XILINX®

Zynq US+ Interior Cabin Monitoring System & DMS - Demo Development

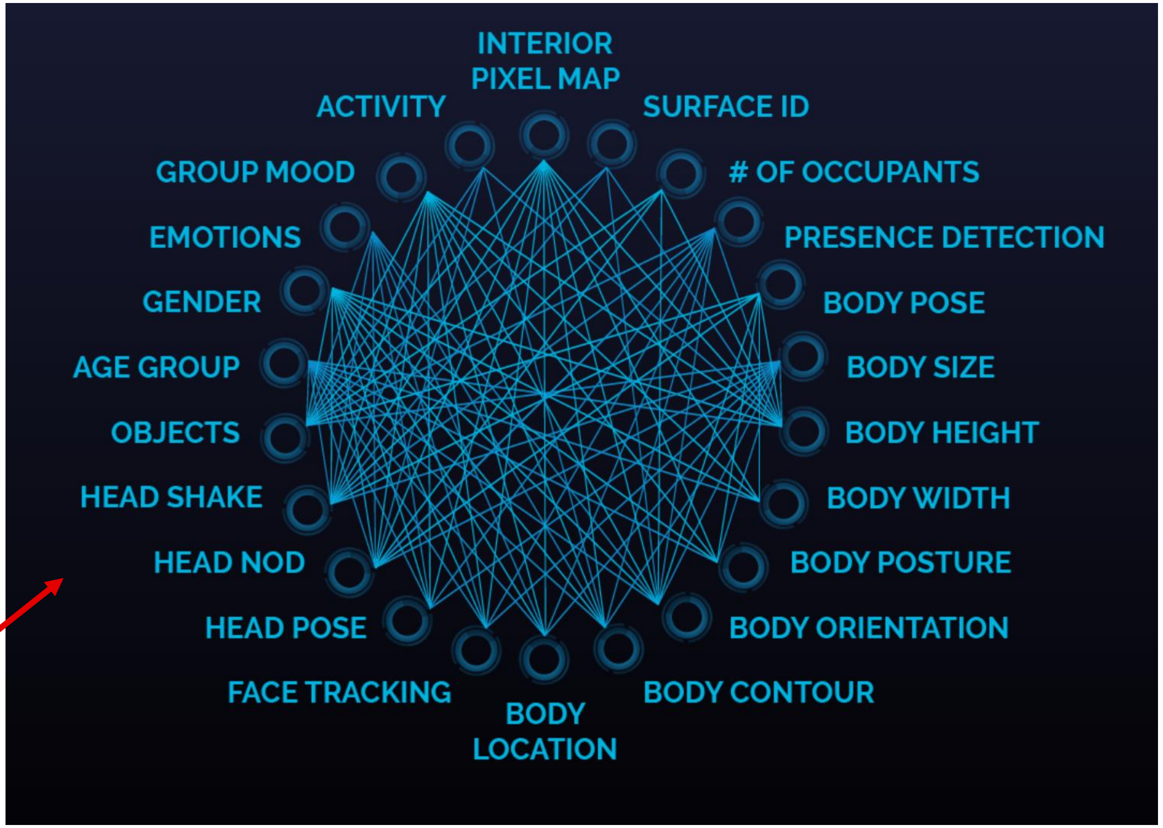
Jon Cory
Machine Learning Specialist FAE

Agenda

- ICMS Introduction
- Model Selection, Design, & Pruning
  - 2D Detection
  - Pose Estimation
  - Face Detection
- Deployment Software
- Distracted Driver Development
- Summary

Interior Cabin Monitoring Systems

- ICMS can involve many tasks:
  - Detecting the location/presence of occupants
    - This could be used to remove seat bladders for occupant detection, or as preprocessing for other tasks
  - Detecting child/infant seats
    - Could be used to ensure no child is left in the vehicle
  - Detecting the body position of the occupants (leaning forward/backward)
    - e.g. to assist with how to deploy air bags
  - Detecting whether the seat belt is buckled
  - Detecting other objects as they might be left behind in a ride share or taxi
  - Gesture detection/recognition for cabin controls
  - From Eyeris: https://www.eyeris.ai/automotive/



