



IETF-123 Hackathon



Interface to In-Network Computing Functions (I2ICF) Project

July 20, 2025, Madrid

Champion: Jaehoon (Paul) Jeong

Members: Yoseop Ahn, Mose Gu, Jiwon Seo, and Jisuk Chae

Department of Computer Science and Engineering at SKKU

Email: {pauljeong, ahnjs124, rna0415, sjw6136, sue030124}@skku.edu

IETF-123 Interface to In-Network Computing Functions (I2ICF)

Champion: Jaehoon Paul Jeong (SKKU)



IETF-123 Interface to In-Network Computing Functions (I2ICF) Hackathon

Professors:

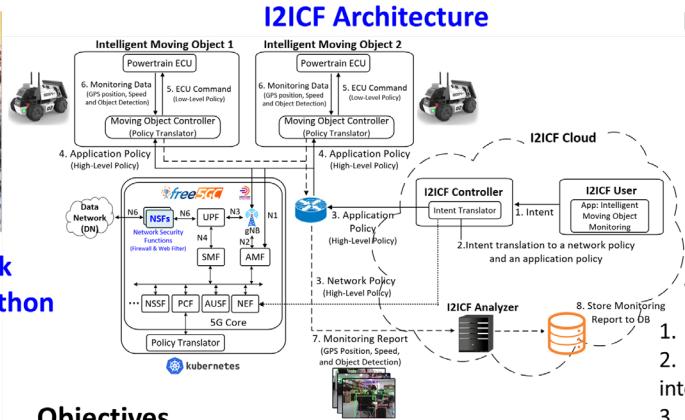
- Jaehoon Paul Jeong (SKKU)
- Younghan Kim (SSU)
- Yong-Geun Hong (DJU)
- Joo-Sang Youn (DEU)
- Yiwen Chris Shen (AJU)

Researchers:

- Jung-Soo Park (ETRI)
- Soohong Daniel Park (Samsung Research)

Students:

- Mose Gu (SKKU)
- Yoseop Ahn (SKKU)
- Jiwon Suh (SKKU)
- Jisuk Chae(SKKU)



Objectives

- To demonstrate Interface to In-Network Computing Functions (I2ICF).
- To develop an integrated framework for intent-driven orchestration of networking, security, camera-based object recognition, and application services in mobile systems (e.g., Robot Car) operating in distributed computing environments.

Future Work

- The intent translation module will be designed and implemented using advanced techniques such as Knowledge Graph Embedding and Reasoning Model.
- A real-time analytics system powered by Machine Learning and Deep Learning will be developed to assess the operational status of Service Functions (SFs) in mobile platforms, with the aim of enhancing safety and security.

I2ICF Developing Environment

- OS: Ubuntu 20.04
- Kubernetes: Micrkok8s v1.32.2
- Object Detection: YOLO v8
- ROS Version: Humble
- GitHub Repository:
<https://github.com/jaehoonaupjeong/I2ICF/tree/main/IETF-123>

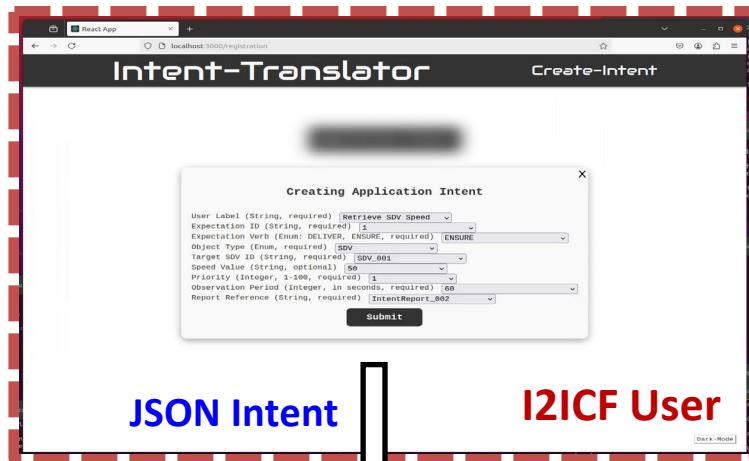
Workflow of the I2ICF Testbed

1. I2ICF User sends an intent to the I2ICF Controller.
2. The I2ICF Controller's Intent Translator converts the intent into Network and Application Policies.
3. The translated Network Policy is forwarded to the wireless network components (e.g., 5G NEF).
4. The High-Level Application Policy is sent to each Moving Object Controller.
5. Each Moving Object Controller translates the received Application Policy and applies it to the Powertrain ECU, adjusting the operational parameters.
6. The Moving Objects monitor operational data (e.g., GPS position and speed) and detect objects or obstacles in front of the Robot Car with YOLOv8.
7. The Moving Object Controllers set these data into a Monitoring Report (e.g., JSON File) and forward it to the I2ICF Analyzer.
8. The I2ICF Analyzer processes the Monitoring Report to assess the performance of the policies and stores the results in the database (DB) for further analysis.

