[Proposal] New architecture for behavior path planner module #3097

satoshi-ota started this conversation in **Design**

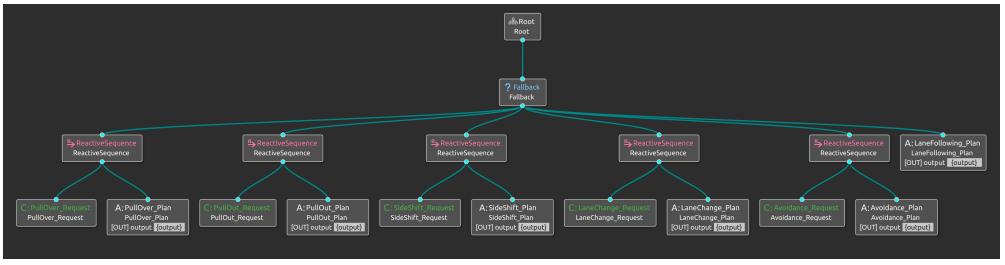
satoshi-ota on Dec 1, 2022 Maintainer edited

We TIER IV would like to propose a new architecture for beahvior path planner module.

Background

The current behavior path planner module architecture is working in the following flow/rules:

- Each module (lane change, avoidance, etc) and function (has_request, is_ready, run, etc) is managed in the BehaviorTree framework
- Module manager checks the necessarity of path modification in a predetermined order.
  - The order don't change in runtime.
- Module manager runs the module which requests path modification at first.
  - while one module is already running, the other module is not able to run simultaneously.



In short, the current architecture has the following two major limitations:

1. Multiple modules cannot run simultaneously, which makes it difficult to achieve complex use cases (e.g. parked vehicle avoidance during lanechange maneuver).
2. Hierarchies between modules can not be changed in runtime, which can not achieve flexible path generation according to use cases.

To overcome these limitations, we TIER IV would like to propose a new architecture for behavior path planner module. It has two major changes.

1. Interface of each module is changed to run multiple modules.
   a. Modules can be operated in series, allowing the manager the flexibility to change the sequence.
2. Replace the Behavior Tree based manager with a simple c++ code for flexible development

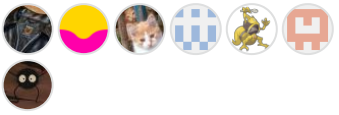
Category

Design

Labels

component:planning

7 participants



a. The interface for each module is still being explored. At this stage, it is difficult for developers to exploit the advantage of the behavior tree (although it may return to the behavior tree in the future).

Details are described below.

Examples of expected use cases achieved by this proposal

- Lane change when an obstacle exists in the target lane
- Pullover/out when an obstacle exists in the target lane
- Obstacle avoidance by lane change

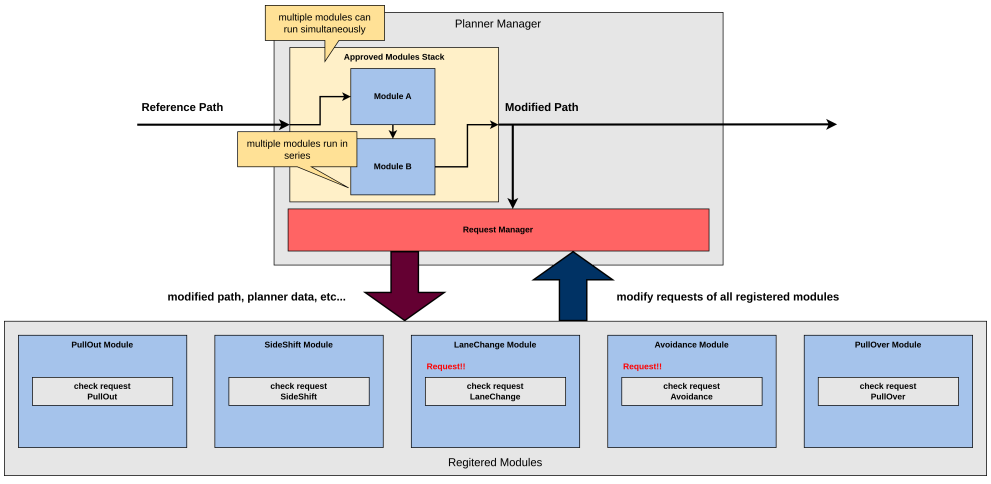
Proposed architecture

documentation here

Overview

The proposed architecture is shown below. The key features are following:

- Multiple modules can run simultaneously in series.
- There is no need to design the hierarchy between modules in advance, and the execution order can be freely designed.



(*Reference Path: the path generated from the centerline of road lanelet.*)

Pros:

- can achive complex use cases
- scalable

Cons:

- more complex than original architecture

Module Interface

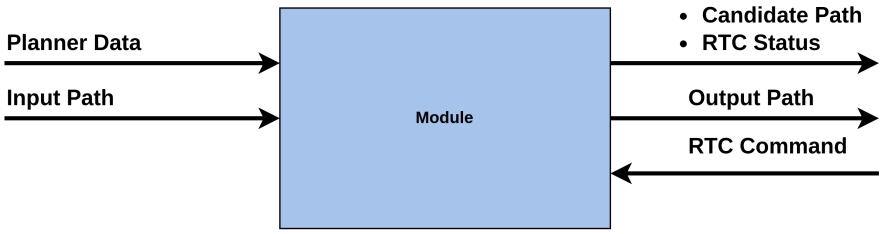
All modules outputs two paths.

- Candidate Path: the modified pathe for operator.

- Output Path: the path which the vehicle drive on.

