RWR 4013

Digital Twins for Smart Cities

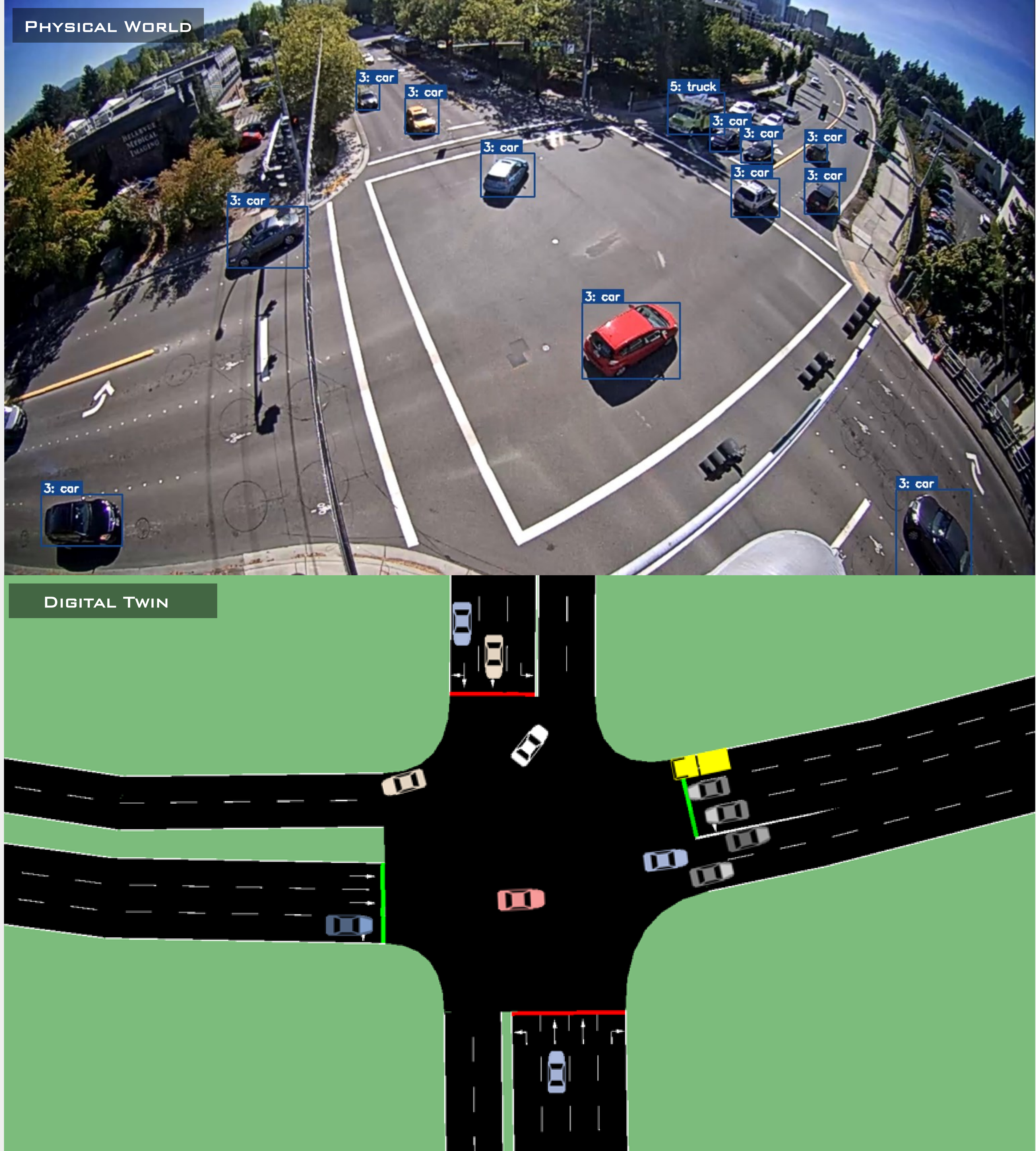
Dr. Ahmad Mohammadi

Week 2 | Session 1:
Computer Vision I
(Object Detection)

