XXIX Congresso de Iniciação Científica UNICAMP 2021

APOIO À INFRAESTRATURA ROBÓTICA BASEADA EM ARCABOUÇO ROS PARA UM MINI VEÍCULO ELÉTRICO

Gabriel Henrique Dutra e Silva - FEM/UNICAMP
Prof.Dr.(orientador) Niederauer Mastelari - FEM/UNICAMP
Prof.Dr.(coorientador) Mauro Koyama - CTI Renato Archer

20 de Agosto de 2021

Palavras-Chave: ROS,VEÍCULO,SENSOR

1 Introdução

O VERDE(Veículo Elétrico Robótico com Diferencial Eletrônico) é um mini veículo robótico que tem o tamanho reduzido de um veículo convencional na escala de 1:5, o veículo é alimentado por baterias recarregáveis de lítio de 28 V e 12 V sendo a primeira destinada a alimentar os motores e a segunda é destinada a alimentar outros componentes do veículo. O VERDE possui tração traseira independente e possui um sistema de direção do tipo Ackermann. Na parte de sensoriamento, o mini veículo possui o sensor "hokuyo"que faz a varredura da área em torno do veículo, podendo assim identificar objetos próximos ao mesmo. Além disso possui encoders nas rodas traseiras e dianteiras, uma IMU que possui um GPS e traz informações importantes sobre o C.G do veículo, tal como a velocidade angular e a aceleração linear nos eixos x,y e z. Tanto sensores quanto atuadores se comunicam uns com os outros através de um computador de bordo fixado na parte superior do chassi, este computador tem instalado o Ubuntu 16.04, o sistema operacional utilizando do arcabouço ROS executa tanto os drivers dos sensores quanto o controle do robô.

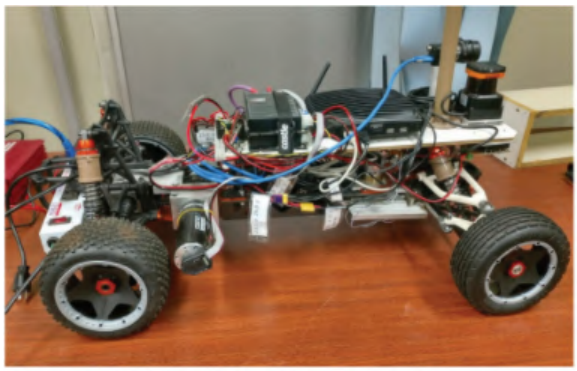

Figura 1: Vista geral do VERDE.

O ROS(Robotic Operation System) como foi citado anteriormente tem participação fundamental na integração dos componentes do VERDE. Segundo Morgan e col(2009), o ROS é um framework que tem como objetivo facilitar a protipagem de robos uma vez que segundo os autores a construção do zero de um sistema de integração seria uma tarefa árdua e nada prática . O funcionamento deste sistema robótico pode ser resumido da seguinte forma, dentro de um "Workspace"serão encontrados tanto pastas utilizadas apenas pelo ROS tais como as pastas "devel"e "build"e também a pasta src onde se encontrarão pacotes criados pelo usuário, estes pacotes conterão nós que são algoritmos escritos também pelo usuário para desempenhar alguma função específica que podem ser escritos tanto em Python quanto C++. Os nós se comunicam-se por meio de tópicos,que podem ser definidos como fluxos de mensagens de um tipo específico(string,int, float etc). Um nó pode publicar mensagens em um tópico, pode coletar mensagens de um tópico ou pode fazer ambos, neste caso, um nó pode,por exemplo, coletar mensagens de um tópico, processar os dados obtidos e depois publicar em um novo tópico.

Os sensores a laser AR500 da Acuity foram instalados na traseira do veículo e integrados ,tal como os outros sensores, no sistema utilizando recursos do ROS, mais especificamente, o driver que fará a ponte entre hardware e software será baseada em pacotes já desenvolvidos para o ROS tais como o Pyserial.

