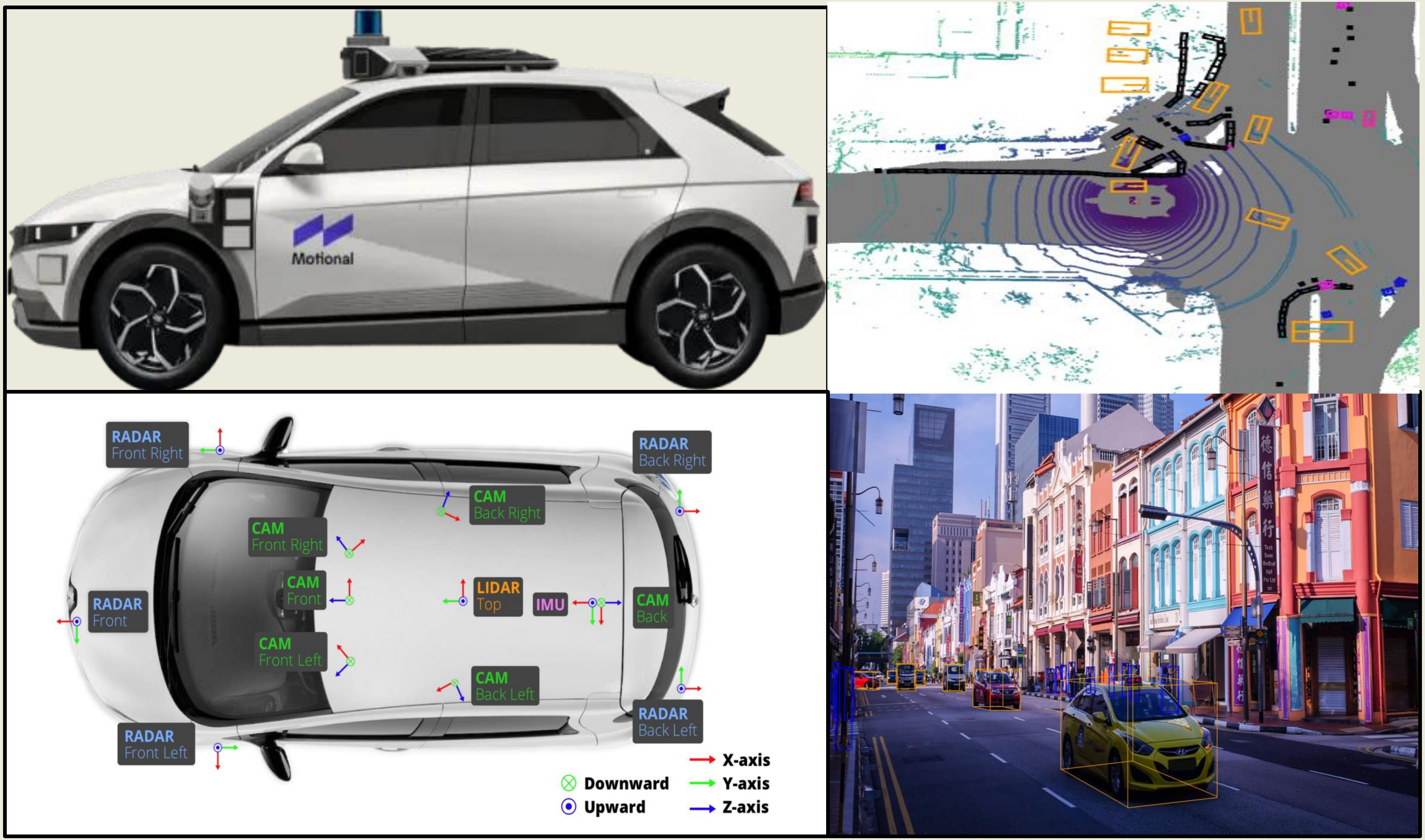
Experimental Design



Our data input requires a wide range of needs in order to capture an entire scene with object detection capabilities.

