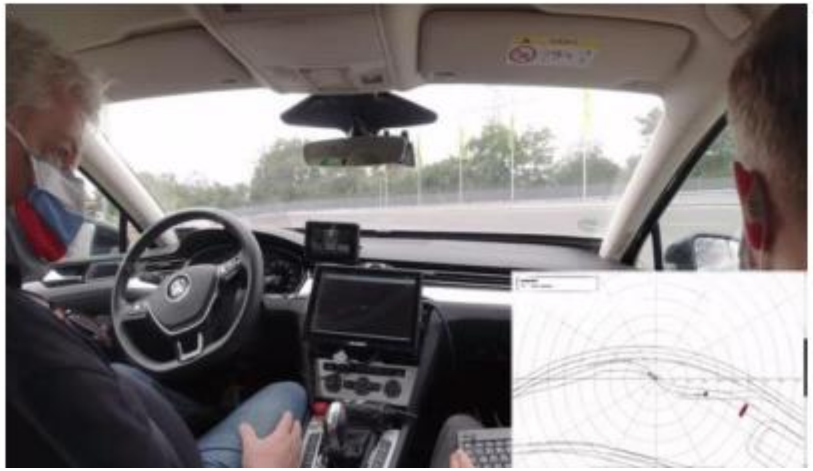
Continental
The Future in Motion

Rapid Prototyping of a Computer Vision Stack for AD using MATLAB/Simulink

Dr. Roxana Florescu & Alexandru Puscasu & Dr. Stephan Kirstein

Driverless Driving Activities



Truck:

- Highway
- Hub to hub

RoboTaxi:

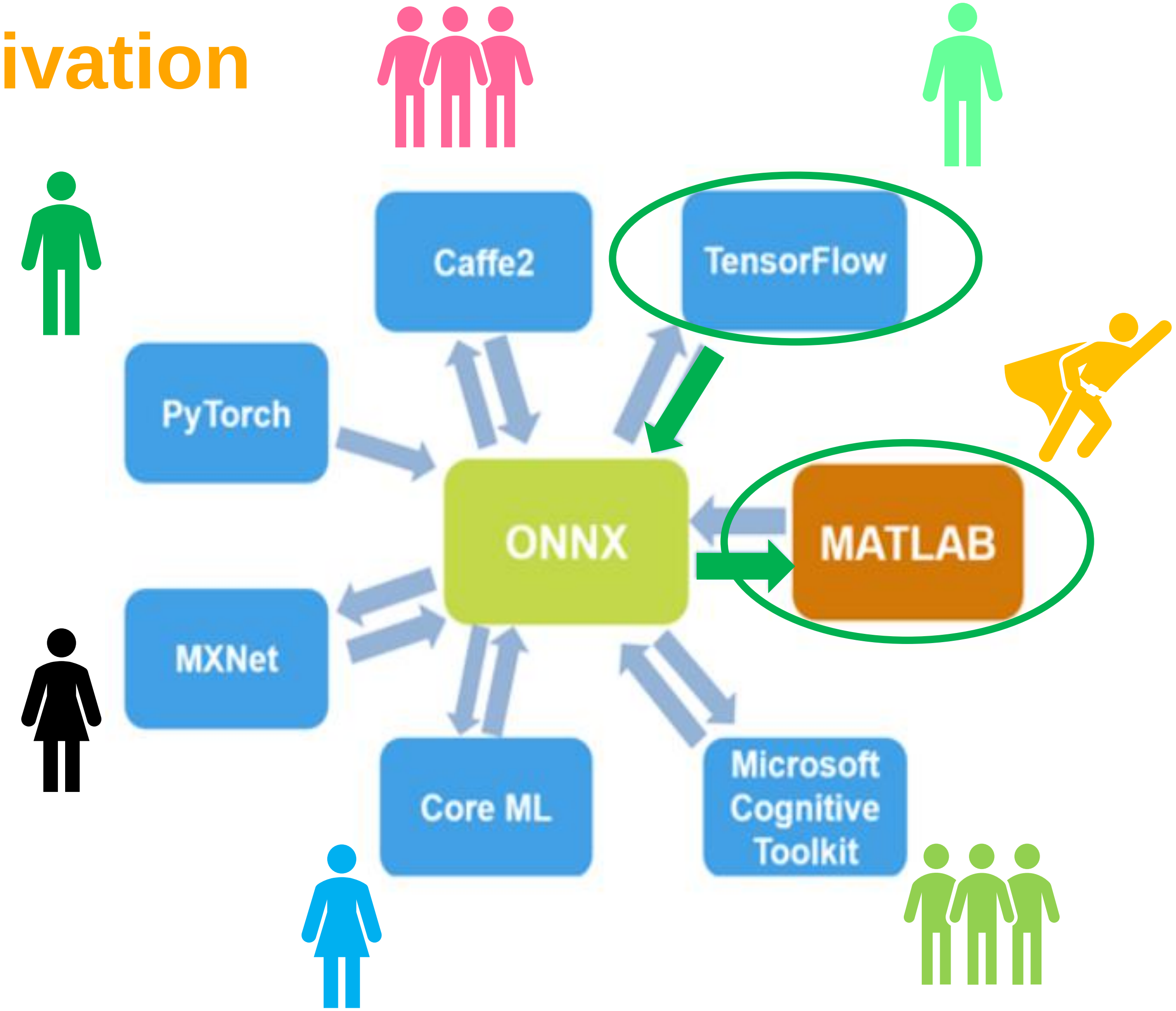
- Inner city

Valet Parking:

- Parking garage

Sensor Technologies: **Camera**, Radar, Lidar, Ultrasonic, IMU

Motivation



- Different AI teams use different toolkits:
  - Many different libraries needed
  - Potentially many version conflicts
  - Often docker containers used
- Open Neural Network Exchange
  - Established standard
  - Opset defines supported layers
- Target of MATLAB prototype:
  - Deployment of networks into vehicle:
    - Image acquisition
    - Preprocessing
    - Network inference on GPU
    - Postprocessing
  - Low number of dependencies
  - C/C++ Code generation

Tool Chain

Image Stream

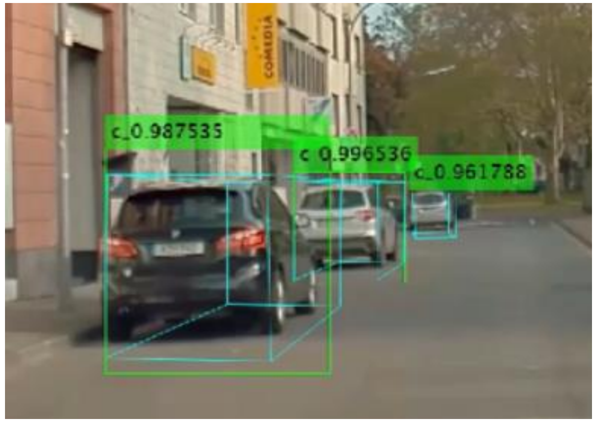
- Video files
- Live camera data
- Measurements