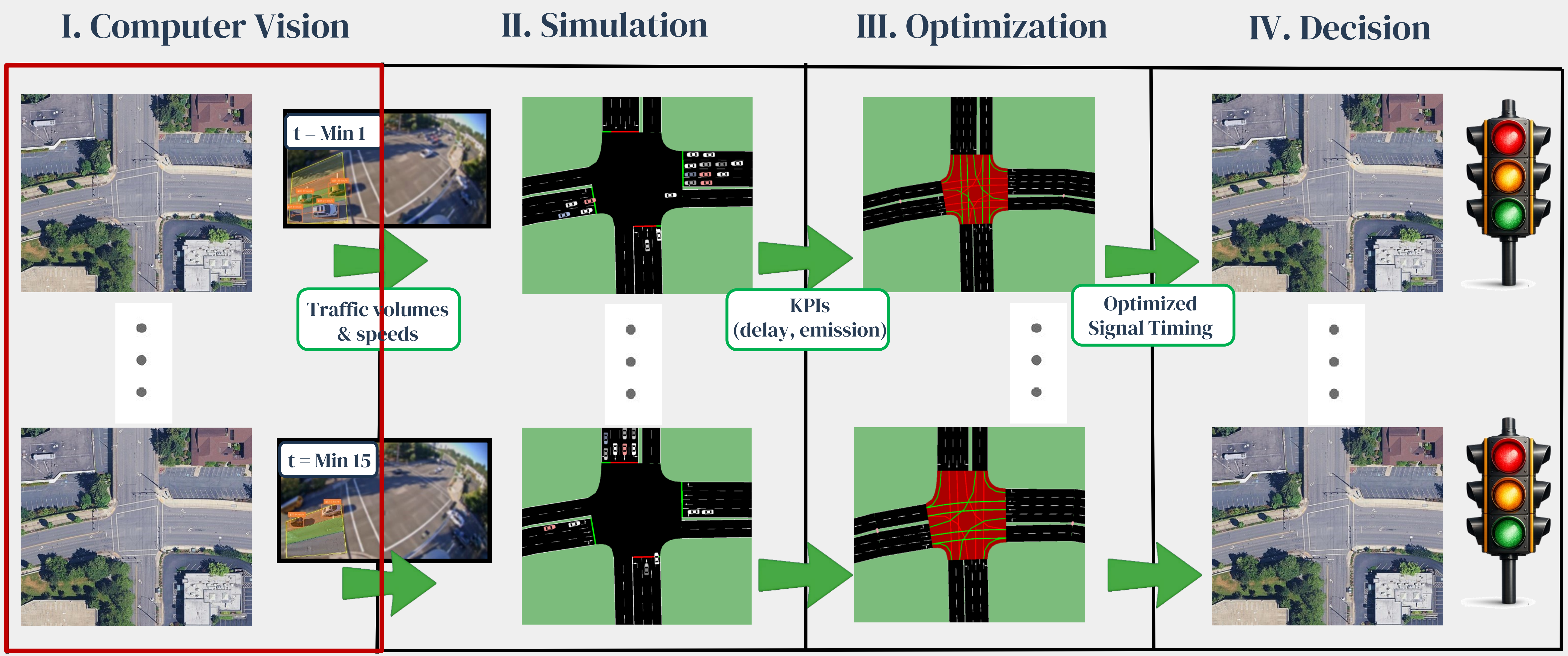
Fall 2026

RoadwayVR

roadwayvr.github.io/DigitalTwinsforSmartCities

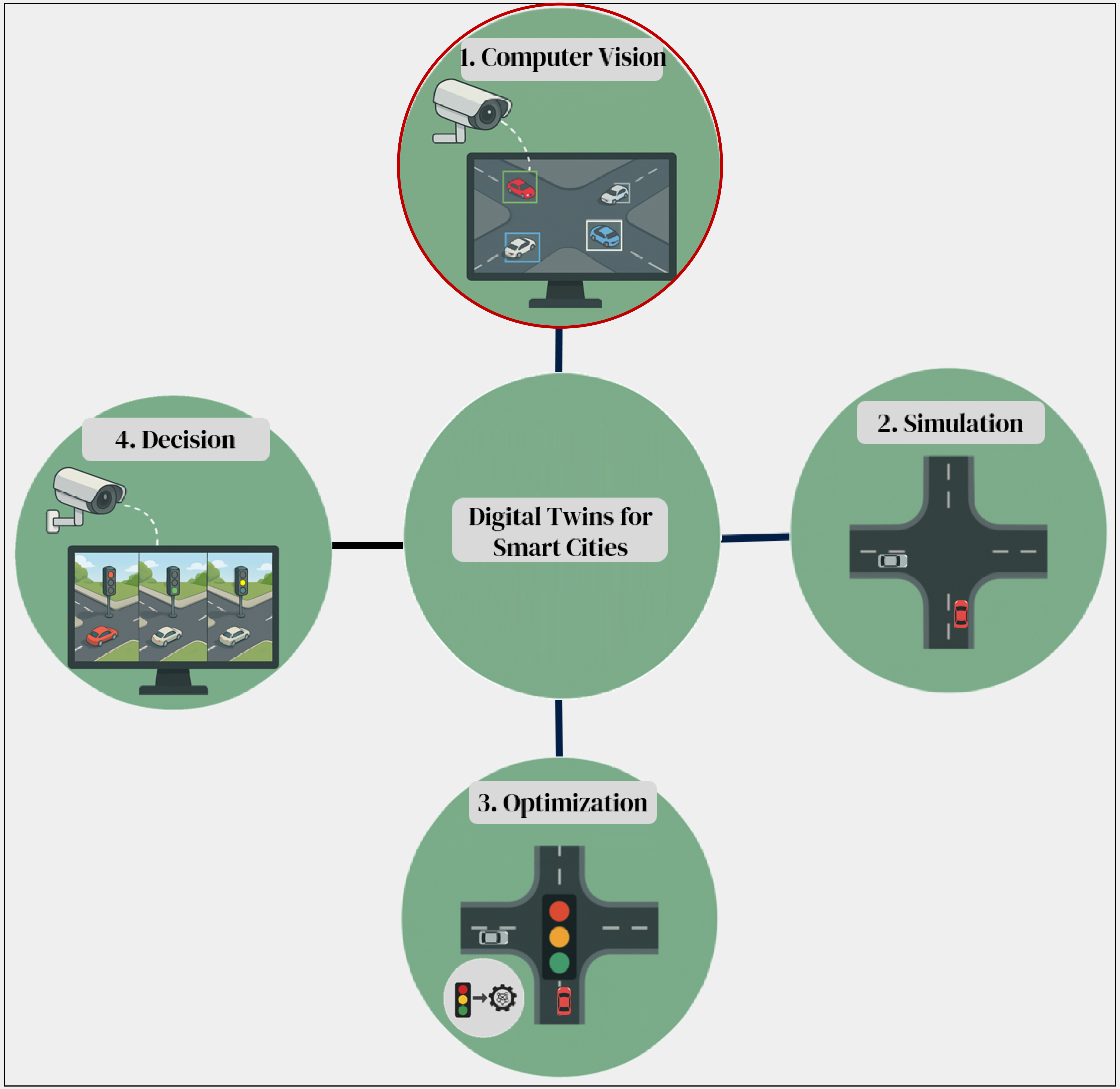


Overview of Course Syllabus in One Shot



Agenda

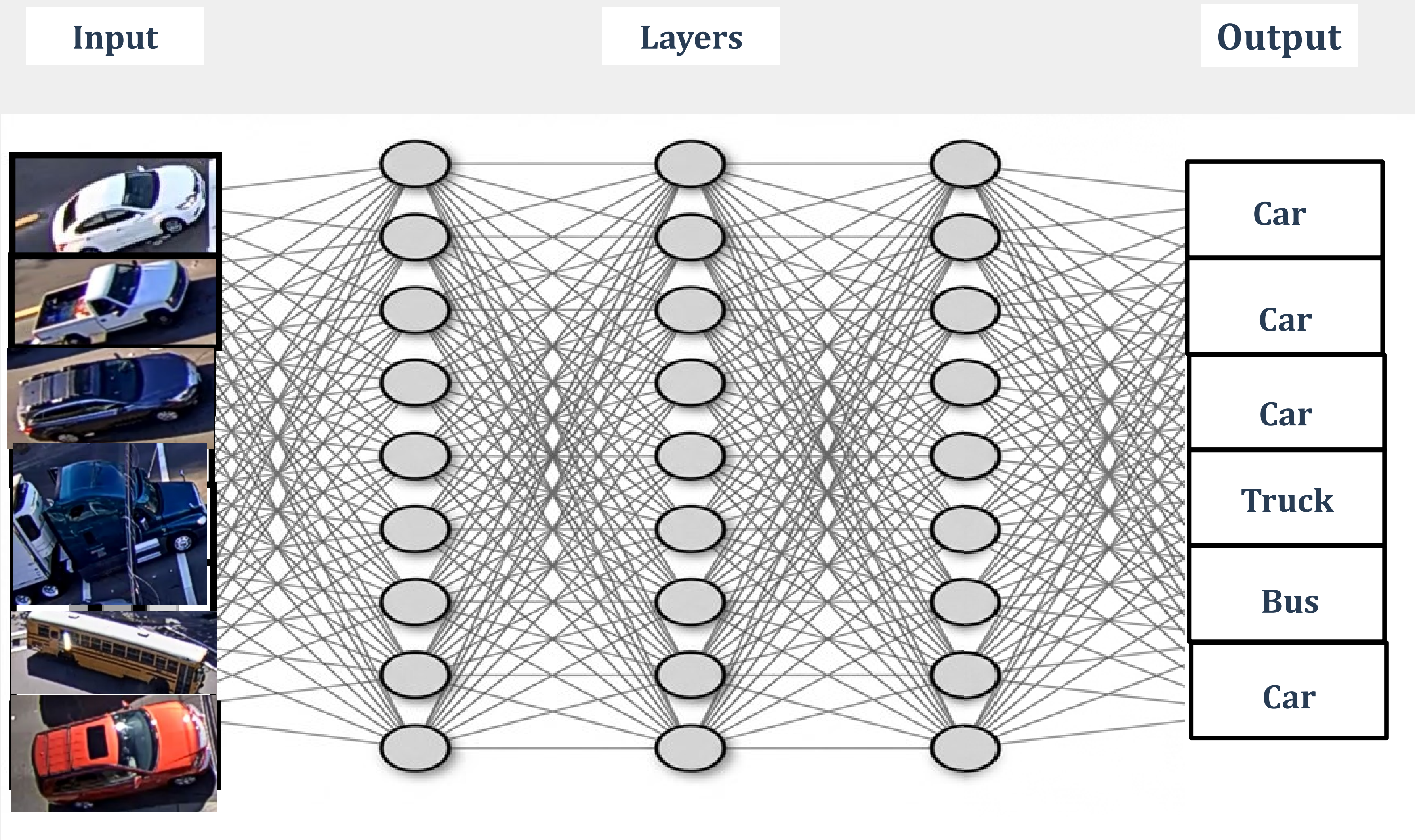
- ❑ Neural Network
- ❑ Convolutional Neural Network (CNN)
- ❑ 3 Stages of CNN
- ❑ 6 Steps of Object Detection using CNN
- ❑ Step 1: Study Area and Video Recording
- ❑ Step 2: Frame Extraction & Dataset Creation
- ❑ Step 3: Image Annotation & Class Definition



