CONTAINERIZED ON-EDGE FRAMEWORK FOR AUTONOMOUS VEHICLES

Bruce Chen, Harold J. Iwen, and Sihai Tang

Background

Edge Computing:

Modern edge computing allows for the implementation of low latency and high proximity to the end user, providing computing power with minimal latency and high responsiveness that cloud cannot achieve.

The edge is a step beyond the traditional cloud computing, taking the power of powerful computation to cover regions requiring additional computational power and or data processing needs.

Use Case

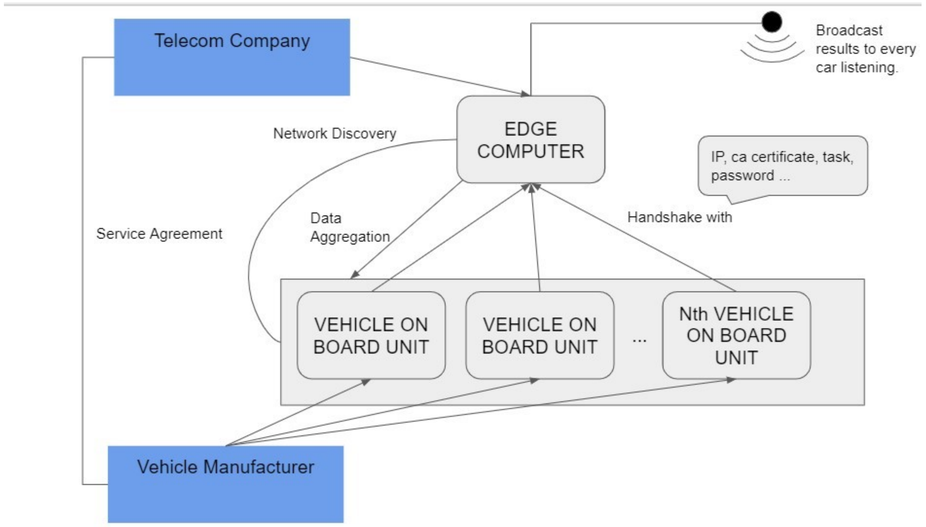
CAV and Edge

To the connected autonomous vehicles, edge is a very important new concept as it is able to minimize the network congestion regardless of the latest cellular technology such as LTE or the encroaching 5G.

By providing the service of edge computing to CAV, we alleviate the pressure of the on-board computation power for the dedicated use of autonomous driving, thus freeing up the vehicle from any overhead in additional tasks.

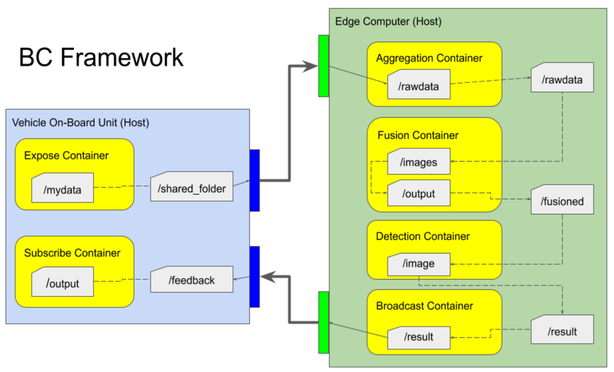
Through computational offloading on the edge, tasks such as amber alerts, accidents, traffic intersection safety and much more can be achieved with minimal impact to the vehicle.

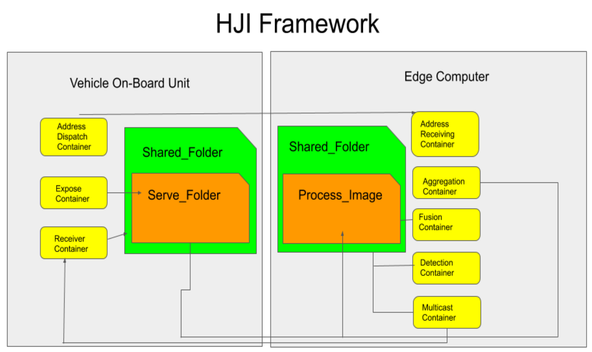
Edge Computing for CAV



A service agreement is made, allowing for future network discoveries. Upon discovering an edge device, autonomous vehicles send identifying information and expose data. The edge device will then process data from the autonomous vehicle and return results if necessary.

Framework Designs

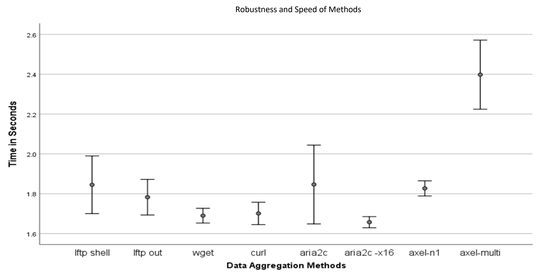




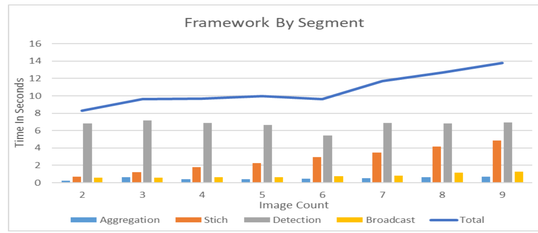
Framework Concept

A framework is all about inverted control. The framework demonstrations will perform the computation offloading task of acquiring 2D image data from multiple vehicles, fusing them as appropriate before running an CNN object detection model on the result. The detection results will then be broadcasted back to the vehicles in question. The process shall be automated according to our framework ideology.

Results



1080P					
Group	Aggregation	Stitch	Detect	Broadcast	Total Time (S)
Above	0.6	2.64696	3.63507	0.00544	6.88747
Below	0.3	1.60079	3.56692	0.00523	5.47294
Cool	0.3	0.97034	3.57243	0.00461	4.84738
First	0.2	1.72102	N/A	N/A	1.92102
Hi	0.3	1.25113	3.61586	0.00539	5.17238
Shaak	0.3	1.72399	3.63339	0.00693	5.66431
Stop1	0.3	1.53299	3.63738	0.00671	5.47708
Stop2	0.3	1.89981	3.61449	0.00449	5.81879
Sub	0.6	3.70769	3.68509	0.00872	8.0015
Tool	0.3	2.01195	3.65668	0.00649	5.97512
Vader	0.4	1.3641	3.61012	0.0046	5.37882
AVERAGE	0.354545455	1.857343	3.622743	0.005861	5.510619091



- Data aggregation is best performed with wget, as it is the most robust command with recursive functionality.
- With MQTT a much faster broadcasting time can be achieved when compared to multicasting.
- Detection was the only framework component that stayed relatively constant with respect to the image quantity.

Acknowledgments

Thank you to the project leader Sihai Tang and program heads Dr. Song Fu and Dr. Qing Yang.