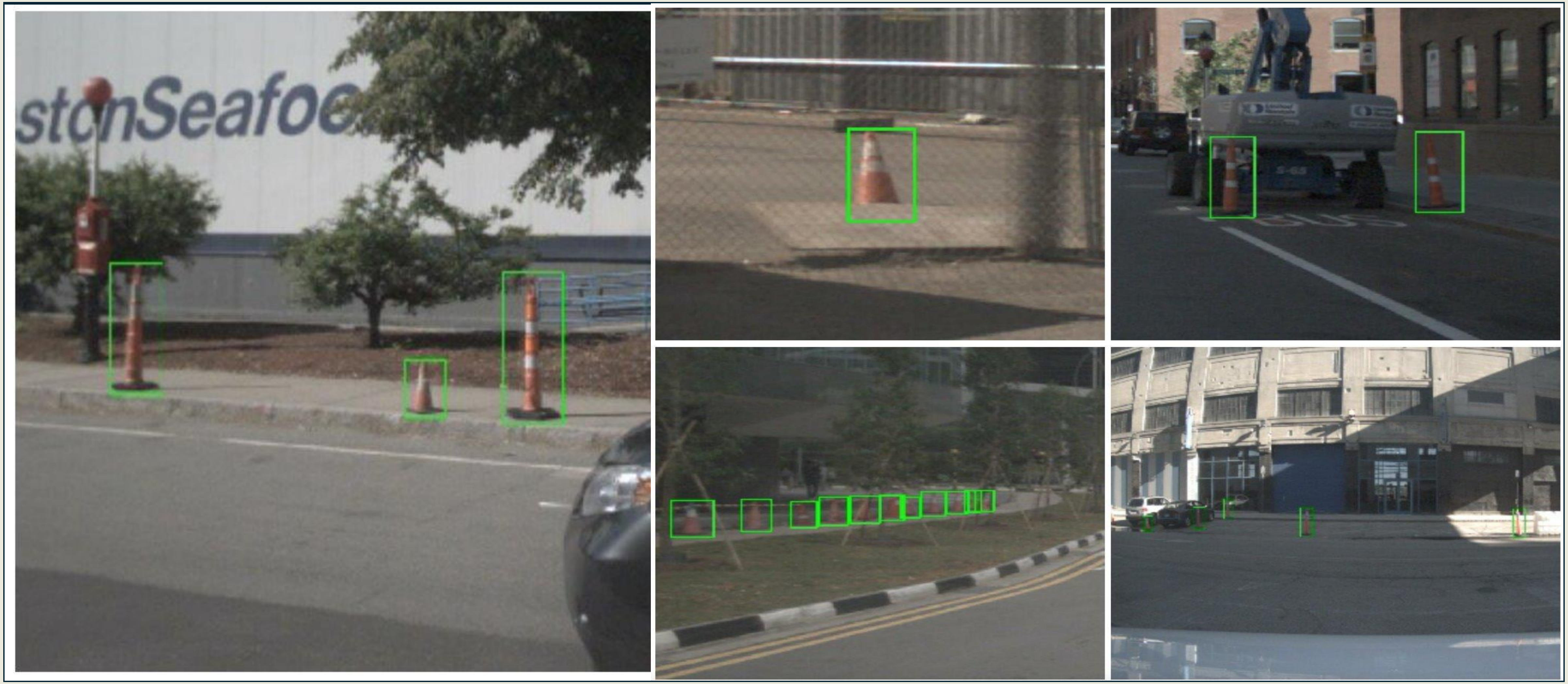
Data Input

- Continuous images of a scene
- Labeled Objects
- Visibility of Objects Annotated
- Intrinsic & extrinsic camera data
- Adaptable for ML models

