APLICAÇÃO DE PLATAFORMA ANDROID NO CONTROLE DE ROBÔS MÓVEIS PARA INSPEÇÃO DE LAVOURAS

Estevam M. Ferreira*, Victor R. F. Miranda†, Mario C. Silva-Jr‡, Leonardo A. Mozelli§, Armando Alves Neto§

*Graduação em Engenharia de Controle e Automação, UFMG, Belo Horizonte, Brasil

†Programa de Pós-graduação em Engenharia Elétrica, UFMG, Belo Horizonte, Brasil

‡CELTA, UFSJ, Ouro Branco, MG, Brasil

§Departamento de Engenharia Eletrônica, UFMG, Belo Horizonte, Brasil

Emails: estevammelo6@gmail.com, victormrfm@gmail.com, mariocupertino@ufsj.edu.br, mozelli@cpdee.ufmg.br, aaneto@cpdee.ufmg.br

Abstract— This paper addresses the use of an *Android smartphone* as an embedded system for navigation control of grounded mobile robots. Based on a control application, it is possible to define geographical goals (known as waypoints) which shall be visited by the robot. GPS data and other inertial sensors integrated at the cell phone are used to control the position, orientation and speed of the platform. Experimental results demonstrate the feasibility of the proposal in a real robot.

Keywords— Agricultural Automation, Mobile Robotics, Precision Agriculture, Android, Control

Resumo— Este trabalho propõe o uso de um *smartphone* com sistema *Android* como sistema embarcado para controle de movimento de um robô móvel terrestre. Através de um aplicativo, será possível definir pontos geográficos, conhecidos como *waypoints*, a serem visitados pelo robô. Dados de GPS e demais sensores inerciais presentes no celular são utilizados para efetuar o controle de posição, orientação e velocidade da plataforma. Resultados experimentais em um robô real são apresentados ao fim do trabalho.

Palavras-chave— Automação agrícola, Robótica móvel, Agricultura de Precisão, Android, Controle

1 Introdução

A crescente produção agrícola brasileira, fruto de investimentos tanto públicos quanto privados, tem conduzido o país a uma posição de destaque internacional cada vez maior. Uma série de fatores favorecem esse crescimento, como clima favorável (que possibilita duas ou mais safras por ano), grandes áreas cultiváveis e farta disponibilidade de água. Visando a manutenção deste cenário, é necessário otimizar a produção, diminuir gastos e melhorar a qualidade dos produtos. Além disso, a presença da robótica permite a melhor utilização dos solos, insumos e defensivos agrícolas, permitindo práticas mais sustentáveis e adequadas do ponto de vista ambiental.

Neste contexto, o presente trabalho apresenta o desenvolvimento de um robô terrestre autônomo, capaz de navegar em ambientes irregulares, típicos de lavouras. Para esse objetivo, avaliou-se a viabilidade do uso de *smartphones* (celulares), equipados com sistema operacional *Android*, para fins de controle de posição, orientação e velocidade do robô.

Considerando que os celulares atuais possuem uma vasta gama de sensores, além de sistema operacional próprio e unidades de processamento para realização de cálculos, sua utilização reduz a complexidade do projeto eletrônico e computacional, quando comparado a soluções que utilizam sensores separados e microcontroladores. A plataforma utilizada nesse projeto é mostrada na Fig.1.

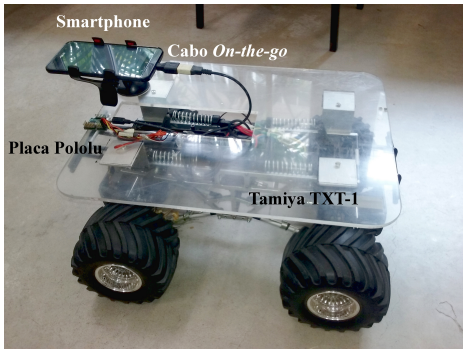


Figura 1: Arcabouço experimental utilizado: *Smartphone Asus Zenfone 2*, conectado a uma placa *Maestro Pololu* através de um cabo *On-The-Go*, embarcado em um robô *Tamiya TXT-1*.