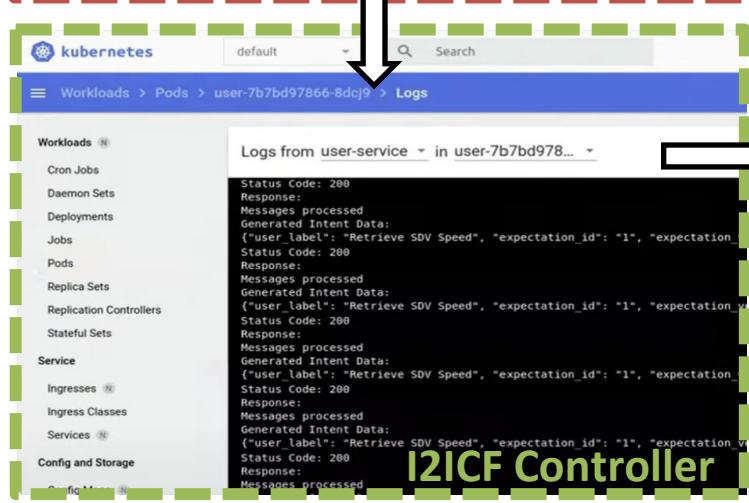
What differences from previous work?

- Previous Goal focused on Intent Translation and Policy Provisioning in Interfaces to In-Network Computing Functions (I2ICF) and its Framework.
 - Creation of a YAML Intent based on 3GPP 28.312 and its Deliverance to Moving Objects.
- Present Goal is to implement a User Equipment (UE)'s data collecting ability and monitoring interface to collect data from the UE which is directed by a user intent.
 - We collect data from a UE (e.g., Robot Car) and deliver it to an Analyzer Component for the framework's closed loop purpose.

