Navegação Visual Autônoma de um Veículo Terrestre em Escala Reduzida

Daniel Veloso Ribeiro, **Cairo Lúcio Nascimento Júnior**, **Wagner Chiepa Cunha**

Instituto Tecnológico de Aeronáutica
Divisão de Engenharia Eletrônica
Laboratório de Máquinas Inteligentes
São José dos Campos - SP
E-mails: dribeiro@ita.br, cairo@ita.br, chiepa@ita.br

Abstract: This article proposes and presents simulations of an autonomous visual navigation system for a small scale four-wheel differential drive vehicle. The navigation system is composed of: 1) a lane keeping controller that process the image of an onboard vehicle camera and corrects the vehicle orientation and speed, 2) a vehicle localization algorithm that uses the Kalman Filter algorithm to estimate the vehicle pose (position and orientation) using data from a RTK GNSS receiver, a digital compass and odometry sensors. Three possible solutions for the lane keeping controller are investigated and compared. The first solution uses a sliding window approach to process the onboard camera image and extract the distance from the center of the vehicle to the center of the lane and the lane curvature radius. Then a proportional controller is used to calculate the desired speed for the vehicle wheels. The second solution uses a deep convolutional neural network to imitate the first solution, that is, the neural network input is the camera image and the network is trained to generate the same output as the proportional controller used in the first solution. The third solution also employs a deep convolutional neural network but in this case the network is trained using the data recorded when a human drove the simulated vehicle through the same simulated environment.

Resumo: Este artigo propõe e apresenta simulações de um sistema de navegação visual autônoma para um veículo diferencial com quatro rodas em escala reduzida. O sistema de navegação é composto por: 1) um controlador de seguimento de faixa que processa a imagem de uma câmera embarcada no veículo para corrigir sua orientação e velocidade, 2) um algoritmo de localização do veículo que usa filtro de Kalman para estimar a pose do veículo (posição e orientação) usando dados de um receptor GNSS RTK, uma bússola digital e sensores de odometria. Foram investigadas e comparadas três possíveis soluções para o controlador de manutenção de faixa. A primeira solução usa uma abordagem com algoritmo de janelas deslizantes para processar a imagem da câmera embarcada, extrair a distância do veículo para o centro da faixa e o seu raio de curvatura. Um controlador proporcional é utilizado, então, para calcular a velocidade desejada para as rodas do veículo. A segunda solução usa uma rede neural convolucional profunda para imitar a primeira solução, isso é, a rede neural recebe a imagem da câmera como entrada e é treinada para gerar a mesma saída que o controlador proporcional usado na primeira solução. A terceira solução também aplica uma rede neural convolucional profunda, mas a rede é treinada usando dados gravados quando um motorista humano dirigiu o veículo no mesmo ambiente simulado.

Keywords: Lane keeping; Autonomous vehicle; Visual navigation; Neural network; Kalman Filter; GNSS RTK .

Palavras-chaves: Seguimento de faixa; Veículo autônomo; Navegação visual; Rede neural; Filtro de Kalman; GNSS RTK.

1. INTRODUÇÃO

Com a popularização dos veículos autônomos nos últimos anos, solucionar o problema de manter o veículo entre faixas através de soluções automáticas têm se tornado mais relevante. Este problema é conhecido como *lane keeping* e é um tipo de seguimento de trajetória. Existem diversas formas de solucionar o problema, aplicando técnicas variadas como redes neurais ou algoritmos de extração de características de uma imagem em conjunto com técnicas de controle (Bing et al., 2018; Jhung et al., 2018; Samuel et al., 2018).

Uma das abordagens para realizar a tarefa de *lane keeping* é a utilização de algoritmos de tratamento de imagem para localizar e extrair as características da faixa que serão utilizadas para gerar uma saída de controle adequada. Existem diversas formas de identificar as faixas, como a detecção de bordas na imagem, algoritmos para detecção de linhas e curvas, método de janelas deslizantes, detecção de cor, entre outros (Yim and Oh, 2003; Lee and Moon, 2018; Aminuddin et al., 2020; Wang et al., 2018). Essas técnicas podem ser, ainda, utilizadas em conjunto, com o intuito de tornar a detecção das faixas mais robusta e, consequentemente, melhorar o controle.

Redes Neurais Artificiais (RNAs) são algoritmos que funcionam como sistemas de computação paralelos (Zurada, 1992) e possuem diversas aplicações. Entre elas a classificação e extração de características de imagens, reconhecimento de fala, regressão, entre outros. As RNAs podem ser utilizadas na solução do problema de direção autônoma de três formas: percepção mediada, reflexo de comportamento e percepção direta (Chen et al., 2015). Na percepção mediada, a rede é treinada para identificar os parâmetros da pista, como sua posição na imagem, raio de curvatura, sinalização por exemplo. No reflexo de comportamento, a rede é treinada para gerar um comando direto de uma imagem de entrada. Por fim, na percepção direta, a rede é treinada para prever a oportunidade de executar ações de direção.

Neste artigo serão abordadas três técnicas para solucionar o problema de *lane keeping* com um veículo de quatro rodas com controle diferencial em escala reduzida. A primeira baseia-se em um algoritmo de tratamento de imagem, que extrai as características da faixa utilizando a técnica de janelas deslizantes. Já a segunda utiliza uma rede neural convolucional profunda de percepção direta, onde a entrada é a imagem não tratada de uma câmera embarcada no veículo e a saída é um comando para os motores, de forma a imitar o primeiro controlador. A terceira solução aplica a mesma topologia de rede neural que a anterior, no entanto, seu objetivo é aprender a controlar o veículo de forma similar a um ser humano.