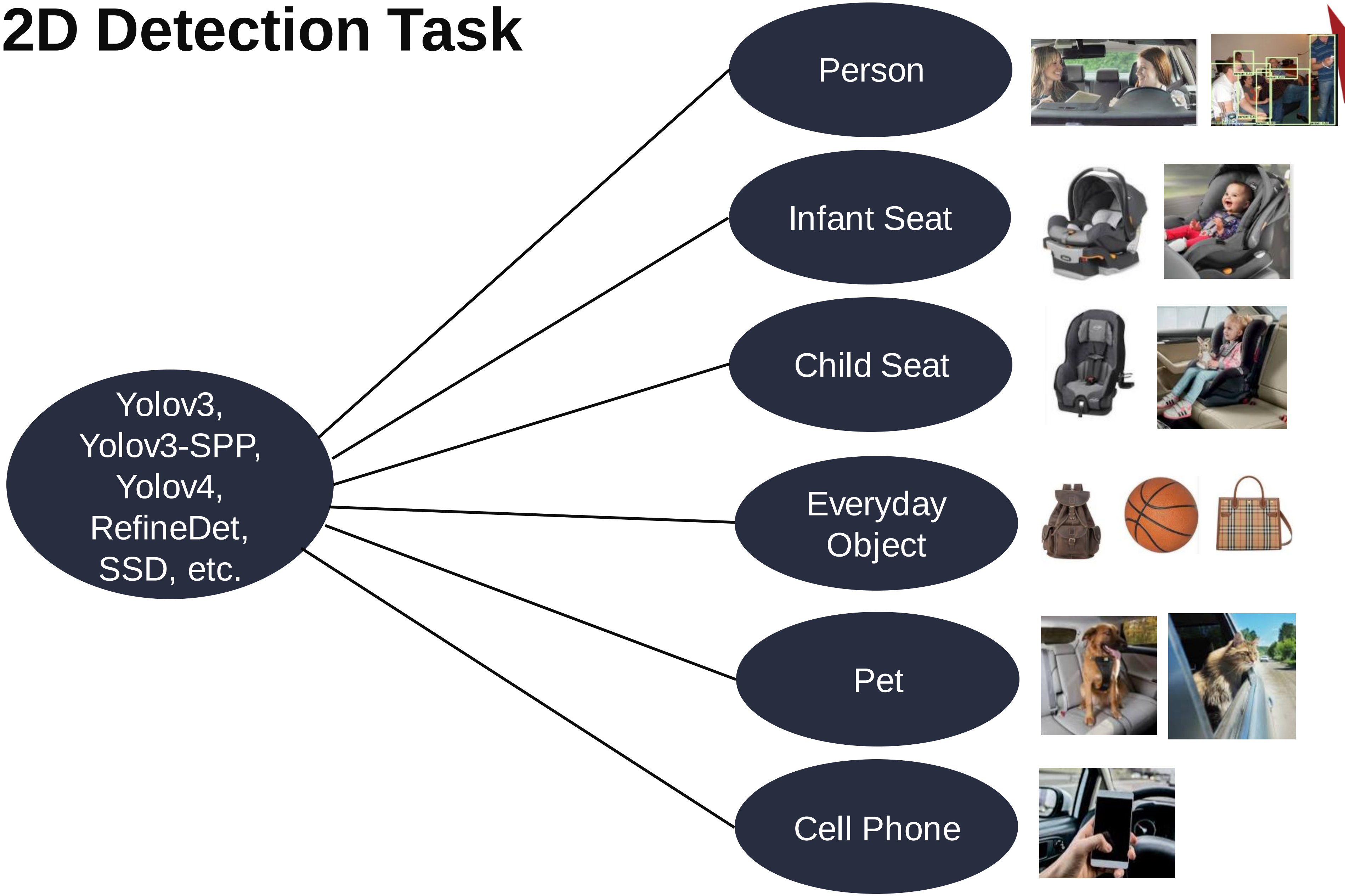
Statement of Intent

- What is intended:
  - To demonstrate how a Xilinx device and DPU can be used to implement ICMS relevant networks
  - To provide a starting point for customer specific and/or joint development
- What is not intended:
  - This is not a fully tested automotive solution
  - This is not intended as licensable IP
- This development was limited to open-source data with limited resources
  1) Open-source data (MS COCO 2017 and SVIRO datasets)
  2) Publicly available models
     - Models should ideally be trained using the production intent imager with real-world in cabin scene
  - 1x engineer working part time for a few weeks

Model Selection Options

- Many models already available from the Vitis AI Model Zoo
- Pose Detection Options:
  - SP Net model
    - Used recursively with a pedestrian detector for pose estimation
    - SPNet can run ~3.5ms, >270 FPS, even on B1600 DPU
  - OpenPose
    - Whole scene pose estimation
    - Can scale input dimensions to run faster, but by default only achieves 1.8 FPS on B1600 DPU
  - Eyeris Upper Body Pose Model
    - DPU deployment/testing in progress
    - Trained on real in-cabin data
- 2D Detection Options:
  - Densebox (face detection)
  - RefineDet (pedestrian)
  - Yolov2/Yolov3/Yolov3-SPP/Yolov4-leaky-relu (general object/pedestrian)
  - SSD (general object/pedestrian)
    - All different types of backbones are possible (VGG, Mobilenet, Resnet, Inception, etc.)