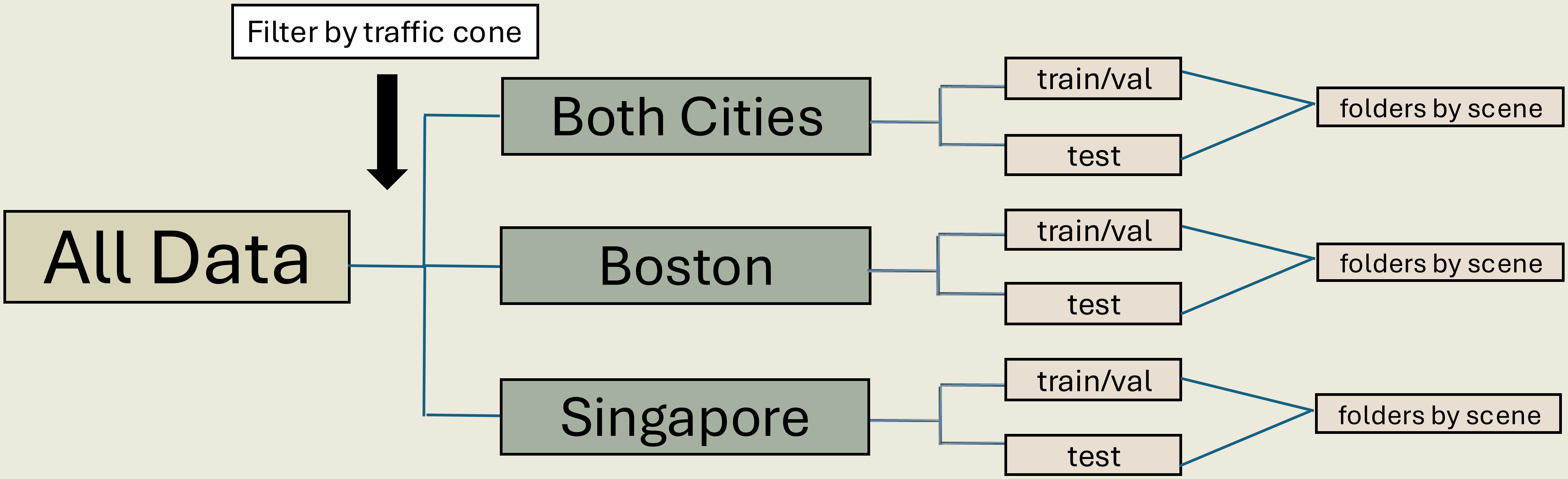

Motional
RADAR
Front Right
RADAR
Back Right
CAM
Back Right
CAM
Front Right
CAM
Front
LIDAR
Top
IMU
CAM
Back
RADAR
Front
CAM
Front Left
CAM
Back Left
RADAR
Back Left
RADAR
Front Left
X-axis
Downward
Y-axis
Upward
Z-axis

Traffic Cones

Source: https://github.com/nutonomy/nuscenes-devkit/blob/master/docs/instructions_nuscenes.md#traffic-cone