O carro e a pista foram modelados no simulador V-REP (E. Rohmer, 2013), com o intuito de reproduzir um veículo real de tamanho reduzido e as ciclovias localizadas nas proximidades do ITA (Instituto Tecnológico de Aeronáutica), onde pretende-se aplicar as soluções propostas serão testadas. O veículo real possui uma câmera USB embarcada, *encoders* no eixo dos motores traseiros, um receptor *Global Navigation Satellite System* (GNSS) com capacidade para *Real time kinematic* (RTK) e uma bússola digital.

Em diversas aplicações de veículos autônomos é interessante que o algoritmo tenha capacidade de localizá-lo em relação ao seu ambiente. Uma das formas mais consagradas de implementar a solução para esta tarefa é através do Filtro de Kalman (FK), um estimador de estados de um sistema dinâmico a partir de medidas corrompidas por ruídos. Para tal, deve-se obter um bom modelo para ruído dos sensores e comportamento dinâmico do sistema (Almeida, 2017).

2. FORMULAÇÃO E SOLUÇÃO DO PROBLEMA

2.1 Lane Keeping

O problema de *Lane Keeping* é um tipo de seguimento de trajetória que consiste em manter um robô ou veículo em torno de um caminho de referência sinalizado por duas faixas no chão. Neste trabalho, tal caminho é definido por duas faixas brancas no asfalto, onde a trajetória de referência é a média entre elas. O ambiente e o veículo foram simulados no *software* V-REP. Com o intuito de tornar a simulação mais próxima do ambiente real, o veículo e o ambiente foram criados com as características encontradas na prática. Tais valores podem ser observados na tabela 1, assim como o ambiente descrito é ilustrado na figura 1.

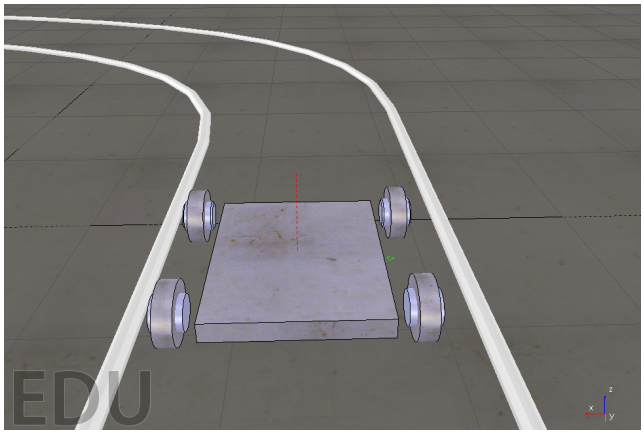

Figura 1. Veículo e trajetória simulada.

Tabela 1. Dimensões do veículo e pista simulados.

Parâmetro	Valor	Unidade
Massa do veículo	12.7	kg
Largura do veículo incluindo rodas	44	cm
Comprimento do veículo	53	cm
Diâmetro das rodas	7.5	cm
Altura total da câmera	45	cm
Torque dos motores	4.5	kg cm
Distância média entre as faixas	85	cm
Largura das faixas	5	cm

2.2 Solução Visual Proporcional

A primeira solução proposta recebe como entrada a imagem de uma câmera embarcada, identifica as faixas utilizando transformações de perspectiva e o algoritmo de janelas deslizantes. Esses dados são utilizados para definir o raio de curvatura médio das faixas, o erro da posição

do veículo em relação ao centro da trajetória de referência e o comando necessário para reduzi-lo. Este algoritmo foi baseado na solução proposta por Dwivedi (2017).

Para que o algoritmo funcione adequadamente, deve-se fazer uma transformação de perspectiva, de forma que as faixas fiquem paralelas quando o trecho for uma reta, simulando uma visão superior. Esse tipo de tratamento é uma modificação geométrica da imagem e é conhecida como *bird eye view* (Pratt, 2007). As características da trajetória serão calculadas na imagem em perspectiva, portanto, é necessário estabelecer uma relação de transformação de *pixels* para metros em ambos os eixos. Essa relação foi calculada através de um padrão quadriculado no ambiente simulado, onde cada elemento possui lados com comprimento de meio metro.

Para facilitar a extração das características da faixa, utiliza-se detecção de bordas, linhas e cor para determinar a região da imagem que delimita o caminho. Esses dados são utilizados para gerar uma imagem binária, contendo apenas os *pixels* de interesse. O algoritmo de janelas deslizantes é, então, utilizado para definir um conjunto de pontos, que serão usados para ajustar um polinômio de segundo grau, representando cada faixa. Com estas funções, calcula-se o raio de curvatura e a distância do veículo em relação ao centro do caminho. O resultado deste processamento é ilustrado na figura 2.

