Current Issues and Future Directions of Autoware Architecture #5032

yukkysaito started this conversation in **Design**

yukkysaito on Jul 25 Maintainer edited

The current Autoware architecture is based on the Autoware Architecture Proposal introduced 4 years ago.

- https://github.com/tier4/AutowareArchitectureProposal.proj
- Autoware Architecture Proposal.pdf

However, at that time, a perfect architecture had not yet been found, and it was considered the best available architecture given the technological standards of the time. As a result, several trade-offs were made, and it became apparent that there are inherent limitations in its performance as an autonomous driving system due to its architecture. Present-day Autoware is widely used in many places and is even progressing towards commercialization in domains with limited operational design domains, which is commendable. However, when considering services in urban environments like taxi services, improving modules within the current Autoware architecture may not achieve complete autonomy.

In this discussion, we aim to explore the constraints of the current Autoware architecture and discuss potential solutions for future re-architecture. We will brainstorm specific constraints and issues with the current architecture and create separate discussion threads for each detailed issue and its potential solutions. This discussion will serve to consolidate individual discussions into actionable insights.

Furthermore, it is crucial to document what trade-offs were made in the architecture and why. Considering how to realistically cover these trade-offs is essential. If dropped elements are critical for realizing autonomous driving as a service, resolving these issues will be key. For instance, aspects that can be covered by remote assistance (Tele Operation) might be less problematic if dropped. Taking these points into account, we aim to advance discussions towards re-architecting Autoware.

Specific examples of constraints and issues that have been evident since the proposal of the Autoware architecture have already been documented in threads.

- The difficulty in handling complex scenarios due to the hierarchical architecture of Planning and the modularization of functionalities.
- The interface between Perception and Planning lacks sufficient information.
- Introduction of route level information in map
- Autonomous driving without HD maps and (WIP).

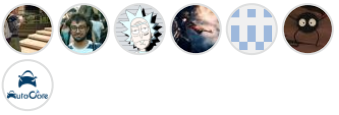
Category

Design

Labels

component:interfac...

7 participants



- Driving through long tunnels (WIP).
- Map representation on a 1000 km scale (WIP).

↑ 2 👍 2

4 comments · 21 replies

Oldest | Newest | Top

mitsudome-r on Jul 26 Maintainer

@yukkysaito
Thanks for posting the discussion.

Here are some of the limitations/issues that I had in my mind:

- **Introduction of route level information in map**: We are working on dynamic loading of Lanelet2 map, but some modules like mission_planner would require a full map anyways to calculate the global route. However, such planner on the other hand should not need detailed geometric information but only need abstracted route graph like you see in Google Map. We could consider updating map interface to support different level of abstraction to be sent to client nodes.
- **Rerouting Mechanism for Planning**: This might be relevant with the handling of complex scenarios that you mentioned, but currently Autoware gets stuck if it fails to lane change or find some obstacle that it cannot avoid. Even in such situation, we would like to have some kind of rerouting feature triggered from behavior planner whenever it thinks that the original route is not feasible.

↑ 1 2 replies

yukkysaito on Jul 31 Maintainer Author

Thank you for raising the issue. I also believe that the two suggestions you mentioned will need to be considered in the future, given the architectural constraints. Could you please start a thread in the GitHub Discussion for a more detailed explanation? I'd like to add it to the task list.

mitsudome-r on Aug 21 Maintainer

@yukkysaito I created the discussion:
https://github.com/orgs/autowarefoundation/discussions/5113

👍 1

liuXinGangChina on Aug 21 Collaborator

Hi,saito-san. @yukkysaito
As a engineer both in L4 and ADAS area, I have used AW since 2018 and witness the rapid grownth of AW. I have several suggestion about AW's Architecture accrding to my experience and the feedback from my collegague

1.Main principle for the design of whole system is "Meet system requirement simply and efficiently"
2.For localization in an open env like urban streets lane-detect, gnss, wheel speed and imu fusion is pretty enough, ndt can be used to provide a extra source for redundancy
3.In terms of perception, in most of the lane driving scenarios, lidar camera radar combination is sufficient, the most important thing is "Squeeze all resources out of the system cpu and gpu to reduce latency of the perception pipeline"
4.Regarding planning just focusing on two scenarios “lane centering” and “lane change” they can cover most of the situation and give passengers a good experience. For the rest of the scenario ,i believe they can be solved by using free-space planning

Have a nice day

↑ 1 👍 1 2 replies

yukkysaito on Aug 21 Maintainer Author

@liuXinGangChina Thank you for your comment.
As someone who has been involved with Autoware since its inception, I am very pleased to hear your feedback.

I agree with your opinions. The current architecture of Autoware has undergone significant functional additions that were somewhat forced compared to the initial proposals. As a result, the code and functions have become more complex, and I believe it is necessary to re-architect the system. I will carefully consider your advice and suggestions as guidelines for this re-architecture process. If you have any specific issues or proposals regarding the current architecture, please let me know.

liuXinGangChina on Sep 4 Collaborator

hello Saito-san @yukkysaito regarding my comment item #3 “reduce latency of the perception pipeline” we have create a task to analyze compute latency of current autoware’s perception pipeline -- issue link

armaganarsln on Aug 21 Collaborator

@yukkysaito I think this discussion is really necessary and hopefully you will have the results that you are looking for. Having said that my thinking "personally" is that we should start looking into E2E architecture. I made a presentation at SPC (strategy planning committee) and discussed it with other people who were against it.

Today also I talked to **@kminoda** san during sensing & perception WG. I believe the technical team is capable of creating a better system but the resources and freedom are not available to them and I am trying to change that by asking questions and bringing these topics up. Hopefully 2025 we will move to General Perception Net and Planning Net architecture and move on from there.