Previous Demonstration

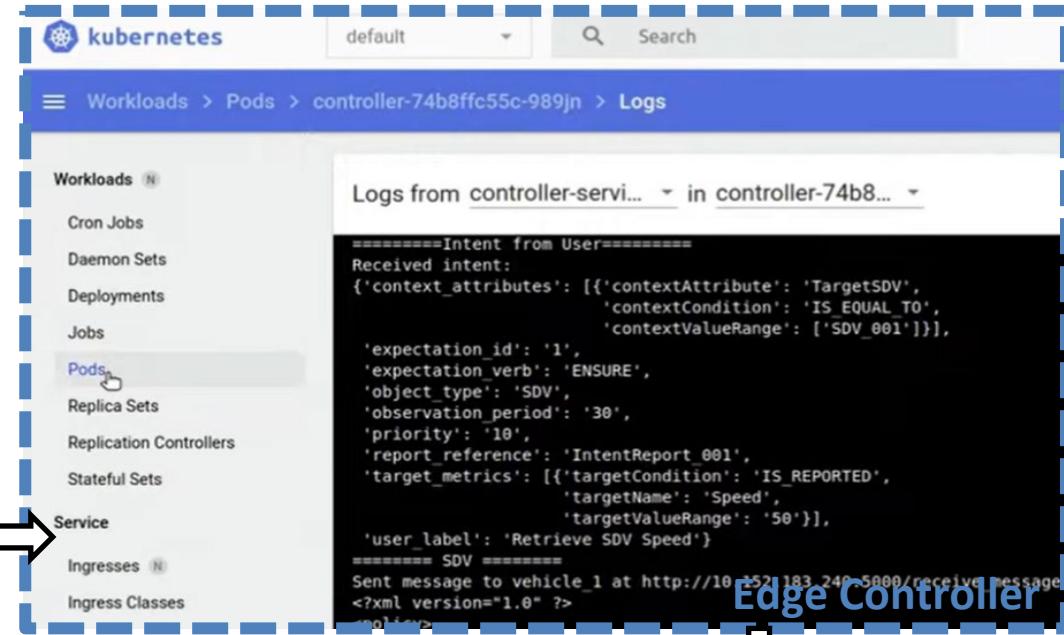


JSON Intent



I2ICF User

I2ICF Controller



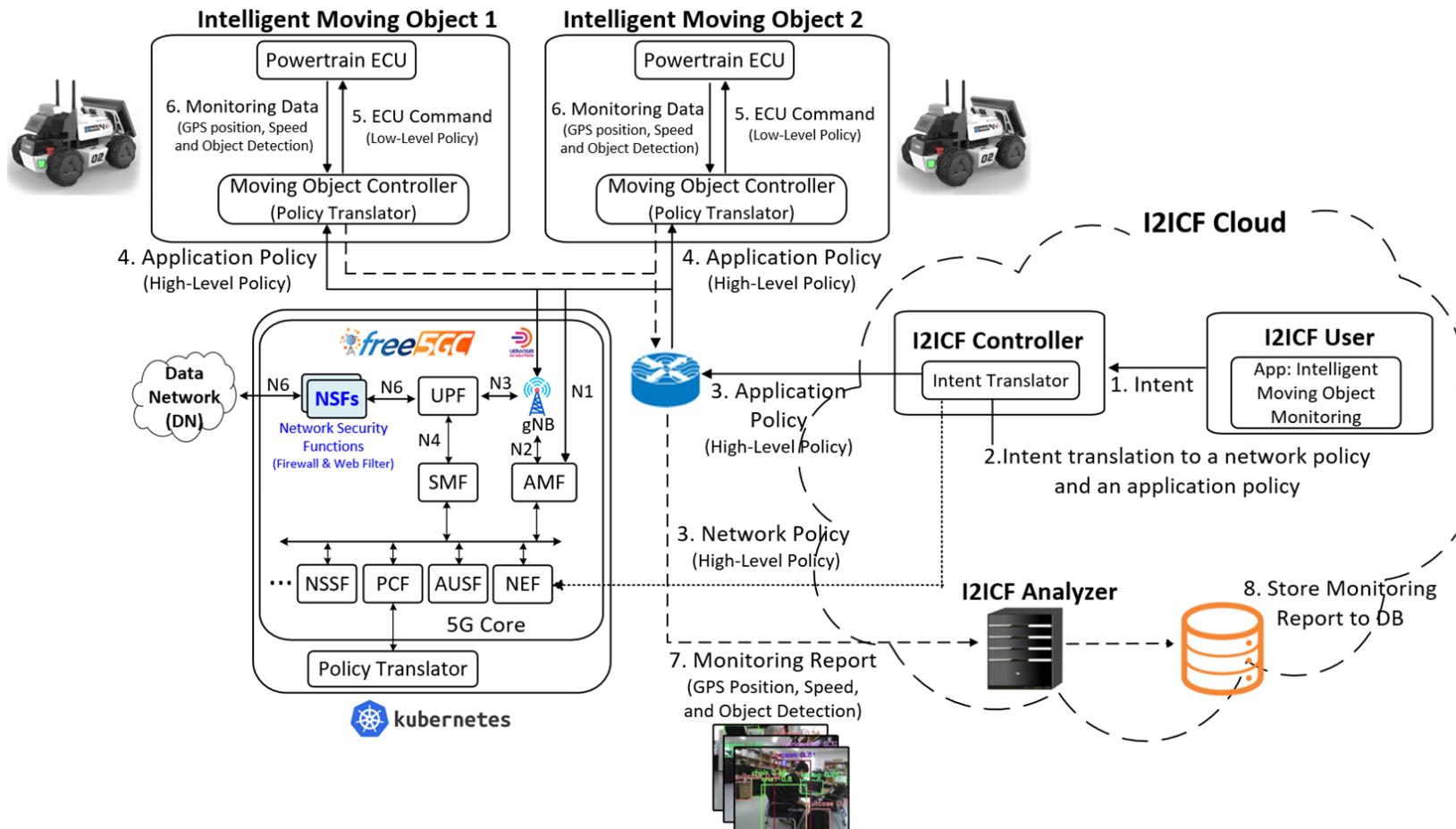
Logs from controller-servi... in controller-74b8...