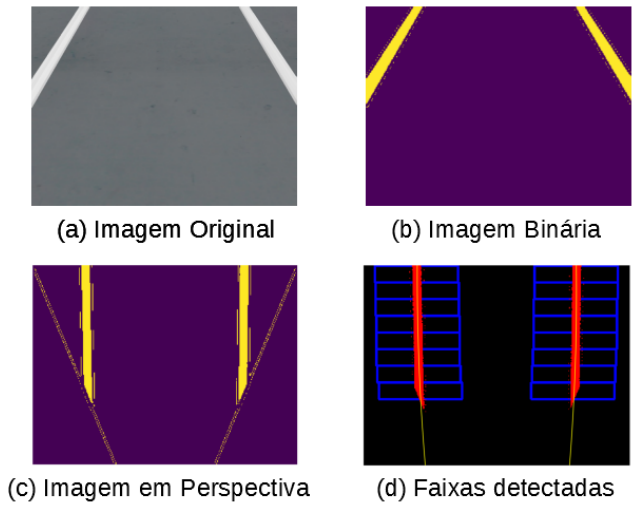

Figura 2. Polinômios ajustados as faixas

Para tornar a resposta do controlador mais parecida com o comportamento humano, a equação (14) foi utilizada para definir a velocidade base do veículo em função do raio de curvatura das faixas, da mesma forma que um motorista faria (Odhams, 2006). Já a equação (15), define a diferença de velocidade entre as rodas em função do erro para o centro da trajetória.

$$V = V_{max} * (1 - e^{-\lambda * R}) \tag{1}$$

$$dV = \frac{2 * V * E_C}{L} \tag{2}$$

onde:

- V é a velocidade base do veículo;
- V_{max} é a velocidade em uma reta;
- λ é uma constante que define a variação da velocidade base;
- R é o raio de curvatura da faixa;
- dV é a diferença de velocidade entre as rodas;
- E_C é o erro em relação ao centro da faixa;
- L é a largura da faixa.

2.3 Solução Neural Proporcional

A segunda solução proposta neste trabalho utiliza uma rede neural convolucional profunda proposta por Bojarski et al. (2016). Se trata de uma rede de ponta-a-ponta para controlar veículos autônomos, cuja entrada é uma imagem sem tratamento e a saída é o ângulo de volante apropriado. Já que o veículo utilizado nas simulações é diferencial, o controle de guinada deve ser feito através da diferença de velocidade entre as rodas. No entanto, a arquitetura utilizada possui apenas uma saída, portanto, foi desenvolvido um mapeamento de velocidade para as rodas, seguindo as equações (16) e (17).

O controlador funciona determinando uma razão entre as velocidades das rodas. Esse valor é uma porcentagem da velocidade base e seu sinal determina a direção de curva, onde valores negativos representam correções para a esquerda e valores positivos para a direita. Portanto, sempre que a inferência da rede é negativa, a velocidade da roda direita é mantida em seu valor base e a roda esquerda tem sua velocidade reduzida até um mínimo de zero e o oposto caso a razão seja positiva.

$$r = \begin{cases} (1 - \frac{V_{direita}}{V_{esquerda}}) & \text{se } V_{esquerda} \geq V_{direita} \\ (-1 + \frac{V_{esquerda}}{V_{direita}}) & \text{se } V_{direita} > V_{esquerda} \end{cases} \tag{3}$$

$$V_{Base} = \frac{V_{Max} * (1 - e^{-\lambda * (1-r)})}{(1 - e^{-\lambda})} \tag{4}$$

Para realizar o treinamento foi criado um conjunto de dados, utilizando imagens não tratadas da câmera e o comando gerado pelo algoritmo da solução anterior associado a ela. Com este método, foram armazenadas 2825 imagens em diversos trechos da pista, compostas por curvas para a esquerda, direita e retas. Para melhorar o treinamento, todas as imagens de curva foram invertidas, assim como os comandos associados. Além disso, 30% dos dados foram contaminados intencionalmente com ruídos do tipo gaussiano e "sal e pimenta", com a intenção de tornar a rede mais robusta a eles. O resultado destas modificações é ilustrado na figura 3.

Utilizando essas técnicas e os conjuntos criados, foram obtidos os resultados de treinamento demonstrados na figura 4. Para validar o treinamento, utilizou-se um conjunto de validação com 20% de imagens retiradas da base de treinamento aleatoriamente, para evitar o viés do ser humano na sua seleção.

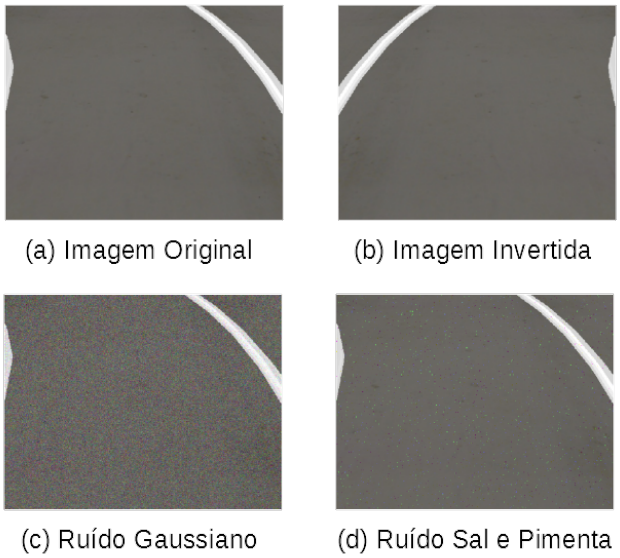

Figura 3. Tratamentos utilizados na base de treinamento

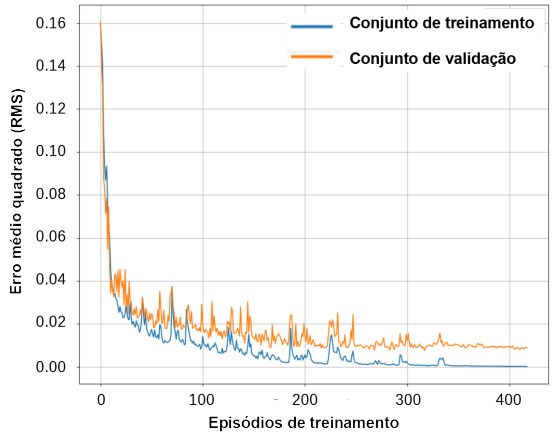

Figura 4. Resultado do treinamento com dados gerados pelo controlador automático

2.4 Neural Manual

A terceira solução proposta neste trabalho utiliza a mesma topologia de rede neural do controlador anterior. No entanto, o objetivo de seu treinamento é diferente. Ao invés de aprender com um algoritmo automático que possui controles consistentes de acordo com a leitura dos sensores, pretendeu-se treiná-la com comandos dados pelo ser humano. Como a pista é relativamente estreita em relação as dimensões do veículo, o controle manual em velocidades altas é difícil, pois qualquer pertubação faz com que o veículo saia da pista. Para evitar esse problema durante a captura das imagens, a velocidade do veículo foi limitada a no máximo 1 km/h.

