V Simpósio de Iniciação Científica 2022
Ciência, Tecnologia e Inovação no Brasil

DESENVOLVIMENTO DE SISTEMA WEB PARA DIAGNÓSTICO DE VEÍCULO AUTÔNOMO E TELE-OPERAÇÃO VIA APP

José Gabriel de Jesus Flores[1] (IC), Giovani Bernardes Vitor (PQ) [1], Rafael Francisco dos Santos(PQ) [1], Willian Gomes de Almeida (PQ) [1]
[1]*Universidade Federal de Itajubá – Campus Itabira*

Palavras-chave: Robótica autônoma, Sistemas inteligentes, Software.

Introdução

Com o aprimoramento da tecnologia ao longo dos anos, diversas tarefas diárias estão cada vez mais fáceis, devido a automatização das máquinas. Partindo desse princípio, um carrinho de golfe foi utilizado como base para um veículo autônomo. Automóvel esse que reconhece o terreno em sua volta, identificando cada objeto durante o seu percurso, realizando decisões inteligentes ao ser capaz de evitar colisões e navegar rotas sem a interferência humana.

De modo a focar na interligação dos comandos, sensores instalados no veículo que se comunicam com a placa de controle central, a pesquisa justifica-se pelo intuito de aperfeiçoar os possíveis déficits funcionais do veículo autônomo diante das limitações operacionais.

Sendo assim, a pesquisa foi constituída por duas etapas, a princípio, através de uma revisão sistemática da literatura teórica – conceitual, de artigos, monografias e trabalhos científicos, a qual buscou conceitos sobre o *framework* ROS2 sobre a perspectiva da robótica autônoma e de quais são os dispositivos que teriam potencial para integrar esta tecnologia. Com filtragem de informações por entre uma análise comparativa de dados, visando as aplicações mais relevantes para a implementação no carrinho de golfe.

Posteriormente a essa filtragem, a equipe se dividiu em dois ramos, qualificando a segunda etapa do projeto. Enquanto os demais membros se voltam à parte de *hardware* do veículo, tal como posicionamento dos sensores, desenvolvimento de suportes para os dispositivos eletrônicos e placas de circuitos elétricos de *driver*.

A outra, a qual o resumo se desenvolverá, parte da construção de uma interface de diagnóstico que realiza o monitoramento ao vivo de todos os parâmetros relevantes e da condição de estado presente do carrinho de golfe. De modo, a tornar possível visualizar esses dados do veículo de forma remota, por meio de uma página web local.

Em sequência, foi criado um aplicativo *android* capaz de controlar com precisão os movimentos lineares e angulares do transporte através de uma interface de celular contendo um *joystick*. Para realizar a interconexão destes dispositivos, foram utilizadas bibliotecas de comunicação *socket*, de modo que para validar as funcionalidades deste aplicativo realizou-se testes de autenticação em ambientes simulados e no próprio veículo.

Metodologia

Desenvolvimento do módulo de diagnóstico *web*.

Tendo como princípio, que um veículo autônomo tem como aplicação a capacidade de coletar o dado de seu ambiente e, de forma simultânea, definir sua localização, direção e a detecção de obstáculos, a fim de alcançar seu percurso com segurança. Uma interface foi desenvolvida com o propósito de acompanhar as informações da situação, em tempo real, do carrinho de golfe.

Este sistema, chamado de “Golfinho Diagnostics”, preliminarmente, faz uso da tecnologia Robot Operating System 2 (ROS2). *Software* este determinado, pela sua quantidade de recursos disponíveis voltado para a criação de aplicativos robóticos, nesse caso no desenvolvimento de um sistema de condução automatizada, no qual proporciona a sua execução ao vivo. A Figura 1 apresenta a arquitetura do módulo de diagnóstico *web*.

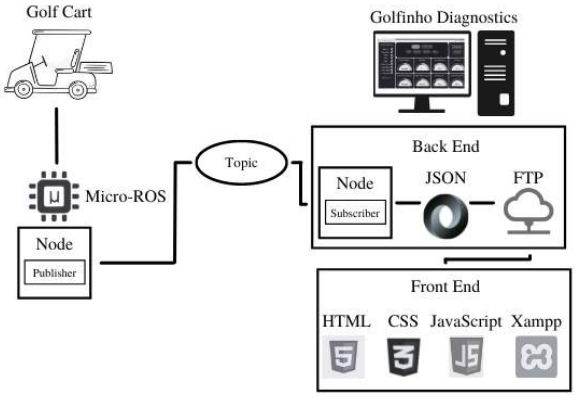

Figura 1 – Arquitetura do módulo de diagnóstico *web*. Fonte: Autor.