MATLAB / Simulink

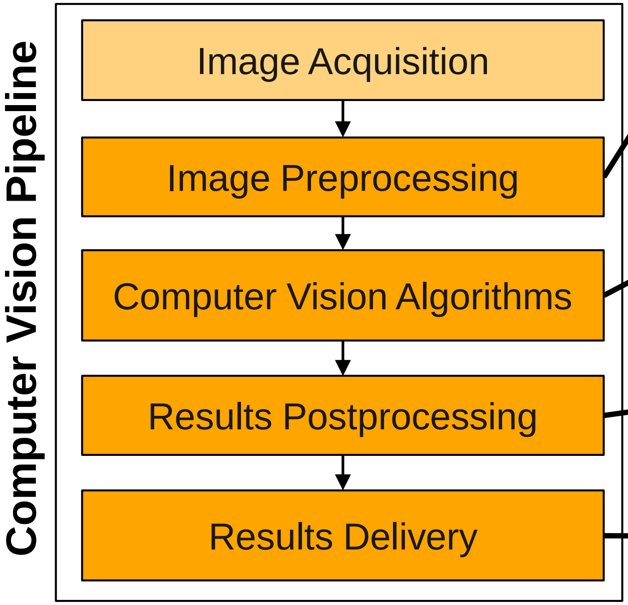
Code generation

CPU/GPU/FPGA

Processed Image



Computer Vision Pipeline Implemented in MATLAB

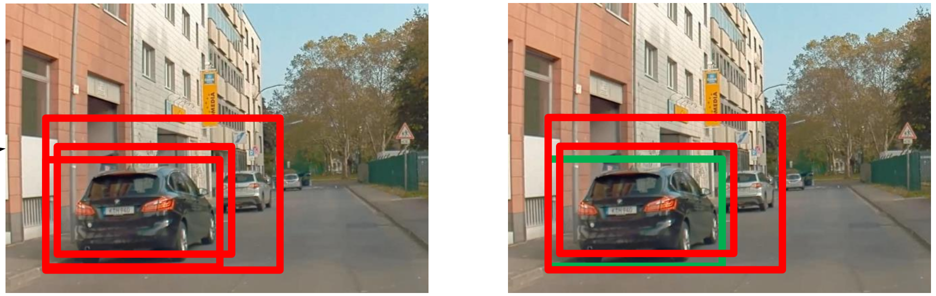


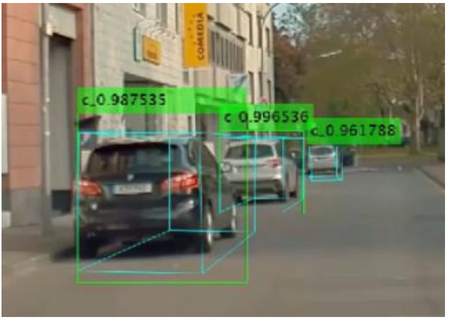
Rescaling

Cropping

Image encoding

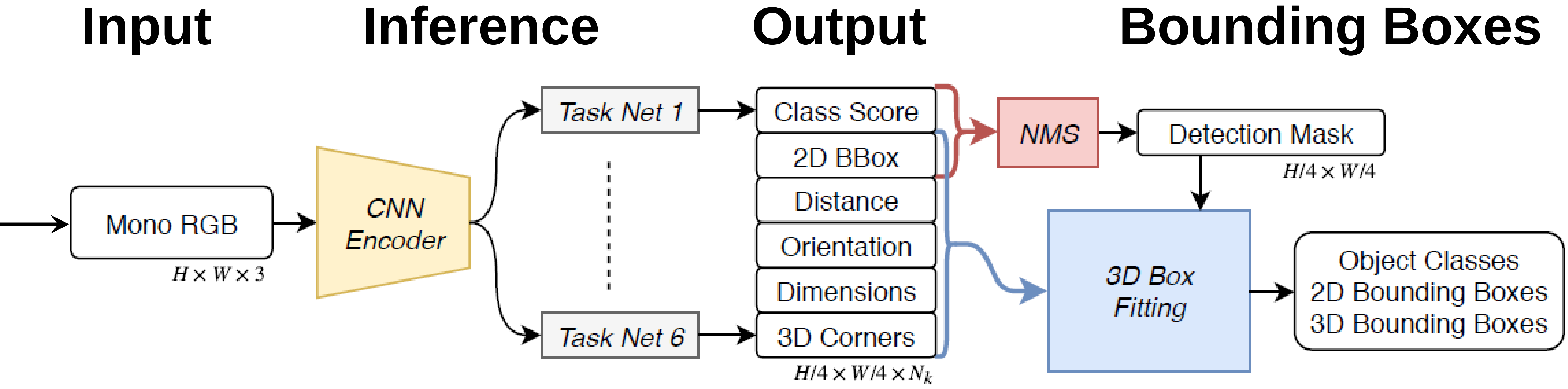
Different custom CNNs like SS3D


Non-Maximum suppression, class/confidence extraction

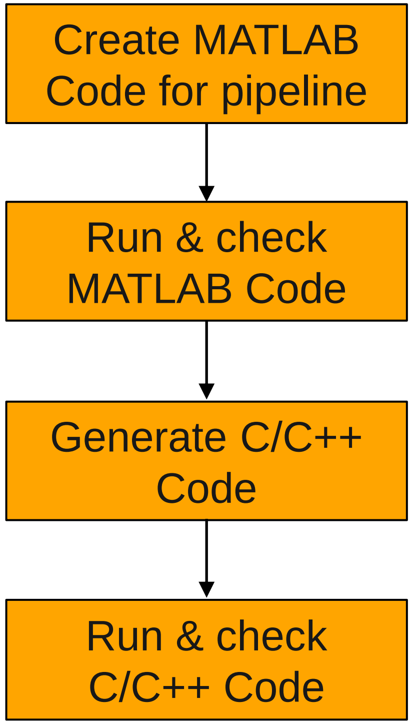


Displaying, video files, export (eCAL - middleware)

