

Arquitetura HIL para teste de sistemas embarcados como *vehicle interface* de veículos autônomos baseados no Autoware

Projeto – Etapa 4

Gabriel Toffanetto França da Rocha

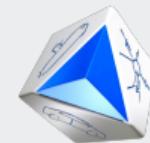
g289320@dac.unicamp.br

Professor Dr. Rodrigo Moreira Bacurau

IM420X – Projeto de Sistemas Embarcados de Tempo Real

Faculdade de Engenharia Mecânica
Universidade Estadual de Campinas

26 de novembro de 2024



Agenda

1 Introdução

2 Proposta

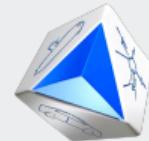
3 Resultados

4 Conclusão

5 Apêndices



Introdução



Contextualização



Figura 1: Veículo Autônomo do LMA.

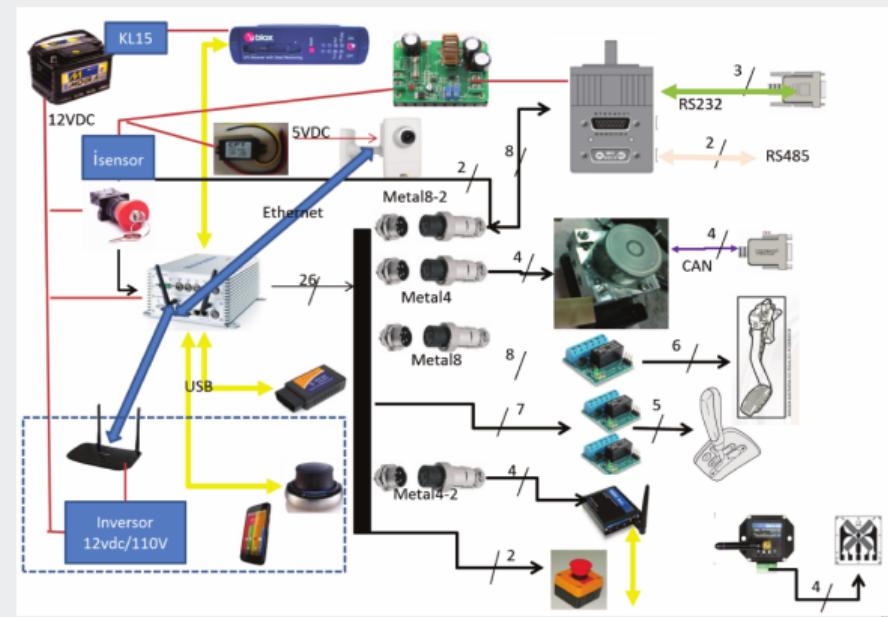


Figura 2: Diagrama de hardware do VILMA01 (??).

Autoware



Autoware

Figura 3: Logomarca do Autoware.



Autoware



Autoware

Figura 3: Logomarca do Autoware.

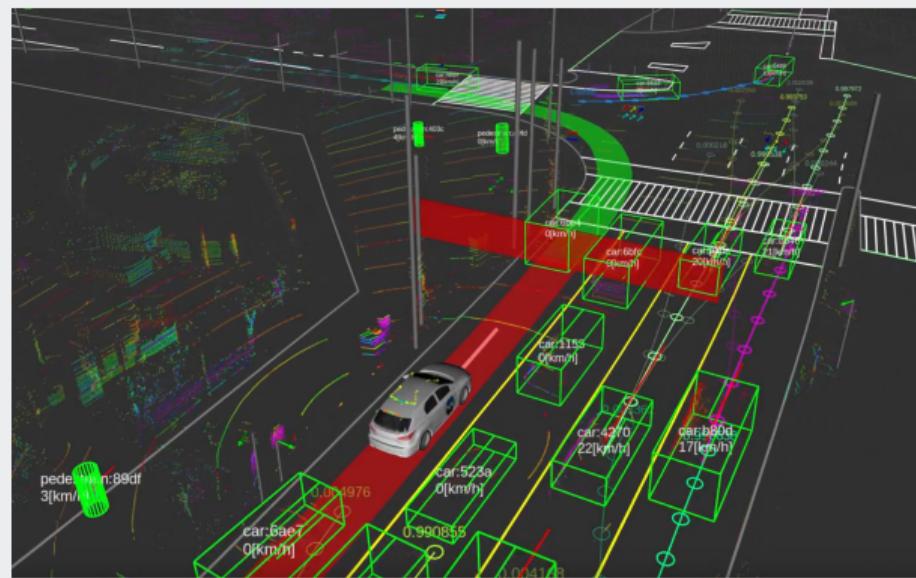


Figura 4: Visualização do Autoware em operação (??)