2D Detection Task



Yolov3, Yolov3-SPP, Yolov4, RefineDet, SSD, etc.

Person

Infant Seat

Child Seat

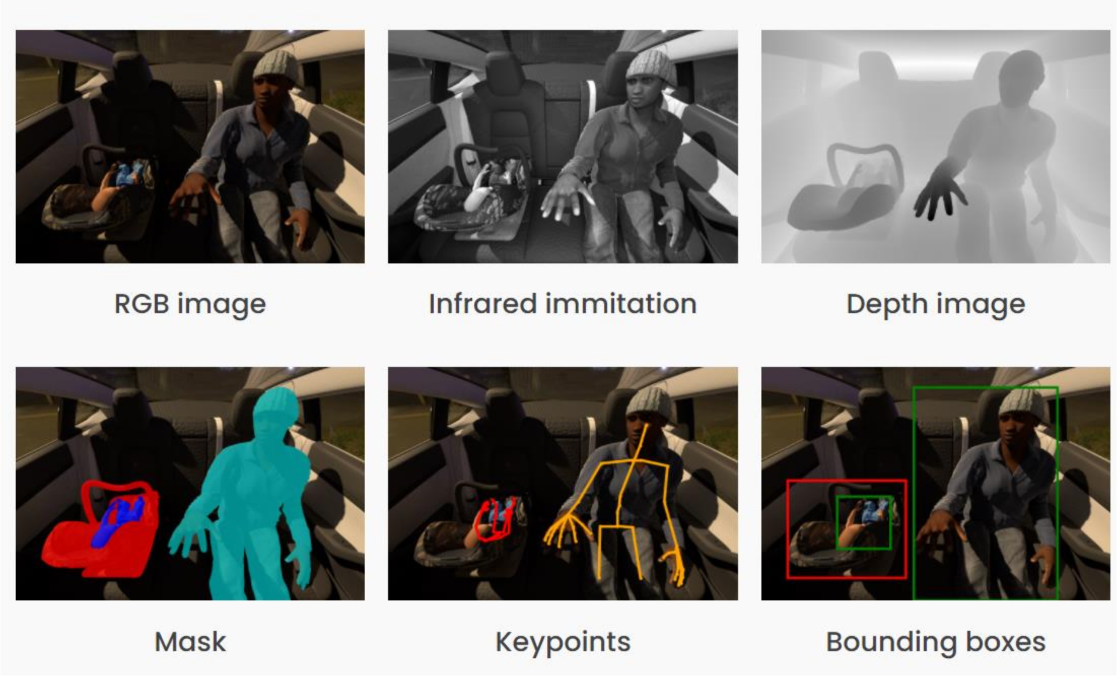
Everyday Object

Pet

Cell Phone

2D Detection Datasets

- SVIRO is a free synthetic dataset (Synthetic Vehicle Interior Rear Seat Occupancy Dataset)
  - https://arxiv.org/abs/2001.03483
  - https://sviro.kl.dfki.de/
  - ~25K Images with 4 classes
    - Person, Infant Seat, Child Seat, Everyday Object
- COCO 2017 is a free detection dataset with real world images, though not taken in the context of the interior vehicle cabin
  - https://cocodataset.org/#home
  - ~123K Images with 80 classes
  - Primary focus was on the pedestrian class (left this class as is)
  - Included the “cell phone” class
  - Combined the cat and dog class into a “pet” class
  - Combined the following classes into the Everyday Object class:
    - Backpack, umbrella, handbag, suitcase, frisbee, sports ball, baseball bat, baseball glove, tennis racket, bottle, cup, banana, apple sandwich, orange, hot dog, pizza, donut, laptop, book



