TEXAS TECH UNIVERSITY SYSTEM™

Autonomous Navigation with Unitree Go1 Quadruped Robot

Implementation of SLAM, Object Detection plus tracking, and Path Planning

Graduate Project Lab

Week 6

Rishikesh

October 15, 2024

1. SLAM Basics

Mathematical Foundations

- **Core Components**:
  - **State Estimation**: Estimating the robot's pose (position and orientation)
  - **Map Building**: Creating a representation of the environment
- **Key Mathematical Concepts**:
  - **Probability Theory**: Handling uncertainty
  - **Bayesian Filters**: Recursive estimation (e.g., Kalman Filter, Particle Filter)
  - **Linear Algebra**: Transformations and coordinate frames

The SLAM Problem Formulation

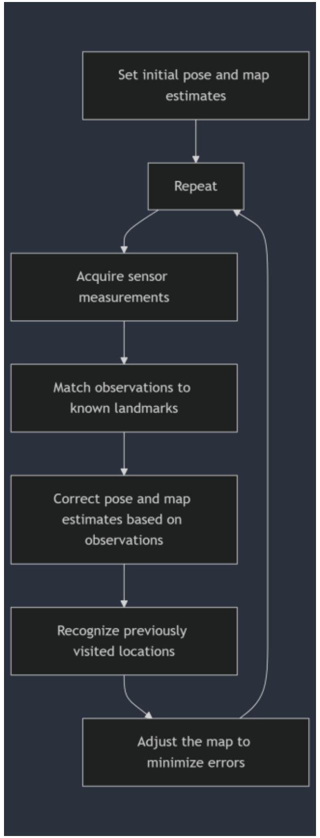
- **State Vector ({x})**:
  - Robot pose and map features
- **Observations ({z})**:
  - Sensor measurements (e.g., LiDAR scans)
- **Controls ({u})**:
  - Robot motion commands
- **Objective**:
  - Estimate the posterior $P(x|z1{:}t,u1{:}t)$

2. Bayesian Approach to SLAM

Bayesian Approach to SLAM

- **Recursive Bayesian Estimation**:
  - **Prediction Step**:
    - $\hat{\mathbf{x}}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_t) + \mathbf{w}_t$
    - $\mathbf{w}_t$: Process noise
  - **Update Step**:
    - $\mathbf{x}_t = \hat{\mathbf{x}}_t + K_t(\mathbf{z}_t - h(\hat{\mathbf{x}}_t))$
    - K_t: Kalman Gain
    - $h(\hat{\mathbf{x}}_t)$: Measurement model
- **Assumptions**:
  - Markov property
  - Gaussian noise (for EKF SLAM)

3. SLAM Algorithm Flowchart



- **Step 1: Initialization**
  - Set $\mathbf{x}_0 = [x_0, y_0, \theta_0]$
  - Empty map $\mathcal{M}$
- **Step 2: Motion Prediction**
  - $\hat{\mathbf{x}}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_t)$
- **Step 3: Sensor Measurement**
  - Obtain $\mathbf{z}_t$
- **Step 4: Data Association**
  - Match $\mathbf{z}_t$ to landmarks in $\mathcal{M}$
- **Step 5: State Update**
  - Compute $\mathbf{x}_t$ and update $\mathcal{M}$
- **Step 6: Map Optimization**
  - Adjust map to minimize residual errors

```
initialize(x_0, M)
for t in 1...T:
    # Motion Prediction
    x_pred = predict_motion(x_{t-1}, u_t)

    # Sensor Measurement
    z_t = get_sensor_data()

    # Data Association
    associations = associate_data(z_t, M)

    # State and Map Update
    x_t, M = update_state_map(x_pred, z_t, associations)

    # Map Optimization (if necessary)
    M = optimize_map(M)

  return go(f, seed, [])
}
```