YAML Intent

Edge Controller

Moving Object
Service Functions

Interface to In-Network Computing Functions (I2ICF) for Moving Objects



Goal of Hackathon Project

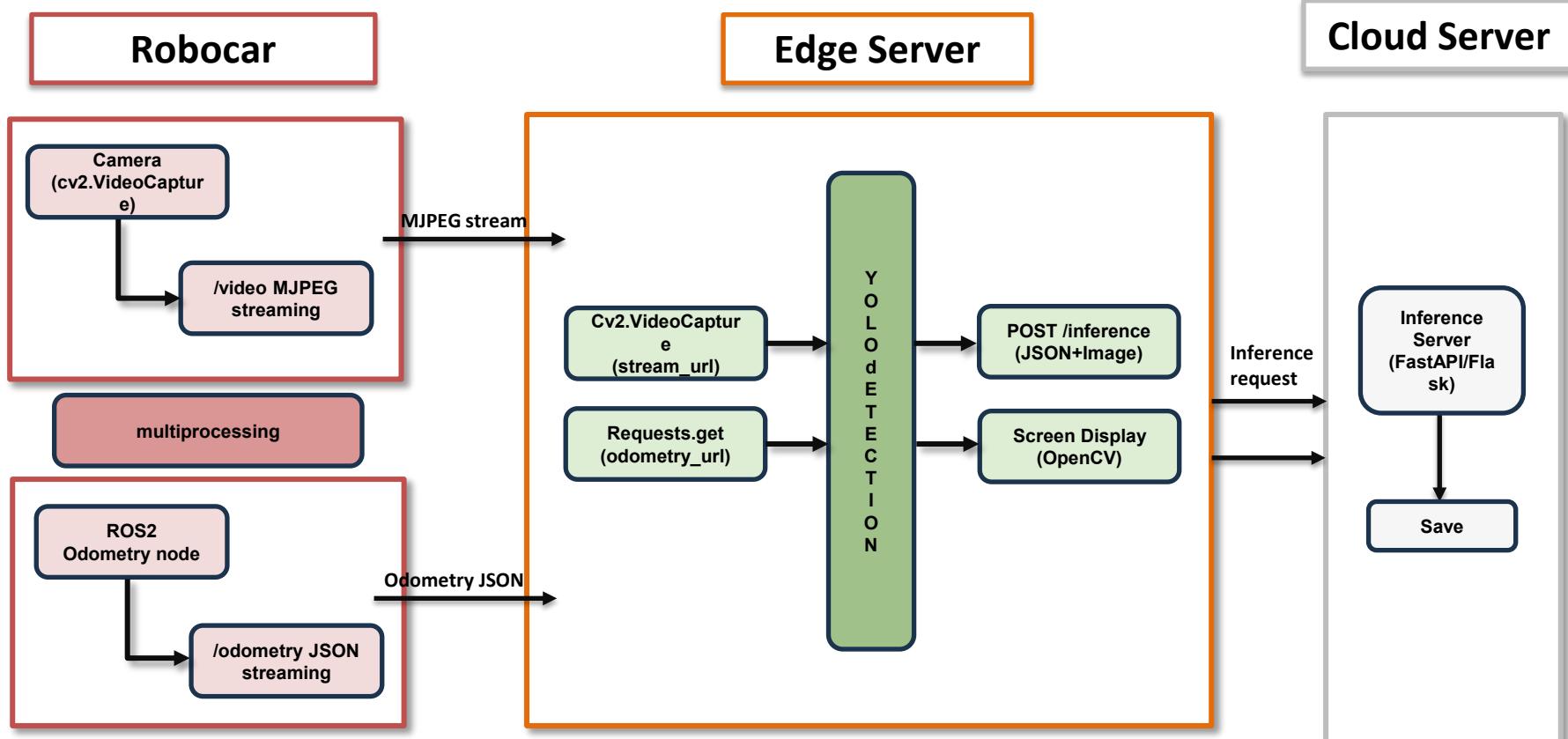
- The goal is to show the feasibility of a framework and interfaces for configuring and monitoring Moving Objects (e.g., robot cars and drones).
 - **I2ICF (Interface to In-Network Computing Functions) Framework**
 - An intelligent framework for moving-objects with an intent translator and closed-loop control.
 - **Three categories of actions** of moving objects: Sensing, Movement, and Actuation.
- Internet Drafts for the I2ICF Project
 - <https://datatracker.ietf.org/doc/draft-jeong-opsawg-i2icf-problem-statement/>
 - <https://datatracker.ietf.org/doc/draft-jeong-opsawg-i2icf-framework/>
 - <https://datatracker.ietf.org/doc/draft-ywj-opsawg-i2icf-data-center-networking/>
 - <https://datatracker.ietf.org/doc/draft-ahn-opsawg-i2icf-cits/>

3 Kinds of Actions of Moving Objects

- Sensing
 - Camera Sensor
 - Odometry Sensor
- Movement
 - Four-wheel movement
 - Source, intermediate positions, and destination movement
- Actuation
 - Sensing data delivery
 - Fire alarm



Goal Flow Diagram



Demonstration

```
agilex@agikex-NUC12WSKi7:~$ ros2 launch limo_base limo_base.launch.py
[INFO] [launch]: All log files can be found below /home/agilex/.ros/log/2025-07-14-14-29-35-182625-agikex-NUC12WSKi7-3201
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [limo_base-1]: process started with pid [3202]
[limo_base-1] Loading parameters:
[limo_base-1] - port name: ttylimo
[limo_base-1] - odom frame name: odom
[limo_base-1] - base frame name: base_link
[limo_base-1] - pub odom tf: 0
[limo_base-1] -use_mcnamu: 0
[limo_base-1] [INFO] [1752474575.296639524] [limo_base_node]: connect the serial port:'/dev/ttylimo'
[limo_base-1] [INFO] [1752474575.299580805] [limo_base_node]: enableCommandedMode :
[limo_base-1] [INFO] [1752474575.299597332] [limo_base_node]: Open the serial port:'/dev/ttylimo'
```

1.LIMO ROS2 Launch

```
agilex@agikex-NUC12WSKi7:~/SDV_Robocar/SDV_Robocar$ python3 robocar_server2.py
[ WARN:0] global ./modules/videoio/src/cap_gstreamer.cpp (1100) open OpenCV | GStreamer warning: Cannot query video position: status=0, value=-1, duration=-1
* Serving Flask app 'robocar_server2'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:8000
* Running on http://192.168.50.82:8000
Press CTRL+C to quit
```