Convolutional Neural Network

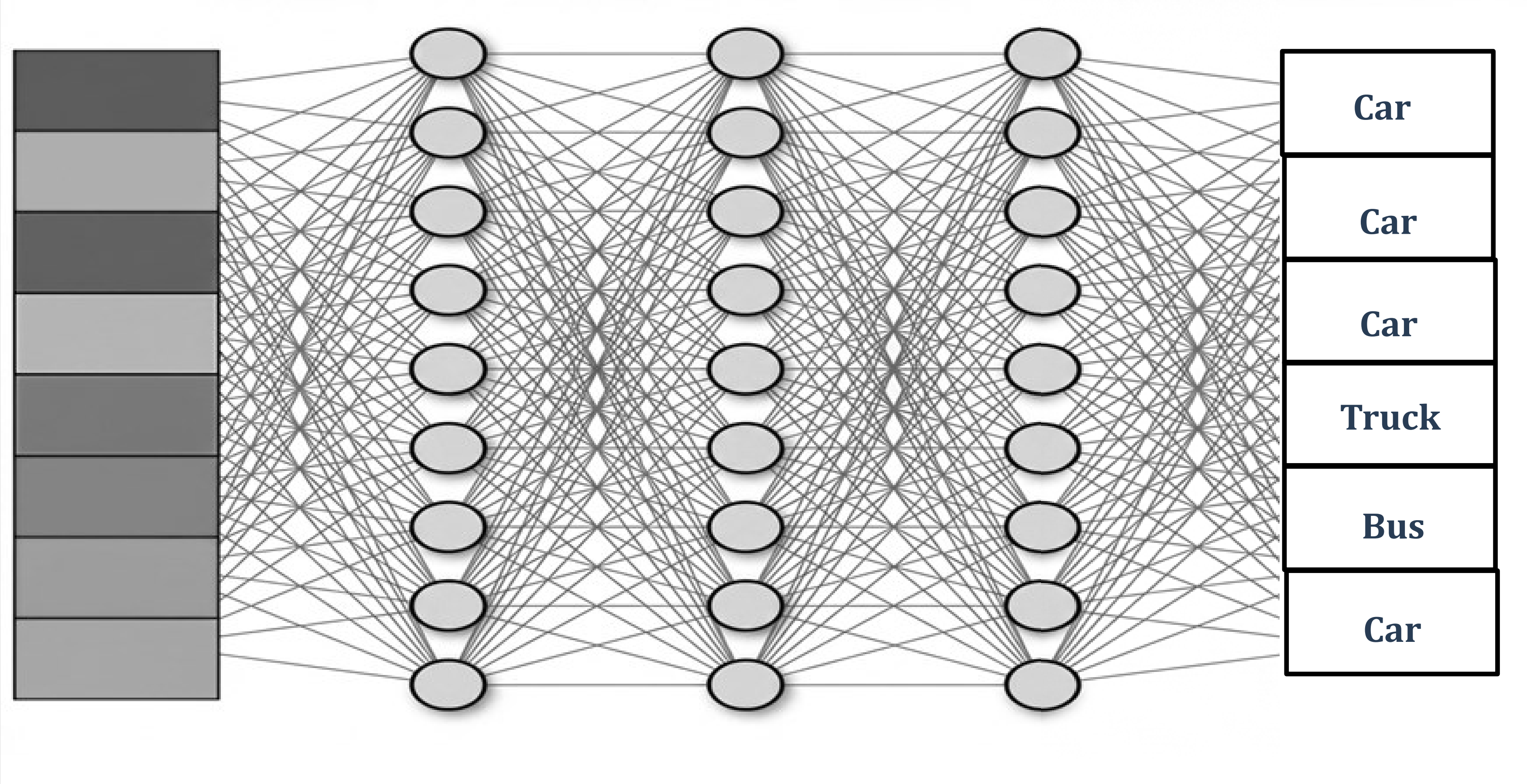
Convolution + $\boxed{\text{Neural Network}}$