Arquitetura

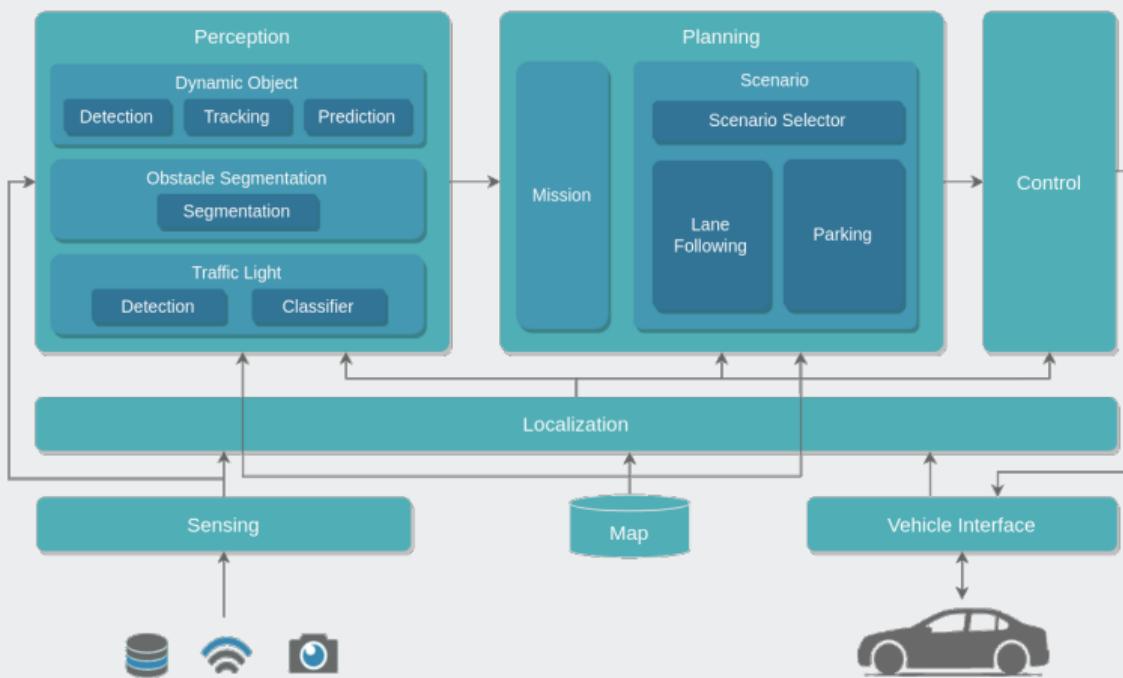
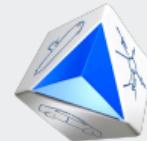
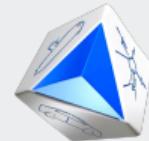


Figura 5: Arquitetura de alto nível (??).

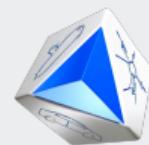


Proposta



Proposta

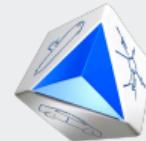
Trazer o Autoware para dentro do microcontrolador de forma confiável e de fácil implementação.



Proposta



Figura 6: Ferramentas para viabilização do projeto.



Proposta

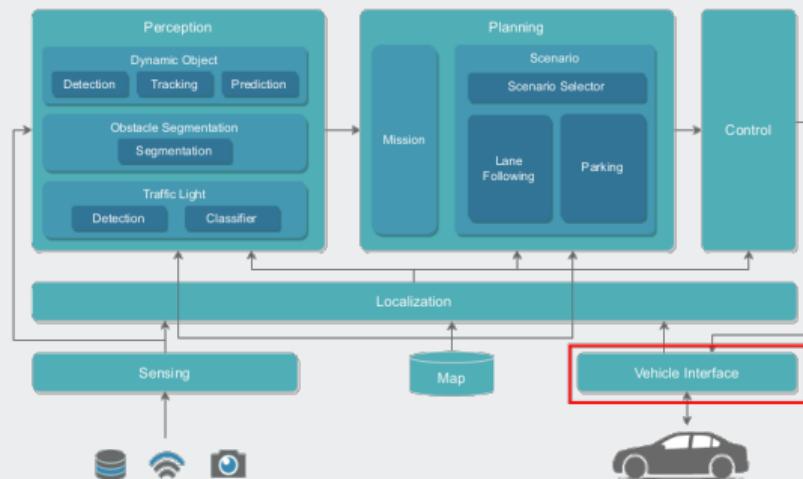
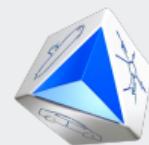


Figura 7: Escopo do projeto na arquitetura Autoware.



Proposta

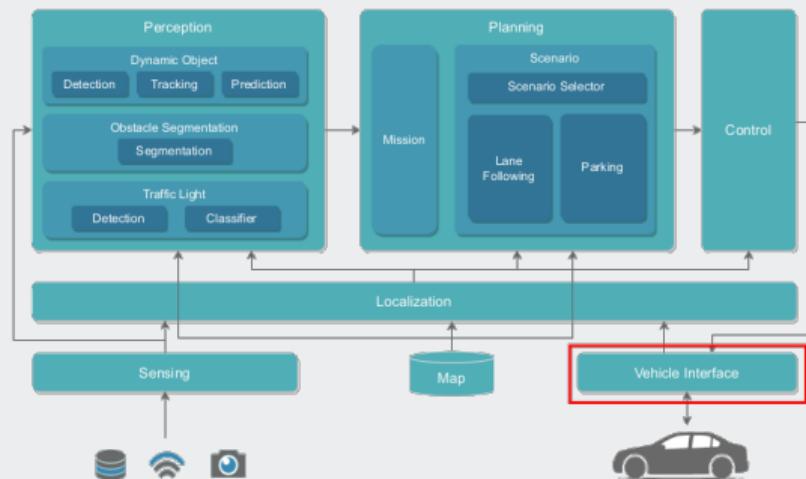


Figura 7: Escopo do projeto na arquitetura Autoware.