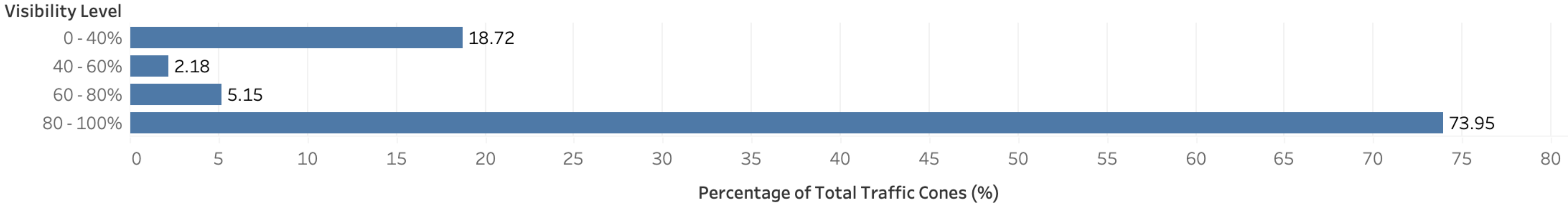


Data Structure

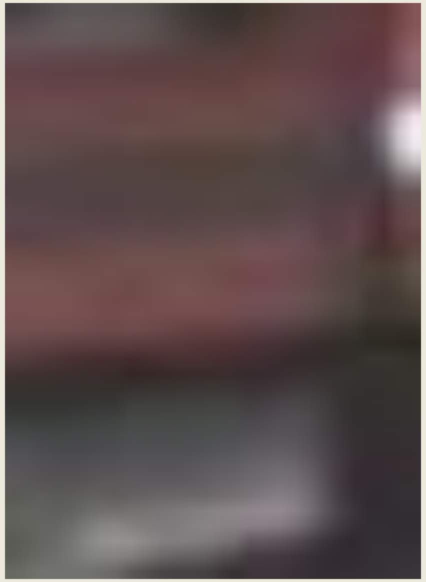


Nearly 74% of Traffic Cones Have High Visibility (80-100% Level)

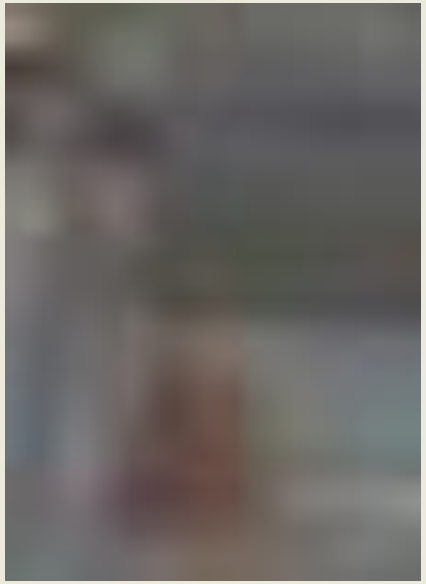
Dataset sourced from NuScenes Mini Dataset--Total traffic cones analyzed: 1378



Sum of Percentage for each Visibility Level. The marks are labeled by sum of Percentage.