Neural Network



Neural Network

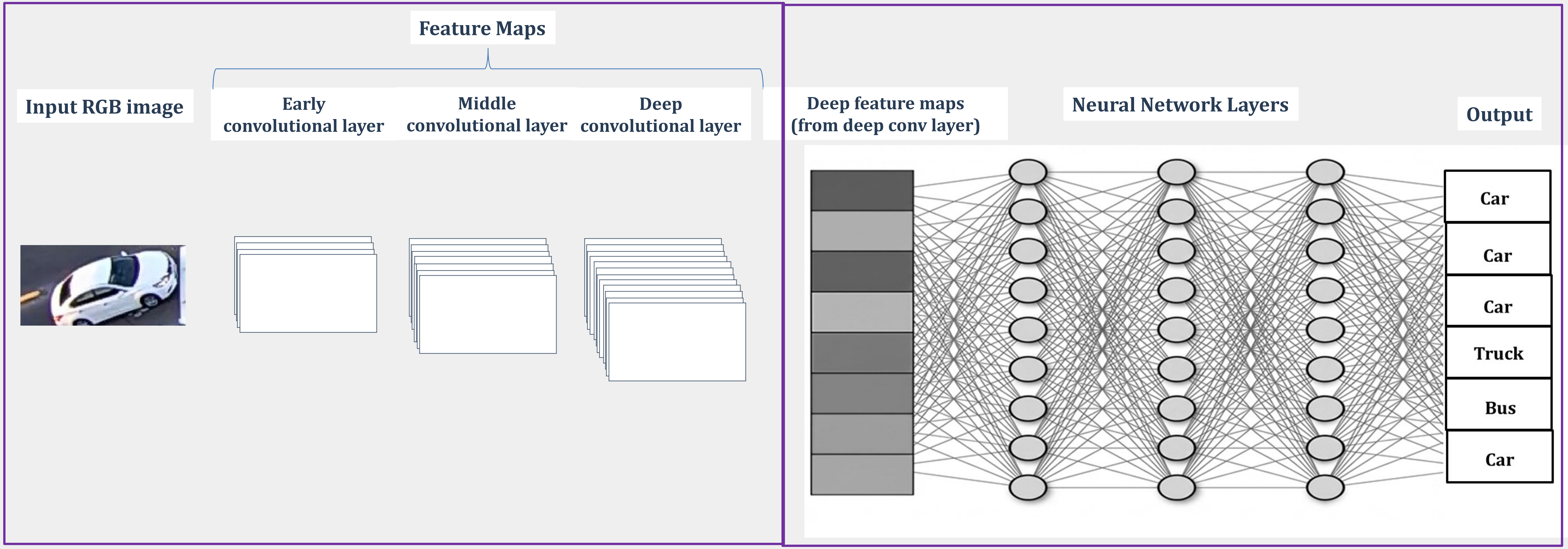
Input Layers Output



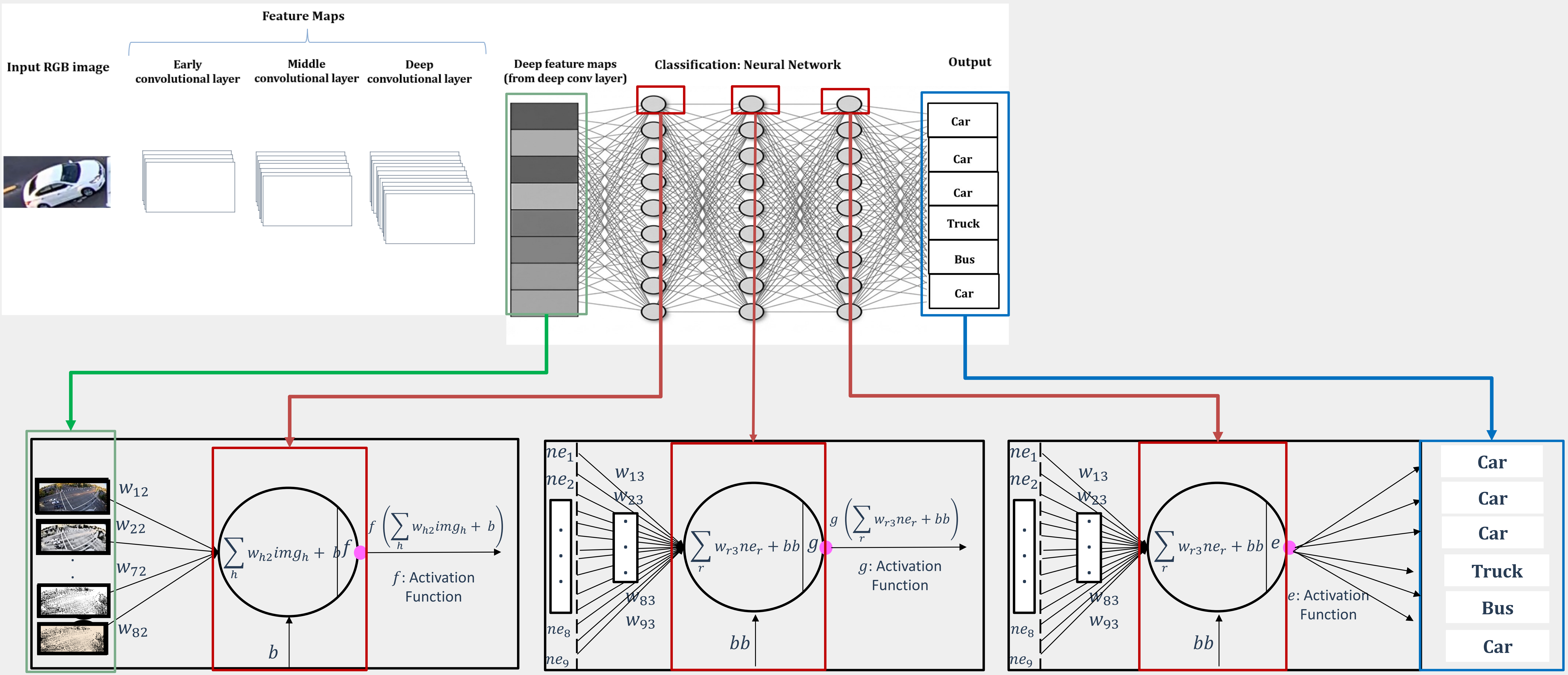
Convolutional Neural Network

Convolutional

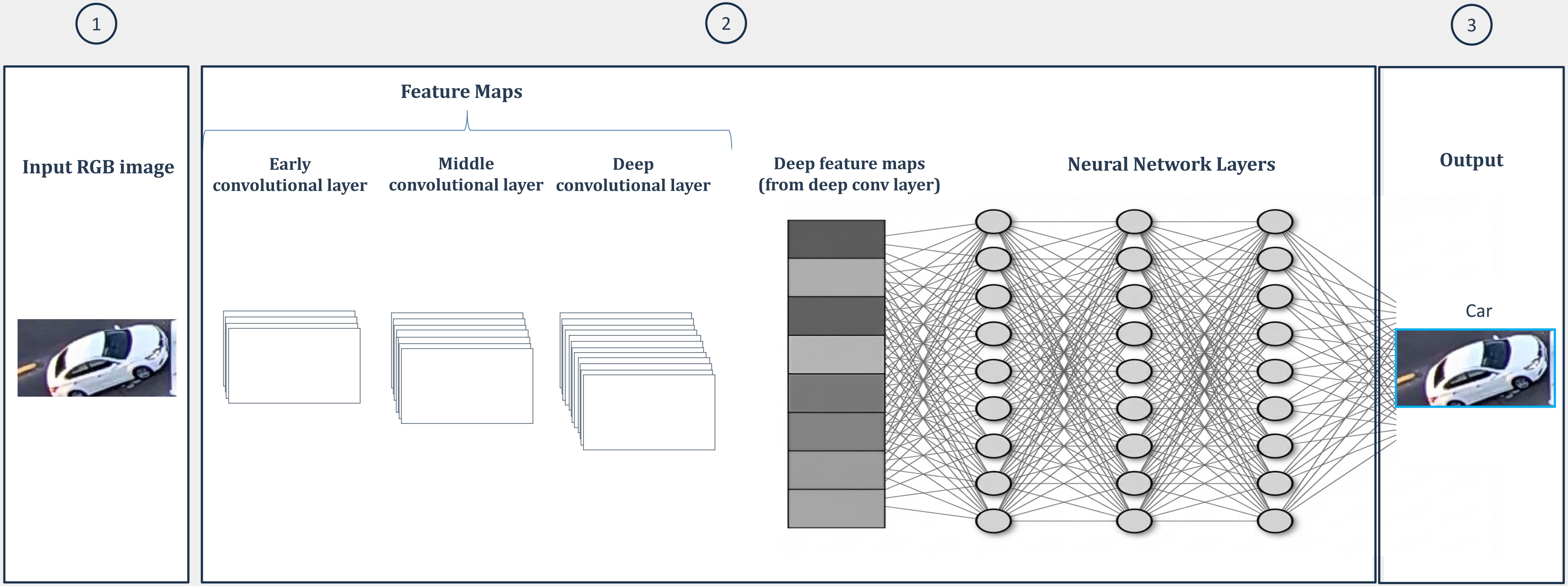
Neural Network



Convolutional Neural Network



3 Stages of Convolutional Neural Network



3 Stages of Convolutional Neural Network

1

Input RGB image

2

Algorithm: CNN

3

Output

Car

Class (car, bus etc)

Confidence score

3 Stages of Convolutional Neural Network

1

Input RGB image

2

Algorithm: YOLO (You Only Look Once)
(Enhanced version of classical CNN)