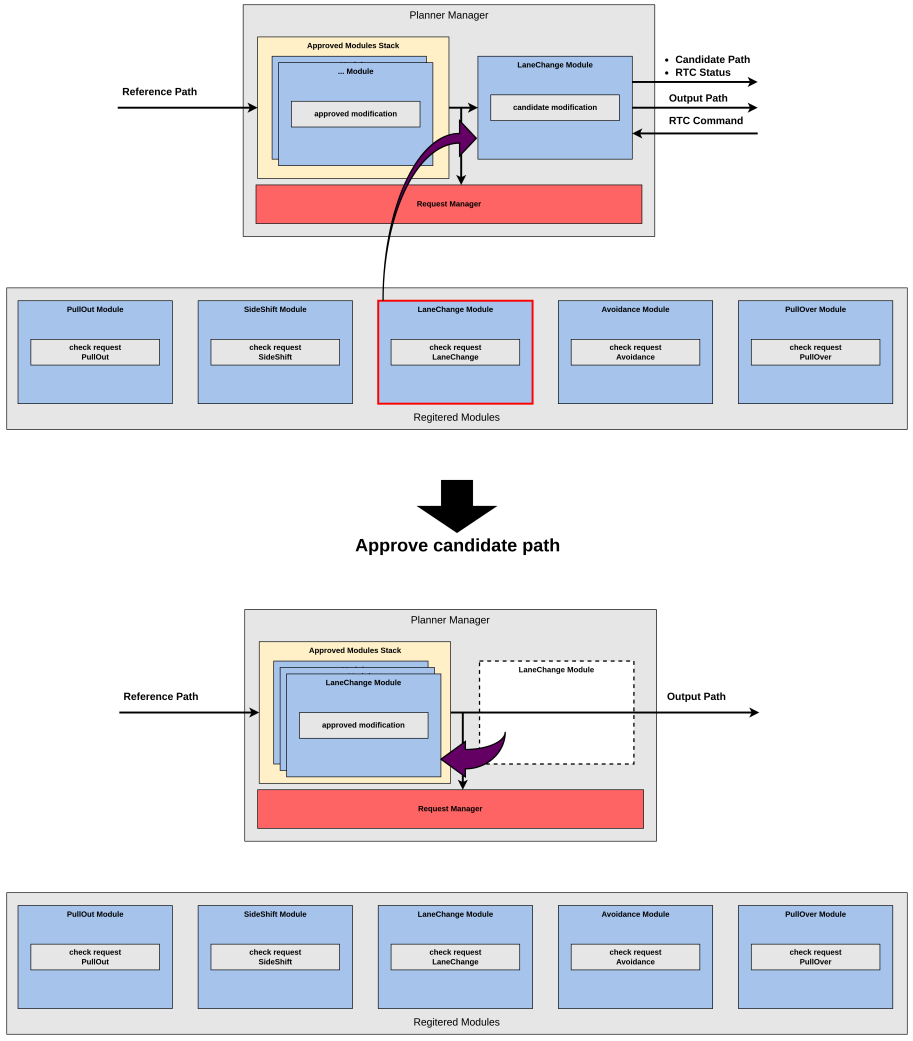
At first, the running module receives the latest data and path, and outputs the modified path as **Candidate Path** and RTC Status. The RTC (Request To Cooperate) is a system that asks the operator's approve to change the path (RTC Status), and the operator checks the candidate path and approves the module's path modification (RTC Command). If the path modification is approved, the module outputs the path as **Output Path**.

- Input Path (autoware_auto_planning_msgs/msg/PathWithLaneId)
- Planner Data
- RTC Command (tier4_rtc_msgs/srv/CooperateCommands)
- RTC Status (tier4_rtc_msgs/msg/CooperateStatus)
- Candidate Path (autoware_auto_planning_msgs/msg/PathWithLaneId)
- Output Path (autoware_auto_planning_msgs/msg/PathWithLaneId)



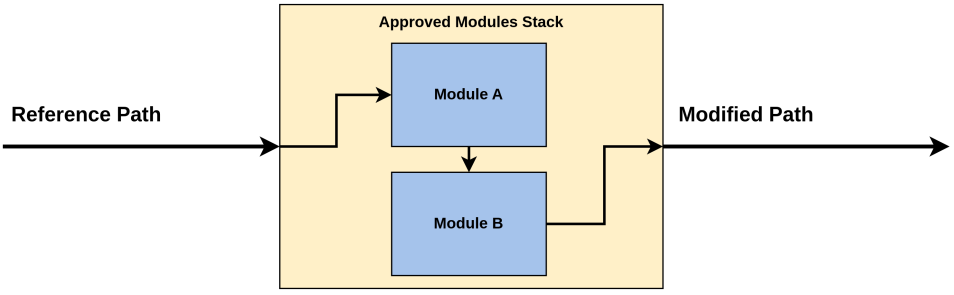
Planner manager

There is no update of input/output topics, but there is new planner manager as a subsititute for BehaviorTree. The manager passes the latest subscribed data to the registered modules and activates module(s) if necessary. The activated module first outputs a candidate path, and if the path modification is approved, the module is moved into Approved Modules Stack by planner manager.



How Approved Module Stack works ?

The Approved Modules Stack (yellow block in the above figure) has a vector or queue structure in which approved modules are stored. This block receives the reference path and outputs modified path as the result of the modification by all stored modules. In this time, all modules run in series, and the output of one module is the input of the next module.



How Request Manager works ?

The Request Manager (red block in above figure) passes the latest data to module checks which registered module requests path modification based on the latest modified path of Approved Modules Stack. If several modules request path modification, it selects the one module among them that has the highest priority to be run. In my mind, the algorithm for prioritizing among the modules can be based on following methods, but it can be made richer in the future.

- designed in advance

- give highest priority to the module whose modification area is in the very front

As first step, we TIER IV would like to use the former one, but if this architecture works well as intended, we will try the latter approach.

Process

There are 5 steps in one planning process:

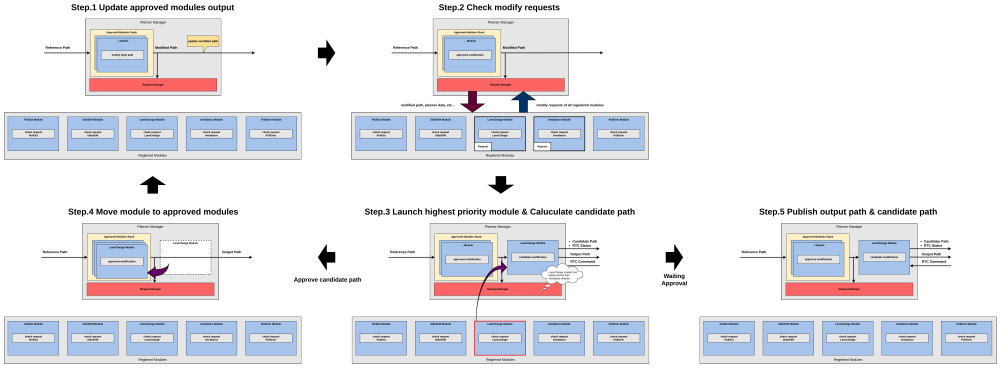
1. Update approved modules output. All registrered modules judge the necessarity of path modification based on this.
2. The Request Manager passes the latest modified path (output of Approved Modules Stack) and subscribed data to all registered modules and recieves path modification requests.
3. The Request Manager selects **ONLY ONE** module to run among those that have requested a change of path. The other module's requests are dismissed in this iterate. The selected (highest priority) module calculates and outputs the candidate path, and waits approval. (If the selected module runs in auto approval mode, the candidate path is approved in this Step3.)
4. If the candidate path is approved, the selected module is moved into Approved Module Stack by manager.
5. Publish output path and candidate path.

and, loop step1~4 until the following conditions are satisfied.

