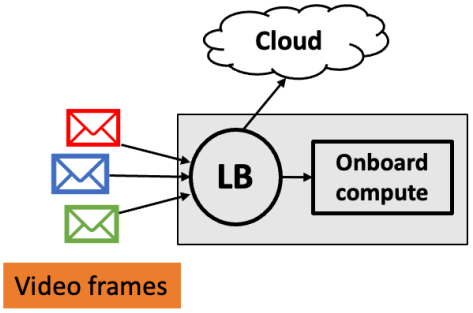
Performance Aware Cloud-level Load Balancer for Connected and Autonomous Vehicles

Progress report: April'24-Jul'25

Praveen Tammana, CSE Dept. IIT-Hyderabad

Idea: Performance-aware load balancer

In this proposal, we work on building a novel onboard load balancer (LB) for AVs connected to an edge cloud. The LB routes video inference requests between onboard and cloud. Onboard has models with low accuracy and resource efficiency. Whereas Edge Cloud runs models with high accuracy. If the vision-based application is delay-sensitive, video frames are forwarded to both the onboard and edge cloud, and the cloud's response is picked only if the response arrives on time (e.g., < ~100 ms). If the application is delay tolerant, then forward frames to one of the two locations based on the network conditions.



The proposal's goal is to explore and build an onboard LB that considers the application's delay tolerance, network delays, edge cloud availability, application accuracy level (high to low), and onboard load. More specifically, we design and develop an onboard intelligent and programmable load balancer for a vision-based object detection application to be deployed on autonomous vehicles. The features of the proposed LB are:

1. Aware of application requirements (e.g., delay tolerance, accuracy)
2. Reroute requests based on network performance
3. Adapt fast to changes (100's of microsecs)

Work packages:

1. **Prototype:** A prototype of an "intelligent onboard load balancer" that splits video frames from an autonomous vehicle (e.g., warehouse rover) cameras between onboard and edge cloud.
2. **Testing:** Test the prototype using TiHAN testbed's outdoor WiFi and Edge cloud lab.

3. **Evaluation:** Evaluate the performance regarding scalability, request completion time, resource overheads, and accuracy.
4. **Publications:** A publication covering the problem statement, different design aspects of the system, and evaluation results.

Work package	**Objectives**	**Status**
Prototype	We drive our research by developing prototypes on two bot-specific platforms: Agribot and Warehousebot. 1. Understand the onboard system workflow 2. Design and develop an onboard only system (both hardware and software) 3. Design and develop building blocks: onboard agents, cloud APIs, and cloud services. 4. Design and develop offboard only system with the building blocks (both hardware and software) 5. Design and develop an intelligent onboard LB that routes requests between onboard and edge cloud offloaded services (offboard).	1. Done 2. Done 3. Done 4. Done 5. In-progress
Testing	1. Develop two applications, one each for two setups. 2. Two applications: Warehousebot vision-based navigation, Agribot tomato detection, and pluck. 3. Two setups: Completely onboard and edge cloud-offload (offboard). 4. Test each prototype with the associated use case, both onboard and offboard in a lab setup. a. Collision detection usecase for Warehousebot. Demos: collision detection onboard, collision detection offloaded to edge cloud.	1. Done 2. Done 3. Done 4. Done

	b. Tomato harvesting usecase for Agribot. Demos: multi-tomato pluck onboard, tomato pluck offboard. 5. Test each prototype with the associated use case, varying network delays, cloud availability, and onboard load.	5. In-progress
Evaluation	Profile onboard and offboard performance for usecases (real time scenario): 1. Onboard vs. Offboard (A/B): CPU, RAM, GPU, Power utilization 2. Onboard vs. Offboard (A and B): LB adaptation to network delays, cloud availability, and onboard load while meeting application requirements. 3. Onboard vs. Offboard (A and B): Req. completion time, Throughput, Impact on usecase.	1. Done 2. In-progress 3. In-progress
Publications	1. **"Leveraging Edge-cloud for Compute Efficient and Safer Autonomous Navigation"**, ACM/IEEE COMSNETS 2025. 2. ***Developing an Affordable Agribot for Efficient Harvesting: A Practical Experience*** 3. **Agribot: Offboard Cloud Architecture for Agricultural Workloads — Profiling and Results** 4. **Understanding the benefits of Onboard and Edge cloud offloads for Vision-based Navigation**	1. Done 2. In-progress 3. In-progress 4. In-progress

1. WP1: Prototype

A prototype of an "intelligent onboard load balancer" that splits video frames from an autonomous vehicle (e.g., warehouse rover) cameras between onboard and edge cloud.

We drive our research by developing prototypes on two bot-specific platforms: Agribot and Warehousebot.