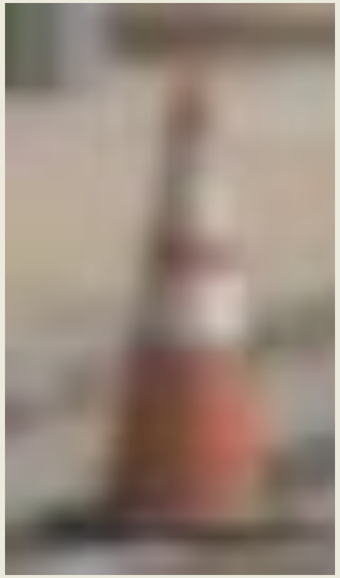


0-40%



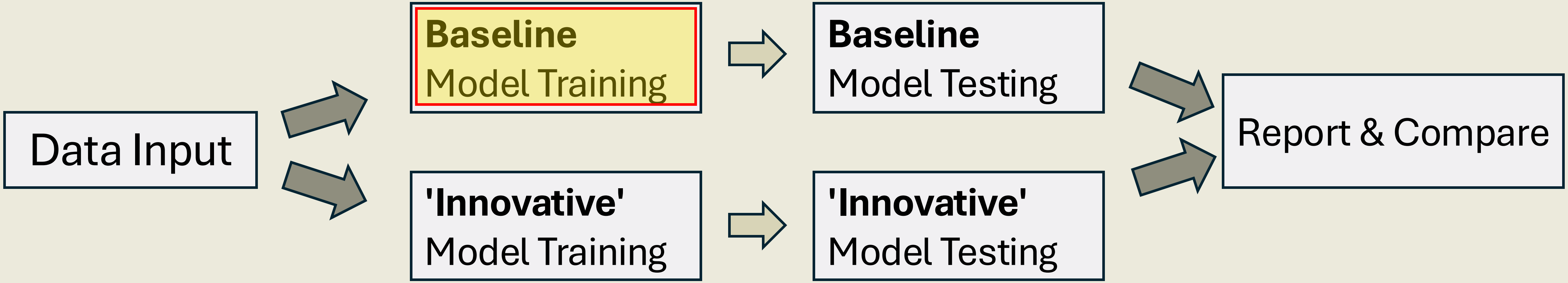
40-60%

60-80%



80-100%

Experimental Design

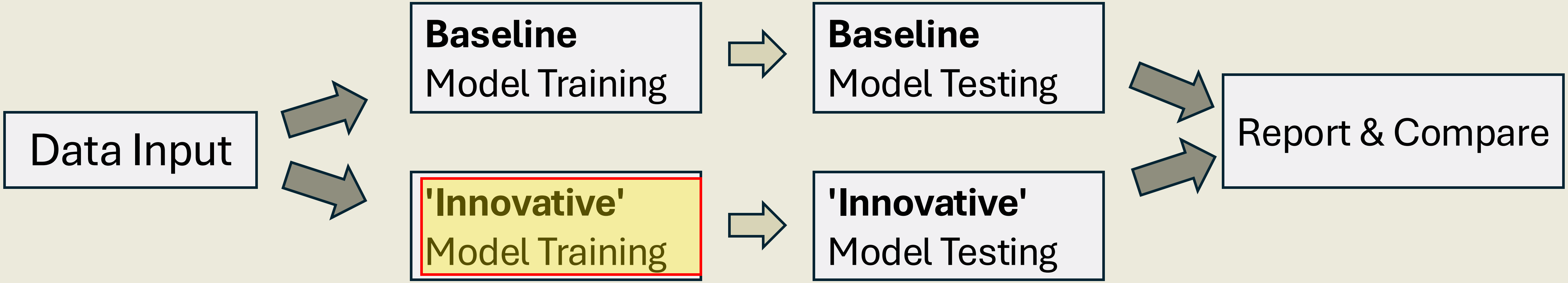


We define a baseline model as known reference, which would not consider obscurement or scene context between single images

You Only Look Once (YOLO)

- A real-time object detection algorithm
- Predict both the class and location of objects in an image
- Achieves high accuracy
- Fast object detection
- Good performance on small objects

Experimental Design



Our 'innovative' approach uses camera location to understand how the images connect to one another within a scene.

How much can partially obscured object detection improve using multi-views? 1.	How do different levels of obscurement with multiple views affect detection accuracy? 2.
What number of scene views do we need for high detection performance? 3.	What features of our objects contribute to detected boxes? 4.