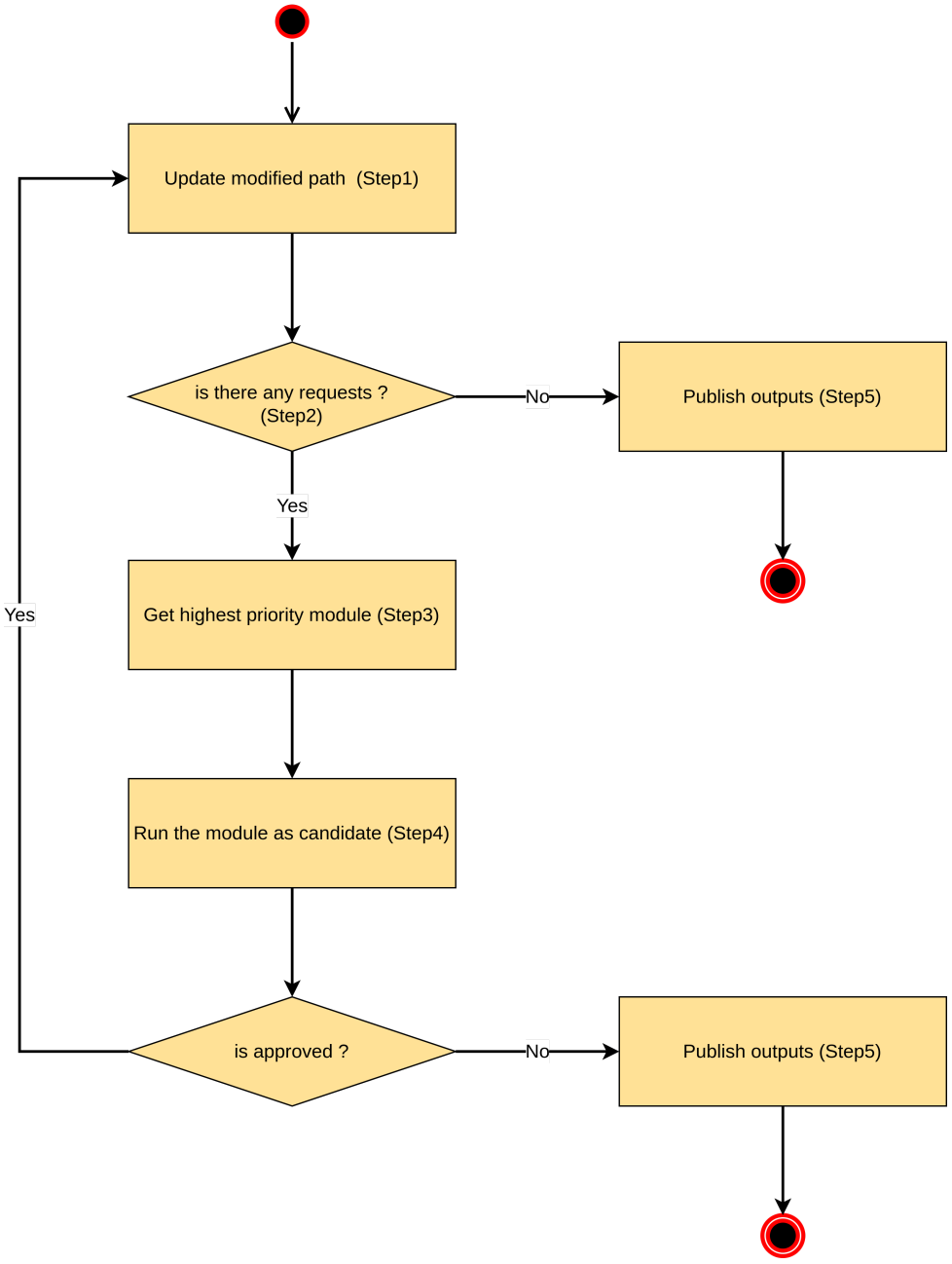
- there is no request of path modification. (check it in Step2)
- highest priority module has not approved. (check it in Step3)

Thus, dismissed requests in step3 may be selected and the module activated in the next iterate.

Finally, behavior path planner module outputs its modified path and candidate path in Step5. (See flowchart)



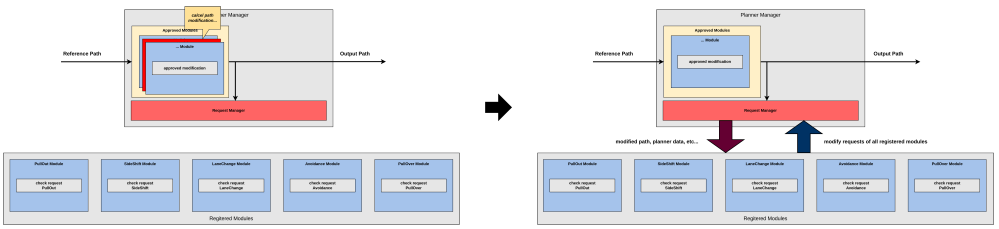
Flowchart



Abort/Expire approved module modification

All modules in Approved Modules Stack determine the decision on the need for a path modification based on the assumption that the approved module's outputs will not change drastically. However, there is a possibility that approved module (e.g. avoidance, lanechange) judges that the modified path is unsafe, and it may revert the changed path or generate another modified path.

In this situation, the paths output by the later modules may also change drastically, so the planner manager removes the module with the drastically path change and all the modules that are stacked at the back of the module. This process is execute in Step1.