O texto está organizado de acordo com as seguintes seções. A Seção 2 trata da revisão bibliográfica. Ao longo da Seção 3 conceitos básicos sobre o tema são apresentados. Já a Seção 4 aborda a metodologia de desenvolvimento, seguida pela descrição da implementação de um aplicativo capaz de efetuar o controle de posição, orientação e velocidade do robô. A Seção 5 mostra a análise dos resultados. Por fim, a Seção 6 trata da conclusão e de direções futuras para o trabalho.

2 Trabalhos Relacionados

A utilização de robôs móveis terrestres autônomos para monitoramento de ambiente interno e externo foi e continua sendo alvo de diversos estudos. Em Martinez et al. (2014), foi proposto o uso de um robô para monitorar condições ambientais de áreas fechadas e, através dessa análise, desenvolver uma aplicação capaz de detectar regiões com temperaturas desagradáveis, visando atuar localmente de forma a normalizar a temperatura. Em Gobor et al. (2013) é proposto um robô para controle do crescimento de ervas daninhas por meio de uma ferramenta de capina rotativa, evitando o uso de produtos químicos.

Dentre desafios da área, é possível mencionar a implementação de sistemas capazes de evitar colisão com obstáculos e a teleoperação de robôs onde há pouca ou nenhuma cobertura de GPS. Al-Aubidy et al. (2013), enfrenta esses dois desafios através de robô móvel monitorado através de uma base que se comunica com o robô através da tecnologia de comunicação sem-fio *Gereral Packet Radio Service* (GPRS), a mesma utilizada para comunicação móvel.

Várias são as pesquisas que se dedicaram ao estudo da utilização de *drones* para o monitoramento ambiental. Como alguns exemplos, temos Valente et al. (2011), Paneque-Gálvez et al. (2014) e Anderson and Gaston (2013). Acredita-se que a utilização de veículos terrestres são complementares a utilização de veículos aéreos uma vez que ambos possuem suas limitações (Grocholsky et al., 2006).

Analisando o atual cenário para monitoramento de ambientes utilizando robôs tele-operados é possível elencar vantagens e desvantagens do projeto proposto. A primeira desvantagem observada é a necessidade do recebimento do sinal de GPS pelo celular para o funcionamento da solução desenvolvida. Entretanto, com o desenvolvimento da tecnologia de GPS esse problema tende a diminuir cada vez mais, ao longo do tempo. Outra desvantagem, é o fato da solução proposta não tratar a existência de obstáculos, entretanto a utilização de um celular com câmera embarcado ao robô favorece a futura adição dessa melhoria ao sistema.

Por fim, uma vantagem do projeto é a inovação do uso de toda a arquitetura de sensores e processamento presente em um *smartphone Android* para fins de controle. Acredita-se que a utilização de uma tecnologia já difundida no mercado é uma característica que estreita o laço existente entre o usuário e o produto produzido. Outra vantagem, é o fato de que se trata de uma plataforma comercial de pequenas dimensões. É sabido que o uso de veículos de tração de grande porte trazem prejuízos ao solo (Ayers, 1994).

3 Fundamentos

3.1 Modelo cinemático do robô

Em De Luca et al. (1998), os autores descreveram um modelo cinemático para veículos com restrições não holonômicas do tipo *Arckerman*, tal qual o utilizado neste trabalho. Esse modelo se baseia nas seguintes equações:

$$\dot{x} = v\cos\theta, \tag{1}$$

$$\dot{y} = v\sin\theta, \tag{2}$$

$$\dot{\theta} = \frac{v}{l}\tan\phi, \tag{3}$$

sendo $\vec{x} = (x, y, \theta)$, no qual (x, y) é a posição do veículo e θ é a sua orientação; ainda, v é a sua velocidade linear, l é a distância entre as rodas frontais e traseiras do veículo e ϕ é o ângulo de esterçamento das rodas dianteiras do veículo. A Figura 2 ilustra o conceito do modelo.