2 Atividades Realizadas

2.1 Compreensão do funcionamento de publishers e subscriber

Como foi dito anteriormente, os nós são algoritmos que se comunicam com o ROS utilizando tópicos, se o nó publica mensagens em um determinado tópico então ele é denominado "publisher"e se coleta mensagens do tópico é denominado "subscriber".

O algoritmo de um publisher escrito em Python pode ser resumido da seguinte forma, em primeiro lugar instância-se o objeto "rospy.Publisher"com os parâmetros como nome do tópico no qual será publicado as mensagens, tipo de mensagem que será publicado e o "queue_size". Depois, configura-se uma frequência qualquer 'N' de publicação em hertz com a linha de comando "rospy.rate(N)". Após estas linhas de comando cria-se um loop que funciona sob a condição do nó estar em funcionamento. Dentro deste loop há dois elementos essenciais, a linha de comando para publicar uma mensagem e a linha "rate.sleep()"que controla a frequência da iteração deste loop garantindo que as mensagens serão publicadas na frequência configurada.

Para um subscriber, também se instância um objeto, neste caso o objeto "rospy.Subscriber", com os parametros como nome do tópico no qual serão coletadas as mensagens, tipo de mensagem coletada e o nome do função que implementará o método de callback. O método de callback deve ser definido antes do objeto que será instanciado e possibilitará a manipulação de dados.

A comunicação entre nós e tópicos pode ser visualizada melhor com a ferramenta rqt_graph como pode ser visto abaixo:

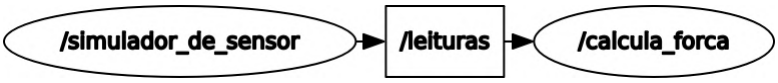

Figura 2: Esquemático da comunicação entre nós.

O nó chamado de "simulador_de_ sensor"é um publisher que publica no tópico "leituras"e o nó "calcula_forca"é um subscriber que processa dados do mesmo tópico.

2.2 Desenvolvimento do driver do sensor

O driver do sensor é um algoritmo em Python que utiliza o módulo Pyserial, este módulo tem a função de permitir a comunicação do sistema com dispositivos que usam comunicação serial. A rotina do programa pode ser resumida na abertura de uma porta serial, em outras palavras, no estanciamento de um objeto do tipo serial.Serial com todos os parametros necessários. Em seguida, os dados que são enviados à porta são decodificados de binário para decimal e normalizados usando uma fórmula descrita no manual do fabricante do sensor. Finalmente, publica-se o valor da distância em milímetros em um tópico que utiliza mensagens

do tipo Float64. Para o funcionamento do driver é necessário definir no próprio nó a porta USB no qual o sensor está conectado e o baud rate que o dispositivo de medição está configurado.

2.3 Desenvolvimento de predefinição de trajetos

Foram escritos três nós, estes nós realizam ,cada um, um tipo de trajeto. Um trajeto é retilíneo, e os outros dois trajetos são curvas no qual diferenciam-se pelo ângulo de esterçamento das rodas e o no sentido que a curva é percorrida(sentido horário ou anti-horário). Os três nós são quase similares, publica-se nos dois tópicos "velocity_controllerLeft/command"e "velocity_controllerRight/command"que controla a velocidade em RPM das rodas traseiras e para controle do esterçamento das rodas publica-se um valor no tópico "position_controller/command". O algoritmo faz a publicação destes valores por um tempo limitado e após este tempo terminar todos os parâmetros de velocidade e esterçamento voltam ao que eram antes de iniciar o trajeto.

2.4 Coleta de dados

A coleta de dados foi feita utilizando o recurso chamado rosbag, a partir do comando "rosbag record"em conjunto com os nomes dos tópicos de interesse e após o término do processo é gerado um arquivo do tipo "bag file"que contém todos os dados que foram publicados nos tópicos em um intervalo de tempo. Utilizando o comando "rosbag play"em conjunto com o nome do bag file criado é possível reproduzir o conteúdo dos tópicos em qualquer computador que tenha instalado o ROS. Além disso, o serviço foi utilizado em conjunto com um outro recurso do ROS chamado de "rqt_multiplot"para gerar um arquivo de texto com todas as medidas dos tópicos e assim usa-lo no Matlab para gerar gráficos.