RGB image

Infrared immitation

Depth image

Mask

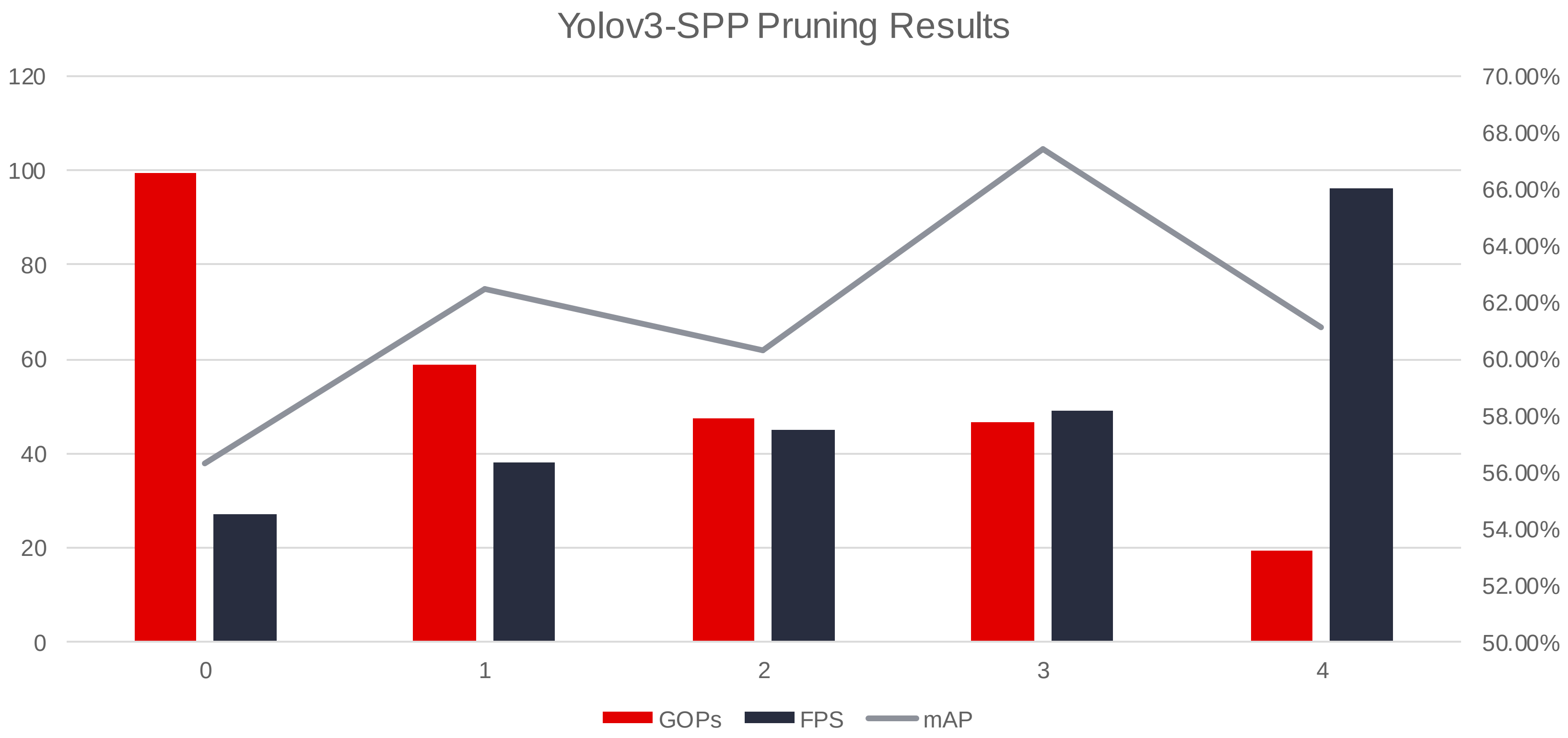
Keypoints

Bounding boxes

Initial Detection Model Development

- For all models
  - Started with Pretrained COCO weights and fine-tuned models
  - Trained in Darknet from AlexeyAB: https://github.com/AlexeyAB/darknet
  - Initial testing perform on a ZCU102 (ZU9 device)
    - Hardware and board image already setup to support Vitis-AI and DPU
    - Further experimentation done on Ultra96 Board
- Started with Yolov4 model
  - Some modifications required to run the model on the DPU (smaller SPP kernel sizes, replace MISH with leaky-ReLU)
  - Achieved high accuracy and model trained easily, however, the Vitis AI Optimizer doesn’t yet support Yolov4
- Researched Yolov4-tiny model
  - Trains well and produces good accuracy
  - There are some incompatibilities in the route layers for converting this model to Caffe/TF/Keras which is required for deployment
- Migrated to Yolov3-SPP
  - More difficult to train, but fully compatible with Vitis AI Optimizer Toolchain
  - Only modifications needed were to reduce # of classes and reduce SPP kernel sizes to 3, 5, & 7
  - Experimented with pruning to show deployment path to ZU3
  - Realized that pruning on SVIRO dataset produced poor “real world” performance -> augmented dataset with COCO images

Yolov3-SPP Pruning



*FPS numbers are for ZCU102 board (ZU9 device) with 3x B4096 DPUs
*Trend is showing that further pruning should be possible and enable deployment on ZU3 device