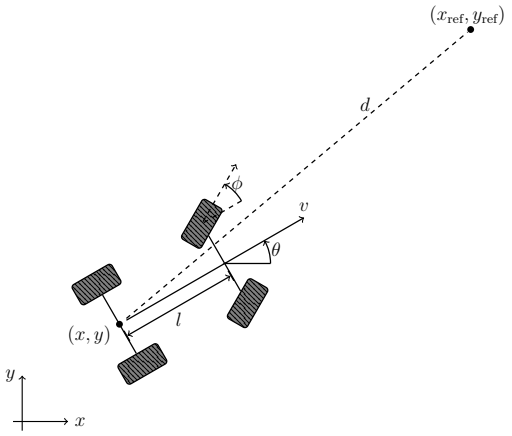


Figura 2: Apresentação da variáveis de processo no controle de posição e orientação.

3.2 Controle de posição e orientação do robô

Por controle de posição, entende-se o conjunto de ações necessárias para que um robô móvel navegue de um ponto geográfico a outro, dentro de um ambiente. Já o controle de orientação são as ações de direcionamento do veículo, atuando sobre as rodas de forma que a orientação do carro esteja apontando para o ponto onde deseja alcançar (alvo).

Ainda conforme ilustrado na Fig. 2, deseja-se que a distância d entre a posição do robô (x, y) e a posição desejado do alvo $(x_{\text{ref}}, y_{\text{ref}})$ seja reduzida para o mais próximo possível de zero. No próximo capítulo serão apresentadas as leis de controle utilizadas para zerar esse erro de distância.

3.3 Fundamentos do desenvolvimento Android

Uma vez que o produto do projeto é um aplicativo *Android*, para controle de navegação de um robô

móvel, faz-se necessário o entendimento de alguns conceitos básicos desta tecnologia.

A linguagem utilizada para programar aplicativos *Android* é o *Java*. O código é compilado pelas ferramentas do *Android SDK* (*Software Development Kit*), juntamente com todos os arquivos de dados e recursos em um único arquivo de sufixo .apk, que é um Pacote *Android* (APK). Esse arquivo contém todo o conteúdo do aplicativo e são os arquivos que os dispositivos desenvolvidos para *Android* usam para instalá-lo. O site para desenvolvedores *Android*, (Android, 2017), foi utilizado como base para essa seção.

O sistema Android implementa o princípio do privilégio mínimo. Sendo assim, cada aplicativo, por padrão, tem acesso somente aos componentes necessários para a execução do seu trabalho e nada mais, impedindo o acesso a partes do sistema para o qual não tem permissão. Com isso, para que o aplicativo tenha acesso a dados de geolocalização (GPS) e acesso a escrita de dados (funcionalidades necessárias para o desenvolvimento do projeto), será necessária a solicitação de permissão de acesso a essas funcionalidades ao usuário.

Atividade ou *Activity* é um dos quatro tipos de componentes de aplicativos, os quais são os blocos de construção de um aplicativo *Android*. Cada componente é um ponto diferente pelo qual o sistema pode acessar seu aplicativo. Cada tipo tem uma finalidade distinta e tem um ciclo de vida específico que define a forma pela qual o componente é criado e destruído. No caso do aplicativo em estudo, será utilizado apenas o tipo de componente chamado Atividade, que representa uma tela única com uma interface de usuário. Serão desenvolvidas duas atividades: A Tela Principal e a Tela de Mapa. Essas duas atividades trocam dados entre si através de uma *Intent*, que será descrita na seção a seguir.

A atividade é disparada através de uma mensagem assíncrona chamada *Intent*. Através de pares de valor-chave é possível passar dados para a atividade que está sendo chamada. Além disso, é possível utilizar filtros de *Intents* de forma a chamar uma Atividade apenas sob condições específicas.