Here is my presentation if you want to have a look:
https://docs.google.com/presentation/d/1SH2r3z56qdC5ErjRzORozKWJH-pnSLfkSeXatfFa9Vk/edit#slide=id.g2efc2e3471b_0_50

I will be very happy to learn your thoughts as well.

↑ 1 👍 2 16 replies

Show 11 previous replies

samet-kutuk on Aug 23 Collaborator

I agree with Doğan's observations. And simply don't agree with Armağan's observations in general.

I think Armağan's understanding of the SDV is incomplete. I'm saying this because I think Armağan got the wrong impression and got confused from that podcast we've recorded with Robert from SOAFEE regarding the notion of bringing the vehicles to a dealership for updating the vehicles. The main idea of the SDV is software freshness and it actually proposes a solution to update the vehicles via OTA updates (which Tesla does) rather than bringing vehicles to a dealership/service center to update the vehicle's software. That was the main idea there.

Traditional way of updating a vehicle's software -> bringing it to a service center for update (a baby update, if you will)
vs.
Modern way of updating a vehicle's software -> get the software updates to the vehicles over the air, over night either when your vehicle is parked at your garage and/or via fleet management software.

An initiative like Co-MLOps can serve perfectly in an SDV setting actually. Accumulating driving data over time, run inference and improve the software continuously, utilizing data platforms like Co-MLOps.

Let's get our facts straight first.

armaganarsln on Aug 23 Collaborator edited

I am not here to discuss SDV because SDV is neither a solution nor a new architecture. It's, as I said before a buzz word which doesn't have benefits for Autoware users right now. It's made up for mass production vehicles by TIER1's which is not our use case currently. It might be in the future maybe 2030 but I am trying to find a solution today.

And lets get our facts straight. Of course I am not against updating the software remotely we already have things in place to do that. I don't understand why you want to discuss SDV so could we please focus on the E2E driving instead?

samet-kutuk on Aug 23 Collaborator

I'm talking about the SDV here to defend some of the activities and commitments we have, that unfortunately you're blatantly attacking at every turn.

It seems like you have very strong opinions on some matters, but in my opinion, they're not backed by a good understanding.
My contention is, if we are smart, we would cross-pollinate the concepts of SDV and data-driven development approach, which are not mutually exclusive - but in a symbiotic relationship.

What I've heard so far about the E2E development paradigm that it requires tons of investment that only very deep-pocketed organizations can undertake; of which the success is not a guarantee.

Besides, I'd like to learn how adopting an E2E development approach would impact the community-driven aspect of Autoware. One might say that all contributors can contribute with their data to this approach, however Autoware has well-established modular architecture, where many contributors are working on improving component stacks. It might not be realistic to expect all Autoware users and developers to change their worldview about how to develop autonomous systems.

I'm not a big believer in moonshot projects - unless you have necessary resources to put behind it and take all the risks. I think where we have left the conversation around E2E development is not to go moonshot way but hybridize and take a gradual development approach. I think everybody appreciates the value of AI models and we should leverage them to the maximum - but considering the realities of resource constraints.

armaganarsln on Aug 23 Collaborator

Well, I am sorry to hear that you think I am attacking SDV. In this thread (here is my original post) : I didn't even mention it until its mentioned as a solution instead of E2E driving research.

And the last time we met at SPC my question was about resource allocation, not the concept itself. And as you said one needs resources to achieve goals that's why I asked whether we could assign our resources to E2E instead of SDV. That's what all was about.

I hope that we could work together and dedicate our resources to E2E driving but besides that I couldn't care less about SDV. Even that I won't use the word anymore not to be affiliated with the topic.

doganulus on Aug 24 Collaborator

> And the last time we met at SPC my question was about resource allocation, not the concept itself. And as you said one needs resources to achieve goals that's why I asked whether we could assign our resources to E2E instead of SDV. That's what all was about.

Currently, I see there are -1 people working on the SDV-related topics in Autoware. One full-time engineer and 3-4 part-time blockers. If you wish, I can happily support you in the SPC to allocate those latter resources somewhere else. You should not force people to work in the working groups they do not believe or understand its mission.

Owen-Liuyuxuan on Sep 4 Collaborator

One of the main architectural considerations I want to raise is that the behavior planning and perception modules should both be "local". They could be vehicle coordinates in lane driving or could be odom coordinates in parking, but generally, they are "local".

1. From the nature of perception and behavior planning in driving, they only involve objects and environments that are around the vehicle instead of in the scale of the world. So fundamentally, whenever our perception results are being transformed into the "global map" frame, or conducting planning using "global map" frame coordinates, we are probably doing some "non-optimal" things
   a. One of the main targets is to make sure planning can work on both raw HDMaps or onboard HDMaps from perception.
   b. Decouple perception and localization as much as possible.
   c. Improving local map interfaces, which will also become a good start for future E2E planning modules.
2. Mission Planner and global map system should be grouped as service providers to hide away global information. Providing **local maps** and goals at some low frequency.

↑ 1 1 reply

doganulus on Sep 5 Collaborator

> They could be vehicle coordinates in lane driving or could be odom coordinates in parking.

Ideally, the lane and free-space planner should be active at the same time. When a lane structure is detected, the lane system overrides the free-space planner outputs while the latter still continues to calculate. This means the free-space planner acts as a fallback mechanism. Such redundancy is valuable for safety.

> Mission Planner and global map system should be grouped as service providers to hide away global information. Providing local maps and goals at some low frequency.

> I agree with that wholeheartedly. The vehicle must be safe without global info, maps, and even sensors. It is a good system architecture that makes it possible.