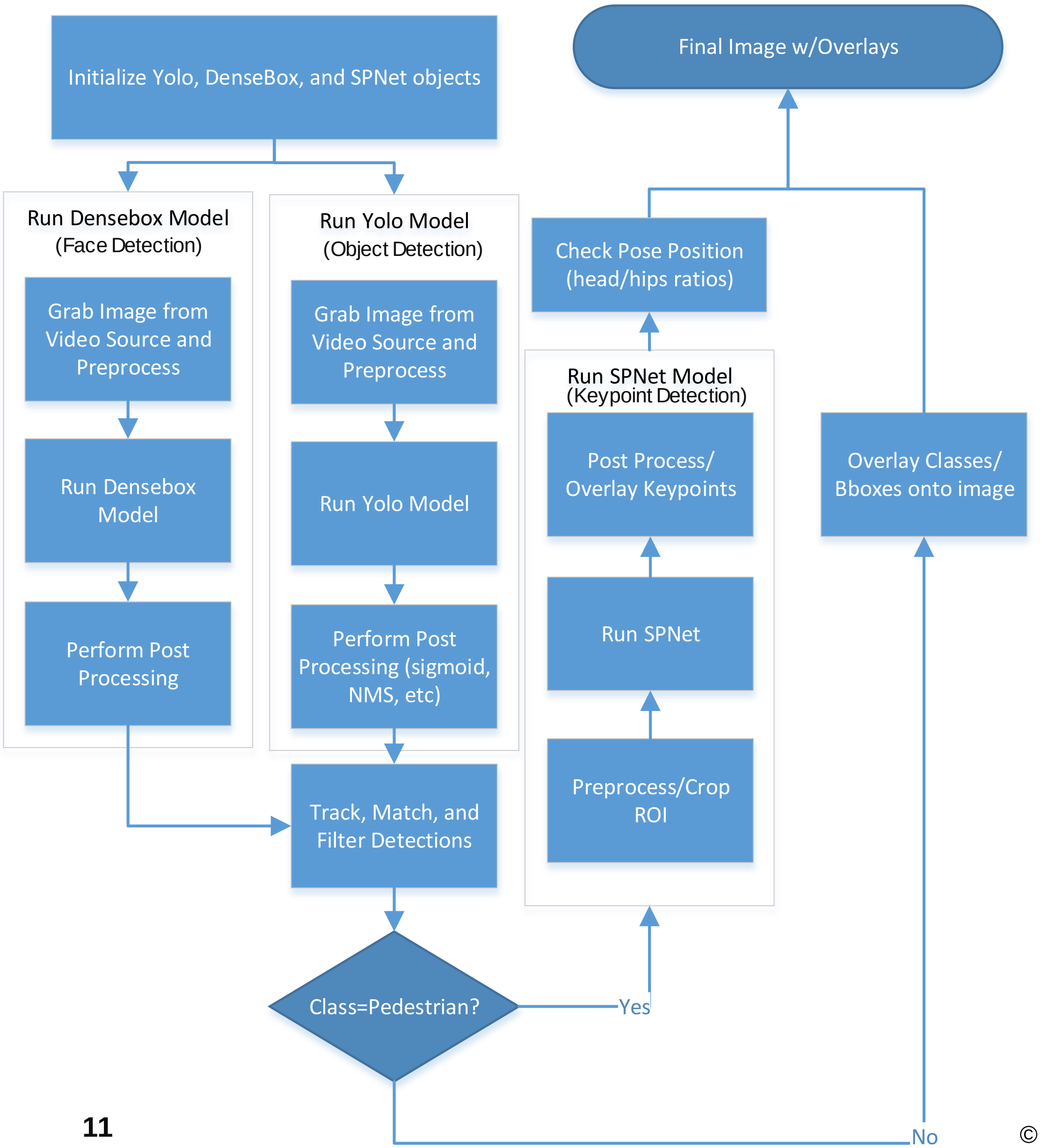
Deployment Software

- Started with the Vitis-AI-Library software for SSD+Pose Detect
  - https://github.com/Xilinx/Vitis-AI/tree/master/Vitis-AI-Library/overview/samples/posedetect
  - Supports video with multithreading
  - Uses the VART APIs
  - Supports key-point detection with SP-Net model