Note

↑ 2 👍 4

6 comments · 17 replies

Oldest | Newest | Top

satoshi-ota on Dec 1, 2022 Maintainer Author

@mitsudome-r @yukkysaito @TakaHoribe @xmfcx

↑ 2

2 replies

satoshi-ota on Dec 5, 2022 Maintainer Author

@xmfcx Do you have any comments ? 😃

xmfcx on Dec 6, 2022 Maintainer

I will read it today!

👍 1

xmfcx on Dec 9, 2022 Maintainer

@satoshi-ota Can you explain what approving means?

> The RTC (Request To Cooperate) is a system that asks the operator's approve to change the path (RTC Status), and the operator checks the candidate path and approves the module's path modification (RTC Command). If the path modification is approved, the module outputs the path as Output Path.

But how often are these sent to an operator?

Or is there any other approving mechanism?

↑ 1

5 replies

xmfcx on Dec 9, 2022 Maintainer

If we sent every small modification in every small module sent to an operator, operator would have too many requests to evaluate.

satoshi-ota on Dec 13, 2022 Maintainer Author

Thanks for your comment 👍

> Or is there any other approving mechanism?

The approving mechanism already exist in autoware, and documentation is here.
Each module requests approval as soon as it needs **HUGE** path modification. (e.g. execute lane change, avoidance, pull over...)

Basically, we do **NOT** assume that it requests approval for a small change in routing, such as a 10 cm path shift.
In addition, auto approving mechanism also exist in current autoware, and we can use it instead of human operator if necessary.

TakaHoribe on Dec 14, 2022 Maintainer

@xmfcx The list of modules using the approval mechanism is:

- Lane change
- Avoidance with large path modification
- Pullover/out
- Intersection entry
- Traffic light stop/go
- crosswalk stop/go
- etc...
  Actually, it is almost all modules in `behavior_path_planner` and `behavior_velocity planner`.

The purpose of the approval mechanism is to increase safety and efficiency in the development. In detail, it is used for:

- make time for a safety operator to check the decision (whether it is really safe)
  - so the process would be "autoware requests a behavioral change, operator checks if the autoware's decision is safe and approves, then the vehicle behavior changes".
- discard unexpected requests, especially requests due to the low performance of the perception. We can then focus on the planning behavior in development.
- Gather data on human decisions.

By default setting, auto-approver exists and it approves every request as soon as it is requested. You can turn off the auto-approve if needed. It can be tested on the planning_simulator.

The concept of the approval should be summarized and writen somewhere in a document, but currently it is not. I'll update the planning design document for the approval.

satoshi-ota on Dec 17, 2022 Maintainer Author

We will wait for comments until next Tuesday 😃
If there are no major concerns, I will implement the prototype 🚀

xmfcx on Dec 19, 2022 Maintainer

Alright, thanks for all the information that you've provided @TakaHoribe and @satoshi-ota !

This change will help Autoware be more flexible and dropping the behavior-tree dependency will reduce the complexity and maybe even reduce the compile/linking times.

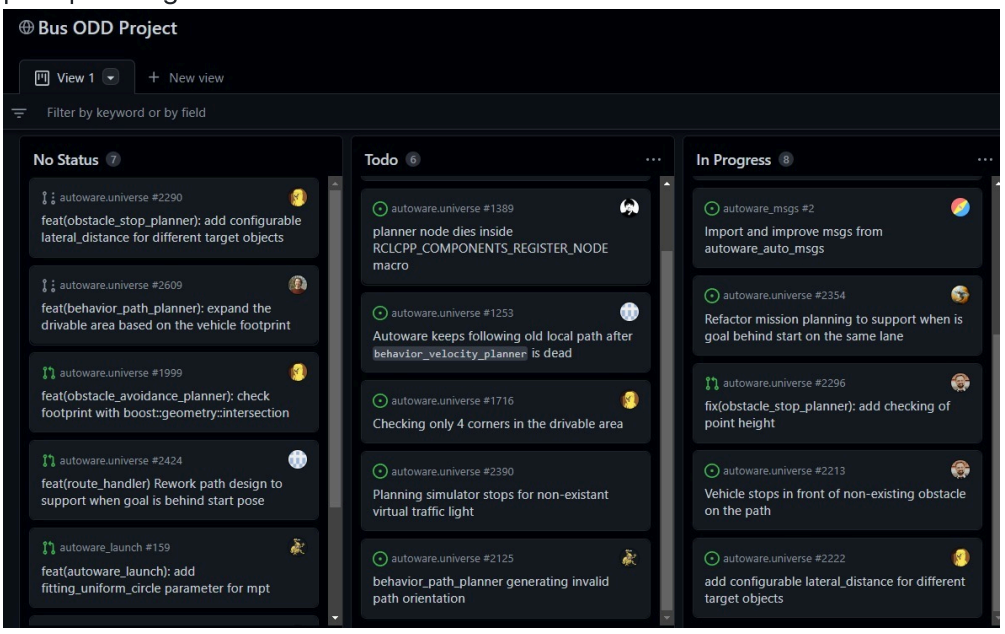
Thanks for working on this! (^ᵕ^)/

❤️ 1

armaganarsln on Jan 14, 2023 Collaborator

Thank you for all the work and i must say its really exiting to have new architectural changes and design discussions. Although its not my area i am a bit concerned of the timing of this proposal.

We are about to finish bus ODD in 3 weeks and there are a bunch of issues in path planning as below:



Also after bus odd, there is a plan to start the new ODD for 2023 with so many new challenges in path planning (because the new ODD includes highway multilane driving, roundabouts, underpasses etc.) and where maybe we will have to change or plan to change the architecture in 2-3 months time.

So i want to understand whether the timing is right for this new proposal or we should focus our efforts to finish the bus odd issues and propose the architecture considering the 2023 urban & freeway high traffic ODD.

↑ 1 4 replies

satoshi-ota on Jan 24, 2023 Maintainer Author

Hi @armaganarsln Thanks for your comment, and appologize for my late reply.

I think we should focus our efforts to finish the bus odd issues, but a few members in TIER IV, are working on developping the new architecture of planner. We TIER IV would like to implement new behavior path planner proto type that based on proposed architecture at 2023 3/E. So we still have few month to improve and fix current issues. In addition, we plan to keep the current behavior path planner we have been using, and we intend to have a transition phase from the previous architecture to the new architecture. cc **@TakaHoribe**

If you have any other concerns, feel free to comment 😄

armaganarsln on Jan 24, 2023 Collaborator

My main concerns are two:

1. There are so many path planning related issues some needs help: https://github.com/autowarefoundation/autoware.universe/milestone/9
2. I want everyone to think and decide on the new architecture when they have time.