Progress:

1. **Objective:** (a) Understand the current onboard system workflow. (b) development.
    a. Interaction among ROS nodes
    b. Control commands to the Actuation workflow
    c. Find ROS nodes running on CPU and GPU
    d. **Warehousebot**: More details are in Section II (Background), Fig. 1: "**Leveraging Edge-cloud for Compute Efficient and Safer Autonomous Navigation**", published in ACM/IEEE COMSNETS 2025.
    e. **Agribot:** More details are in Sections VI (End To End System Architecture And Workflow, Fig. 12: **"*Developing an Affordable Agribot for Efficient Harvesting: A Practical Experience*,"** *unpublished manuscript.*
2. **Objective:** (a) Design and develop building blocks: onboard agents, cloud APIs, and cloud services. (b) development of offboard system with the building blocks.
    a. Onboard agents offload compute to the edge cloud
    b. APIs for Inference-as-a-Service
    c. Edge cloud services process inference requests
    d. **Warehousebot:** More details are in Section IV (Design), Fig. 2: **"Leveraging Edge-cloud for Compute Efficient and Safer Autonomous Navigation"**, ACM/IEEE COMSNETS 2025.
    e. **Agribot:** More details are in Section V (System Architecture, Design and Implementation), Fig. 2: "**Agribot: Offboard Cloud Architecture for Agricultural Workloads — Profiling and Results**," *unpublished manuscript.*

In-progress:

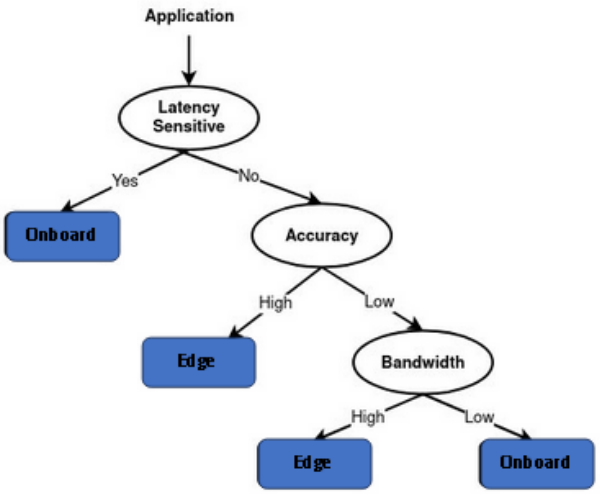
1. **Objective:** Design and develop an intelligent onboard LB that routes requests between onboard and cloud offloaded services (offboard).
    a. DLB: function (application requirements, telemetry feed)
    b. Application requirements
        i. Application delay tolerance

ii. Application accuracy level (high to low)

c. Telemetry feed
   i. Network delays
   ii. Edge cloud availability
   iii. Onboard load

f. LB design is in progress. Our design is inspired by the ideas discussed in the papers below:
   i. Section IIIA, in the paper "Bumblebee: Application-aware adaptation for edge-cloud orchestration", published in SEC'22 (link)
   ii. Section V, in the paper "Leveraging Cloud Computing to Make Autonomous Vehicles Safer", published in IROS'23 (link)
   iii. Section III, in the paper "DL^3: Adaptive Load Balancing for Latency-critical Edge Cloud Applications", published in CNSM'24 (link)

The key idea is to make the LB aware of the application requirements of a request and route the request either to the onboard or the edge cloud for processing.



For instance, when a request arrives, the load balancer checks the requirements of the application. If it is a request for a latency-sensitive application, the request has to be processed onboard. If not, it checks for other requirement accuracy. If the request is for a high-accuracy application, it has to be routed to the edge cloud for processing. Otherwise, the request is routed to either the onboard or the edge cloud based on the load on the onboard unit, the network conditions to the edge cloud, or the availability of the edge cloud.

2. WP2: Testing

Test the prototype using the TiHAN testbed's outdoor WiFi and Edge cloud lab.

Progress: Developed two applications, one each for two setups. The two applications are Warehousebot vision-based navigation, Agribot tomato detection and pluck. The two setups are: completely onboard and edge cloud-offload (offboard). Test each prototype with the associated use case.