Para iniciar um componente de aplicativo, é necessário antes que o sistema *Android* leia o arquivo AndroidManifest.xml (o arquivo de "manifesto") do aplicativo para o sistema saber da existência do componente. É nesse arquivo que todos os componentes de aplicativos criados serão referenciados. Não apenas componentes, no manifesto também serão declaradas as permissões utilizadas (geolocalização e escrita de dados) e recursos de *hardware* utilizados (placa controladora de servomecanismos).

Ao executar um aplicativo o sistema cria uma *thread* principal que é responsável por operações de interface de usuário. Essa thread não deve ser sobrecarregada com outras operações além das de interface de usuário, sob o risco do aplicativo apresentar atrasos na atualização de conteúdo da tela, depreciando a responsividade do aplicativo.

Para outras operações é recomendado a utilização de *threads* secundárias. Elas não possuem acesso direto à fila de execução da *thread* principal e portanto não é possível chamar operações de atualização da tela diretamente por eles. São necessários *Handlers* para isso, que são objetos que postam uma operação de User Interface (UI) na fila de operações a serem executadas pela *thread* principal.

Em um projeto anterior o mesmo robô foi utilizado para fins de locomoção autônoma utilizando os mesmos princípios empregados no presente projeto. O projeto anterior teve sucesso em realizar o controle do robô utilizando um *notebook* embarcado ao robô e sensores dedicados à localização geográfica do robô. Algoritmos de controle em *Python* foram implementados nesse *notebook*.

4 Desenvolvimento

Nesta seção, serão descritos os métodos utilizados na implementação do robô controlado através de um aplicativo *Android*. O desenvolvimento do protótipo do sistema seguiu algumas premissas:

1) o nível da API *Android* mínima escolhida foi a 14, devido a sua cobertura de aproximadamente 100% de dispositivos ativos na *Google Play Store*;

2) não considerou-se nesse projeto a existência de obstáculos no ambiente;

3) não há derrapamento nas rodas do robô;

4) o robô se movimenta apenas para frente, com velocidade constante pré-definida.

Para desenvolvimento do aplicativo, foi utilizado o ambiente integrado chamado *Android Studio*, produzido pela *Google*. Já o computador utilizado foi um *Toshiba Satellite* com processador *Intel Core I7* e equipado com sistema operacional *Windows 7*.

4.1 Arquitetura

A solução proposta conta com um *smartphone Android* conectado via USB a uma placa controladora de servomecanismos *Mini Maestro Pololu*. No celular, foi programado o aplicativo responsável pela aquisição dos dados de geolocalização, orientação e velocidade. O sinal de controle foi enviado via USB para a placa controladora, que por sua vez aciona as entradas de controle do robô. Essa arquitetura é ilustrada na Fig. 3.

Dois tipos de sensores do celular foram utilizados neste projeto: magnetômetros e GPS. Os sensores de orientação medem, em graus, a atitude