2. Run robocar_server2.py

Demonstration

```
(venv) wego@b4u:~/신학프로젝트/SDV_Robocar$ ``C
(venv) wego@b4u:~/신학프로젝트/SDV_Robocar$ python3 k8s_server.py
* Serving Flask app 'k8s_server'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:8080
* Running on http://172.30.88.170:8080
Press CTRL+C to quit
'C(venv) wego@b4u:~/신학프로젝트/SDV_Robocar$ python3 desktop_yolo_stream2.py
[전송 성공] 2025-07-14_15-58-41
[전송 성공] 2025-07-14_15-58-43
[전송 성공] 2025-07-14_15-58-44
[전송 성공] 2025-07-14_15-58-45
[전송 성공] 2025-07-14_15-58-46
[전송 성공] 2025-07-14_15-58-48
[전송 성공] 2025-07-14_15-58-49
[전송 성공] 2025-07-14_15-58-50
[전송 성공] 2025-07-14_15-58-51
[전송 성공] 2025-07-14_15-58-52
[전송 성공] 2025-07-14_15-58-54
[전송 성공] 2025-07-14_15-58-55
[전송 성공] 2025-07-14_15-58-56
[전송 성공] 2025-07-14_15-58-57
[전송 성공] 2025-07-14_15-58-58
[전송 성공] 2025-07-14_15-59-00
```

```
(venv) wego@b4u:~/신학프로젝트/SDV_Robocar$ python3 k8s_server.py
* Serving Flask app 'k8s_server'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:8080
* Running on http://172.30.88.170:8080
Press CTRL+C to quit
[2025-07-14_15-58-41] 이미지 저장 완료 → inference_logs2/images/2025-07-14_15-58-41.jpg
[2025-07-14_15-58-41] 수신 완료 | 경제 수: 1
127.0.0.1 - - [14/Jul/2025 15:58:41] "POST /inference HTTP/1.1" 200 -
[2025-07-14_15-58-43] 이미지 저장 완료 → inference_logs2/images/2025-07-14_15-58-43.jpg
[2025-07-14_15-58-43] 수신 완료 | 경제 수: 1
127.0.0.1 - - [14/Jul/2025 15:58:43] "POST /inference HTTP/1.1" 200 -
[2025-07-14_15-58-44] 이미지 저장 완료 → inference_logs2/images/2025-07-14_15-58-44.jpg
[2025-07-14_15-58-44] 수신 완료 | 경제 수: 2
127.0.0.1 - - [14/Jul/2025 15:58:44] "POST /inference HTTP/1.1" 200 -
[2025-07-14_15-58-45] 이미지 저장 완료 → inference_logs2/images/2025-07-14_15-58-45.jpg
[2025-07-14_15-58-45] 수신 완료 | 경제 수: 0
127.0.0.1 - - [14/Jul/2025 15:58:45] "POST /inference HTTP/1.1" 200 -
[2025-07-14_15-58-46] 이미지 저장 완료 → inference_logs2/images/2025-07-14_15-58-46.jpg
[2025-07-14_15-58-46] 수신 완료 | 경제 수: 0
127.0.0.1 - - [14/Jul/2025 15:58:47] "POST /inference HTTP/1.1" 200 -
[2025-07-14_15-58-48] 이미지 저장 완료 → inference_logs2/images/2025-07-14_15-58-48.jpg
[2025-07-14_15-58-48] 수신 완료 | 경제 수: 0
127.0.0.1 - - [14/Jul/2025 15:58:49] "POST /inference HTTP/1.1" 200 -
[2025-07-14_15-58-49] 이미지 저장 완료 → inference_logs2/images/2025-07-14_15-58-49.jpg
```

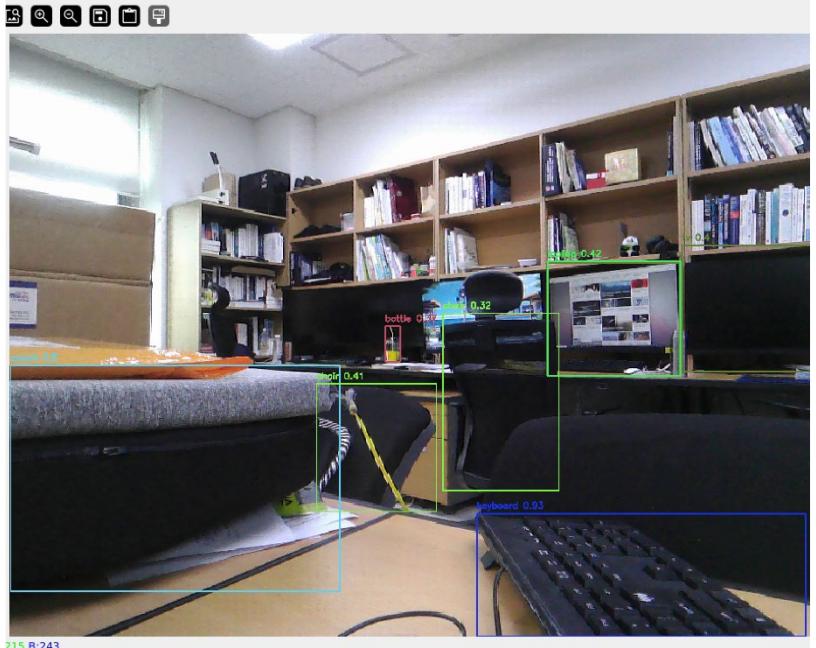
3. Run desktop_yolo_stream2.py
& k8s_server.py

```
es Terminal JUL 14 14:50 agilex@agilex-NUC12WSK17:~/SDV_Robocar/SDV_Robocar
agilex@agilex-NUC12WSK17:~/SDV_Robocar/SDV_Robocar$ python3 robocar_server2.py
[ WARN:0] global ./modules/videoto/src/cap_gstreamer.cpp (1100) open OpenCV | GStreamer warning: Cannot query video position: status=0, value=-1, duration=-1
* Serving Flask app 'robocar_server2'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:8000
* Running on http://192.168.56.82:8000
Press CTRL+C to quit
192.168.56.167 - - [14/Jul/2025 14:49:38] "GET /video HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:39] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:41] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:43] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:44] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:44] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:45] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:45] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:46] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:46] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:47] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:47] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:48] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:48] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:49] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:49] "GET /odometry HTTP/1.1" 200 -
192.168.56.167 - - [14/Jul/2025 14:49:50] "GET /odometry HTTP/1.1" 200 -
```