3

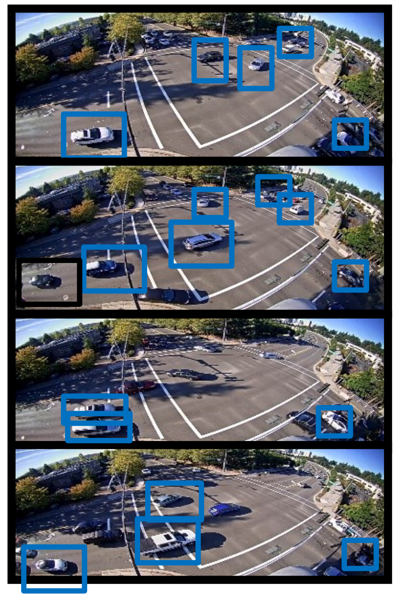
Class (car, bus etc)

Confidence score

Bounding Box

6 Steps in Convolutional Neural Network for Object Detection

➢ **Output: Object Detection**

1 **Input**	2 **Algorithm**	3 **Output**
	**Algorithm: YOLO**	 **Class (car, bus, etc)** **Confidence score** **Bounding Box**
Step 1: Study area and video recording Step 2: Frame extraction & dataset creation	Step 5: YOLO Trainer on Mini Datasets Step 6: Understanding the Result	Step 3.Image annotation & class definition Step 4.Dataset partitioning (train validation test)

Step 1: Study Area and Video Recording

Drone View (Easiest for tracking)

Blocking: Low - vehicles rarely overlap

View angle: Top-down - shapes look consistent across the scene

Distance clarity: High clarity - spacing is clear → stable tracks

Streetlight / Corner View (Moderate)

Blocking: Medium - vehicles can overlap; poles/trees may occlude

View angle: Elevated corner - far objects appear smaller

Distance clarity: Medium clarity - small/far vehicles are harder

Pedestrian View (Hardest for tracking)

Blocking: High - frequent overlap between cars/people

View angle: Ground-level - strong near-vs-far perspective changes

Distance clarity: lowest clarity

Step 1: Study Area and Video Recording

Video Recording Requirements:

- ❑ **Camera stability:** The camera should be fixed during recording (no zoom).
- ❑ **Mounting:** The camera should be mounted on a stable support (tripod/pole) to minimize vibration.
- ❑ **Frame rate:** Video should be recorded at a constant frame rate (recommended ≥ 30 FPS).
- ❑ **Resolution:** Video should be recorded at a resolution sufficient for object visibility (recommended ≥ 1080p).
- ❑ **Field of view:** The recording should include the full area of interest (intersection + approaches) with clear lane markings/stop lines where possible.
- ❑ **Lighting conditions:** The camera should be positioned to avoid extreme glare and backlighting when feasible.
- ❑ **Data integrity:** Videos should be stored without additional editing (no filters, no speed changes); original files are maintained.
- ❑ **Metadata:** Each recording should be documented with time/date, duration, camera height.

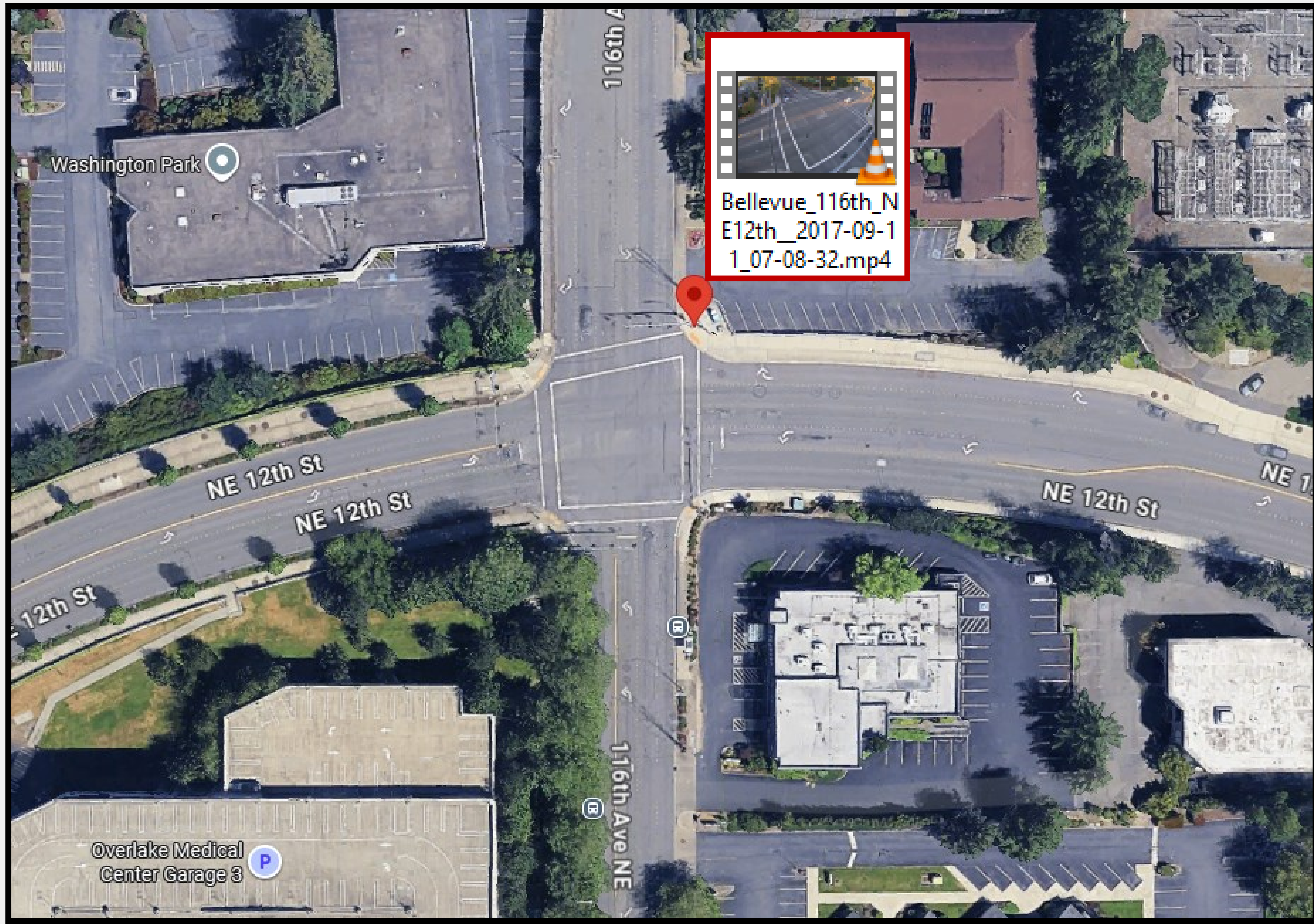
Step 1: Study Area and Video Recording

Dataset Summary

- Download Week2.Material.Zip
- Extract it
- In Folder “1.Study area & video recording”
- Open InputVideo.mp4

Key specifications

- **City:** Bellevue, Washington (USA)
- **Coordinates:** 47.62293830628107, -122.18554859584597
- **Date:** September 2017
- **View:** Streetlight / Corner View
- **Duration:** 3 minutes