DETR + 3D

Multi-view
image inputs

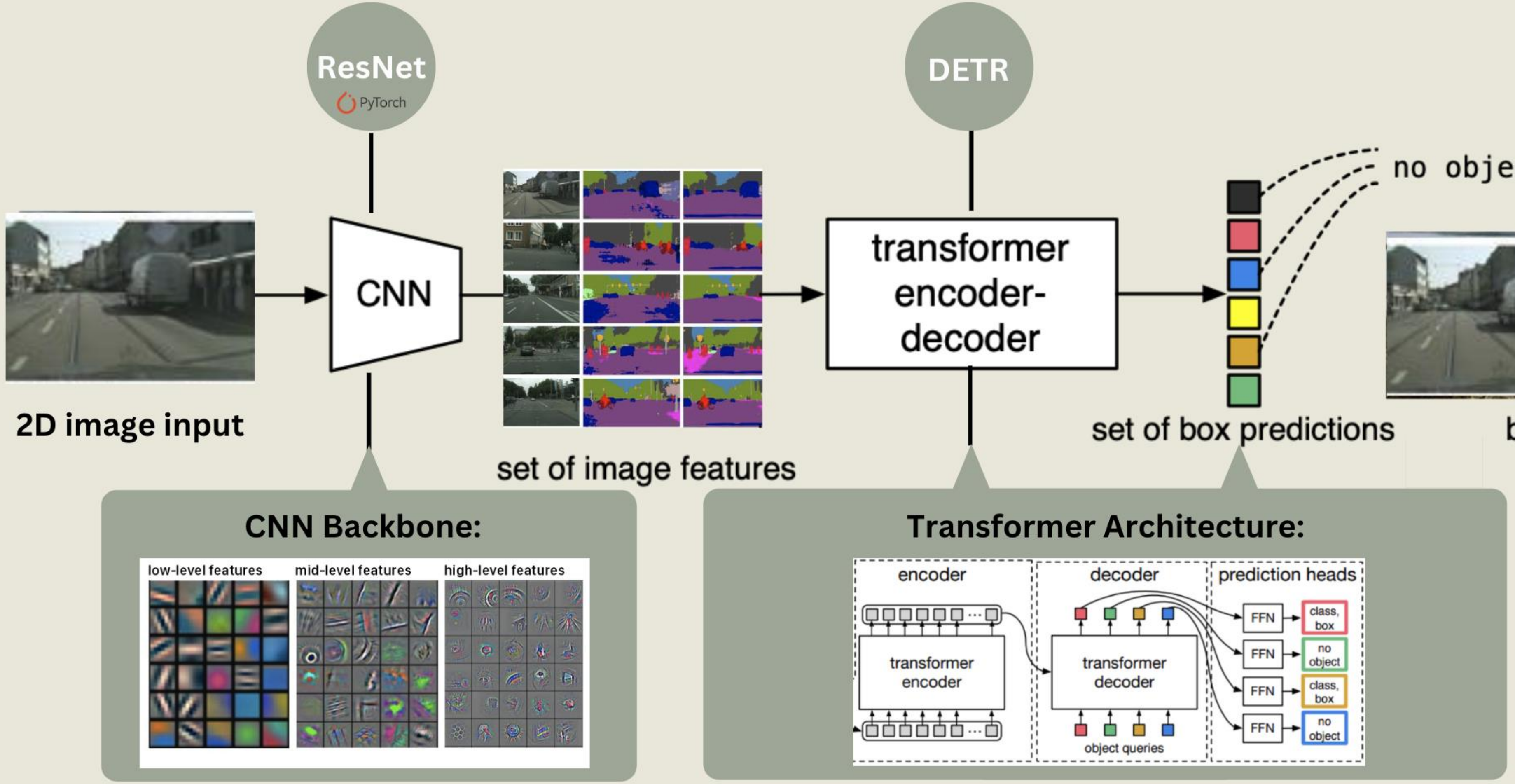
Obscured
object detection

Efficient
+ light-weight

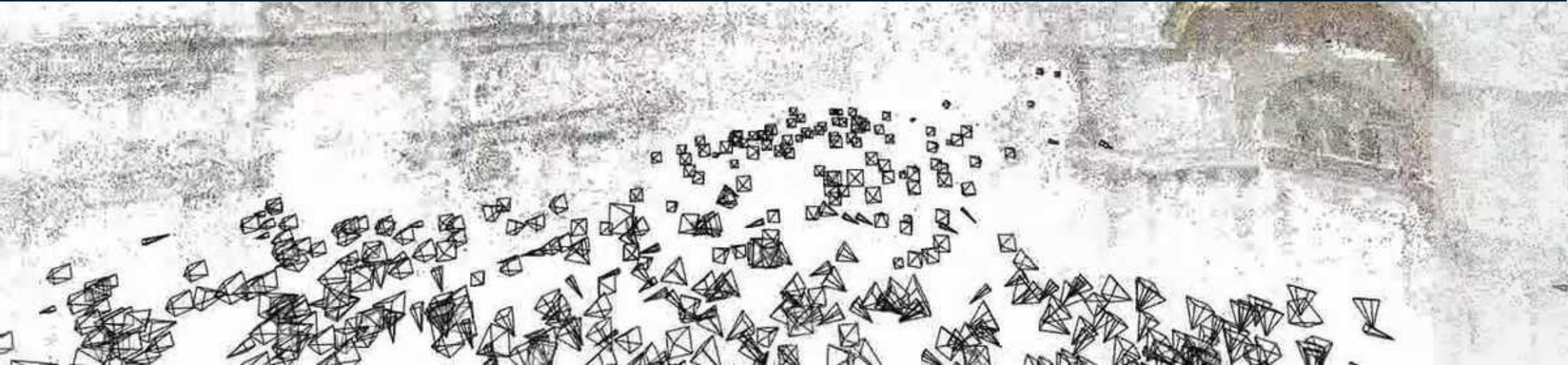
- Detects 3D objects directly from 2D multi-view images using transformer-based object queries