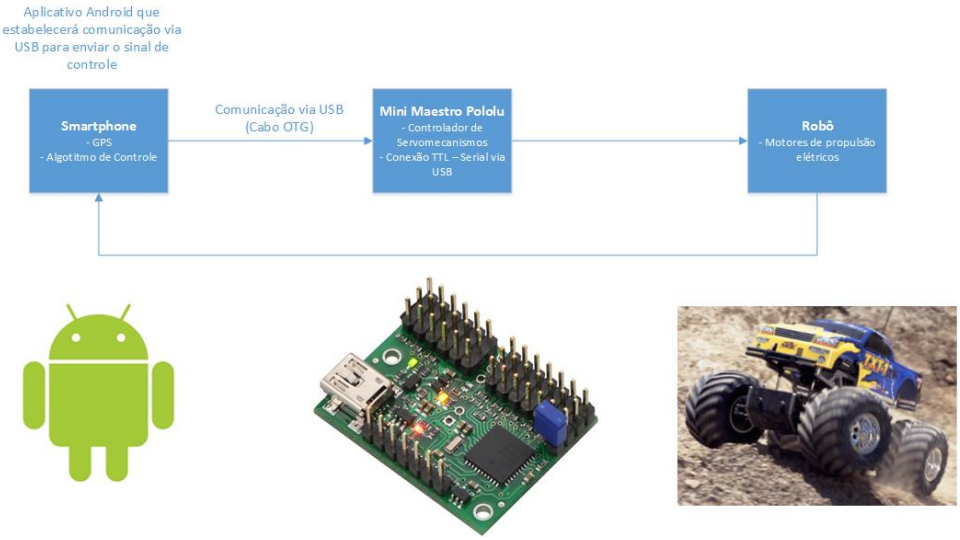


Figura 3: Arquitetura do sistema: smartphone, placa controladora de servomecanismos e robô. *Fontes:* https://www.pololu.com/product/1352; https://developer.android.com

θ do celular em relação aos eixos (x, y) da Terra, de modo que $\theta = 0°$ indica a direção Norte, $\theta = 180°$ o Sul, $\theta = 90°$ o Leste e $\theta = 270°$ o Oeste. A taxa de aquisição de dados configurada para esse caso foi de aproximadamente 5 Hz. Já para o GPS, a taxa de aquisição escolhida foi de aproximadamente 1 Hz, a máxima fornecida pelo dispositivo. A qualidade dos dados de geolocalização depende principalmente do número de satélites disponíveis para uma determinada localização.

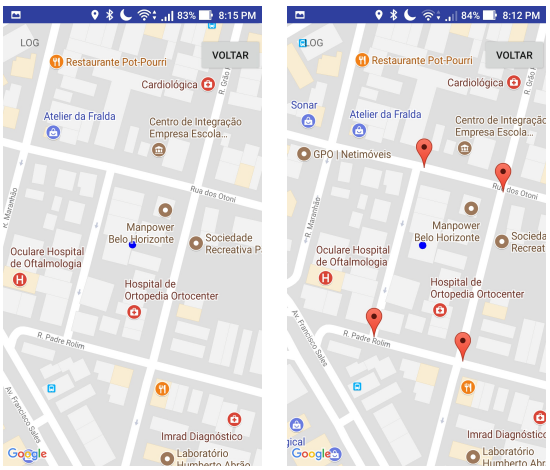
4.2 O aplicativo

O aplicativo foi concebido de forma apresentar apenas uma simples Tela de Mapas. Através de um filtro de *Intent* foi possível condicionar a inicialização do aplicativo à conexão da placa controladora de servomecanismos. Dessa forma, ao conectar a placa, o aplicativo é automaticamente iniciado. Uma vez na Tela Inicial, o usuário será notificado, através de um modal, de que não existem alvos geográficos para o robô alcançar. Para adicionar novos alvos geográficos, o usuário deverá inicializar a Atividade de Mapa através do modal ou através do botão "Mapa" na tela inicial.

Ao inicializar a Atividade de Mapa, o aplicativo automaticamente centralizará o mapa na posição atual do usuário. Através de cliques de longa duração no mapa, o usuário cria pinos indicando que o alvo geográfico foi adicionado na Lista de Alvos Geográficos que o robô irá percorrer. Uma vez adicionados todos os alvos desejados, o usuário deverá clicar no botão de "Voltar" presente no canto superior direito da Tela de Mapa. A Tela Principal irá se abrir, exibindo a latitude e longitude de cada um dos alvos geográficos escolhidos. Ao clicar em "Iniciar", o controle do robô será iniciado e ele percorrerá os alvos um a um até chegar no último alvo, onde o robô estacionará esperando por novos alvos. Esse fluxo de operação é retratado na Figura 3.

4.3 Tela de mapa

Através da API do *Google Maps* para *Android*, foi criada uma nova tela que se trata de um mapa renderizado. Para tanto, cria-se uma chave da aplicação para o acesso à API. É através dela que a aplicação é reconhecida e autenticada para se ter acesso aos métodos de renderização e utilização do Mapa. Feito isso, ao inicializar o Mapa, automaticamente é utilizada a localização do usuário para centralizar o mapa e aumentar o *zoom* na localização do usuário. O resultado é apresentado na Figura 4(a).