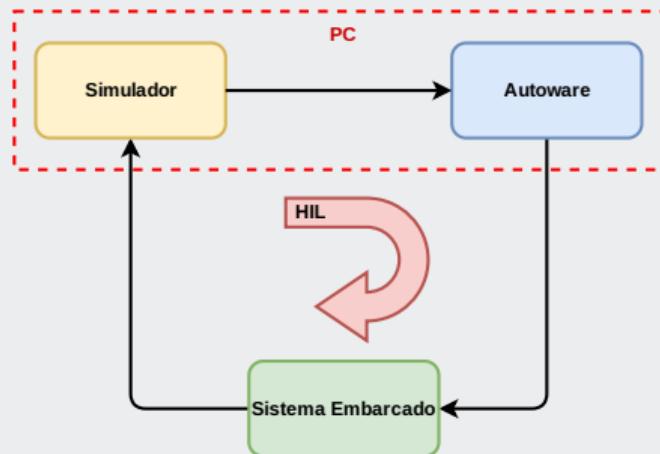


Figura 8: Arquitetura de teste do *hardware*.

Vehicle interface

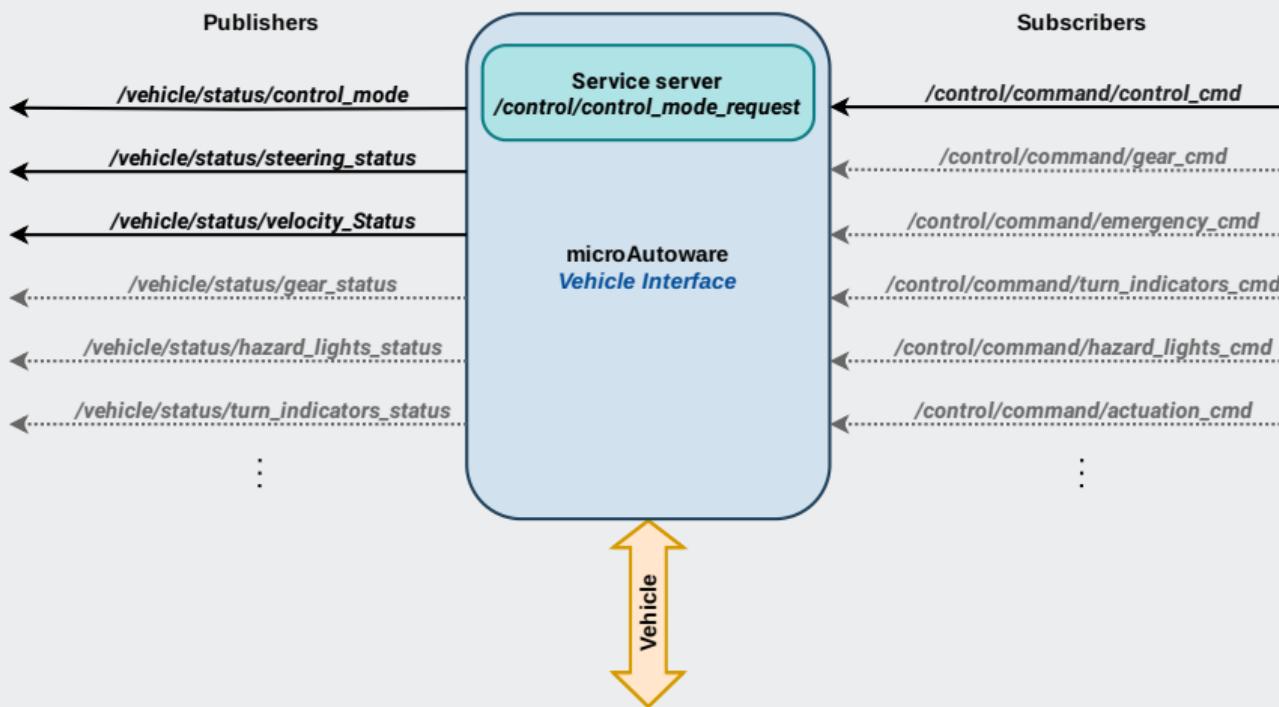
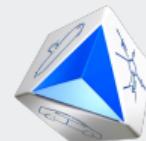


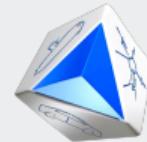
Figura 9: Diagrama de tópicos da *vehicle interface*.



Requisitos

Requisitos funcionais

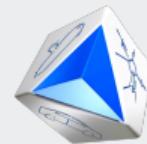
- Comunicação com o Autoware;
- Controle da aceleração, frenagem e direção do veículo;
- Controle dos faróis e luzes de sinalização (seta) do veículo;
- Teleoperação do veículo por um *joystick* em *hardware*;
- Troca do modo de operação por meio da *switch* do *joystick*;
- Subscrição por meio do micro-ROS em todos os tópicos necessários do Autoware;
- Publicação a partir micro-ROS em todos os tópicos necessários do Autoware;
- Comunicação a partir micro-ROS em todos os serviços necessários do Autoware.