O conjunto de treinamento e validação foi levantado no mesmo percurso da solução anterior. No entanto, a frequência de captura das imagens foi reduzida para evitar informação repetida devido a baixa velocidade. Ao final da captura dos dados o conjunto contou com 1850 imagens, foi feito o mesmo tratamento de duplicação e adição de ruídos do conjunto anterior. Com estes dados, obteve-se o resultado de treinamento da figura 5.

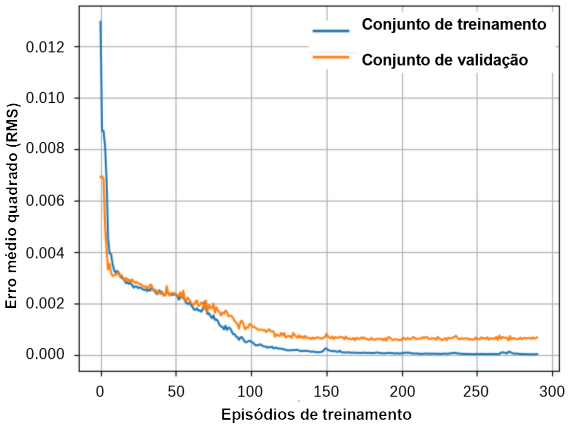

Figura 5. Resultado do treinamento com dados gerados pelo ser humano

2.5 Sistema de Localização

O veículo utilizado para as simulações é terrestre com quatro rodas e controle diferencial. Desta forma, existem três graus de liberdade no modelo: as coordenadas (x, y) no plano local e o ângulo de guinada (θ). O filtro de Kalman, utilizado para estimar estes três estados, funciona com duas etapas: propagação e atualização. Na etapa da propagação, o modelo dinâmico do veículo é utilizado para estimar os seus estados de acordo com as entradas dadas, além da incerteza na estimação e o ganho de Kalman. Já a atualização, mapeia a inovação na correção para os estados estimados, reduzindo a sua incerteza. Para cada etapa, é necessário utilizar sensoriamento adequado, neste trabalho será utilizado um conjunto de três sensores: GPS RTK, bússola digital e odometria de rodas.

A solução para posicionamento usando GPS RTK é uma técnica de posicionamento diferencial, o que significa que existem um ou mais sensores coletando sinais para aumentar a sua acurácia e confiabilidade (Parkinson and Spilker, 1997a). Isto é alcançado compensando a influência dos erros das órbitas dos satélites e da atmosfera. Dentre os receptores utilizados deve existir pelo menos um que se mantém fixo por um longo tempoo qual é chamado de base. O objetivo deste é diminuir a incerteza em relação a sua própria posição, para fornecer dados de correção aos dispositivos móveis. Uma ilustração desse processo pode ser vista na figura 6. No RTK, a base também é utilizada para sincronizar a fase da portadora do sinal recebido, fornecendo maior resolução, em níveis centimétricos, da medida de posição do receptor móvel (Langley, 1998). Em função das distorções na propagação da onda de alta frequência, o erro relativo da solução RTK depende da distância entre a base e o receptor móvel. Portanto, recomenda-se mantê-los em distâncias menores que 10 km.

Para que o sistema de localização funcione adequadamente é necessário um modelo fiel da dinâmica do veículo, segundo Corke (2011),as equações que representam a variação dos estados para um veículo diferencial são dadas por

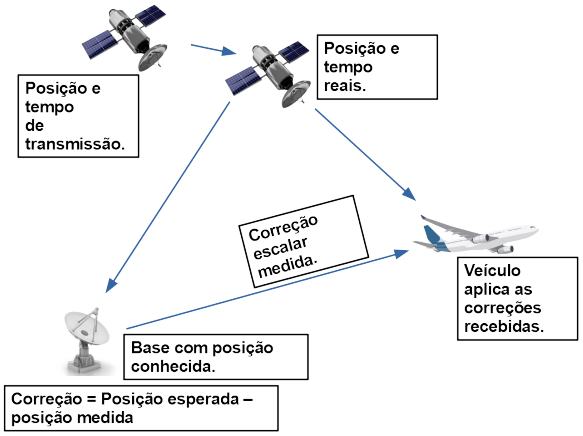

Figura 6. Esquema de GPS diferencial. Adaptado de (Parkinson and Spilker, 1997b)

(5),(6) e (7), onde d_R é a distância percorrida pela roda direita, d_L é a distância percorrida pela roda esquerda e W é a distância entre as duas rodas do veículo. Estas equações serão utilizadas na etapa de propagação do filtro, através da leitura de odômetros para obter o deslocamento das rodas e uma bússola digital para definir a orientação.