Of course everyone free to concentrate on different topics but i think its not fair to the engineers who don't have time to comment on this new architecture module.

So could we postpone this item to next month or March and focus on bus odd related issues?

satoshi-ota on Jan 25, 2023 Maintainer Author

I got it. I would like to plan the new architecture developping, taking into account the impact on BUS ODD. cc **@mitsudome-r** **@xmfcx** **@angry-crab**

mitsudome-r on Jan 26, 2023 Maintainer

@armaganarsln
Thanks for the feedback.
I agree that we should wait to make the modification on the current package until Bus ODD project is settled. (or develop as a separate package so that it wouldn't disturb the project if TIER IV has urgent needs).

> Of course everyone free to concentrate on different topics but i think its not fair to the engineers who don't have time to comment on this new architecture module.

This is a valid point. However, this discussion has been there since December 1st. I've also mentioned it in one of the ASWG calls, but we only have Fatih and your reaction so far. If there are engineers who has interest but doesn't have time to look into the details, I would love to see them posting the comment that they want TIER IV to wait for discussion.(Just like you just did in your comment 😄) That way, we know that there are people who are interested in the topic and set a call to have a further discussion when they have time.

@satoshi-ota
Although we mentioned this discussion in the past ASWG, we only said that we will be working on prototype back then, but we never had chance to make conclusion about whether the new architecture is acceptable. Now that we have prototype implementation, it might be better if we have a call either in ASWG or as a separate call to provide your findings and make decision about whether we want to go with this new architecture or not.

satoshi-ota on Apr 7, 2023 Maintainer Author edited

@xmfcx @mitsudome-r @mehmetdogru @TakaHoribe
Hi, we TIER IV are working on developing new behavior path manager based on this proposal. I guess that few bugs still remain, but the new manager is already merged in main branch, and you can use that by switching flag. This manager makes it possible to execute multiple scene module simultaneously, for example, to execute avoidance maneuver during lane change (:arrow_down_small: movie) in some situation.

NOTE: BT based manager is used as default.

And, we TIER IV would like to use this manager in autoware as default in near future. Then, I want to ask you about what we should do before that. I suppose that we TIER IV should...

- [x] maintain and update documents about each modules and manager.
- [x] confirm that the new manager PASSes all **TIER IV internal** scenarios that BT based manager already passed.
- [x] confirm that the new manager PASSes all **AWF** scenarios.
- [x] decide whether we continue to support BT based manager.

I want your options about that.

- Do you have any suggestion or concern abot this manager update?
- When should we do that manager update? Can we update as soon as possible if the above criterion are satisfied?

Please feel free to left comments.

simplescreenrecorder-2023-04-07_20.13.01.mp4

↑ 1 ❤️ 1 3 replies

mehmetdogru on Apr 10, 2023 Maintainer

@satoshi-ota Thank you for working on this!

In my opinion, we can make the new manager default one as soon as it passes all the scenarios as you mentioned. After doing this it is a good idea to support BT based manager as well for sometime (duration to be determined) to have a back-up plan just in case if we will have major issues with the new one in different/more complex real life scenarios than the ones we have for simulations. And so hopefully we can complete the manager update transition without any glitch.

satoshi-ota on Apr 10, 2023 Maintainer Author

Hi @mehmetdogru Thanks for your positive comments.

> In my opinion, we can make the new manager default one as soon as it passes all the scenarios as you mentioned. After doing this it is a good idea to support BT based manager as well for sometime (duration to be determined) to have a back-up plan just in case if we will have major issues with the new one in different/more complex real life scenarios than the ones we have for simulations.

I completely agree with you. Since this is a large update, we should have an alternative plan to use BT based manager in case problems arise, as you said. On the other hand, I think we also need to be careful to decrease maintenance costs since multiple managers coexits.

First and foremost, I think it is important for everyone to know how the manager works, so I will make it a priority to maintain the documentation about that.

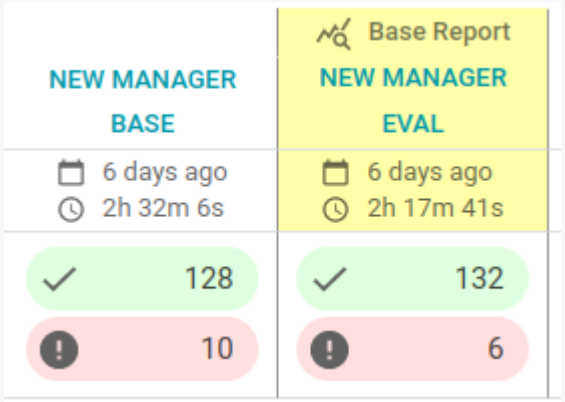
And, I will wait a bit for other member's opinion.

👍 1

satoshi-ota on Jun 22, 2023 Maintainer Author

AWF SCENARIO TEST RESULTS

- **NEW MANAGER EVALUATION RESULT: 131/138 (ran 2023/06/13)**
- BASE LINE EVALUATION: 128/138 (ran 2023/06/13)



I confitmed that new behavior path planner achieve most of all scenarios in awf catalog. And basically, all of the failed scenario that new behavior path planner can't pass are failed on old behavior path planner, too. Then, I think those fails don't related to behavior path planner architecture, and we can transit to new behavior path planner manager. What do you think?

satoshi-ota on Apr 13, 2023 Maintainer Author edited

How to use the new manager?

For now, we can switch the NEW and BT based manager by flag in Cmake file. And, BT based is used by default.
If we wanna try to use the new one, please build with flag `COMPILE_WITH_OLD_ARCHITECTURE` setting `FALSE`.

```
cmake_minimum_required(VERSION 3.14)
project(behavior_path_planner)

find_package(autoware_cmake REQUIRED)
autoware_package()

find_package(OpenCV REQUIRED)
find_package(magic_enum CONFIG REQUIRED)

set(COMPILE_WITH_OLD_ARCHITECTURE TRUE)
...
```

What can we do on the new manager?