(a) Tela de Mapa antes da inicialização.

(b) Tela de Mapa após a seleção de quatro alvos.

Figura 4: Tela do aplicativo para seleção de pontos alvo (*waypoints*).

A seleção dos alvos é feita através de um clique longo na tela, ao fazê-lo um *pin* é adicionado no

local indicando que o alvo já foi adicionado. Após a seleção de quatro alvos o resultado é exibido na Figura 4(b). Após selecionar os alvos o usuário deverá clicar em voltar. Através de um *Intent*, será chamada novamente a Tela Principal. As informações dos alvos selecionados (latitude e longitude dos pontos) são passadas para a Tela Principal através de valores chave, conforme apresentado na seção 3.3.

4.4 Controle de velocidade

Para a implementação do controle de velocidade, considerou-se o modelo de aceleração longitudinal do veículo

$$ma(t) = \alpha u_v(t) + f(v(t)) + h(g), \tag{4}$$

sendo m é a massa do robô (aproximadamente 5 [Kg]), a sua aceleração e α é uma constante de proporcionalidade, que relaciona o torque do motor com a entrada de aceleração u_v. Além disso, $f(v)$ representa uma função que modela perdas relacionadas à velocidade do veículo (tais como resistência do ar, atrito viscoso e outros), e $h(g)$ incorpora efeitos relacionados à aceleração da gravidade (tais como a inclinação e rugosidade do terreno).

Conforme dito anteriormente, foram desconsiderados a inclinação do terreno e outros efeitos de perturbação, de modo que $h(g) = 0$. Assim sendo, linearizando a equação (4) em torno de um ponto de operação desejado, tem-se:

$$m\dot{\widetilde{v}}(t) \approx \alpha\widetilde{u_v}(t) - b\widetilde{v}(t), \tag{5}$$

sendo b uma constante que incorpor as perdas por atrito relacionadas a velocidade do robô, $\widetilde{v}(t) = v(t) - v^{op}$ (velocidade menos a velocidade de operação), e $\widetilde{u_v}(t) = u_v(t) - u_v^{op}$ (ação de controle menos a ação de controle de operação).

Aplicando-se a Transformada de Laplace, obtém-se:

$$ms\widetilde{V}(s) = \alpha\,\widetilde{U_v}(s) - b\widetilde{V}(s) \tag{6}$$

$$G(s) = \frac{\widetilde{V}(s)}{\widetilde{U_v}(s)} = \frac{\alpha}{(ms+b)} \approx \frac{120}{(5s+6000)}. \tag{7}$$

Aqui, os valores de α e b foram obtidos por meio de ensaios, cujos pontos de operação foram $v^{op} = 1,5m/s$ e $u_v^{op} = 1440$ pulsos PWM.

Um controlador proporcional foi adotado para controle velocidade, conforme:

$$u_v = K_v\left(v_{\text{ref}} - v\right), \tag{8}$$

cujo ganho proporcional é mostrado na Tabela 1.

4.5 Controle de orientação

Conforme apresentado na seção 3.2, a variável controlada é θ. O valor de referência é θ_{ref}, que é calculado através dos valores de latitude e longitude de referência e medido. Foi utilizado um controlador proporcional, de forma que:

Tabela 1: Parâmetros dos controladores de velocidade, orientação e posição.

parâmetro	K_v	K_θ	d_{min}
valor	50	8	5 [m]

$$u_\theta = K_\theta\left(\theta_{\text{ref}} - \theta\right) \tag{9}$$

cujo valor foi definido experimentalmente, conforme Tabela 1, e $(\theta_{\text{ref}} - \theta)$ é o valor do erro de orientação.

A *API* de sensores do *Android* utiliza como referência para a medição dos ângulos de orientação o eixo y, e que ângulos acima de 180° serão representados por seu equivalente valor negativo. Dessa forma, quando o robô está apontando para uma direção à direita do alvo (ou seja, $\theta_{\text{ref}} < \theta$) haverá um sinal de controle negativo. Já quando o robô está apontando para uma direção à esquerda do alvo ($\theta_{\text{ref}} > \theta$) haverá um sinal de controle positivo. Um sinal positivo (negativo) de controle provoca um esterçamento nas rodas do robô para a direita (esquerda). Sendo assim, o sinal de controle atuará sempre esterçando as rodas do robô de forma a orientá-lo em direção ao alvo.