$$\dot{x} = \frac{1}{2}(dR + dL)cos(\theta) \tag{5}$$

$$\dot{y} = \frac{1}{2}(dR + dL)sen(\theta) \tag{6}$$

$$\dot{\theta} = \frac{(dR - dL)}{W} \tag{7}$$

No presente trabalho, a etapa de atualização fará a estimação do erro entre os estados gerados na propagação e os estados verdadeiros do veículo. Para isso, deve-se criar um sistema de erro conforme demonstrado na equação (8). Aplicando as equações para a dinâmica do veículo, e, considerando que a orientação fornecida pela bússola geralmente possui erros pequenos o suficiente para serem desconsiderados (dos Santos et al., 2013), obteve-se o modelo de erro demonstrado em (9).

$$X^E(t) = \begin{bmatrix} x^{NAV}(t) - x^V(t) \\ y^{NAV}(t) - y^V(t) \\ \theta^{NAV}(t) - \theta^V(t) \end{bmatrix} \tag{8}$$

$$X^E(t) = \begin{bmatrix} 0.5(\omega_R + \omega_L)cos(\theta^V) \\ 0.5(\omega_R + \omega_L)cos(\theta^V) \\ 0.5(\omega_R - \omega_L) \end{bmatrix} \tag{9}$$

Assim, a propagação do sistema discretizado é dada pelo seguinte conjunto de equações:

$$\begin{bmatrix} X^E_{k+1} \\ Y^E_{k+1} \\ \theta^E_{k+1} \end{bmatrix} = \begin{bmatrix} X^E_k \\ Y^E_k \\ \theta^E_k \end{bmatrix} + \begin{bmatrix} cos(\theta_k)^V & cos(\theta_k)^V \\ sen(\theta_k)^V & sin(\theta_k)^V \\ \frac{1}{W} & -\frac{1}{W} \end{bmatrix} \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} \tag{10}$$

Utilizando a simplificação mencionada, o sistema de erro formado se torna linear, então é possível utilizar o Filtro de Kalman sem linearização. O algoritmo utilizado neste trabalho é descrito a seguir.

Passo 1: Inicialização do Filtro de Kalman:

Para inicializar o filtro, o usuário deve fornecer os seguintes dados: estados iniciais do veículo e a matriz de covariância do erro destes estados.

$$X_0^{NAV-} = \begin{bmatrix} x_0 \\ y_0 \\ \theta_0 \end{bmatrix} \tag{11}$$

$$P_0^{NAV-} = \begin{bmatrix} \alpha_1 & 0 & 0 \\ 0 & \alpha_2 & 0 \\ 0 & 0 & \alpha_3 \end{bmatrix} \tag{12}$$

Inicializa-se a contagem para o tempo do filtro.

Passo 2: Atualização do Filtro de Kalman:

Quando a contagem de tempo para o filtro for um múltiplo da taxa de amostragem de atualização, deve-se realizar a leitura do GPS e bússola. Em seguida, esses dados são usados para calcular o ganho do filtro e atualizar os estados.

$$Y_K^E = X_k^{NAV-} - \begin{bmatrix} x_K \\ y_K \\ \theta_K \end{bmatrix} \tag{13}$$

$$G_K = P_K^- H^T [H P_K^- H^T + R_K]^{-1} \tag{14}$$

$$P_K^+ = [I_{3x3} - G_K H] P_K^- \tag{15}$$

$$X_K^{NAV+} = X_K^{NAV-} - G_K Y_K^E \tag{16}$$

Passo 3: Propagação do Filtro de Kalman:

Quando o tempo for um múltiplo da taxa de amostragem de propagação, deve ser feita a leitura do odômetro para calcular a posição do veículo, baseando-se no modelo dinâmico e nas leituras da odometria. Nesta etapa, faz-se a propagação da matriz de covariância dos erros do modelo.

$$X_{K+1}^- = X_K + \begin{bmatrix} cos(\theta_k)^V & cos(\theta_k)^V \\ sen(\theta_k)^V & sin(\theta_k)^V \\ \frac{1}{W} & -\frac{1}{W} \end{bmatrix} \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} \tag{17}$$

$$P_{K+1}^- = A_d P_K^+ (A_d)^T + B_D Q_K (B_D)^T \tag{18}$$

2.6 Variância dos sensores embarcados

Para que a simulação seja fiel ao ambiente onde pretende-se aplicar o algoritmo de localização, foram levantados os histogramas dos sensores utilizados no robô real, para determinar suas variâncias. No caso do GPS, sua antena foi colocada em uma posição fixa, livre de obstruções, por duas horas, enquanto os dados recebidos foram armazenados para levantar os histogramas. Os resultados obtidos podem ser vistos nas figuras 7, 8 e 9.