4. Setting up SLAM for Software Development

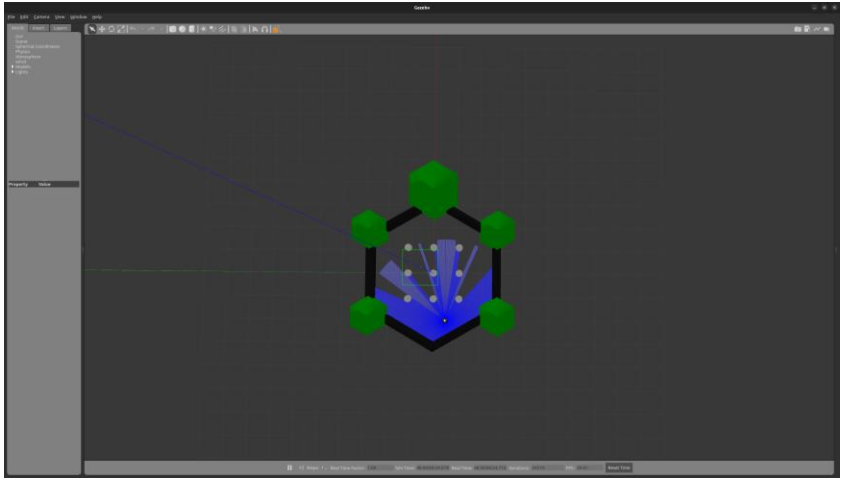
Installation and Setup

- **Step 1: Install ROS 2 Humble**
  - Update system packages
  - Add ROS 2 repository and keys
  - Install ROS 2 desktop packages
- **Step 2: Create ROS 2 Workspace**
  - `mkdir -p ~/ros2_ws/src`
  - Initialize workspace with `colcon build`
- **Step 3: Clone Necessary Packages**
  - `turtlebot3`, `turtlebot3_msgs`, `turtlebot3_simulations`
  - Use compatible branches (e.g., `humble-devel`)

Installing Dependencies

- **Use `rosdep` to Install Dependencies**
  - `rosdep update`
  - `rosdep install --from-paths src --ignore-src -r -y`
- **Build the Workspace**
  - `colcon build --symlink-install`
- **Source the Workspace**
  - `source ~/ros2_ws/install/setup.bash`
  - Add to `~/.bashrc` for automatic sourcing

5. Launching SLAM



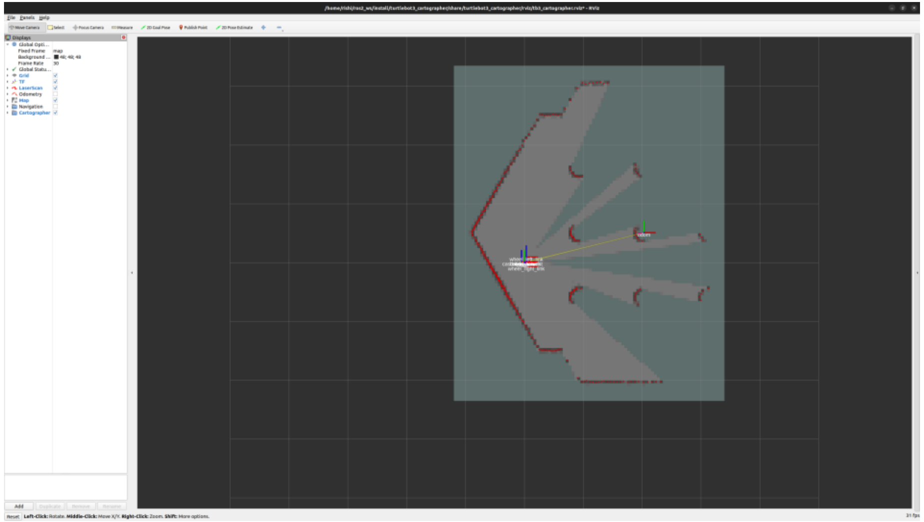
Gazebo: Simulates the robot and environment

- **Set the Robot Model Environment Variable**
  - `export TURTLEBOT3_MODEL=burger`
- **Terminal 1: Launch Gazebo Simulation**
  - `ros2 launch turtlebot3_gazebo turtlebot3_world.launch.py`
- **Terminal 2: Launch Cartographer SLAM Node**
  - `ros2 launch turtlebot3_cartographer cartographer.launch.py use_sim_time:=True`

6. Visualization in Real Time

- **Visualizing with RViz**:
  - `rviz2 -d ~/ros2_ws/src/turtlebot3_cartographer/rviz/turtlebot3_cartographer.rviz`
- **RViz Displays**:
  - **Robot Model**: Visual representation of TurtleBot3
  - **Map**: 2D occupancy grid map being built in real-time
  - **Laser Scan**: Sensor data visualization
- **Interacting with RViz**:
  - Zoom, pan, and rotate the view
  - Add or remove display elements

RViz: Visualizes the robot's pose and the map being built



7. COCO DATASET

What is COCO?