Robocar_server when stream success

Demonstration

YOLO Detection



YOLO Detection

images	2025-07-14 오후 3:59	파일 폴더
2025-07-14_15-58-41.json	2025-07-14 오후 3:58	JSON 월본 파일 239KB
2025-07-14_15-58-43.json	2025-07-14 오후 3:58	JSON 월본 파일 239KB
2025-07-14_15-58-44.json	2025-07-14 오후 3:58	JSON 월본 파일 239KB
2025-07-14_15-58-45.json	2025-07-14 오후 3:58	JSON 월본 파일 233KB
2025-07-14_15-58-46.json	2025-07-14 오후 3:58	JSON 월본 파일 233KB

Logs

What we learned

- We implemented a Monitoring Interface for collecting and analyzing the Moving Objects' data in the I2ICF Framework in Wireless Networks.
- We demonstrated Intent-Based Networking (IBN) for the configuring and monitoring of Moving Objects through the I2ICF Framework.

Next Steps

- We will support **three categories of actions** of moving objects:
Sensing, Movement, and Actuation.
- In IETF 124, we will adopt **an Intent Translator** in advances from a Rule-based scheme to an AI-based scheme (e.g., Large Language Model: LLM) on Kubernetes Container Orchestration System.
- Also, we will design **YANG Data Models for three categories of actions of Moving Objects.**
 - Refer to <https://datatracker.ietf.org/doc/draft-jeong-opsawg-i2icf-framework/>

Open-Source Project for I2ICF

[URL] <https://github.com/jaehoonpauljeong/I2ICF/tree/main/IETF-123>

The screenshot shows a GitHub repository page for the project `I2ICF / IETF-123 / SDV_Robocar`. The repository contains several files and sub-directories, including `main`, `BoF`, `IETF-120`, `IETF-121`, `IETF-122`, and `IETF-123`. The `IETF-123` directory is expanded, showing the `SDV_Robocar` folder which contains files like `README.md`, `desktop_yolo_stream.py`, `k8s_server.py`, and various Python scripts for robocar detection and control.

A recent commit by `rma0415` titled "Update README.md" is highlighted. The commit message includes the text "Update README.md". The commit was made to the `main` branch and has a timestamp of "Last commit message".

The footer of the page states: "This is the IETF-123 Hackathon Project for IETF-123 Hackathon."

Demonstration Video Clip for I2ICF

[URL] <https://www.youtube.com/watch?v=OKZS2LH-OmA>



4. The Robot can transfer the GPS, Speed, and Image data to I2ICF Analyzer

I2ICF Hackathon Team

- **Professors:**

- Jaehoon Paul Jeong (SKKU)
- Younghan Kim (SSU)
- Yong-Geun Hong (DJU)
- Joo-Sang Youn (DEU)
- Yiwen Chris Shen (AJU)

- **Researchers:**

- Jung-Soo Park (ETRI)
- Soohong Daniel Park (Samsung Research)

- **Students:**

- Mose Gu (SKKU)
- Yoseop Ahn (SKKU)
- Jiwon Suh (SKKU)
- Jisuk Chae(SKKU)

Hackathon Team Photo



Appendix

Set-up and Detailed-Procedure

LIMO ROS2 Specification



Physical appearance of the LIMO ROS2 platform

Specifications:			
Dimensions	Weight	Payload	Ground Clearance
322 x 220 x 251mm	4.8kg	4kg	24mm
Steering Structure	Screen	IPC	Depth Camera
40N·m	7 inches	Intel NUC	Orbbec Dabai
LiDAR	Battery	Operating Time	Standby Time
EAI T-mini Pro	10Ah 12V	2.5hours	4hours
System	Speed	ROS Version	
Ubuntu 22.04	1m/s	ROS2 Humble	
Control Distance	Control Method		
10m	Mobile App / Command		

Key hardware specifications of LIMO ROS2

System Architecture & Technology Stack

Input data:

- Real-time MJPEG video stream from [/video](#)
- Odometry data via [/odometry](#)

Output Data (sent every 1 second):

- YOLOv8 object detection results
- Robot speed and GPS (converted from odometry)
- Base64-encoded JPEG image

Storage Example (SERVER):

- inference_logs2/2025-07-14_16-22-01.json
- Inference_logs2/images/2025-07-14_16-22-01.jpg

All sensor data is processed every second and saved as structured JSON with detection results and image.