Assim sendo, a arquitetura da interface parte de um *node* presente no microROS do veículo, em que sua função é publicar os parâmetros da condição do carrinho de golfe em determinados *topics*. Deste modo, o *software* do Golfinho Diagnostics, realizado pelo editor de código-fonte do Visual Studio Code, tem como disposição um *node* inscrito nestes *topics* para coletar todos os dados fornecidos pelo automóvel. Com isso, por meio das dependências de código aberto Vckpg (MICROSOFT, 2014) e JsonCpp (OPEN-SOURCE-PARSERS, 2015) realiza-se o tratamento dessas informações, na qual um arquivo JSON vai ser criado a partir delas. Tecnologia utilizada para facilitar a transferência dos dados do *backend* para o *frontend*.

Desta forma, a ferramenta FTP proporciona o carregamento do arquivo JSON para a página *web*, na qual sua construção foi idealizada pela linguagem de marcação HTML5 associada à personalização feita pelo CSS3, dispondo do XAMPP como servidor local. Tecnologias da *web design* estabelecidas pela sua viabilidade, eficácia e responsividade para serem exibidas em diferentes resoluções. Além disso, há o uso da linguagem de programação JavaScript para extrair as informações do arquivo JSON, na qual resulta em uma página *web* dinâmica que apresenta as informações da situação simultaneamente a do carrinho de golfe.

Desenvolvimento do aplicativo de tele-operação.

Dando prosseguimento aos estudos de robótica autônoma é proposto o desenvolvimento de um dispositivo de tele-operação. Isto é, um operador que, por meio de um ambiente remoto, controla um robô a distância. Neste cenário, irá permitir que o veículo navegue em circunstâncias extremas que demandam intervenção humana, como paradas anormais ou percursos incorretos.

A partir desses conceitos, um aplicativo *android*, denominado "Golfinho Teleop", foi realizado para controlar os movimentos lineares e angulares do carrinho de golfe.

O *backend* desta aplicação foi realizada no ambiente de desenvolvimento integrado (IDE) Android Studio na linguagem de programação JAVA. Enquanto, o *frontend* utilizou-se da dependência Virtual-joystick-android (BRUN, 2015) para criação da interface.

Em sequência, realiza-se a integração deste dispositivo ao ROS2 através de uma conexão socket. Em que, se conectam, por intermédio de uma mesma rede *wi-fi*.

Possibilitando, a criação de um *node* neste mesmo *framework* para receber os dados do aplicativo e efetuar um tratamento dessas informações. Para só então, publicá-las em um novo *topic*. Isto posto, a placa de controle microROS do veículo possui um *node* inscrito neste mesmo *topic*, na qual permite o carrinho de golfe receber os comandos enviados pelo aplicativo.

Resultados e discussão

Funcionamento e validação do módulo de diagnóstico *web*.

A execução do sistema *web* é realizada a partir da inicialização do painel de controle do XAMPP, no qual se habilita os módulos de serviço para hospedagem local. Em seguida, deve-se abrir um navegador da internet e digitar "localhost" na barra de endereços. Esta ação irá direcioná-lo para a página do diagnóstico *web*, como mostrado na Figura 3.

Figura 3 – Interface do módulo de diagnóstico *web*. Fonte: Autor.

Os parâmetros exibidos para o usuário estão separados na interface. Ao lado esquerdo, a grade de serviço apresenta, por meio de caixas de seleção, se os componentes instalados no veículo estão funcionando da maneira esperada. Além disso, neste mesmo bloco há as coordenadas da localização do veículo.

Ao centro superior da interface há um painel maior com botões que indicam a situação em que o veículo se encontra, em estado manual, autônomo ou desligado. E logo abaixo, a direção atual do automóvel, podendo ser para frente, para trás ou no neutro.