Step 2: Frame Extraction & Dataset Creation

➢ **Task: Extract frames from the sample video to create an image dataset.**

Procedure

1. Open the MP4 in a video player/editor.

2. Extract frames at a fixed interval (every 6 seconds for this exercise) - Take Screenshot

3. **I included 1.jpg as a sample (do not remove or modify it).**

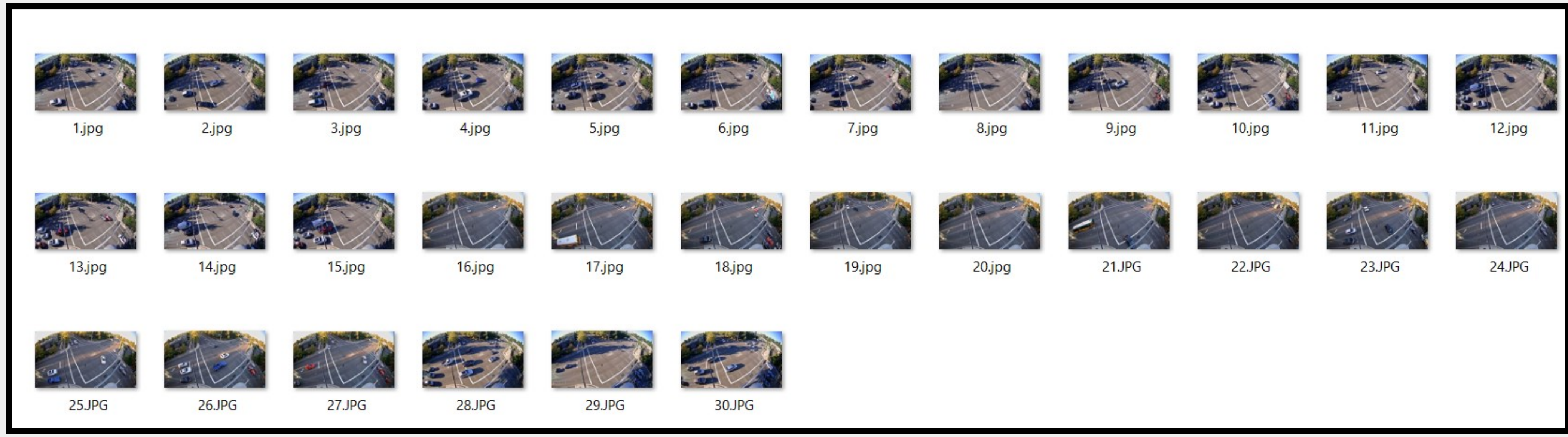
4. Save frames to Folder: “2.Frame extraction & dataset creation\images”

5. Start saving your extracted frames from 2.jpg, then 3.jpg … up to 30.jpg.

Expected output

➢ 30 images (one dataset folder)

➢ Calculation: 3 min = 180 s → 900 / 6 ≈ 30 frames



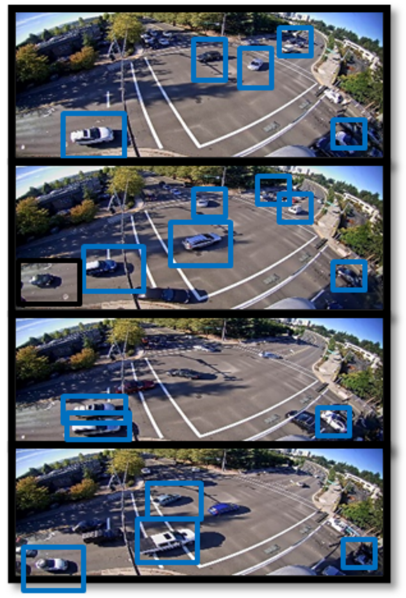
Stage 1 Completed

1 **Input**

2 **Algorithm**

3 **Output**

Algorithm: YOLO



Class (car, bus, etc)
Confidence score
Bounding Box

Step 1: Study area and video recording
Step 2: Frame extraction & dataset creation

Step 5: YOLO Trainer on Mini Datasets
Step 6: Understanding the Result

Step 3.Image annotation & class definition
Step 4.Dataset partitioning (train validation test)

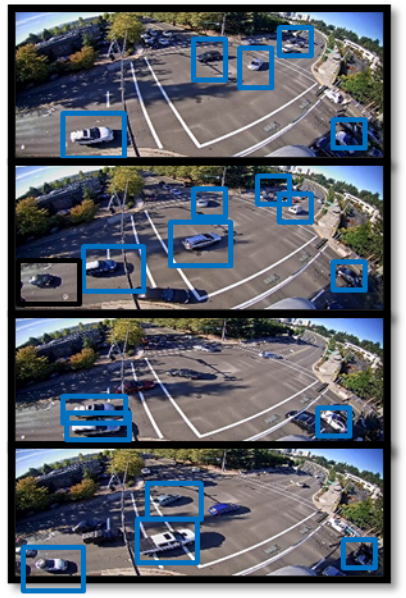
Stage 3: In Progress

1 **Input**

2 **Algorithm**

3 **Output**

Algorithm: YOLO



Class (car, bus, etc)
Confidence score
Bounding Box

Step 1: Study area and video recording
Step 2: Frame extraction & dataset creation

Step 5: YOLO Trainer on Mini Datasets
Step 6: Understanding the Result

Step 3.Image annotation & class definition
Step 4.Dataset partitioning (train validation test)

Step 3: Image Annotation & Class Definition

Goal: Annotate image dataset which means giving labels manually.

Procedure

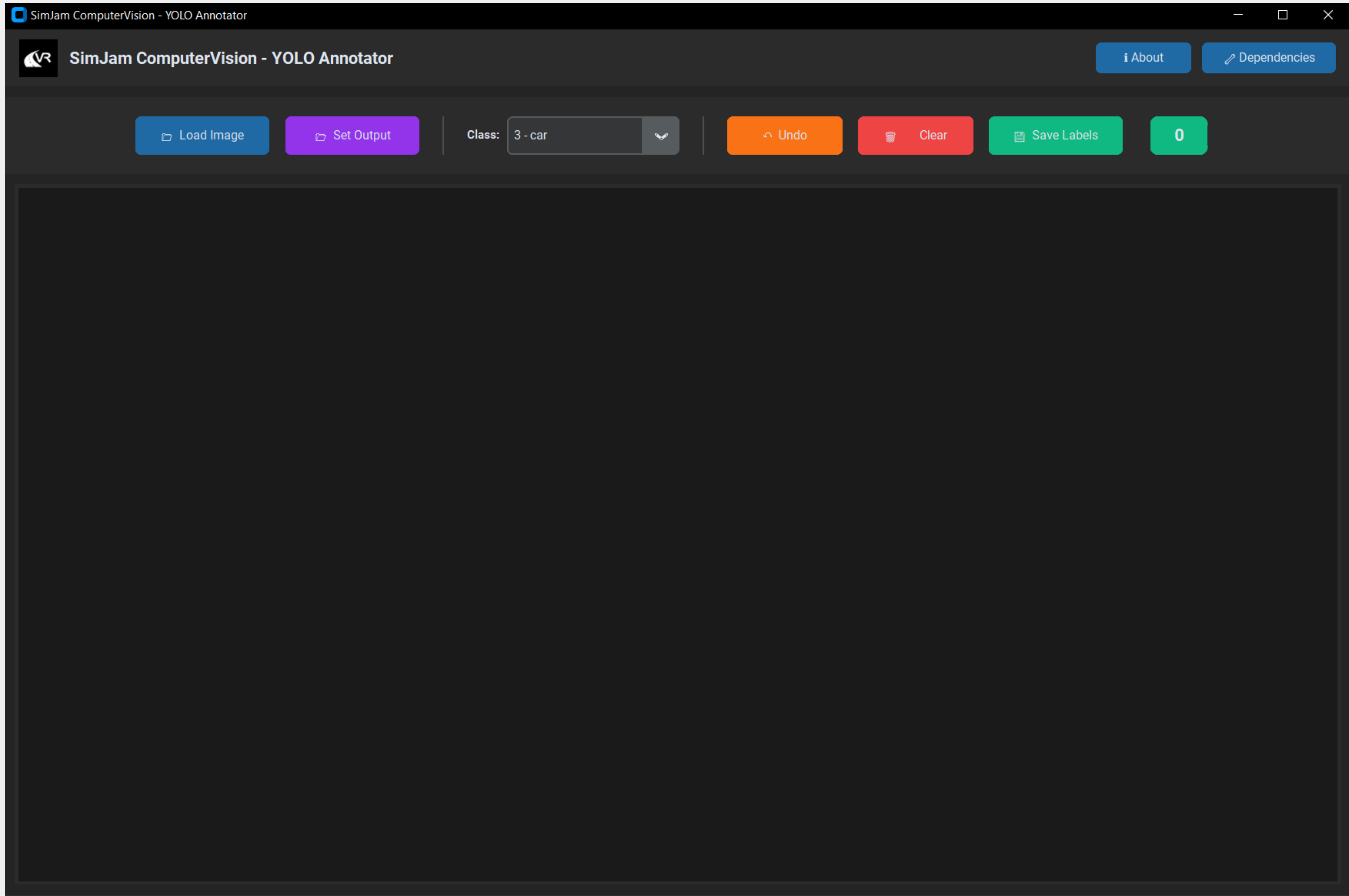
1. Open Folder “3.Image annotation & class definition”