Requisitos

Requisitos funcionais

- Comunicação com o Autoware;
- Controle da aceleração, frenagem e direção do veículo;
- Controle dos faróis e luzes de sinalização (seta) do veículo;
- Teleoperação do veículo por um *joystick* em *hardware*;
- Troca do modo de operação por meio da *switch* do *joystick*;
- Subscrição por meio do micro-ROS em todos os tópicos necessários do Autoware;
- Publicação a partir micro-ROS em todos os tópicos necessários do Autoware;
- Comunicação a partir micro-ROS em todos os serviços necessários do Autoware.



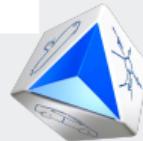
Requisitos

Requisitos funcionais

- Comunicação com o Autoware;
- Controle da aceleração, frenagem e direção do veículo;
- Controle dos faróis e luzes de sinalização (seta) do veículo;
- Teleoperação do veículo por um *joystick* em *hardware*;
- Troca do modo de operação por meio da *switch* do *joystick*;
- Subscrição por meio do micro-ROS em todos os tópicos necessários do Autoware;
- Publicação a partir micro-ROS em todos os tópicos necessários do Autoware;
- Comunicação a partir micro-ROS em todos os serviços necessários do Autoware.

Requisitos não-funcionais

- A *vehicle interface* deve ser construída na forma de um pacote portátil para outros microcontroladores STM32;
- O interfaceamento com o veículo deve ser intercambiável com diferentes configurações;
- Deve-se garantir sincronização de *timestamp* entre o Autoware e o microcontrolador;
- O sistema embarcado deve abstraír o veículo como um sistema *Drive-By-Wire* (DBW) para o Autoware.



Requisitos

Requisitos funcionais

- Comunicação com o Autoware;
- Controle da aceleração, frenagem e direção do veículo;
- Controle dos faróis e luzes de sinalização (seta) do veículo;
- Teleoperação do veículo por um *joystick* em *hardware*;
- Troca do modo de operação por meio da *switch* do *joystick*;
- Subscrição por meio do micro-ROS em todos os tópicos necessários do Autoware;
- Publicação a partir micro-ROS em todos os tópicos necessários do Autoware;
- **Comunicação a partir micro-ROS em todos os serviços necessários do Autoware.**

Requisitos não-funcionais

- A *vehicle interface* deve ser construída na forma de um pacote portátil para outros microcontroladores STM32;
- O interfaceamento com o veículo deve ser intercambiável com diferentes configurações;
- Deve-se garantir sincronização de *timestamp* entre o Autoware e o microcontrolador;
- O sistema embarcado deve abstraír o veículo como um sistema *Drive-By-Wire* (DBW) para o Autoware.

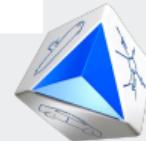


Diagrama de blocos

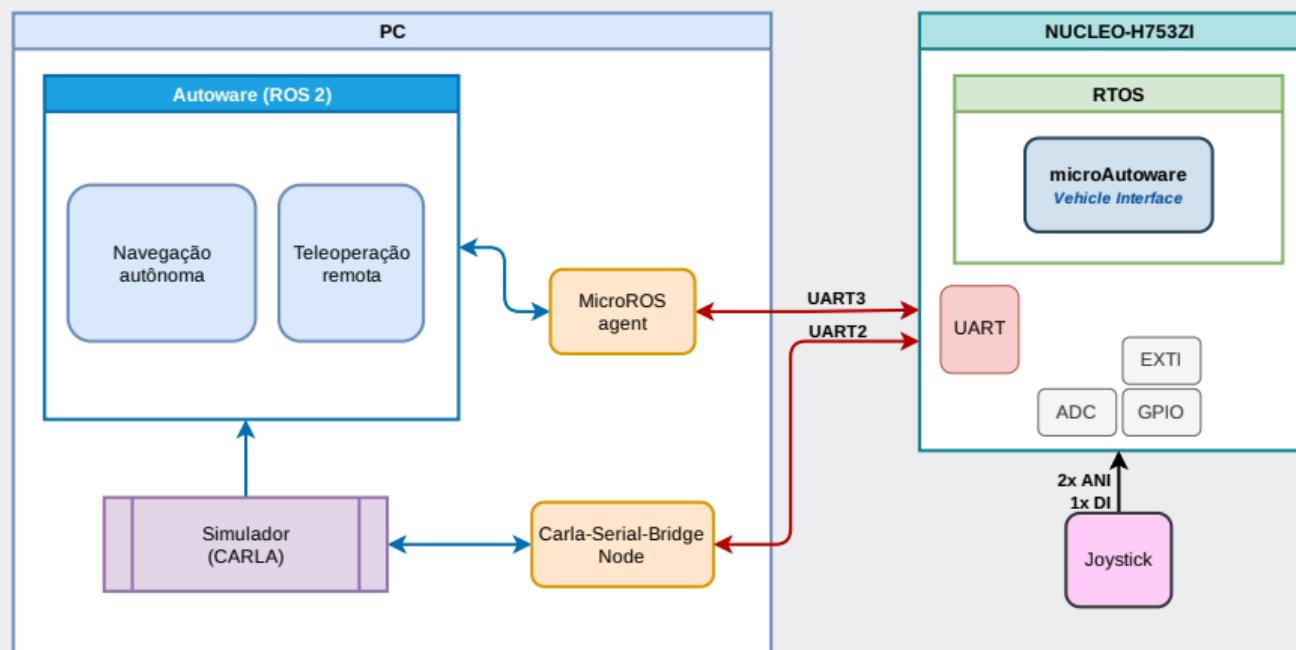
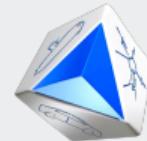
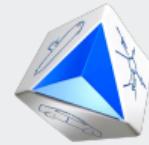