1. **Warehousebot vision-based navigation application**
    a. **Collision detection use case:** Demo link for collision detection onboard, collision detection offloaded to edge cloud.
    b. **Onboard design and development:** More details are in Section II (Background), Fig. 1: "**Leveraging Edge-cloud for Compute Efficient and Safer Autonomous Navigation**", ACM/IEEE COMSNETS 2025.
    c. **Offboard design:** More details are in Section IV (Design), Figure 2: **"Leveraging Edge-cloud for Compute Efficient and Safer Autonomous Navigation"**, ACM/IEEE COMSNETS 2025.
2. **Agribot tomato detection and pluck application**
    a. **Harvesting tomatoes use case:** Demo link for multi-tomato pluck onboard, tomato pluck offboard.
    b. **Onboard design and development:** More details are in Sections VI (End To End System Architecture And Workflow), Fig. 12: **"*Developing an Affordable Agribot for Efficient Harvesting: A Practical Experience*,"** *unpublished manuscript.*
    c. **Offboard design:** More details are in Section V (System Architecture, Design and Implementation), Fig. 2: "**Agribot: Offboard Cloud Architecture for Agricultural Workloads — Profiling and Results**," *unpublished manuscript.*

In-progress: Test each prototype with the associated use case varying network delays, cloud availability, and onboard load.

3. WP3: Evaluation

Profile onboard and offboard performance for use cases (real-time scenario).

1. **Progress:**
    a. Onboard vs. Offboard (A/B): CPU, RAM, GPU, Power utilization, Request completion time, and Inference latency.
    b. Results for Warehousebot vision-based navigation application results are in **"**Leveraging Edge-cloud for Compute Efficient and Safer Autonomous Navigation**"**, ACM/IEEE COMSNETS 2025.
        i. CPU, GPU, and RAM Consumption: Section VI(A), Fig. 4.
        ii. Energy Consumption: Section VI(C), Fig. 7 and 8
        iii. Request Completion Time: Section VI(B), Fig. 5 and 6
    b. **Results for Agribot – Harvesting application results are in** "Agribot: Offboard Cloud Architecture for Agricultural Workloads — Profiling and Results," *unpublished manuscript*.
        i. CPU, GPU & RAM Consumption: Sections VI(C)(Evaluation), Fig. 3.
        ii. ROS Node CPU & RAM Consumption: Sections VII(B)(Results), Table 1.
        iii. Energy Consumption: Sections VIII(D) (Energy Consumption), Fig. 8.
        iv. Inference latency: Sections VII(A) (Results), Fig. 4.
2. **In-progress:**
    a. Onboard and Offboard (A and B): Experiment setup to test the performance of the onboard LB under variable network delays, edge cloud availability, and onboard load while meeting application requirements.
        i. Impact on two use case applications: Warehousebot vision-based navigation application and Agribot Harvesting.
        ii. Environment:
            1. Introduce random network delays
            2. On/off edge cloud availability
            3. Vary the onboard load
        iv. Metrics:
            1. Request latency (Avg, P99, P9999)
            2. Request throughput
            3. Accuracy - onboard vs. offboard
            4. Onboard resource utilization (CPU, GPU, Power, Memory)
            5. LB's adaptation to network delays and edge cloud availability
                a. Deepdive: feedback control loop time to react and retract
                b. LB's overhead - telemetry
        v. End-to-End metrics: Time to finish action (stop, start)
        vi. Use case specific metric: Measure the distance from the object. Vary network delays and cloud availability

4. WP4: Publications

1. **Progress**:
    a. Prashanth P S, Hrushikesh S, Amrit Kumar, Yuvraj Chowdary Makkena, Praveen Tammana, **"Leveraging Edge-cloud for Compute Efficient and Safer Autonomous Navigation"**, ACM/IEEE COMSNETS 2025.
    b. Prashanth P S, Ranjitha K, Ankit Sharma, Arjun Temura, Rinku Shah, Praveen Tammana, "DL^3: Adaptive Load Balancing for Latency-critical Edge Cloud Applications", Published in IEEE CNSM'24.
2. **In-progress:**
    a. Yuvraj Makkena, Vidya Vepoori, Amit Kumar Patel, Gurusai Nidhish Chadive, Yashwanth Yasp, Prateek Kumar, Kirtpreet Kaur, Anil Kumar Sharma, Praveen Chandrahas, and Praveen Tammana,**"Developing an Affordable Agribot for Efficient Harvesting: A Practical Experience,"** *unpublished manuscript.*
    b. Yuvraj Makkena, Vidya Vepoori, Amit Kumar Patel, Praveen Chandrahas, and Praveen Tammana,"**Agribot: Offboard Cloud Architecture for Agricultural Workloads — Profiling and Results(IEEE)**," *unpublished manuscript*
    c. Mohana Bathula, Prashanth P S, Yuvraj Makkena, and Praveen Tammana, "**Understanding the benefits Onboard and Offboard**"*, unpublished manuscript.*
    d. Prashanth P S, Hrushikesh J S, Amit Doda, Abed Mohammad Kamuluddin, Satananda Burla, Praveen Tammana,"**Bypassing the CPU: GPU-Centric Video Inference for Edge Offloaded Applications**", submitted in *International Conference on AI-ML Systems* (link).