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

Over 100 Classes:

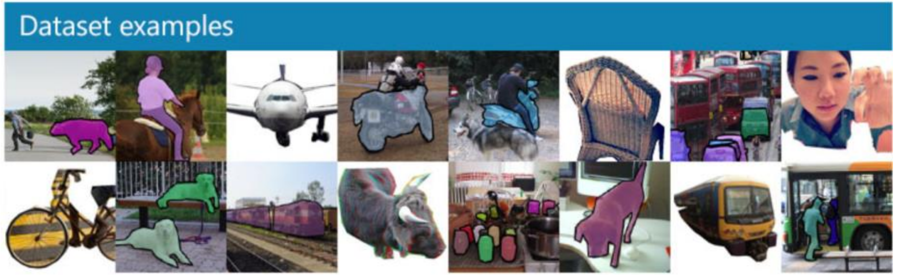
- Humans
- Animals
- Vehicles
- Household Items etc.

Over 200,000 Images

Very Large when trained for object detection (80 MB)

COCO

Common Objects in Context



8. YOLOV8 Object Detection

Inference with a Pre-Built Model

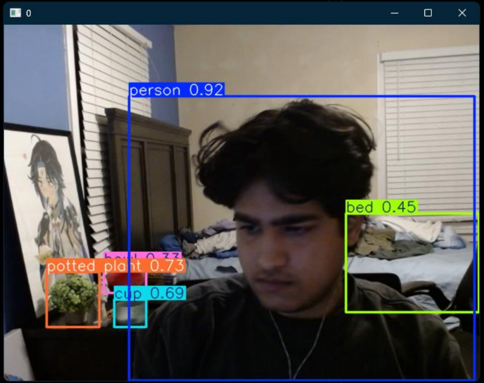
```python
from ultralytics import YOLO

# Load model
model = YOLO("yolov8s.pt")

# Inference
results = model(source=0, show=True)
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SERIAL MONITOR JUPYTER

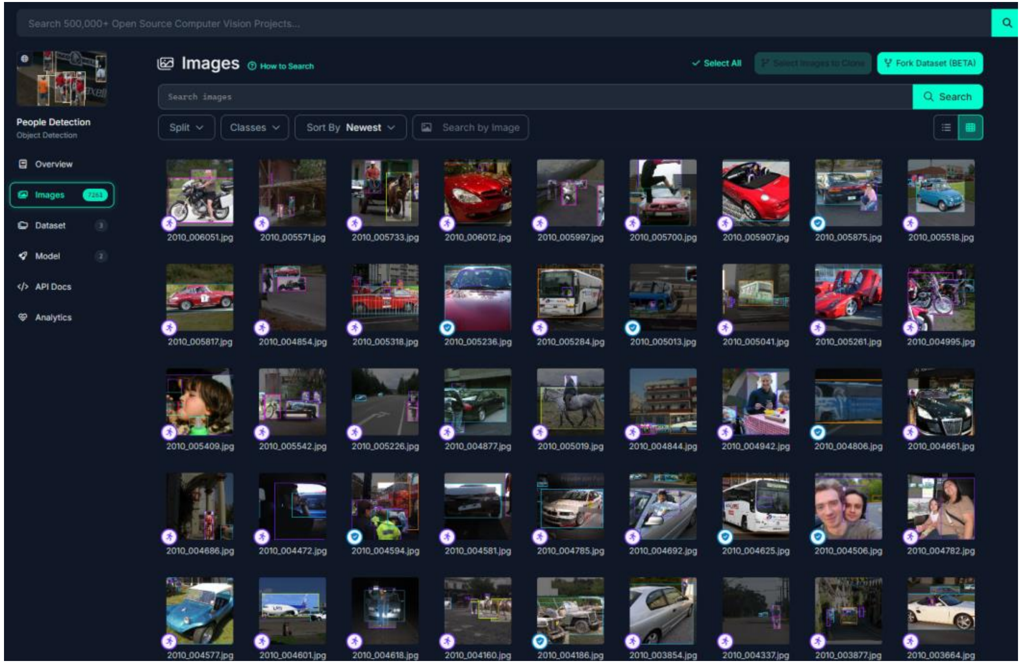
```
0: 480x640 1 person, 2 cups, 1 potted plant, 1 bed, 9.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 2 cups, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 2 cups, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 2 beds, 10.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 9.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 2 beds, 12.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 2 cups, 1 bowl, 1 potted plant, 1 bed, 12.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 14.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 11.5ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 12.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 12.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 12.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 10.5ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 12.0ms
0: 480x640 1 person, 1 cup, 1 potted plant, 1 bed, 10.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 11.0ms
0: 480x640 1 person, 2 cups, 1 bowl, 1 potted plant, 1 bed, 12.0ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 10.5ms
0: 480x640 1 person, 1 cup, 1 bowl, 1 potted plant, 1 bed, 11.0ms
```



OUTLINE

9. Training an Object Detection Model from Scratch

Roboflow: Open-Source Image Annotation and Dataset Library