2. Inside the Folder → Create a folder “images”

3. Create a folder “labels”

4. Copy and paste all images from step 2 to folder “images”

5. Run application “SimJamCVAnnotator.py”



Step 3: Image Annotation & Class Definition

Goal: Annotate image dataset which means giving labels manually.

Procedure

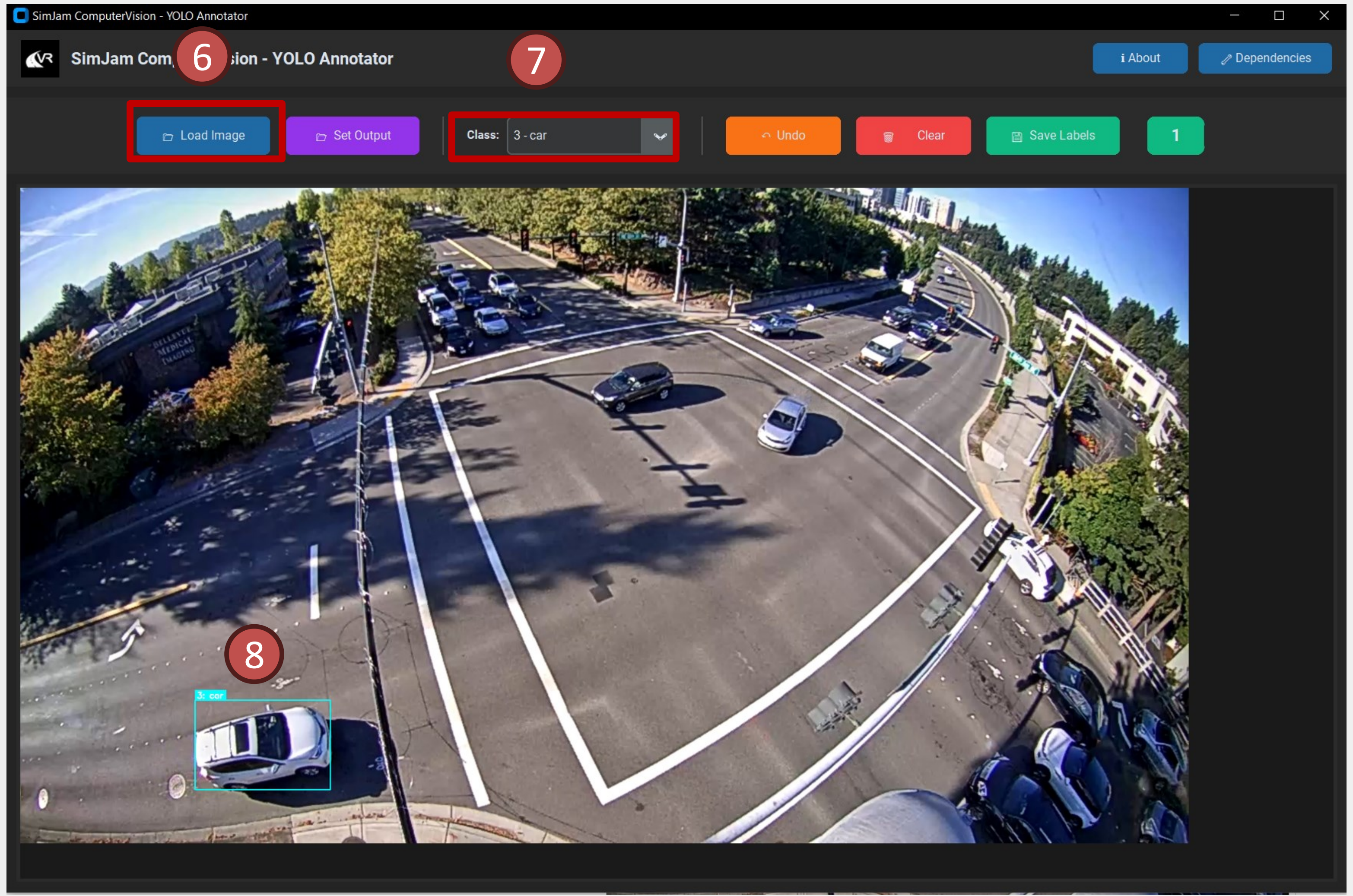
6. Click Load Image → open the folder “3. Image Annotation & Class Definition\images” → select

Image 1.

Note: Your image may look different from the example below.