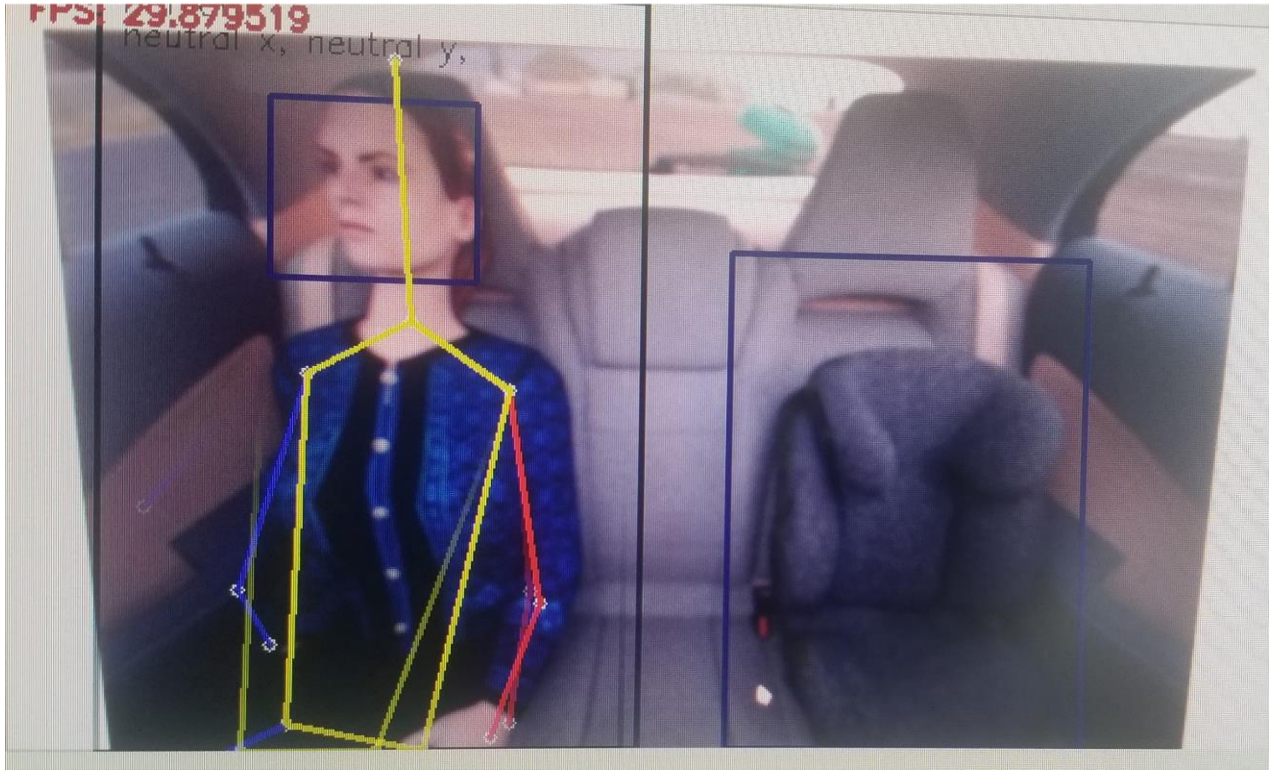
- Modified the software to use Yolov3-SPP instead of SSD for the detection stage
  - Modified the number of classes from 1 to 6
  - Modified to display bounding boxes with labels as well as keypoints
  - Added Face Detection using the Densebox 320x320 model

- Added tracking software to aid with continuity of detections
  - Detections are tracked from frame to frame
  - Detections are matched to existing known detections based on class, size, and position
  - Includes programmable thresholds (e.g. number of frames):
    - If a detection exceeds the maximum threshold without a match it is deleted
    - A detection is not displayed until it meets a minimum threshold

Software Flow



Final Image w/Overlays
Initialize Yolo, DenseBox, and SPNet objects
Run Densebox Model (Face Detection)
Grab Image from Video Source and Preprocess
Run Densebox Model
Perform Post Processing
Run Yolo Model (Object Detection)
Grab Image from Video Source and Preprocess
Run Yolo Model
Perform Post Processing (sigmoid, NMS, etc)
Track, Match, and Filter Detections
Class=Pedestrian?
Yes
No
Check Pose Position (head/hips ratios)
Run SPNet Model (Keypoint Detection)
Post Process/ Overlay Keypoints
Run SPNet
Preprocess/Crop ROI
Overlay Classes/ Bboxes onto image