SS3D Network



Deploying on Different Platforms



- One MATLAB pipeline different deployments:
  - CPU vs. GPU
    - Differences in results
    - Differences in runtime
    - MATLAB vs. C/C++ code generation
  - Embedded Platform (Nvidia AGX)
  - Offline/Live data
- Target:
  - C/C++ with GPU support

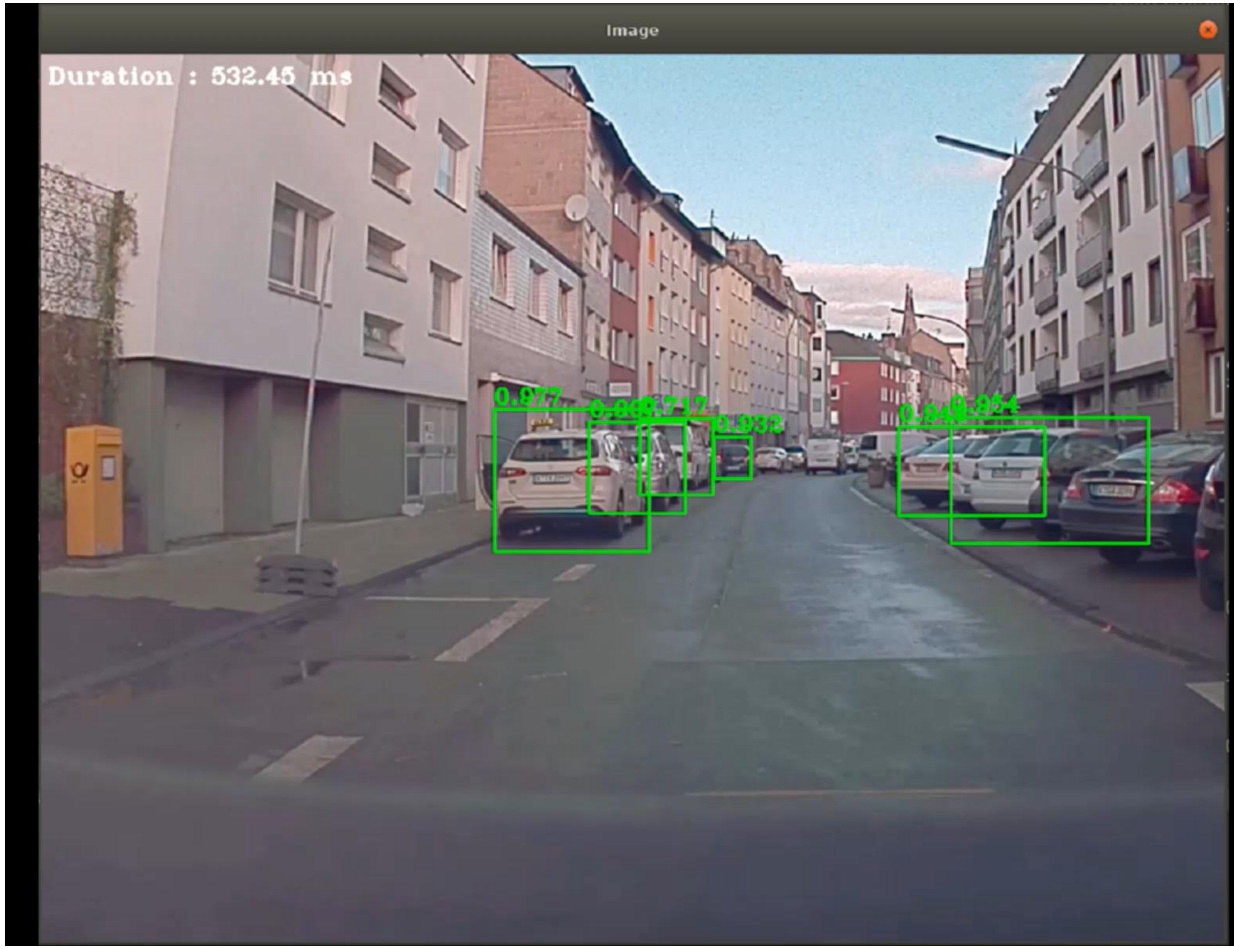
Deploying on Different Platforms