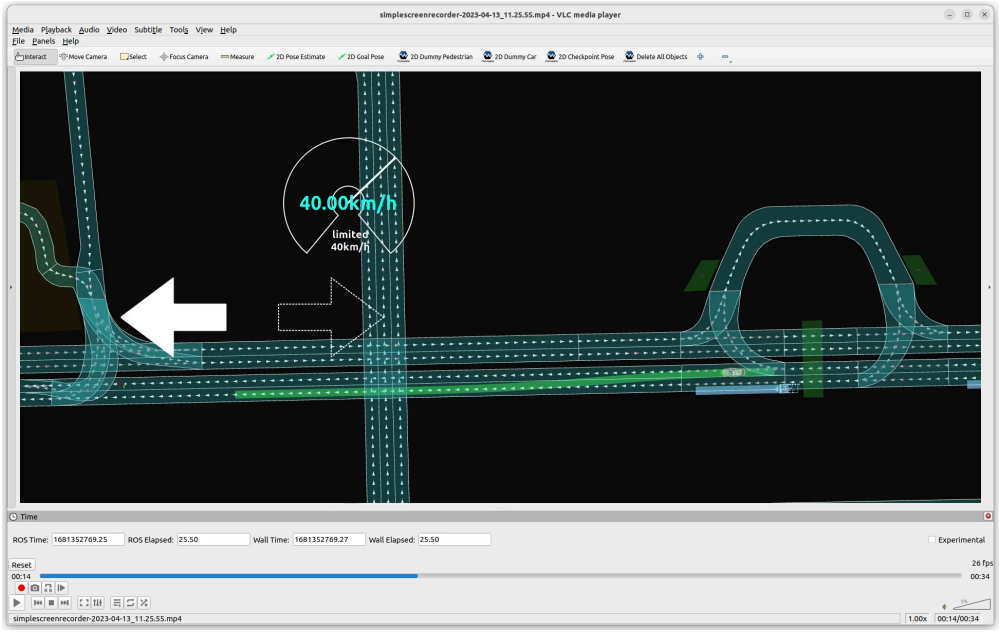
Run two scenarios to compare the behavior of NEW manager with BT based one.

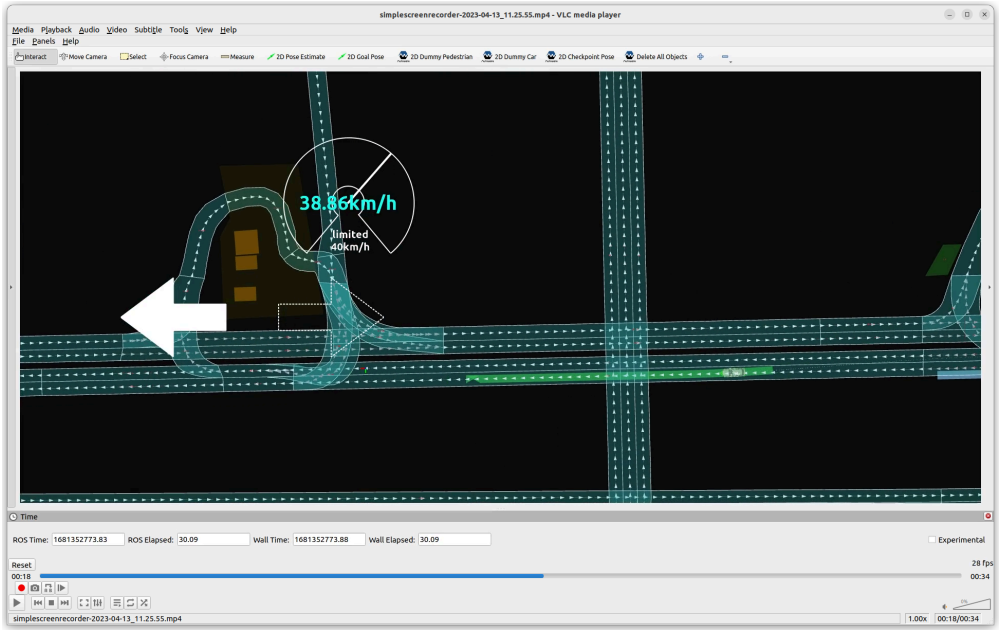
Scenario1. need to execute lane change in avoidance manauver.

BT based manager

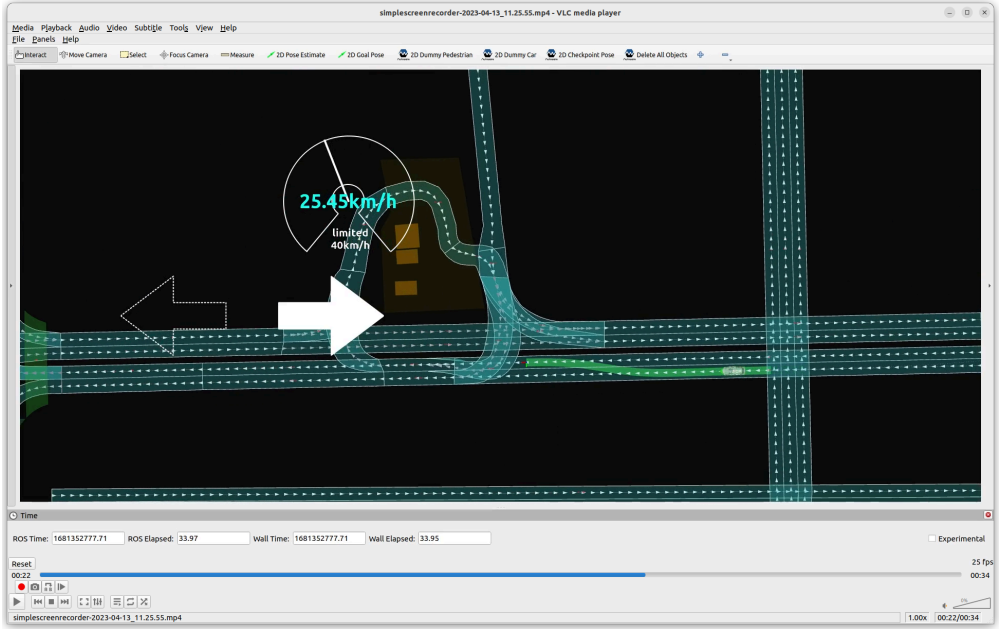
Since BT based manager execute all module execlusively, lane change module can't run while avoidance module is running. Then, lane change module has to wait the end of avoidance module process.

This issue may cause vehicle stuck and over ride (OR) in worst case. For example, in following scene, lane change can do anything despite of the fact that the goal is on the right side lane and the ego is getting closer.