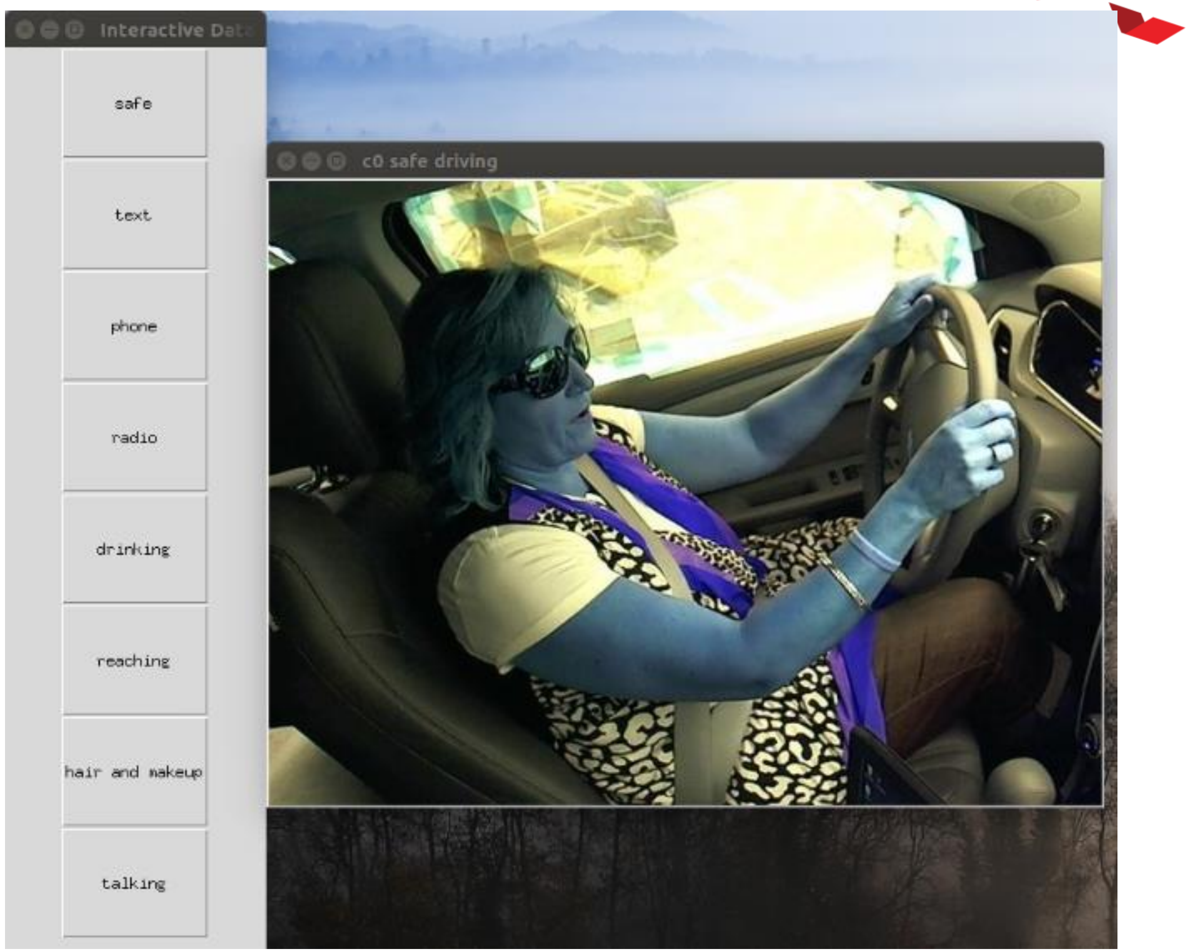


Distracted Driver Model Development

- Goal is to evaluate the feasibility of deploying distracted driver detection model
  - Potentially use a region of interest from the ICMS camera or
  - Add a second camera pointed at the driver and run this model alongside the other tasks
- In 2016, State Farm produced a distracted driver dataset with 10 classes:
  - safe driving, texting/talking cell phone with right/left hand (4 classes), doing hair/makeup, operating the radio, reaching behind, and talking with passenger
  - Consists of ~25000 labelled images and ~75000 test images
- Started by processing the images/labels into tfrecords
- Trained DenseNetX from the Vitis AI Tutorials on this dataset
  - Working through some deployment issues with global average pooling layer
- Also attempted training Keras ResNet50 on the dataset
  - Model was initially highly overfitted because of limited amount of labelled data