```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="3Xj3eINZEW8xa3RNGle1")
project = rf.workspace("leo-ueno").project("people-detection-o4rdr")
version = project.version(8)
dataset = version.download("yolov8")
```

10. Training Results

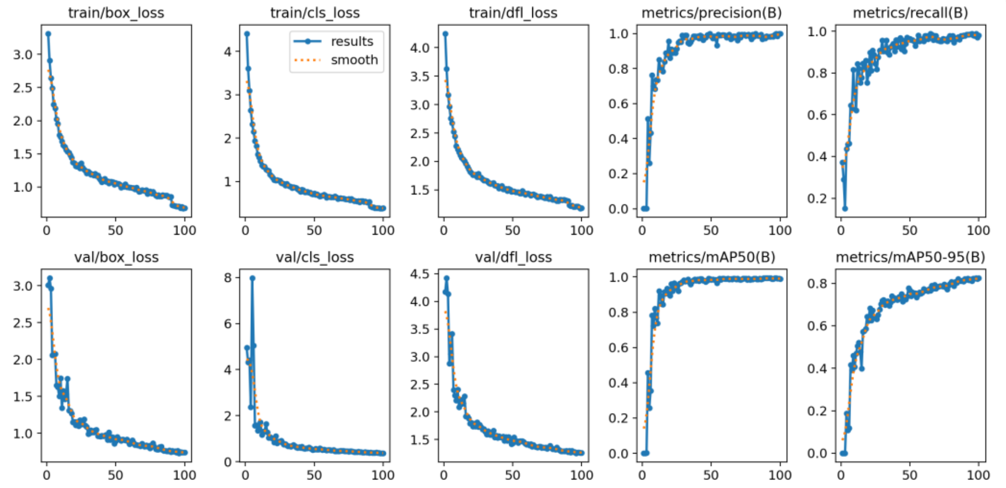
```python
if __name__ == "__main__":
    # Train the model
    results = model.train(data="D:\Documents\Object_Training\data.yaml", epochs=100, imgsz=640)
```

Python

```
New https://pypi.org/project/ultralytics/8.2.72 available  Update with 'pip install -U ultralytics'
Ultralytics YOLOv8.2.68  Python-3.9.5 torch-2.0.0+cu117 CUDA:0 (NVIDIA GeForce RTX 2070, 8192MiB)
engine\trainer: task=detect, mode=train, model=yolov8n.yaml, data=D:\Documents\Object_Training\data.yaml, epo
Overriding model.yaml nc=80 with nc=1