Method	NDS ↑	mAP ↑
Mono3D	0.429	0.366
DHNet	0.437	0.363
PGD [40]	0.448	0.386
DD3D [37] †	0.477	0.418
DETR3D (Ours) #	0.479	0.412

DETR + 3D


ResNet
PyTorch
DETR
CNN
transformer encoder-decoder
no object (ø)
no object (ø)
2D image input
set of image features
set of box predictions
bipartite matching loss
CNN Backbone:
low-level features
mid-level features
high-level features
Transformer Architecture:
encoder
decoder
prediction heads
transformer encoder
transformer decoder
object queries
FFN
class, box
no object

What's next?



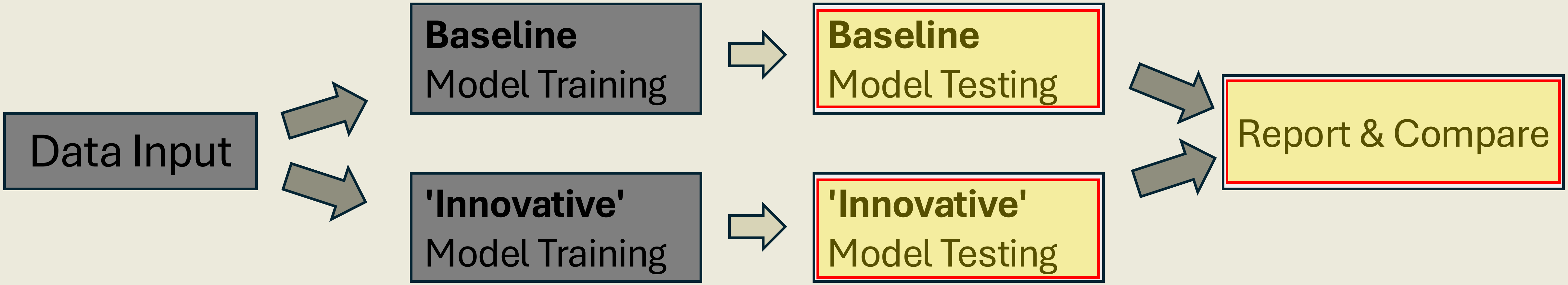
1. How much can partially obscured object detection improve using multi-views?

2. How do different levels of obscurement with multiple views affect detection accuracy?

3. What number of scene views do we need for high detection performance?

4. What features of our objects contribute to detected boxes?

Spring Semester



We will test both our models and analyze the results so we can answer specifically the **<u>interaction between multiple view and object visibility performance</u>**.

TIMELINE

Dec 2024

Training procedures complete

Jan 2025

Testing of models and evaluation metrics (Q2)

Feb 2025

Find 'goldilocks' scene view number (Q3)

March 2025

Analysis of results tor Q2 and 3

April 2025

Present results at symposium

APPENDIX

DETR3D Processes and Description

Multi-view 2D Inputs (Camera Intrinsic and Extrinsic Data)	Input: Input multi-view images from different angles of the scene.
Stage 1: CNN Backbone & FPN Feature Extraction	Stage 1: CNN(ResNet) generates feature maps from each camera view. Feature Pyramid Network(FPN) helps capture multi-scale feature mappings
Stage 2: Transformer (Feature Refinement)	Stage 2: Transformer queries the feature maps using attention mechanisms and optimizes set-to-set loss, as it learns the geometric encoding.
Stage 3 + Output: 3D Bounding Box Prediction & Class labels	Stage 3 and Output: Outputs 3D bounding boxes directly from 2D images, after having combined classification and bounding box regression.