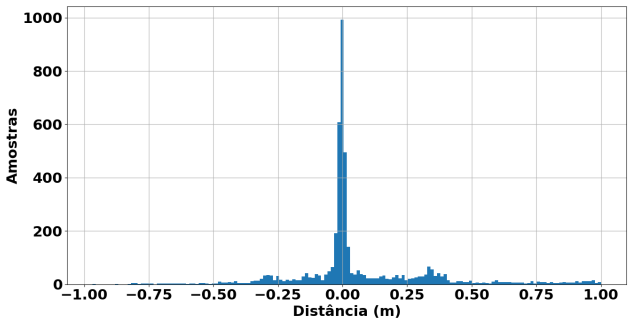

Figura 7. Histograma do erro na latitude

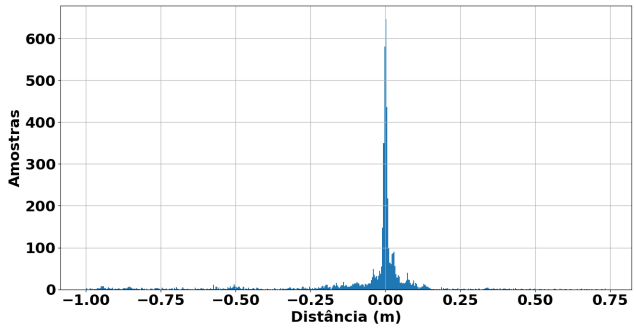

Figura 8. Histograma do erro na longitude

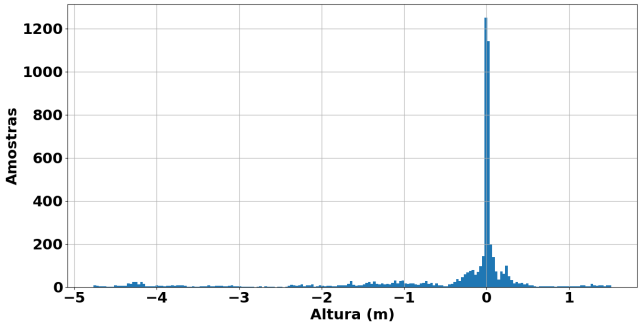

Figura 9. Histograma do erro na altitude

Para determinar a variância da bússola digital, ela foi calibrada e em seguida mantida em uma posição fixa por duas horas, armazenando as diversas leituras obtidas. O histograma resultante destes dados é demonstrado na figura 10.

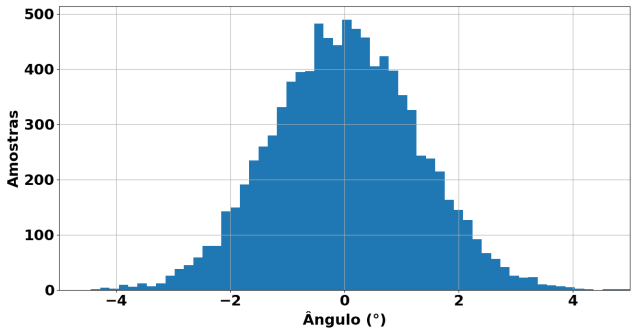

Figura 10. Histograma do erro na orientação

As variâncias e desvios padrões associados aos histogramas levantados são demonstradas na tabela 2. O robô real ainda não possui os sensores de odometria instalados, portanto não é possível utilizar dados reais. Assim sendo, serão estimados valores razoáveis para a simulação.

Tabela 2. Covariâncias e desvios padrões.

Variável	Variância (σ^2)	Desvio Padrão (σ)
GPS_{lat}	0.0674 m^2	0.2597 m
GPS_{long}	0.0472 m^2	0.2173 m
GPS_{alt}	1.2476 m^2	1.1170 m
Bússola	2.9231 ^{o2}	1.7097 ^{o}
Odometria	0.0025 m^2	0.05 m

3. RESULTADOS

Para validar os algoritmos de localização e controle, foram feitos testes no ambiente de simulação, onde modelou-se um veículo, com características similares ao que espera-se encontrar na prática. Para o sistema de sensoriamento, foram simulados os seguintes sensores: uma câmera virtual, um GPS, uma bússola digital e um odômetro em cada eixo dos motores traseiros. Com o intuito de tornar a simulação mais realista, os sensores foram corrompidos com ruído gaussiano de variância similar às observadas nos sensores reais. Além disso, o torque dos motores e dimensões do veículo tiveram seus valores ajustados para serem iguais aos valores nominais dos dispositivos reais.

3.1 Resultados dos controladores

Para verificar o desempenho dos controladores projetados, dois testes foram realizados. No primeiro, a simulação foi iniciada com o veículo fora do centro da trajetória e verificou-se a relação entre a distância do veículo para a trajetória de referência em função do tempo. Tal experimento é ilustrado nas figuras 10, 11 e 12.

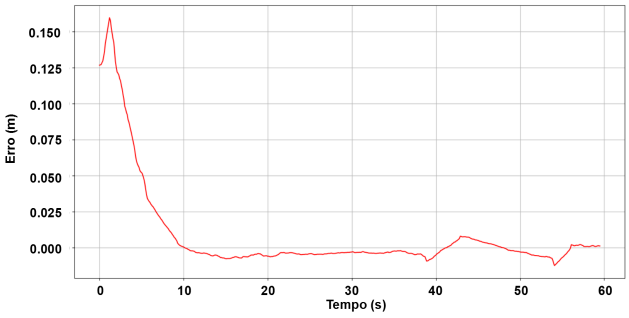

Figura 11. Distância do veículo em relação ao centro da pista com o controlador analítico

Já no segundo teste, uma trajetória fechada com diversas curvas para ambos lados foi construída. As três soluções foram aplicadas com diversos parâmetros para velocidade máxima, um trecho do percurso é ilustrado na figura 14. Os controladores deram cinco voltas na pista para cada velocidade. Desta forma, foram calculados os erros médios quadráticos em relação à trajetória de referência.