A grade superior a direita indica qual pedal está sendo pressionado, o freio ou o acelerador. Enquanto, os demais blocos da interface representam através de velocímetros os dados numéricos do veículo. Referindo-se a bateria do carrinho de golfe, a da placa de controle do sistema, a frenagem, a aceleração, o esterçamento, a velocidade linear e a velocidade da roda direita e esquerda respectivamente.

Para a validação do sistema *web*. Inicialmente realizaram-se testes com o *frontend*, na qual manualmente alterava-se os parâmetros do veículo no arquivo JSON. De modo, a visualizar se a interface gráfica correspondia dinamicamente aos novos dados

inseridos. Logo, observou-se o comportamento esperado.
Em suma, o último teste envolveu a integração de todo o sistema *web backend* e *frontend* com a placa de controle microROS do veículo, como mostrado na Figura 4. Este ensaio, consistiu em enviar sinais diretamente para a placa mãe do veículo. A fim de averiguar se o módulo de diagnóstico iria efetuar sua função prevista. Desta forma, constatou-se que a cada iteração entre as tecnologias o módulo de diagnóstico era capaz de atualizar simultaneamente os dados que eram gerados na placa mãe.

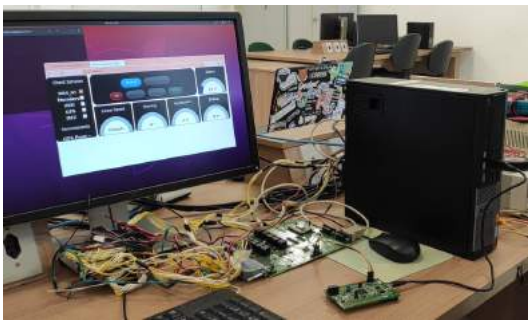

Figura 4 – Ensaio do módulo de diagnóstico *web*. Fonte: Autor.

Funcionamento e validação do aplicativo de tele-operação.
Para efetuar o aplicativo de tele-operação é necessário um dispositivo *android* com a versão 7.0 ou superior. Satisfazendo este requisito introdutório, o usuário deve conectar seu aparelho a um computador que contenha a IDE Android Studio com os arquivos do app. Dessa forma, deve-se realizar a execução do programa em seu dispositivo de destino.
A partir destes passos, já é possível iniciar o aplicativo como qualquer outro. Ao abri-lo será apresentado uma interface, conforme a Figura 4.

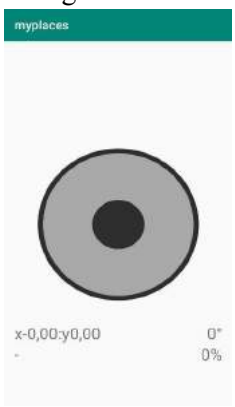

Figura 4 – Interface do aplicativo de tele-operação. Fonte: Autor.

Nesta interface gráfica inicial há um *joystick* ao centro da imagem. E logo abaixo, quatro parâmetros referentes ao controle que indicam respectivamente, as coordenadas de posição (x,y), o ângulo em graus e a intensidade em porcentagem. Além de outra variável que também exibe as coordenadas de posição, mas apenas após estabelecer a conexão ao ROS2.
Com o aplicativo aberto é necessário executar o *node* do *framework* ROS2, em uma mesma rede *wi-fi,* para eles se conectarem. Permitindo, que um operador a distância utilize o *joystick* presente na interface do app para conduzir a distância o carrinho de golfe. A Figura 5 apresenta o diagrama de funcionamento do aplicativo de tele-operação.

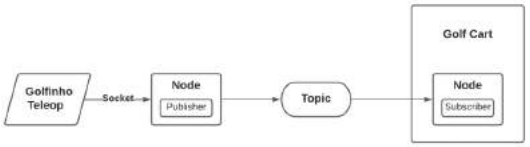

Figura 5 – Diagrama de funcionamento do aplicativo de tele-operação. Fonte: Autor.

A fim de validar o aplicativo de tele-operação, o primeiro experimento realizado é com propósito de verificar a conexão estabelecida entre o *node* do ROS2 e o aplicativo de tele-operação. Para isso, foram executadas ambas as ferramentas, e posteriormente, a movimentação do *joystick*. Resultando nas publicações, realizadas pelo *node,* em um *topic* das coordenadas enviadas pelo aplicativo. Como visto na Figura 6.