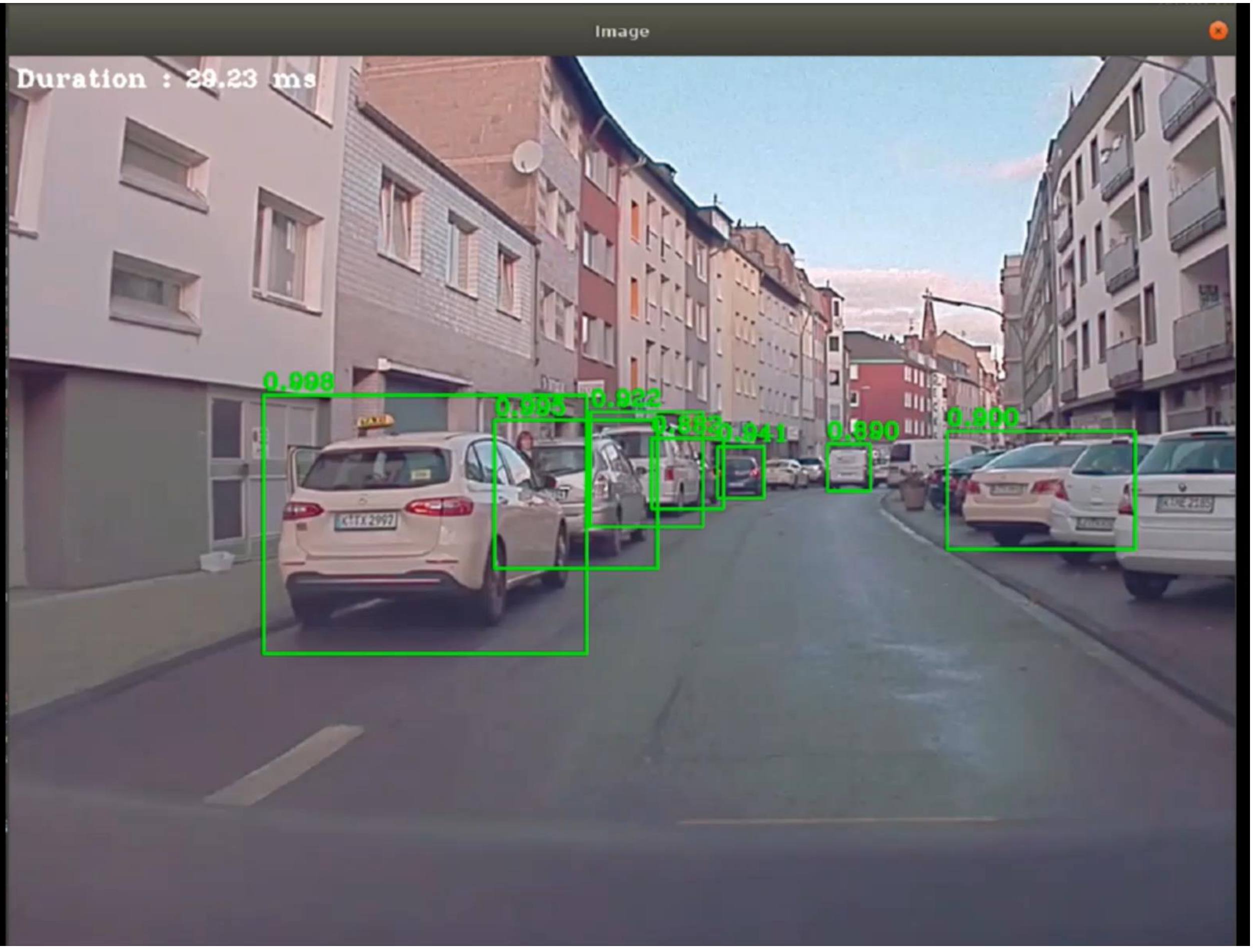
I. **CPU**
   - Input Image Stream:
     - Offline image stream (video/eCAL)
     - Live Image Stream
   - Hardware Details: **Intel Xeon Gold 6132 CPU @2.6GHz 2 x 14 Cores**

II. **GPU**
   - Input Image Stream:
     - Offline image stream (video/eCAL)
     - Live Image Stream
   - Hardware Details: **NVIDIA RTX 2080 Ti 1350 MHz**