Figura 10: Diagrama de blocos atualizado.



Resultados



Inicialização do Autoware

Inicialização do Autoware

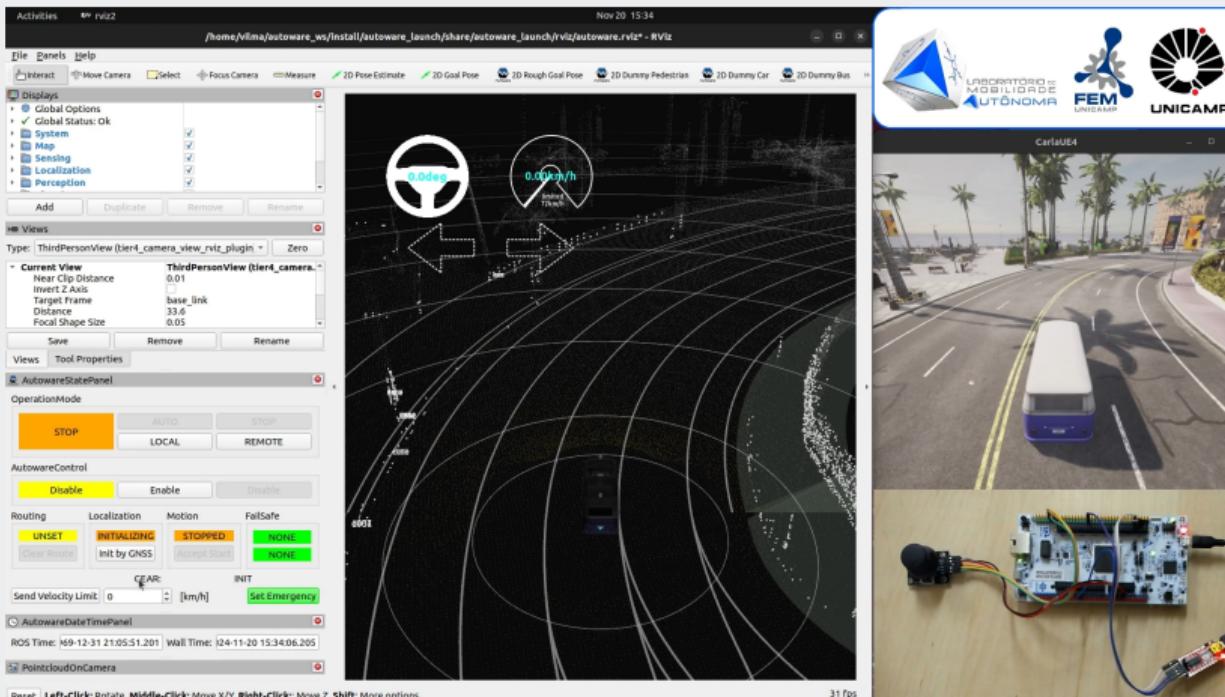


Figura 11: Inicialização.