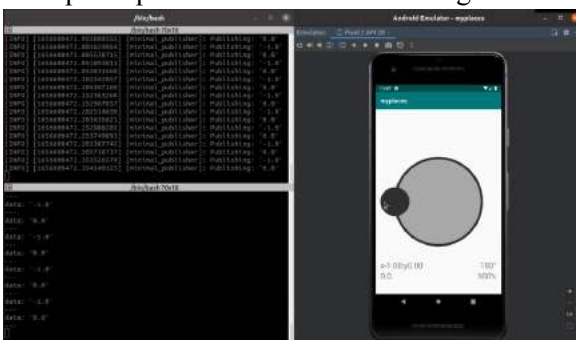

Figura 6 – Resultado do experimento de conexão do aplicativo de tele-operação. Fonte: Autor.

Em seguida, realizou-se testes para autenticar o aplicativo de tele-operação conduzindo um objeto em um ambiente virtual. Os ensaios foram feitos no simulador *turtlesim*, disponibilizado pela documentação do ROS2. E logo, constatou-se que a aplicação desempenhou sua função nestes casos de maneira correta.
Para finalizar, sucedeu-se o teste em ambiente real, mostrado na Figura 7. No qual, é baseado em verificar se o aplicativo controla os movimentos lineares e

angulares do carrinho de golfe. Desta forma, foi observado, após a conexão ser estabelecida entre o app e o ROS2, que ao movimentar o *joystick*. O direcionamento do veículo respondia corretamente, em tempo real, aos comandos recebidos.

Figura 7 – Ensaio do aplicativo de tele-operação com carrinho de golfe. Fonte: Autor.

Conclusões

Com o desenvolver desta pesquisa, inúmeros desafios foram abraçados, englobando um conjunto de habilidades e competências relevantes na engenharia e nas áreas do conhecimento, da robótica móvel e dos veículos inteligentes. Ato que permitiu ao grupo a experiência e a capacitação necessária dentro de seu escopo de estudo, direcionar a problemática "desenvolvimento de sistema *web* para diagnóstico de veículo autônomo e tele-operação via app" ao seu objetivo e funcionalidade prática. Com divisão da equipe entre dois pólos, *hardware* e *software,* foi possível, além de capacitar a comunicação em alto nível com o veículo, resultando no desenvolvimento de novas ferramentas para o carrinho de golfe, a criação de uma interface de diagnóstico *web*, assim como um aplicativo de celular *android.*

Incorporando os parâmetros mais relevantes do automóvel na interface de diagnóstico *web*, de modo a acompanhar os status do veículo durante todo seu percurso, verificando assim se os componentes estão funcionando da maneira adequada. O *software* criado pode identificar um caso de falha antes da mesma causar maiores adversidades. Uma vez que se toma as medidas apropriadas há a possibilidade de se evitar um contratempo ou até um acidente.

De modo a controlar os movimentos, a distância, do carrinho de golfe, o aplicativo de celular *android,* possibilita que o veículo consiga se locomover em caso de falha em uma navegação autônoma, como, por exemplo, um percurso errado ou uma parada espontânea. Desta forma, em situações onde houver trajetos perigosos e ocorrer uma falha no automóvel pode-se evitar a intervenção humana direta, por intermédio deste dispositivo de tele-operação, criando uma maior segurança para a proteção dos responsáveis pelo veículo.

A longo prazo, espera-se que o aplicativo *android* evolua quanto a sua conexão, "quebrando" a dependência perante a rede wi-fi, trabalhando de outras maneiras. Propondo seguimento dos testes das ferramentas desenvolvidas com o objetivo da sua validação.

Agradecimento

Agradeço à Vale por financiar esta pesquisa e a todos os integrantes que contribuíram para a realização do projeto.

Referências

MICROSOFT. Vcpkg: Overview. [*S. l.*], 2018. Disponível em: <https://github.com/microsoft/vcpkg>. Acesso em: 5 out. 2022.

OPEN-SOURCE-PARSERS. JsonCpp. [*S. l.*], 2014. Disponível em: <https://github.com/open-source-parsers/jsoncpp>. Acesso em: 5 out. 2022.

BRUN, Damien. Virtual-joystick-android. [*S. l.*], 2015. Disponível em: <https://github.com/controlwear/virtual-joystick-android>. Acesso em: 5 out. 2022.