É importante notar que haverá casos em que o erro de orientação poderá exceder o valor de 180°. Essa situação só ocorrerá quando apenas um dos ângulos for negativo. Apenas para essa situação, o valor negativo em questão será transformado no seu valor equivalente positivo. Um exemplo dessa situação é o caso onde $\theta_{\text{ref}} = 170°$ e $\theta = -170°$, obtendo um erro de 340°. Convertendo θ obtém-se $\theta = 190°$ e o erro será de $-20°$. Obtendo um sinal de controle negativo, e esterçamento da roda para a esquerda.

4.6 Controle de posição

O controle de posição foi feito através de um Controlador *On / Off*. Definiu-se uma circunferência, com raio definido na Tabela 1, que corresponde à distância mínima que o robô deve estar do seu alvo para ser considerado sucesso. Ao adentrar essa circunferência o robô irá parar e esperar por 5 segundos para se locomover para o próximo alvo (caso exista). A distância entre o robô e o alvo foi calculada usando a *API* de localização do *Google*.

5 Resultados

Nesta seção, serão apresentados os resultados do trabalho. Para os testes, foi utilizado um *smartphone Asus Zenfone 2* comunicando-se, através de um cabo *On-The-Go*, com uma placa controladora de servomecanismos, modelo *Pololu*. Esse

conjunto foi embarcado em um automodelo da *Tamiya*, o modelo *TXT-1*. O arcabouço completo é mostrado na Fig. 1. Os experimentos foram feitos em área aberta, livre de obstáculos.

Inicialmente, foi realizado um experimento para validar o modelo longitudinal de navegação do robô. O resultado é apresentado na Fig. 5. Observou-se que, para o ponto de operação escolhido, o modelo representa satisfatoriamente o sistema real.

Feita a validação, procedeu-se a um primeiro experimento com 4 pontos-alvo, dispostos em terreno plano com variações no tipo de solo, entre concreto liso e grama irregular. Na Fig. 6, observa-se que o robô atingiu todos os alvos em sequência. A Figura 7 apresenta os valores de referência e medidos para a orientação e velocidade do robô durante o primeiro teste.

Observa-se que, para o primeiro alvo, a velocidade não chega a 1,5 m/s, já que o robô alcança o alvo antes disso, parando posteriormente durante 5 segundos. Em seguida, avança para o segundo alvo, atingindo a velocidade de referência. Entre o segundo e o terceiro alvos, o robô passa por um gramado irregular, o que adiciona perturbações ao controle de velocidade. Por fim, ao avançar para o quarto e último alvo, o robô atinge e mantém a velocidade de referência.

No segundo teste, foram utilizados 5 alvos, mas dessa vez em parte do terreno inclinado (subidas e descidas). Assim como antes, o veículo também enfrentou variações de solo, conforme pode ser visto na Fig. 8. Nesse experimento, verifica-se a efetividade do controle, mesmo frente as perturbações causadas pelo relevo do nosso terreno, algo típico também em lavouras. Os valores de velocidade e orientação podem ser observados na Fig. 9.

5.1 Análise da aquisição de dados dos sensores

A fim de avaliar a viabilidade do uso de sistemas *Android* para fins de controle, a confiabilidade dos dados fornecidos foi averiguada. Para isso, mede-se a frequência dos dados fornecida pelo dispositivo e compara-se com a taxa de amostragem desejada, considerando a porcentagem dos dados fora dessa taxa.

Por padrão, o sensor de orientação possui taxa de amostragem de aproximadamente 20 Hz. Entretanto, para esta aplicação, uma taxa de apenas 5 Hz foi suficiente, sendo feita decimação desse sinal. Já para o GPS, a taxa máxima fornecida foi de 1 Hz.