3.2 Comparação das soluções propostas

Para comparar as soluções propostas, foi feita uma simulação onde o veículo percorreu uma pista, composta por diversas curvas, com cada controlador e diversos ajustes de velocidade máxima, por cinco voltas. Foram medidos o

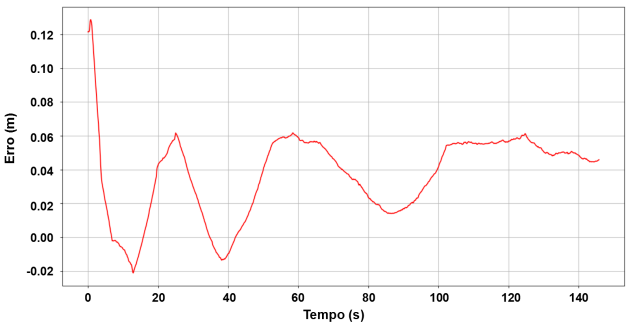

Figura 12. Distância do veículo em relação ao centro da pista da rede, treinada com dados obtidos pelo ser humano

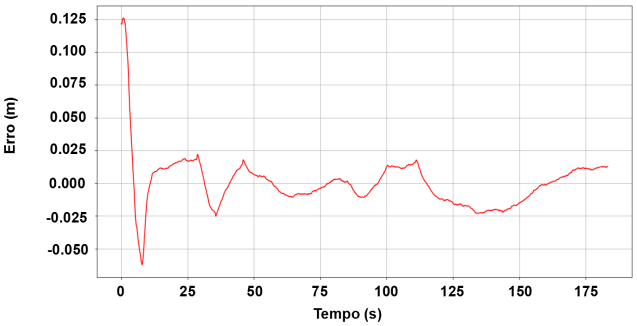

Figura 13. Distância do veículo em relação ao centro da pista da rede treinada com dados obtidos pelo controlador analítico

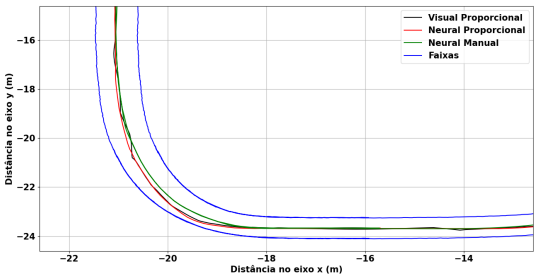

Figura 14. Trajetória do veículo com cada controlador e velocidade média de $2.7km/h$

erro médio quadrático em relação a trajetória de referência e o tempo médio de processamento de cada solução.

Conforme observado na figura 15, todas as soluções propostas apresentam erros proporcionais a velocidade média do veículo, no entanto, o controlador proporcional mantém-se com melhor desempenho em todos os caso. No entanto, os controladores baseados em rede neural possuem melhor tempo de processamento, conforme visto na tabela 3, o que pode ser vantajoso para sua integração em sistemas complexos que necessitam utilizar a CPU do computador embarcado para realizar outras tarefas, desde que exista memória RAM suficiente para carregar a rede neural.

É importante ressaltar, ainda, que a construção do banco de dados para o controlador neural manual apresenta custo maior em relação ao neural proporcional, já que depende do piloto para geração dos dados. Portanto, apesar do desempenho e custo computacional similar, o seu treinamento é uma desvantagem.

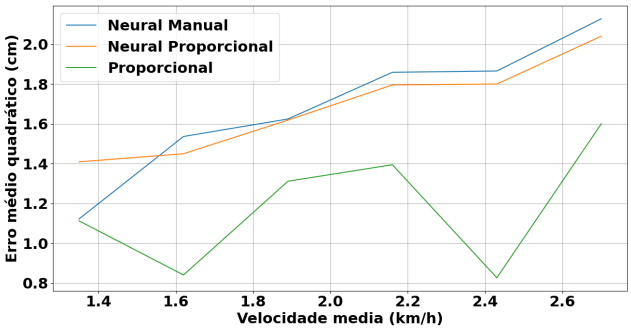

Figura 15. Erro médio quadrático em relação ao centro da pista para os três controladores

Tabela 3. Frequência média de processamento dos controladores.

Controlador	Tempo de processamento médio
Proporcional	106.4 ms
Neural proporcional	12.44 ms
Neural manual	12.53 ms

3.3 Sistema de Localização

Para testar o sistema de localização, o veículo foi simulado realizando um trajeto com uma curva. O teste foi executado com todas as condições ideais, as covariâncias dos sensores informadas para o filtro estavam corretas e o erro na posição inicial estava dentro da faixa de incerteza informada. Desta forma foram obtidos os resultados da figura 17. Em seguida, foi feito um teste com o erro na posição inicial maior do que a estimativa da incerteza da posição inicial e na estimativa do ruído dos sensores. No caso onde o filtro foi inicializado com a posição acima da incerteza informada, o filtro convergiu rapidamente após a fase de atualização. Já na situação onde o ruído do sensor foi subestimado, a resposta do algoritmo é degradada e pode chegar a divergir se o erro for muito grande.