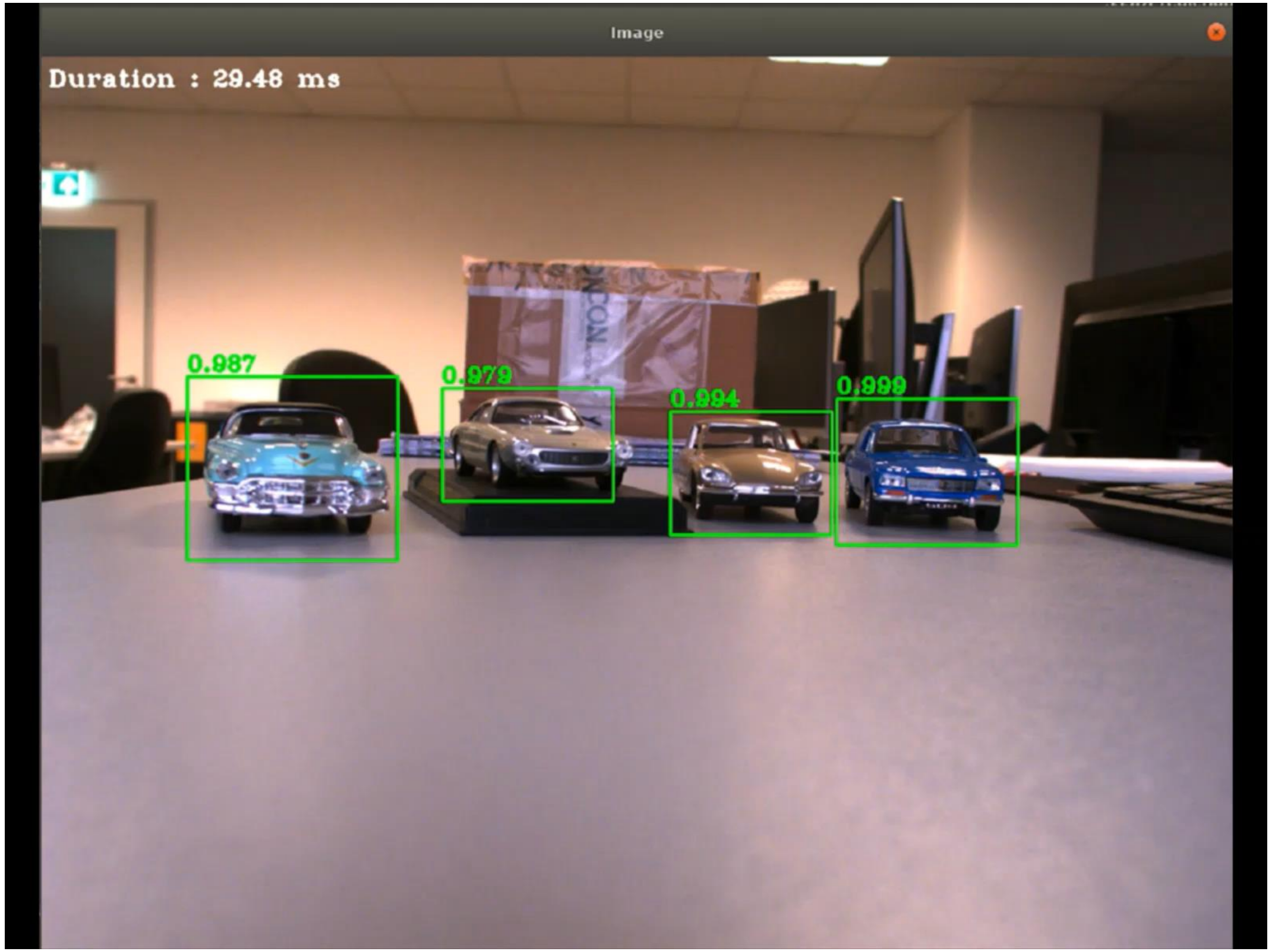
III. **Dedicated HW**
   - Input Image Stream:
     - Offline image stream (video/eCAL)
   - Hardware Details: **NVIDIA AGX Board**

Results CPU vs. GPU





Results on Live Camera Stream (GPU)