Controle manual

Controle manual

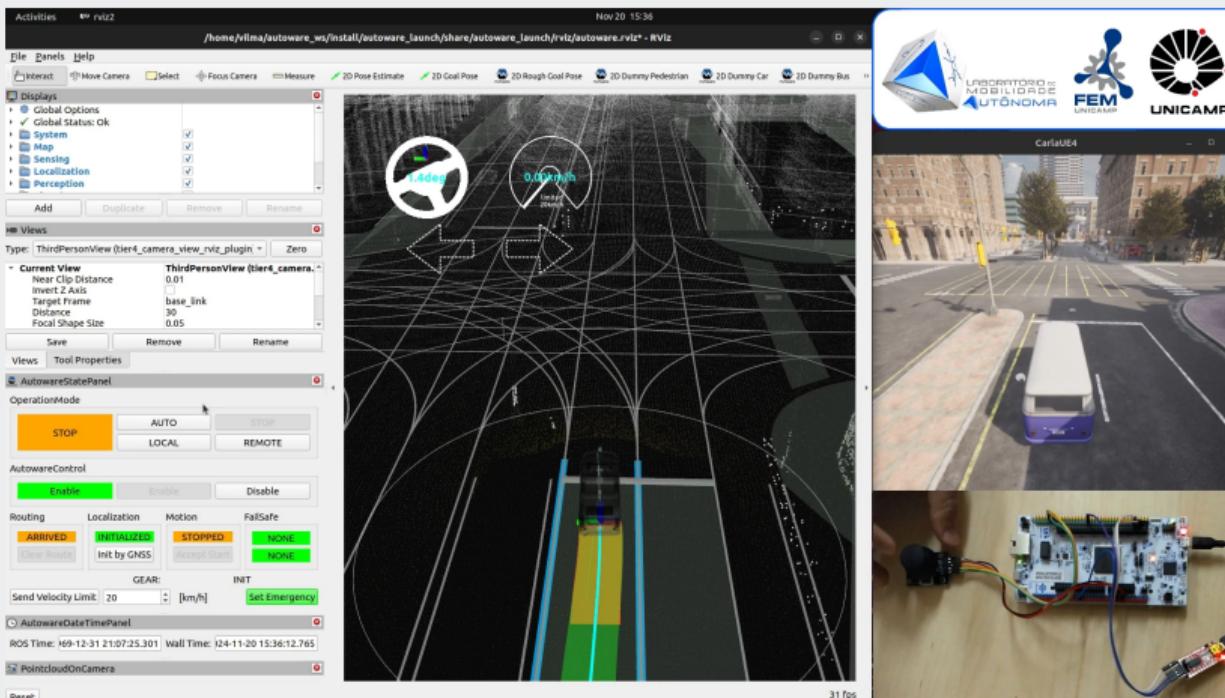


Figura 12: Controle manual.

Controle autônomo

Controle autônomo

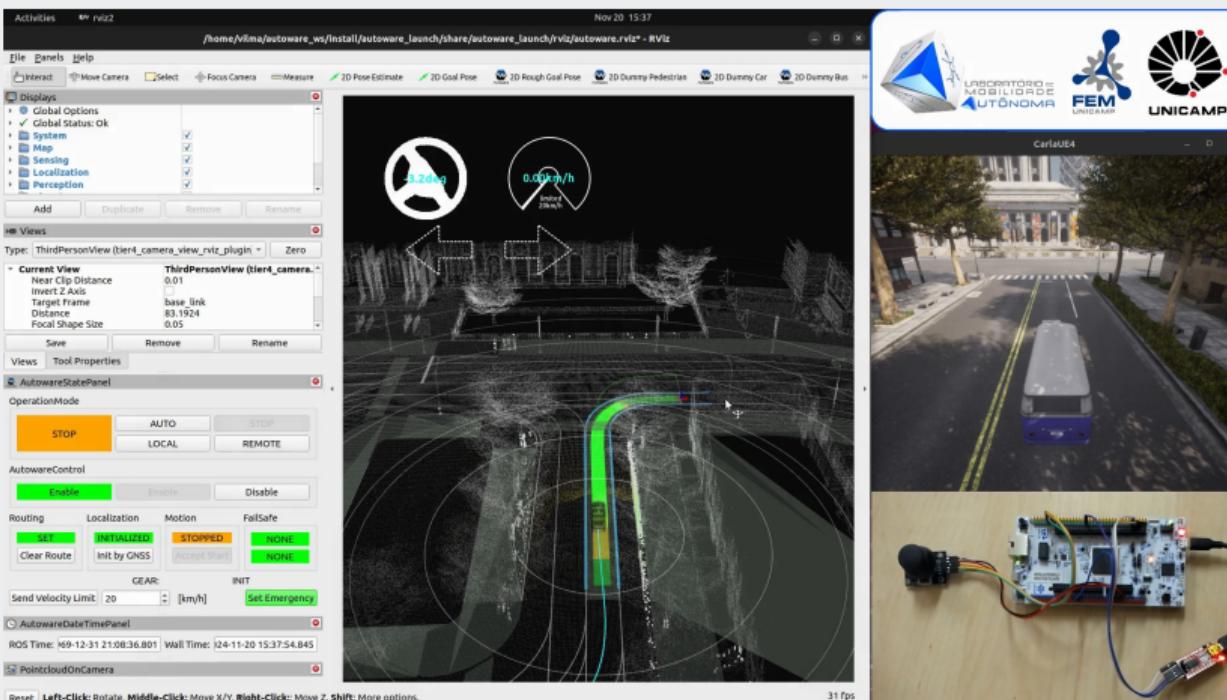


Figura 13: Controle autônomo.