In the end, the remaining distance for lane change quite a bit short.



new manager

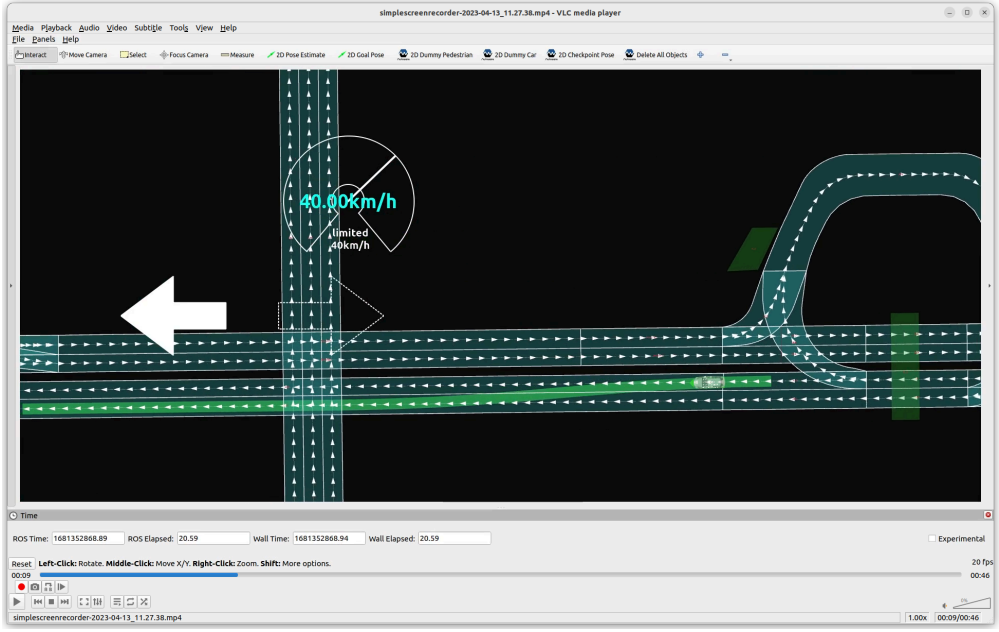
On the other hand, new manager can execute lane change module as soon as the avoidance module decides the output path. So, the ego can execute lane change from the middle of the avoidance path.

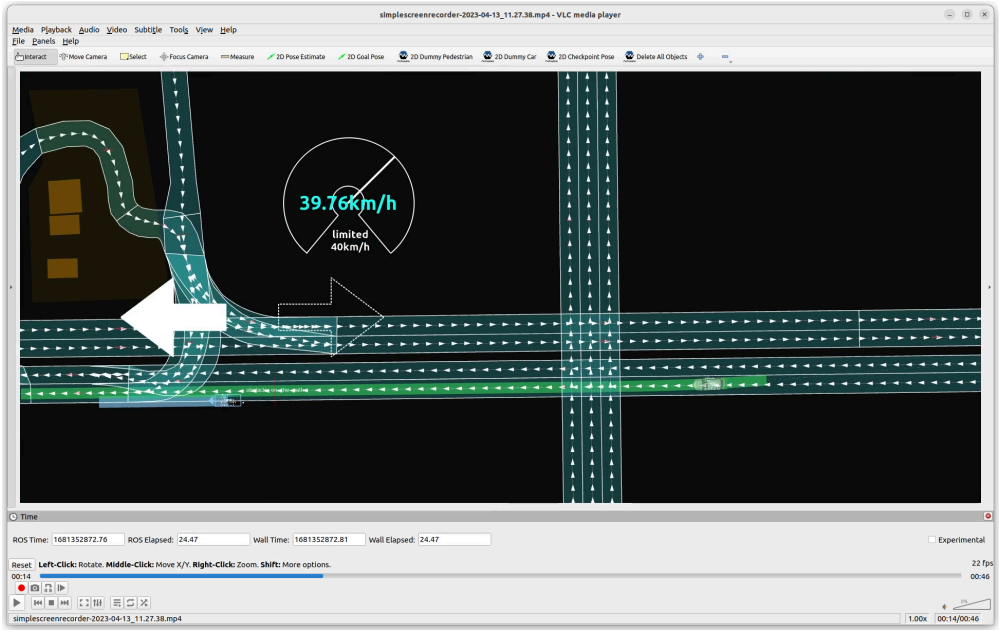
simplescreenrecorder-2023-04-13_11.20.20.mp4

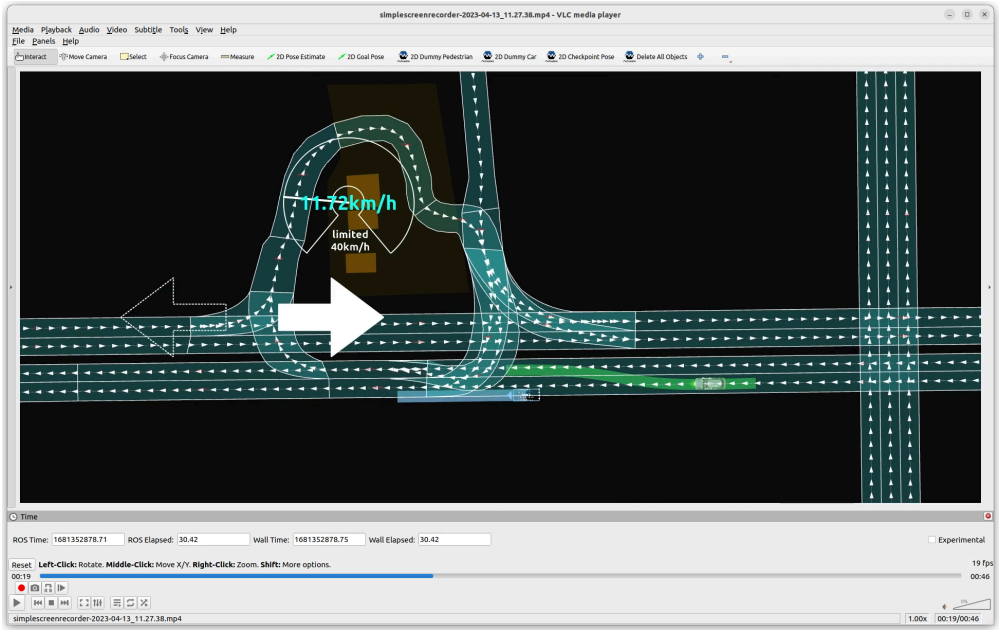
Scenario2. need to execute avoidance in lane change maneuver.