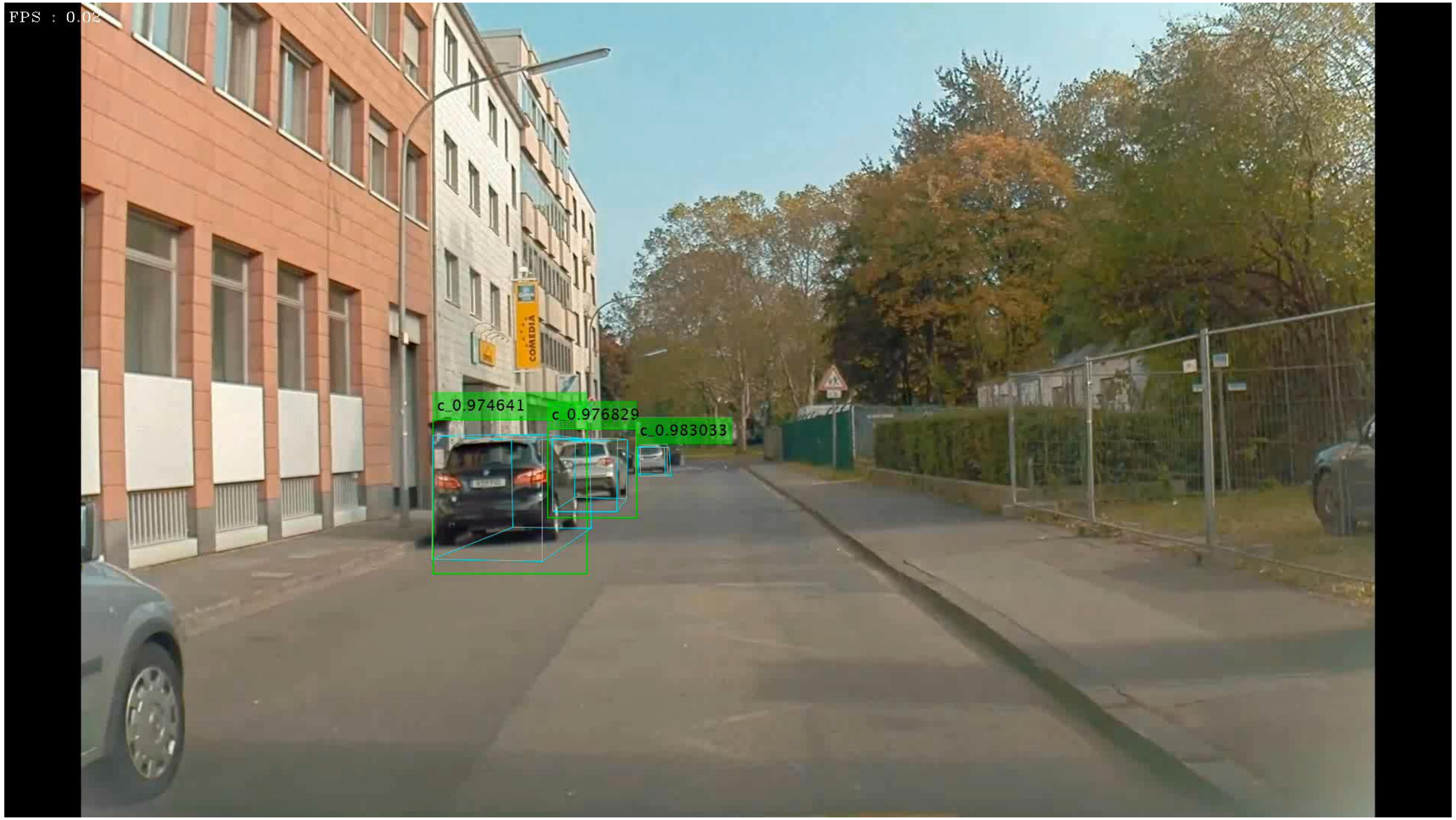


```
Duration : 29.48 ms
```

Inference Time Comparison

Hardware	MATLAB Code		C++ Code	
#Run	CPU [ms]	GPU [ms]	CPU [ms]	GPU [ms]
1	299.4	58.4	290.056	29.170
2	290.5	65.8	289.359	29.411
3	275.5	59.1	288.077	29.379
4	277.5	58.7	273.604	29.726
5	308.1	59.6	301.058	29.468
Overall Average Time	290.2	60.32	288.430	29.43

Results on Nvidia AGX



- Nvidia AGX
  - ARM64 cores
  - Restricted GPU
- 6-7 frames/s possible with cuDNN