2.5 Montagem

O sensor de distância AR500 da acuity que está sendo usado possui um range de 5 cm de acordo com o fabricante e possui algumas zonas onde não lê corretamente as medidas, neste caso o sensor mede corretamente em um intervalo de 65mm a 115mm de distância da face do laser voltada ao chão, fora destes limites a distância obtida é zero. Tendo estas informações, para abarcar o maior número de possíveis situações de compressão da suspensão os lasers foram instalados na seguinte configuração, a face lateral esquerda de ambos os sensores estão a aproximadamente 11,5 cm do eixo do veículo e a distancia de ambas as faces voltadas ao chão dos sensores estãa a uma altura de aproximadamente 9 cm.

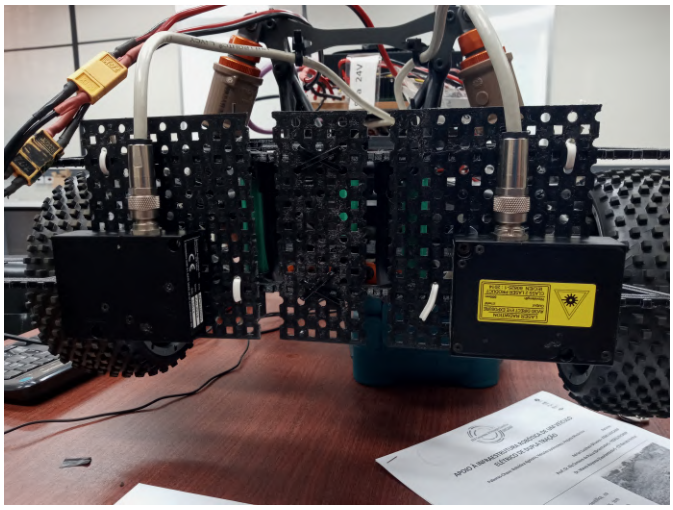

Figura 3: Sensores instalados na traseira do veículo

3 Resultados e Discussão

Para verificar o funcionamento dos sensores fez-se o VERDE se movimentar em uma trajetória retilínea em três velocidades diferentes, neste caso 20,40 e 80 RPM nas rodas traseiras do mini veículo, no meio do trajeto foi colocado um obstáculo por onde o VERDE passaria, as medidas obtidas pelos sensores durante o trajeto podem ser vistos na figura abaixo:

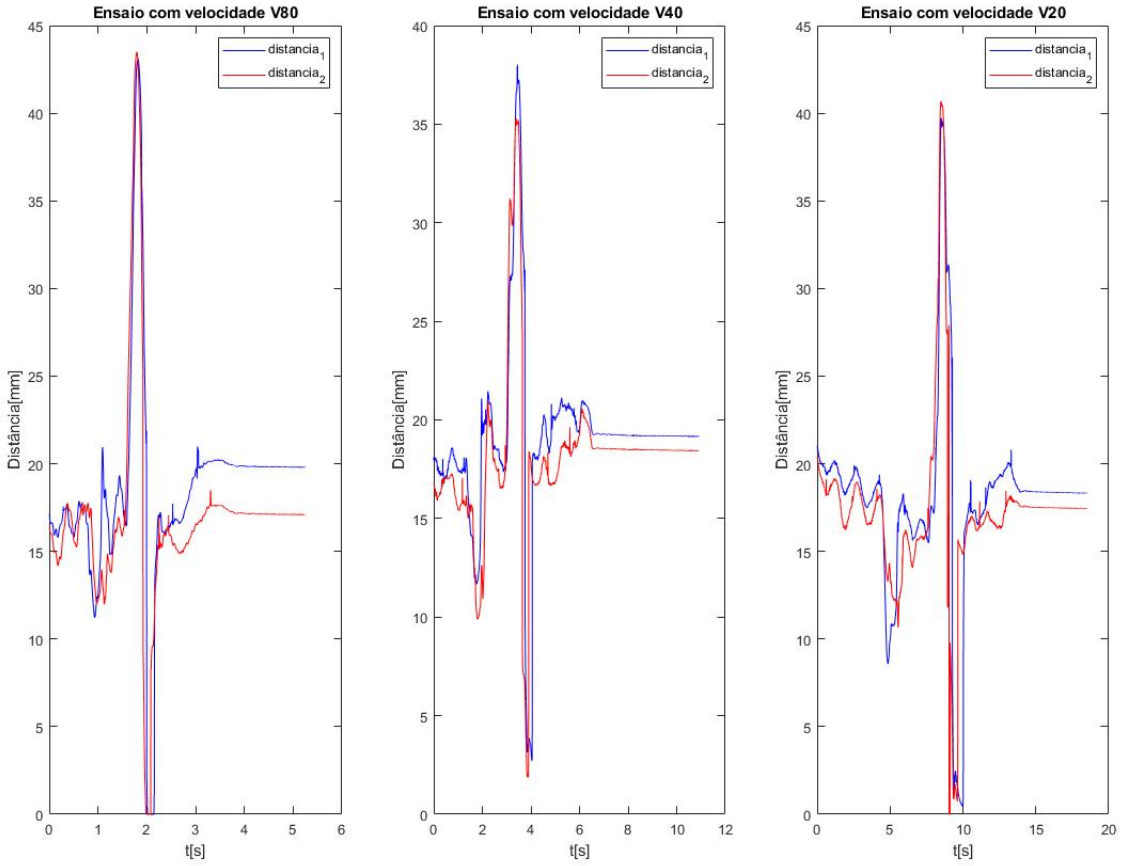

Figura 4: Grafico da leitura dos dois lasers em função do tempo.

O solo pelo qual o mini veículo se movimento é relativamente liso e mesmo antes do VERDE cruzar o obstáculo as medidas não são constantes, uma possível explicação para tal fenômeno passa por dois fatos, o primeiro é que a resolução do sensor é baixa na faixa de μm e o segundo é que o o veículo vibra ao se movimentar. A maior variação ,evidentemente, se refere ao momento quando o VERDE passa pelo obstáculo, num primeiro momento leva a ascensão da traseira em relação ao solo e depois uma brusca diminuição da altura dos sensores. O momento onde as medidas se tornam aproximadamente constantes é quando o VERDE para de se movimentar, é interessante notar que após o percurso terminar a posição de equilíbrio do amortecedor não é a mesma que do início do trajeto, este fenômeno revela uma "histerese"no sistema de amortecimento. Pelos gráficos é perciptível que existe um "offset"entre os dois lasers, neste caso há uma diferença de no máximo 2 mm, a variação é basicamente não perceptível a olho nú.

4 Conclusão

A pesquisa realizada tornou possível uma visão geral do funcionamento do ROS, possibilitando a um entendimento na criação de nós, publisher, subscriber e vários outros elementos importantes que compõe este framework. Além disso permitiu explorar boa parte da capacidade de sensoriamento do VERDE. O sensor possui um range relativamente abaixo e em situações mais extremas como a passagem de uma roda em uma depressão podem levar a leituras erradas. Ainda assim, o sensor possui uma alta precisão de medição, tendo uma resolução de até 5 μm.

5 Bibliografia

[1] - Morgan e col(2009), ROS: an open source Robot Operating System,ICRA(2009), Disponível em https://www.willowgarage.com/sites/default/files/icraoss09-ROS.pdf - Acesso em 20/08/2021

[2] - Sobre Publishers e Subscribers - http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28python%29 - Acesso em 30/08/2021

[3] - Documentação do sensor AR500 - Disponível em: https://www.acuitylaser.com/product/laser-sensors/short-range-sensors/ar500-laser-position-sensor/ - Acesso em 22/08/2021

[4] - Sobre o Pyserial - Disponível em: https://pyserial.readthedocs.io/en/latest/pyserial_api.html - Acesso em 22/08/2021