7. From the Class dropdown (top bar), select Car.

8. Left-click and drag to draw a bounding box that tightly covers a car.



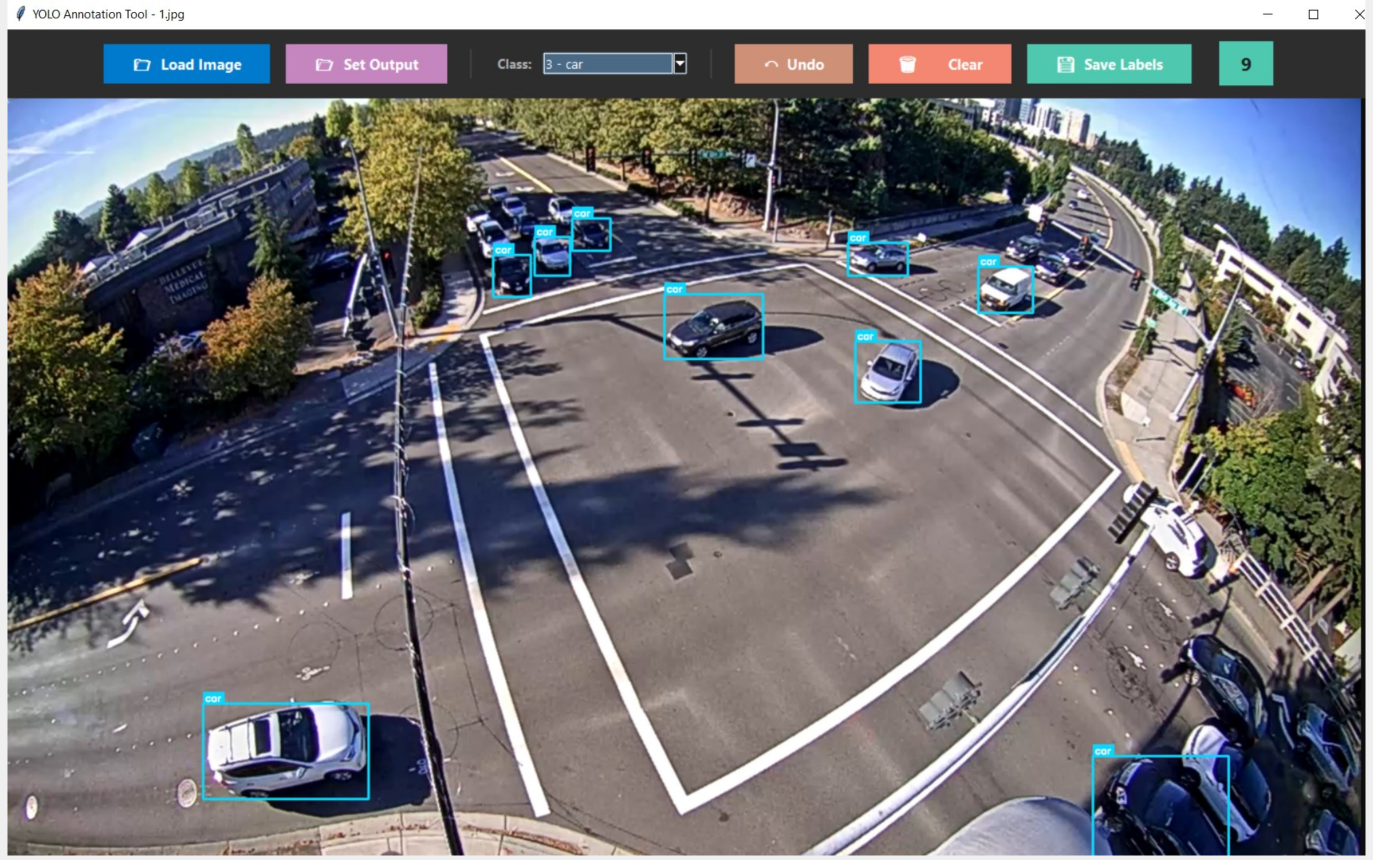
Step 3: Image Annotation & Class Definition

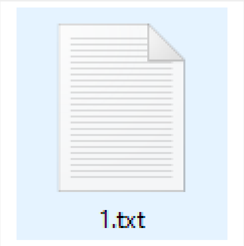
9. The bounding box should tightly fit the car, with minimal background space.

Step 3: Image Annotation & Class Definition

10. Repeat this for all other cars **(Do not label other vehicles (e.g., vans, buses, trucks, etc.)** in the image.

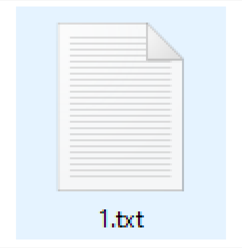
11. When finished, select “Output Directory” and set it to the 3. Image Annotation & Class Definition\labels, then save the labels as “1” (creates 1.txt).





Step 3: Image Annotation & Class Definition

12. Open the stored txt file



Class

Bounding box coordinate

```
3 0.206296 0.851916 0.113805 0.133101
3 0.522878 0.301045 0.071959 0.076655
3 0.652522 0.358885 0.045757 0.087805
3 0.736801 0.251916 0.039890 0.059233
3 0.642745 0.208362 0.041064 0.040418
3 0.372702 0.232056 0.032069 0.047387
3 0.430192 0.177700 0.028940 0.045993
3 0.362534 0.182230 0.025029 0.046690
3 0.403402 0.203833 0.030896 0.041115
3 0.387955 0.164808 0.022683 0.034146
3 0.377200 0.140418 0.021510 0.031359
3 0.412397 0.144599 0.022292 0.036934
3 0.364685 0.122997 0.017599 0.028571
3 0.859014 0.936934 0.095033 0.144251
3 0.900274 0.900000 0.065702 0.154007
3 0.896754 0.775261 0.064920 0.155401
3 0.970278 0.858188 0.050059 0.128920
3 0.980055 0.969338 0.042237 0.071080
3 0.770434 0.226481 0.027376 0.039024
3 0.751662 0.199303 0.028940 0.037631
3 0.789793 0.209408 0.017599 0.025784
```

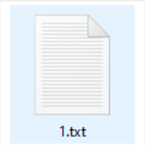
0.206296 = x_center (normalized, 0–1)

0.851916 = y_center (normalized, 0–1)

0.113805 = width of the box (normalized, 0–1)

0.133101 = height of the box (normalized, 0–1)

Step 3: Image Annotation & Class Definition



Class

Bounding box coordinate

```
3 0.206296 0.851916 0.113805 0.133101
3 0.522878 0.301045 0.071959 0.076655
3 0.652522 0.358885 0.045757 0.087805
3 0.736801 0.251916 0.039890 0.059233
3 0.642745 0.208362 0.041064 0.040418
3 0.372702 0.232056 0.032069 0.047387
3 0.430192 0.177700 0.028940 0.045993
3 0.362534 0.182230 0.025029 0.046690
3 0.403402 0.203833 0.030896 0.041115
3 0.387955 0.164808 0.022683 0.034146
3 0.377200 0.140418 0.021510 0.031359
3 0.412397 0.144599 0.022292 0.036934
3 0.364685 0.122997 0.017599 0.028571
3 0.859014 0.936934 0.095033 0.144251
3 0.900274 0.900000 0.065702 0.154007
3 0.896754 0.775261 0.064920 0.155401
3 0.970278 0.858188 0.050059 0.128920
3 0.980055 0.969338 0.042237 0.071080
3 0.770434 0.226481 0.027376 0.039024
3 0.751662 0.199303 0.028940 0.037631
3 0.789793 0.209408 0.017599 0.025784
```

0.206296 = x_center (normalized, 0–1)

0.851916 = y_center (normalized, 0–1)

0.113805 = width of the box (normalized, 0–1)

0.133101 = height of the box (normalized, 0–1)

Step 3: Image Annotation & Class Definition

Goal: How to show the x_center and y_center on the image (Red Circle)

13. Open **Image 1** in **GIMP**.

14. **Top menu bar → Image → Image Scale →** Record number of pixels for Width and Height

Width: 2557

Height: 1435

Now, by multiplying x_center and y_center (in previous slide) to W and H, you find pixel coordinate for centre of the car

- x_px = x_center * W = 0.206296 * 2557 = 527
- y_px = y_center * H = 0.851916 * 1435 = 1223

15. Move the cursor to the car center (red dot) and confirm the status bar shows approximately (527, 1223).

Step 3: Image Annotation & Class Definition

Goal: Show how to turn (x_center, y_center, width, height) into the four box corners.

14. Convert to corner coordinates:

x1 = (x_center - width/2) * W

y1 = (y_center - height/2) * H

x2 = (x_center + width/2) * W

y2 = (y_center + height/2) * H

Then draw a rectangle from (x1, y1) to (x2, y2).

(x1, y1) = top-left corner of the bounding box

(x2, y1) = top-right corner

(x1, y2) = bottom-left corner

(x2, y2) = bottom-right corner