Trocade modo de controle

Trocade modo de controle – Joystick

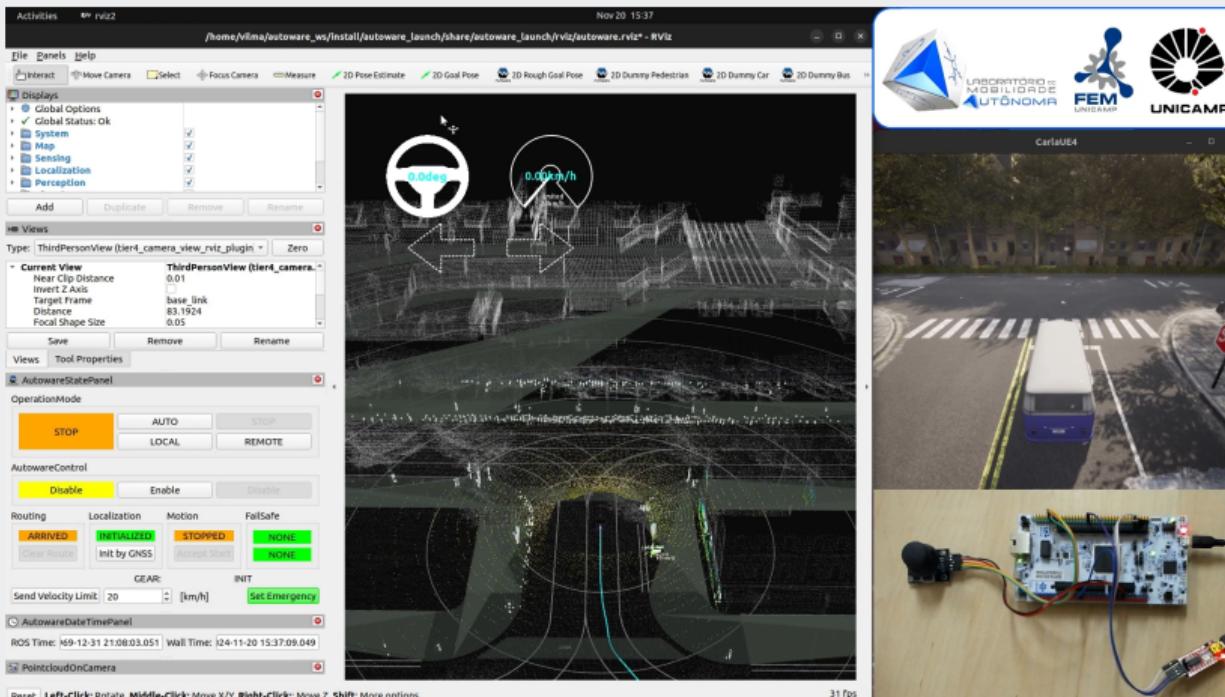


Figura 14: Troca do modo de controle pelo joystick.

Troca de modo de controle

Troca de modo de controle – Autoware

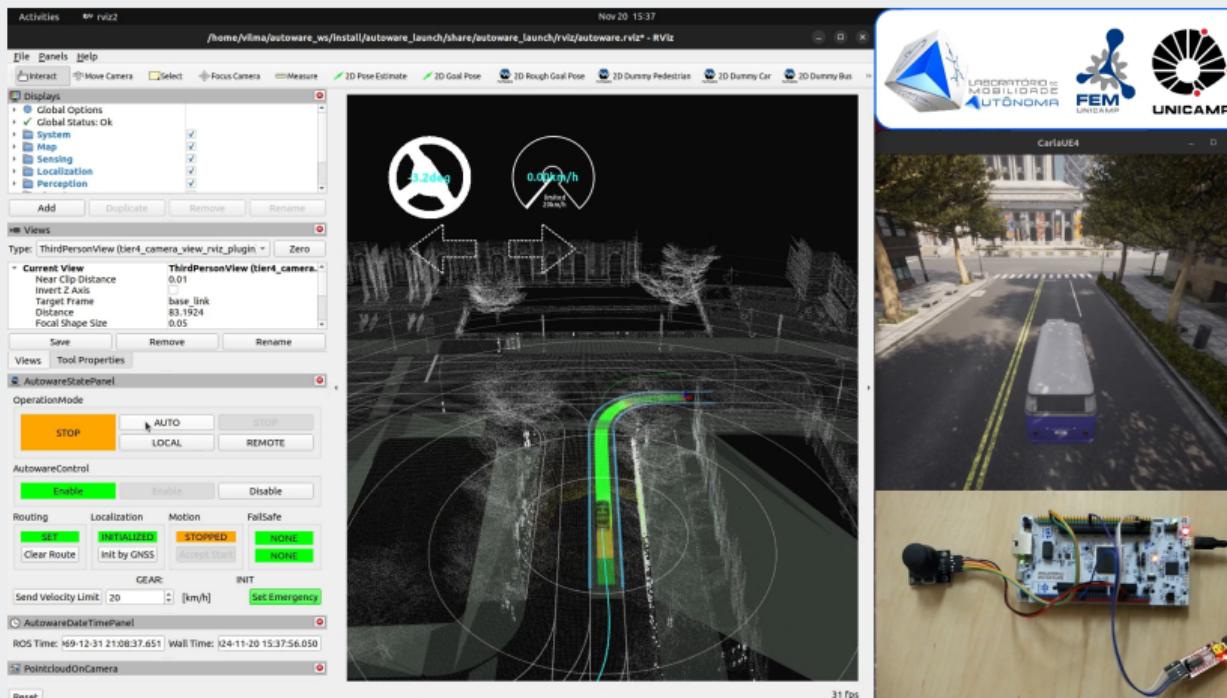


Figura 15: Tomada de condução para modo manual e troca de modo de operação pelo Autoware.