Programming Language:

- Python 3.x

Core Libraries & Frameworks:

- ROS2 Humble (odometry data)
- Flask (HTTP API server)
- OpenCV (video processing & display)
- Ultralytics YOLOv8 (object detection)
- Multiprocessing (ROS2 + Flask parallel execution)
- Base64 JSON (data encoding & structuring)
- Requests (HTTP transmission)

A lightweight software stack built with Python and ROS2 for real-time sensing and inference.

Datasets

- ETH -> Bird Eye View
- HEV-I(Honda Egocentric View-Intersection Dataset) -> outdoor dataset
- UCY -> outdoor dataset
- **AI Hub robot perspective driving video (advanced) - Social navigation robot driving -> Suitable for indoor navigation**
- **Test Dataset 60k, Validation Dataset 6k**

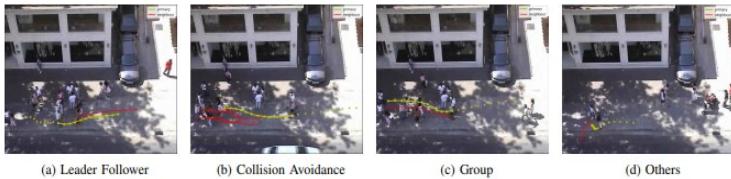
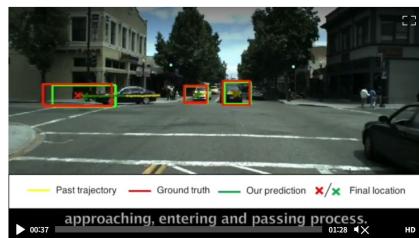
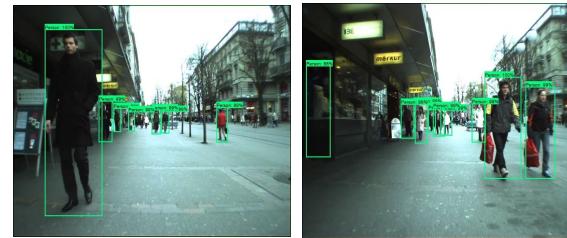


Fig. 9: Sample scenes from our benchmark. In each of the samples, we illustrate a different social interaction between the primary pedestrian (yellow) and the corresponding interacting neighbours (red) in real world datasets.



Hyperparameter

```
AMP: running Automatic Mixed Precision (AMP) checks...
AMP: checks passed
train: Fast image access (ping: 0.10 ms, read: 1650.8510.8 MB/s, size: 277.0 KB)
train: Scanning C:\Users\cowltnr31\data\indoor_preprocessed\train\labels.cache... 63938 images, 4098 backgrounds, 0 corrupt: 100% [██████████] | 63997/63997 [00:00<?, ?it/s]
train: C:\Users\cowltnr31\data\indoor_preprocessed\train\images\RGB-D\image_39956_RB2_PL08_D1_SN05_1_0037.jpg: 1 duplicate labels removed
train: C:\Users\cowltnr31\data\indoor_preprocessed\train\images\RGB-D\image_49228_RB2_PL09_D3_P2_SN05_1_0042.jpg: 1 duplicate labels removed
train: C:\Users\cowltnr31\data\indoor_preprocessed\train\images\RGB-D\image_49232_RB2_PL09_D3_P2_SN05_1_0046.jpg: 1 duplicate labels removed
val: Fast image access (ping: 0.10 ms, read: 1452.9317.4 MB/s, size: 327.7 KB)
val: Scanning C:\Users\cowltnr31\data\indoor_preprocessed\val\labels.cache... 6007 images, 282 backgrounds, 0 corrupt: 100% [██████████] | 6067/6067 [00:00<?, ?it/s]
Plotting labels to C:\Users\cowltnr31\PycharmProjects\ultralytics\runs\detect\train26\labels.jpg...
optimizer: 'optimizer=auto' found, ignoring 'lr=0.01' and 'momentum=0.937' and determining best 'optimizer', 'lr0' and 'momentum' automatically...
optimizer: SGD(lr=0.01, momentum=0.9) with parameter groups 57 weight(decay=0.0), 64 weight(decay=0.0005), 63 bias(decay=0.0)
Image sizes 640 train, 640 val
Using 8 dataloader workers
Logging results to C:\Users\cowltnr31\PycharmProjects\ultralytics\runs\detect\train26
Starting training for 50 epochs...

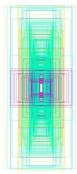
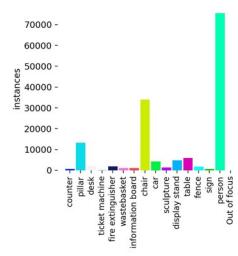
Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
1/50 3.636 0.7181 1.037 0.975 54 640: 100% [██████████] | 4000/4000 [12:18<00:00, 5.41it/s]
Class Images Instances Box(P R mAP50 mAP50-95): 100% [██████████] | 190/190 [00:24<00:00, 7.69it/s]
all 6067 13416 0.756 0.778 0.812 0.69

Epoch GPU_mem box_loss cls_loss dfl_loss Instances Size
2/50 3.836 0.686 0.732 0.9495 80 640: 37% [████] | 1469/4000 [04:25<07:55, 5.32it/s]
```