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
  2                  -1  1      7360  ultralytics.nn.modules.block.C2f             [32, 32, 1, True]
  3                  -1  1     18560  ultralytics.nn.modules.conv.Conv             [32, 64, 3, 2]
  4                  -1  2     49664  ultralytics.nn.modules.block.C2f             [64, 64, 2, True]
  5                  -1  1     73984  ultralytics.nn.modules.conv.Conv             [64, 128, 3, 2]
  6                  -1  2    197632  ultralytics.nn.modules.block.C2f             [128, 128, 2, True]
  7                  -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  8                  -1  1    460288  ultralytics.nn.modules.block.C2f             [256, 256, 1, True]
  9                  -1  1    164608  ultralytics.nn.modules.block.SPPF            [256, 256, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 12                  -1  1    148224  ultralytics.nn.modules.block.C2f             [384, 128, 1]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 14             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 15                  -1  1     37248  ultralytics.nn.modules.block.C2f             [192, 64, 1]
 16                  -1  1     36992  ultralytics.nn.modules.conv.Conv             [64, 64, 3, 2]
 17            [-1, 12]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 18                  -1  1    123648  ultralytics.nn.modules.block.C2f             [192, 128, 1]
...
TensorBoard: Start with 'tensorboard --logdir runs\detect\train12', view at http://localhost:6006/
Freezing layer 'model.22.dfl.conv.weight'
AMP: running Automatic Mixed Precision (AMP) checks with YOLOv8n...
AMP: checks passed
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
train: Scanning D:\Documents\Object_Training\dataset\RGB\train\labels.cache... 652 images, 0 backgrounds, 0 c
albumentations: Blur(p=0.01, blur_limit=(3, 7)), MedianBlur(p=0.01, blur_limit=(3, 7)), ToGray(p=0.01), CLAHE
val: Scanning D:\Documents\Object_Training\dataset\RGB\valid\labels.cache... 186 images, 0 backgrounds, 0 cor
Plotting labels to runs\detect\train12\labels.jpg...
optimizer: 'optimizer=auto' found, ignoring 'lr0=0.01' and 'momentum=0.937' and determining best 'optimizer',
optimizer: AdamW(lr=0.002, momentum=0.9) with parameter groups 57 weight(decay=0.0), 64 weight(decay=0.0005),
TensorBoard: model graph visualization added
Image sizes 640 train, 640 val
Using 8 dataloader workers
Logging results to runs\detect\train12
Starting training for 100 epochs...

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      1/100     2.34G      3.312      4.414      4.248         33        640: 100%|██████████| 41/41 [00:07<
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 6/
                   all        186        186    0.00124      0.371   0.000862    0.00026
```



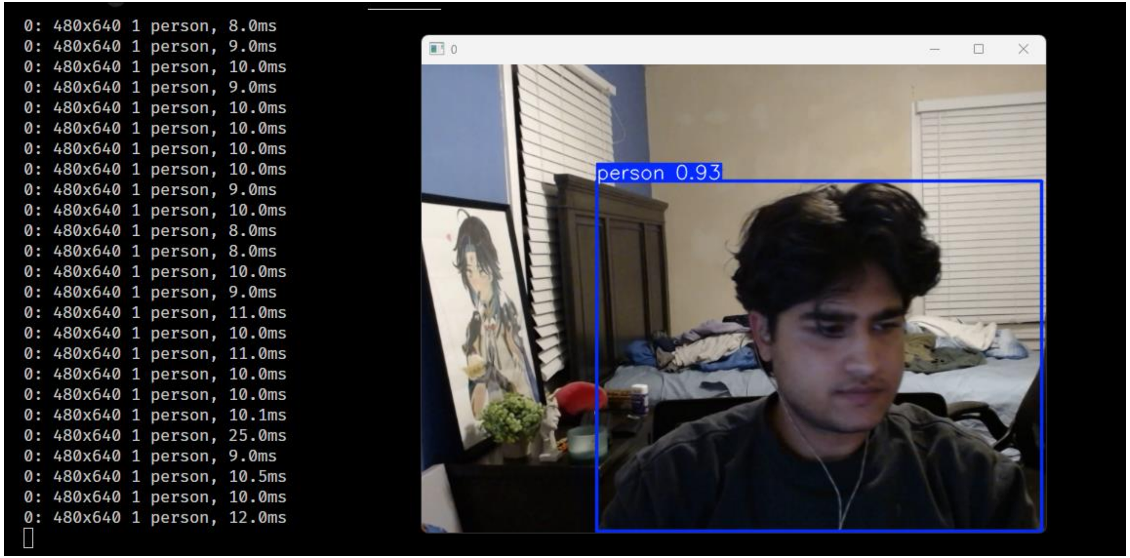

```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     98/100     2.31G      0.703     0.3785      1.201         12        640: 100%|██████████| 41/41 [00:05<
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 6/
                   all        186        186      0.984      0.988      0.991      0.825

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
     99/100     2.31G     0.6771     0.3779      1.174         12        640: 100%|██████████| 41/41 [00:05<
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 6/
                   all        186        186          1      0.971       0.99      0.821

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
    100/100     2.32G     0.6845     0.3845      1.182         12        640: 100%|██████████| 41/41 [00:05<
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 6/
                   all        186        186          1      0.981      0.988      0.825

100 epochs completed in 0.211 hours.
Optimizer stripped from runs\detect\train12\weights\last.pt, 6.3MB
Optimizer stripped from runs\detect\train12\weights\best.pt, 6.3MB
```

11. New Light Weight Model!!



TEXAS TECH UNIVERSITY SYSTEM™