Fail-safe: perda comunicação com Autoware



Modo de emergência

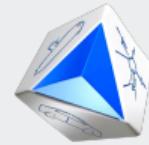
Modo de emergência: Perda de sinais do veículo



Repositório no GitHub



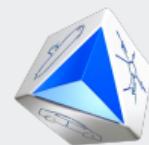
Conclusão



Conclusão

Problemas encontrados

- Perda de resposta do serviço
`control_mode_request;`



Conclusão

Problemas encontrados

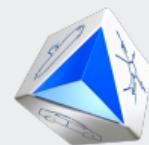
- Perda de resposta do serviço
`control_mode_request;`
- Perda de deadline causada pela instabilidade do recebimento de comandos;



Conclusão

Problemas encontrados

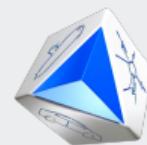
- Perda de resposta do serviço
`control_mode_request;`
- Perda de deadline causada pela instabilidade do recebimento de comandos;
 - Acentuado quando o PC é sobrecarregado.



Conclusão

Problemas encontrados

- Perda de resposta do serviço
`control_mode_request;`
- Perda de deadline causada pela instabilidade do recebimento de comandos;
 - Acentuado quando o PC é sobrecarregado.
 -



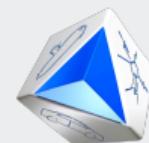
Conclusão

Problemas encontrados

- Perda de resposta do serviço control_mode_request;
- Perda de deadline causada pela instabilidade do recebimento de comandos;
 - Acentuado quando o PC é sobrecarregado.
-

Trabalhos futuros

- Portabilidade do microAutoware para outros microcontroladores;



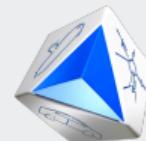
Conclusão

Problemas encontrados

- Perda de resposta do serviço control_mode_request;
- Perda de deadline causada pela instabilidade do recebimento de comandos;
 - Acentuado quando o PC é sobrecarregado.
-

Trabalhos futuros

- Portabilidade do microAutoware para outros microcontroladores;
- Implementação de bibliotecas auxiliares para controle de acelerador e freio;



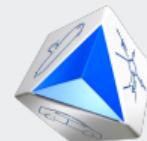
Conclusão

Problemas encontrados

- Perda de resposta do serviço control_mode_request;
- Perda de deadline causada pela instabilidade do recebimento de comandos;
 - Acentuado quando o PC é sobrecarregado.
-

Trabalhos futuros

- Portabilidade do microAutoware para outros microcontroladores;
- Implementação de bibliotecas auxiliares para controle de acelerador e freio;
- Compatibilidade com diferentes periféricos de comunicação com o agente do micro-ROS;
 - e.g. Ethernet.



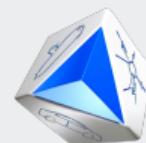
Conclusão

Problemas encontrados

- Perda de resposta do serviço control_mode_request;
- Perda de deadline causada pela instabilidade do recebimento de comandos;
 - Acentuado quando o PC é sobrecarregado.
-

Trabalhos futuros

- Portabilidade do microAutoware para outros microcontroladores;
- Implementação de bibliotecas auxiliares para controle de acelerador e freio;
- Compatibilidade com diferentes periféricos de comunicação com o agente do micro-ROS;
 - e.g. Ethernet.
- Protocolos de detecção de erros.



Conclusão

Problemas encontrados

- Perda de resposta do serviço `control_mode_request`;
- Perda de deadline causada pela instabilidade do recebimento de comandos;
 - Acentuado quando o PC é sobrecarregado.
-

Trabalhos futuros

- Portabilidade do microAutoware para outros microcontroladores;
- Implementação de bibliotecas auxiliares para controle de acelerador e freio;
- Compatibilidade com diferentes periféricos de comunicação com o agente do micro-ROS;
 - e.g. Ethernet.
- Protocolos de detecção de erros.

- Foi possível integrar ao sistema embarcado a função de *vehicle interface* para o Autoware.

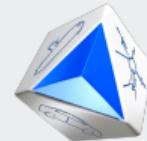


Cronograma

| Atividade/Semana | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|--|---|---|---|---|---|---|---|---|---|
| Proposta do projeto | | ■ | | | | | | | |
| Projeto de <i>hardware e software</i> | | | ■ | ■ | | | | | |
| Integração do STM com o micro-ROS | | | ■ | | | | | | |
| Integração do micro-ROS com o Autoware | | | | ■ | ■ | | | | |
| Implementação das tarefas do sistema embarcado | | | | | ■ | ■ | ■ | | |
| Construção do ambiente de testes | | | | | | ■ | ■ | | |
| Realização dos testes | | | | | | | ■ | ■ | |
| Escrita do relatório | | ■ | ■ | ■ | ■ | ■ | ■ | ■ | |

Tabela 1: Cronograma de atividades.

- Semana 2: Apresentação Etapa 1
- Semana 4: Apresentação Etapa 2
- Semana 7: Apresentação Etapa 3
- Semana 9: Apresentação Final



Apêndices



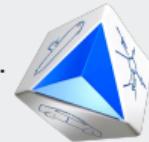
Apêndices



Figura 16: Vídeo de validação do microAutoware na Integra.



Figura 17: Repositório do microAutoware no GitHub.



Obrigado!

Dúvidas?