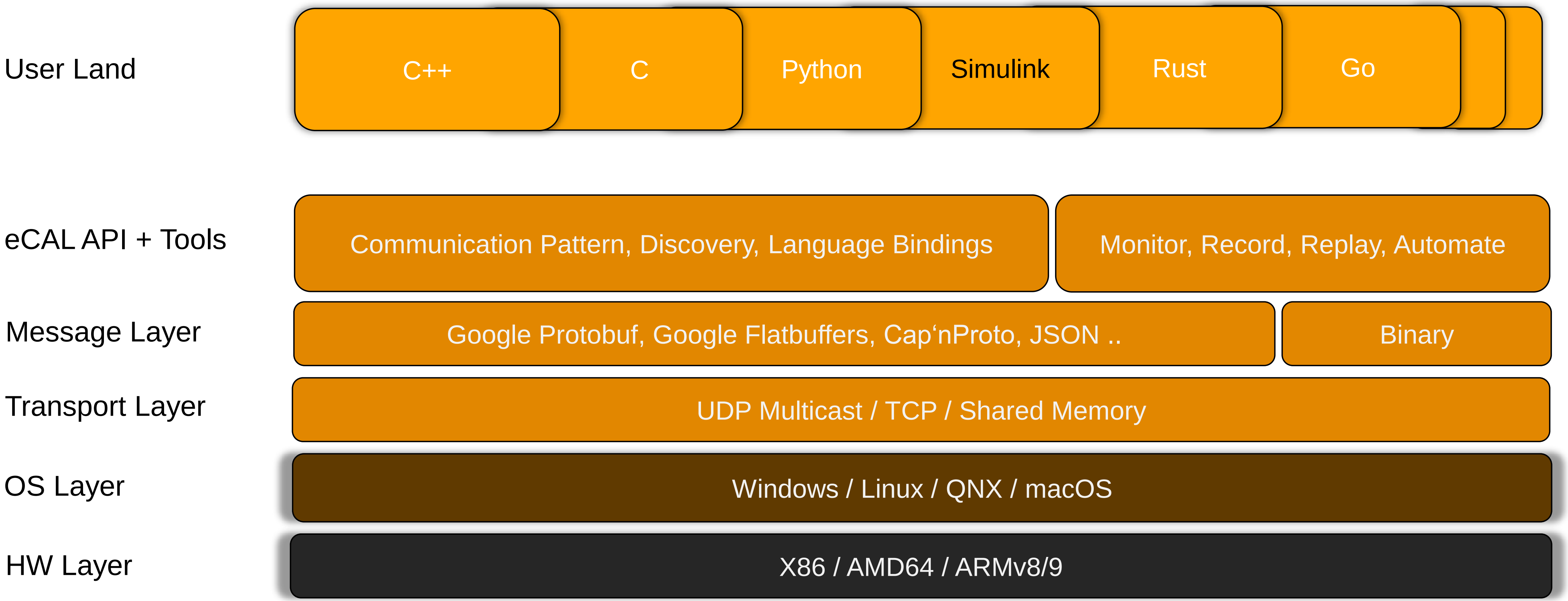
Summary

- ONNX + MATLAB/Simulink enables deployment of several CNNs to target hardware
  - Fastes inference time with C/C++ Code generation
  - Comfortable switch between different targets (CPU/GPU/Embedded)
  - MATLAB support helped to overcome challenges
    - Suggestion of alternative functions
    - Missing CNN operations added

- Suggestions:
  - MATLAB or specific ONNX Opsets aren’t supporting all CNN operations (Link)
  - Code generation: some functions still missing (for pre-/postprocessing)
  - GMSL camera needed for Nvidia AGX → only offline tests
  - GigE camera grabbing with large time variation → another C/C++ API used

Thank you for the attention!

eCAL Architecture overview



User Land: C++, C, Python, Simulink, Rust, Go

eCAL API + Tools: Communication Pattern, Discovery, Language Bindings | Monitor, Record, Replay, Automate

Message Layer: Google Protobuf, Google Flatbuffers, Cap‘nProto, JSON .. | Binary

Transport Layer: UDP Multicast / TCP / Shared Memory

OS Layer: Windows / Linux / QNX / macOS

HW Layer: X86 / AMD64 / ARMv8/9