BT based manager

As mentioned above, avoidance module can't run while the lane change module is running on BT based manager. In the following scene, the avoidance module do nothing despite of the fact that perception module already detected a parked-vehicle on lane change target lane. And, the avoidance module start running after the ego get quite a bit closer to the vehicle.

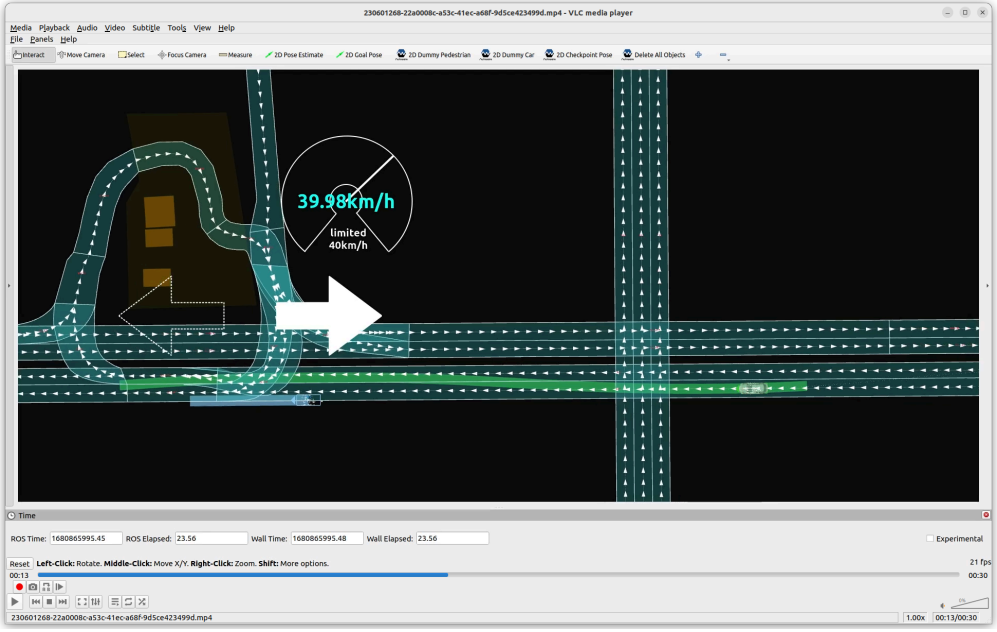




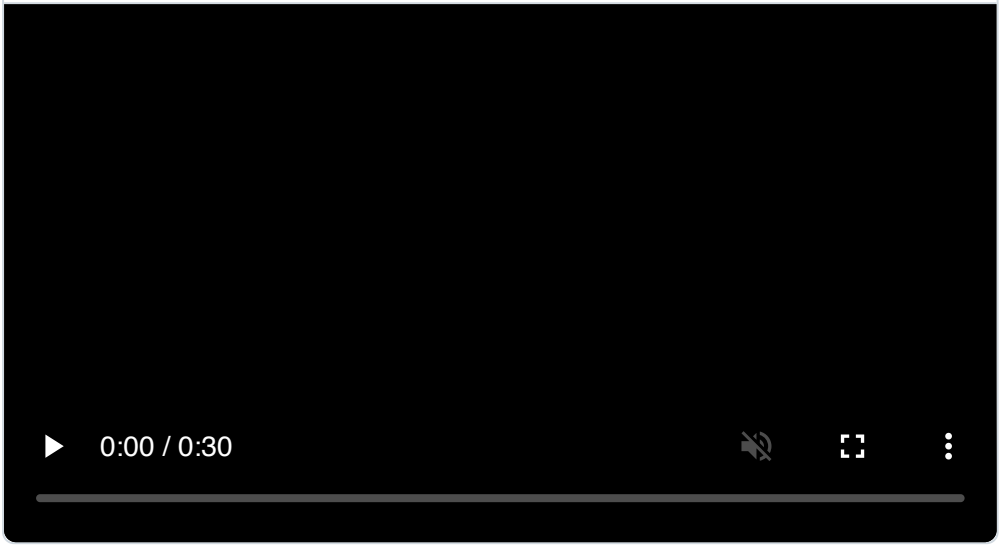


new manager

Avoidance module can run as soon as the perception module detects objects that should be avoid even if it is the lane change maneuver.



230601268-22a0008c-a53c-41ec-a68f-9d5ce423499d.mp4



↑ 1

0 replies

sisaha9 on Jun 22, 2023 edited

@satoshi-ota Could you expand on behavior differences that would occur with this new behavior planner that can't be achieved by running the old behavior tree architecture but having each node in the behavior tree running a separate action client. This would be very similar to how Navigation2 achieves it and would allow for running nodes without blocking each other (by returning the BT::Running instead of Success or Failure) but still maintain the nice UI of Behavior Tree

↑ 1

3 replies

sisaha9 on Jun 22, 2023 edited

> Hierarchies between modules can not be changed in runtime, which can not achieve flexible path generation according to use cases

I guess this point would still be tricky though I could see a way of auto generating the string of an XML file based off current priorities and using the `createTreeFromText` function to change the tree during operation

satoshi-ota on Jun 22, 2023 Maintainer Author

Hi **@sisaha9** ✋

Thanks for asking and your advice 👍 cc **@TakaHoribe**

> Could you expand on behavior differences that would occur with this new behavior planner that can't be achieved by running the old behavior tree architecture but having each node in the behavior tree running a separate action client.

Actually, we think it is not impossible to construct the logic to achieve complex scenarios on Behavior Tree, as you said.

> I guess this point would still be tricky though I could see a way of auto generating the string of an XML file based off current priorities and using the createTreeFromText file to change the tree during operation

However, if we implement all logic for what we want to do on BT (e.g. change the order of execution between modules according to the situation), the tree would become very complicated and difficult to maitain. Additionaly, we think we can't say that it is definitely appropriate to use BT as an interface for modules, and we have decided to allow designing module and manager specifications without being bound by the BT configuration at one time.

sisaha9 on Jun 22, 2023

I see. Thanks for the clarification **@satoshi-ota**

👍 1