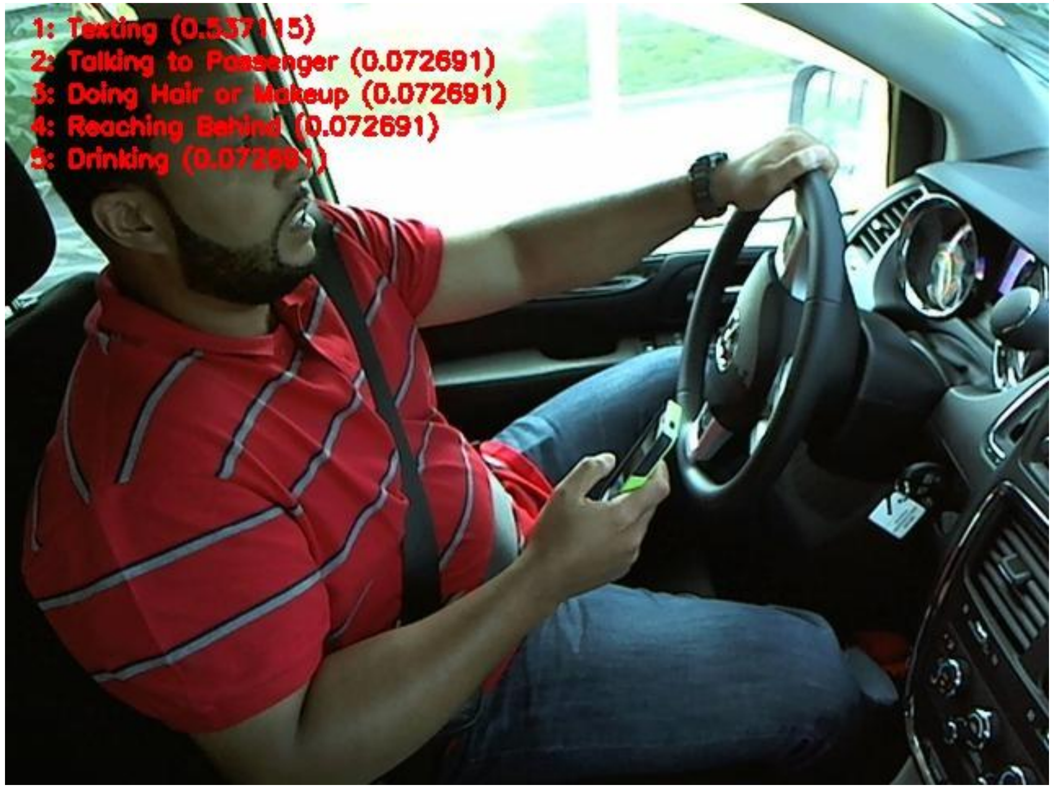
Distracted Driver Dataset Improvements

- Labeled an additional 25,000 images from the test dataset
  - Used the trained DenseNetX model to classify the images into categories
  - Wrote a python GUI utility to assist with correcting and wrong labels
  - Combing the left/right classes for cell phone and texting together
- Added horizontal flips which results in a total of ~100K samples



Distracted Driver Model Development

- Trained Keras ResNet50 on new dataset using TensorFlow 2.3 in Vitis AI 1.3
- Floating point model validation accuracy reached 98.03% for Top1
  - 9000 images in validation set
- Quantized model accuracy:
  - Top1 : %97.88
  - Top5 : %99.94
- Model was deployed on the ZCU102 and can still operate in conjunction with ICMS models at >30 FPS



```
1: Texting (0.537115)
2: Talking to Passenger (0.072691)
3: Doing Hair or Makeup (0.072691)
4: Reaching Behind (0.072691)
5: Drinking (0.072691)
```

Other/Future Work

- Already Ported to Ultra96V2 platform using MIPI input sensor
  - Input scaling and CSC performed on streaming input video using VPSS and Vitis Vision Libraries
  - Achieved ~11-12FPS (B2304 DPU@250/500MHz) and possible to achieve higher performance with further pruning

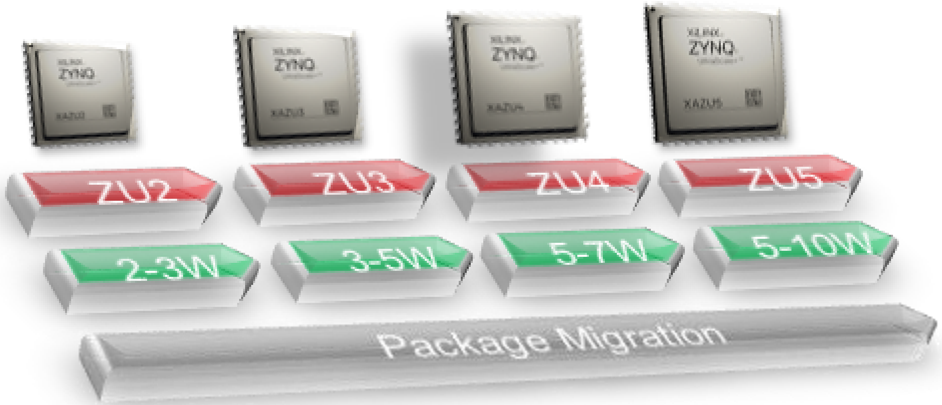
- Working on deploying partner (Eyeris) model for upper body pose detection (trained on better dataset for cabin monitoring)

- Planning to port to RGB-IR sensor and integrate with either ZCU102 or ZCU104 board using GMSL or FPD-Link III FMC
  - Investigating OmniVision and On-Semi sensors
    - E.g. OV2778, OV2312, AR0239, etc.
  - Potential for model retraining with IR data

- Explore Drowsy Driver Algorithms
- Explore Gesture Recognition
- Planning to implement in Ford Escape demo vehicle

Summary

- Vitis AI provides valuable components for rapid demo development
  - Pre-trained models from the Vitis AI Model Zoo
  - Vitis AI Library example code for rapid model deployment
  - Pre-built ZCU102 and ZCU104 board images for hardware testing
    - no FPGA design or custom Linux build needed
- Vitis AI Optimizer enabled >3x acceleration of the detection model
  - Further optimization possible
- Xilinx Zynq UltraScale+ enables a scalable platform for ICMS/DMS custom chip down solutions



Thank You