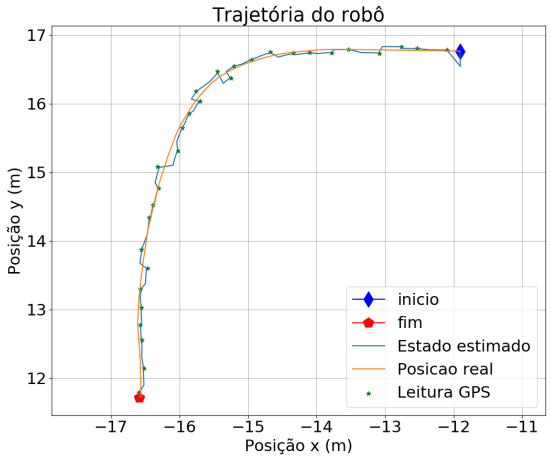

Figura 16. Resultado do sistema de localização

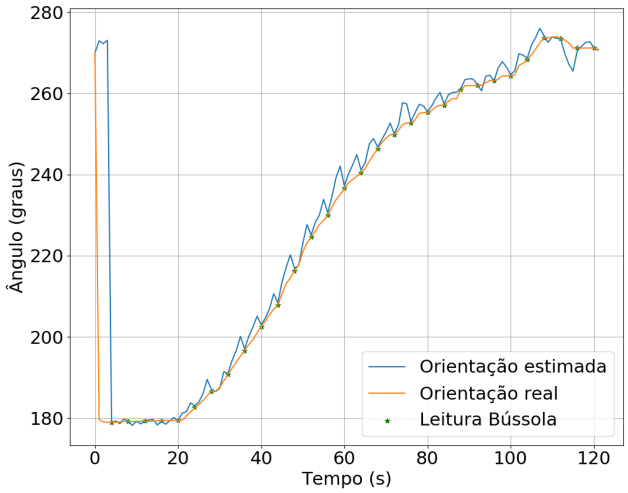

Figura 17. Resultado do sistema de localização com erro na posição inicial maior que a incerteza inicial

4. CONCLUSÃO

Foram apresentados três possíveis soluções para o problema de *lane keeping* de um veículo autônomo diferencial com sistema de localização. Todos os controladores testados foram capazes de realizar a tarefa proposta, se manter e centralizar o veículo entre as duas faixas em um trecho com curvas variadas. Cada solução demonstra certas vantagens e desvantagens na sua aplicação. Em relação à sua capacidade de seguimento de trajetória, o controlador visual proporcional centraliza o veículo mais rapidamente que os outros e apresenta menor erro médio quadrático para o centro da pista até mesmo em velocidades mais altas, no entanto, necessita de mais tempo do processador para gerar a saída. Já os dois controladores baseados em rede neural apresentam erro médio para o centro da pista e tempo de processamento similares, se diferenciando na facilidade de implementação, onde o neural manual apresenta maior grau de dificuldade, já que necessita de um piloto humano para geração do banco de treinamento. No futuro, pretende-se aplicá-los no veículo real para determinar se tais resultados também são válidos fora do ambiente de simulação.

AGRADECIMENTOS

Os autores do artigo agradecem à CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) pelo apoio financeiro concedido ao primeiro autor na forma de uma bolsa de mestrado.

REFERÊNCIAS

Almeida, H.P. (2017). *Navegação autônoma de um veículo terrestre em escala reduzida usando GPS/INS de baixo custo*. Master's thesis, Instituto Tecnológico de Aeronáutica.

Aminuddin, N.S., Ibrahim, M.M., Ali, N.M., Radzi, S.A., Saad, W.H.M., and Darsono, A.M. (2020). A new approach to highway lane detection by using hough transform technique. *Journal of Information and Communication Technology*, 16(2).

Bing, Z., Meschede, C., Huang, K., Chen, G., Rohrbein, F., Akl, M., and Knoll, A. (2018). End to end learning of spiking neural network based on r-stdp for a lane keeping vehicle. *2018 IEEE International Conference on Robotics and Automation (ICRA)*.

Bojarski, M., Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Zieba, K. (2016). *arXiv preprint arXiv:1604.07316*.

Chen, C., Seff, A., Kornhauser, A., and Xiao, J. (2015). DeepDriving: Learning affordance for direct perception in autonomous driving. *2015 IEEE International Conference on Computer Vision (ICCV)*, 468–474.

Corke, P. (2011). *Robotics, Vision and Control*. Springer Nature, Berlin.

dos Santos, D.S., Nascimento, C.L., and Cunha, W.C. (2013). Autonomous navigation of a small boat using IMU/GPS/digital compass integration. *2013 IEEE International Systems Conference (SysCon)*, 468–474.

Dwivedi, P. (2017). Automatic lane detection for self driving cars. URL `https://towardsdatascience.com/https-medium-com-priya-dwivedi-automatic-lane-detecti`

E. Rohmer, S. P. N. Singh, M.F. (2013). Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework. In *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*. Www.coppeliarobotics.com.

Jhung, J., Bae, I., andTaewoo Kim, J.M., Kim, J., and Kim, S. (2018). End-to-end steering controller with cnn-based closed-loop feedback for autonomous vehicles. *2018 IEEE Intelligent Vehicles Symposium (IV)*.

Langley, R.B. (1998). GPS rtk. *GPS World*, 9, 70–72.

Lee, C. and Moon, J.H. (2018). Robust lane detection and tracking for real-time applications. *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS*.

Odhams, A.M.C. (2006). *Identification of Driver Steering and Speed Control*. Ph.D. thesis, Queen's College.

Parkinson, B.W. and Spilker, J.J. (1997a). *Progress In Astronautics and Aeronautics: Global Positioning System: Theory and Applications*, volume 1. American Institute of Aeronautics & Astronautics.

Parkinson, B.W. and Spilker, J.J. (1997b). *Progress In Astronautics and Aeronautics: Global Positioning System: Theory and Applications*, volume 2. American Institute of Aeronautics & Astronautics.

Pratt, W.K. (2007). *Digital image processing: PIKS Scientific inside*. Wiley-Interscience, 4th ed., newly updated and rev. ed edition.

Samuel, M., Mohamad, M., Hussein, M., and Saad, S.M. (2018). Development of lane keeping controller using image processing. *International Journal of Computing and Network Technology*.

Wang, Z., Ren, W., and Qiu, Q. (2018). Lanenet: Real-time lane detection networks for autonomous driving. *Computing Research Repository*, abs/1807.01726.

Yim, Y.U. and Oh, S.Y. (2003). Three-feature based automatic lane detection algorithm (tfalda) for autonomous driving. *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS*.

Zurada, J.M. (1992). *Introduction to Artificial Neural Systems*. West Publishing Company, Minnesota.