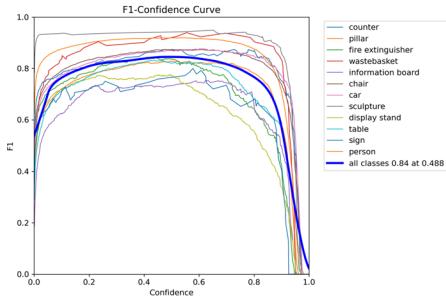
항목	값
model	YOLOv8s (Pretrained)
Task	Object Detection
Input Size	640 x 640
Epochs	50
Batch Size	16
Optimizer	SGD (momentum = 0.937)
Learning Rate	0.01
Weight Decay	0.0005
Train Images	63,997
Validation Images	6,067

Result

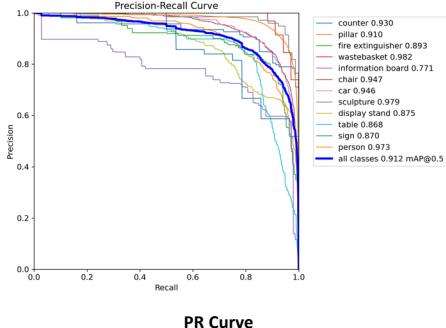
- YOLOv8s fine-tuning performance



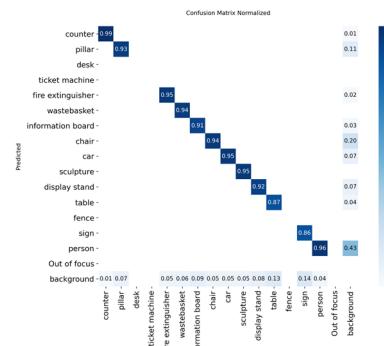
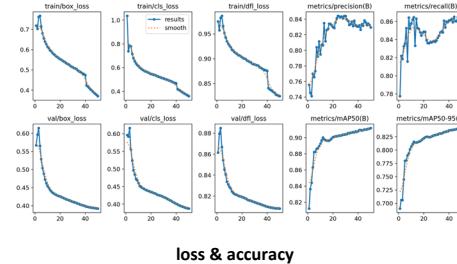
F1 score variation by confidence threshold



Precision-Recall Curve



Class distribution in training data



Normalized confusion matrix
showing per-class prediction accuracy

Result

- YOLOv8s fine-tuning performance

Model summary (fused): 72 layers, 11,131,776 parameters, 0 gradients, 28.5 GFLOPs						
Class	Images	Instances	Box(P)	R	mAP50	mAP50-95
all	6067	13416	0.83	0.867	0.912	0.841
counter	81	81	0.784	0.984	0.93	0.845
pillar	604	815	0.798	0.855	0.91	0.855
fire extinguisher	155	167	0.838	0.838	0.893	0.78
wastebasket	34	34	0.923	0.912	0.982	0.915
information board	55	145	0.657	0.828	0.771	0.623
chair	1171	2029	0.846	0.901	0.947	0.904
car	427	705	0.844	0.897	0.946	0.911
sculpture	78	78	0.935	0.949	0.979	0.933
display stand	314	485	0.83	0.722	0.875	0.794
table	401	426	0.839	0.81	0.868	0.801
sign	28	28	0.743	0.786	0.87	0.841
person	4202	8423	0.92	0.917	0.973	0.895

Speed: 0.1ms preprocess, 1.8ms inference, 0.0ms loss, 0.6ms postprocess per image

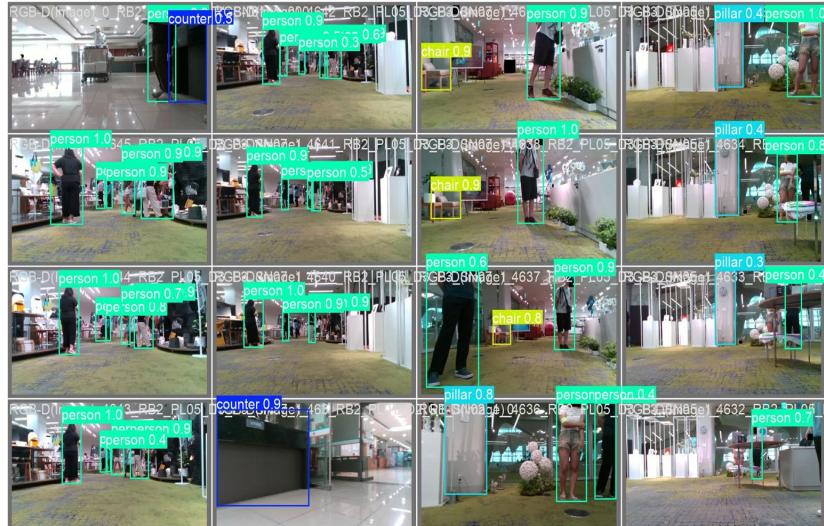
-> Epoch 50 => mAP50: 0.912/ mAP50-95: 0.841

Result

- Prediction vs Ground Truth



ground truth on validation images



predicted bounding boxes on validation images

Result

- Real-time result visualization via camera streaming