Em um teste realizado, observou-se que das 704 amostras de orientação, 678 apresentaram intervalo de tempo de até 0,21 segundos, valor próximo do desejado. A média dos valores de diferença de tempo entre amostras para esse sensor foi de 0,215 segundos, enquanto que o desvio padrão foi de 0,05 segundos. Já para o GPS, observou-se que 131 das 137 amostras apresentam diferença de tempo inferior a 1,1 segundos. A média dos valores de diferença de tempo entre amostras foi de 1,07 segundos e o desvio padrão foi de 0,11 segundos.

6 Conclusões

Esse artigo tratou do desenvolvimento de um robô autônomo terrestre, controlado unicamente por um *smartphone Android*, visando automatizar o processo de coleta de dados em lavouras. Para esse fim, foram estabelecidos dois objetivos: (i) avaliação da viabilidade do uso de *smartphones Android* para fins de controle de um robô autônomo terrestre, e (ii) consolidação de um sistema de comunicação entre o aplicativo instalado no celular e o robô. Ambos objetivos foram alcançados com sucesso, uma vez que o robô se mostrou capaz navegar para pontos geográficos determinadas pelo usuário, baseado apenas em sensores de um celular. Além disso, o processamento desses dados e os cálculos relacionados ao controle também foram realizados inteiramente no aparelho.

A principal limitação encontrada no projeto diz respeito ao controle de velocidade. Uma vez que os dados de velocidade são obtidos através de GPS, com baixa precisão e taxa de aquisição, o controle foi afetado. Entretanto, os dados de GPS se mostraram suficientes para atender ao requisito de projeto.

A solução proposta abre espaço para uma série de melhorias. A primeira delas é a utilização da câmera do celular para a implementação de funcionalidades relacionadas a desvio de obstáculos ou mesmo odometria visual. Outra melhoria interessante seria a expansão da solução de forma a abranger outros sistemas operacionais tais como *iOS* e *Windows Phone*, seja criando aplicativos específicos para cada um desses sistemas operacionais, seja migrando o projeto atual para algum ambiente de desenvolvimento multiplataforma.

Agradecimentos

Esse artigo foi desenvolvido dentro do contexto do projeto "*APIAR: desenvolvimento de robôs para inspeção de lavouras*"(APQ-03433-15), Demanda Universal da Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG). Agradecemos ainda ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio.

Referências

Al-Aubidy, K. M., Ali, M. M., Derbas, A. M. and Al-Mutairi, A. W. (2013). GPRS-based remote sensing and teleoperation of a mobile

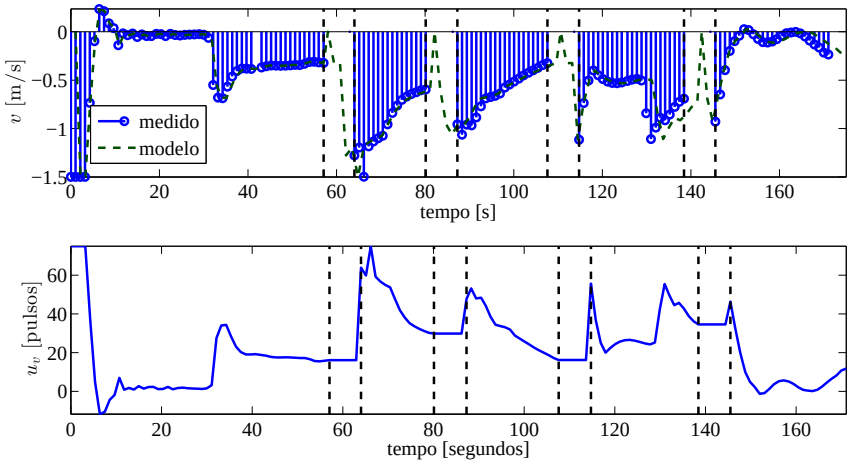


Figura 5: Validação do modelo longitudinal: velocidade do robô em torno do ponto de operação escolhido.

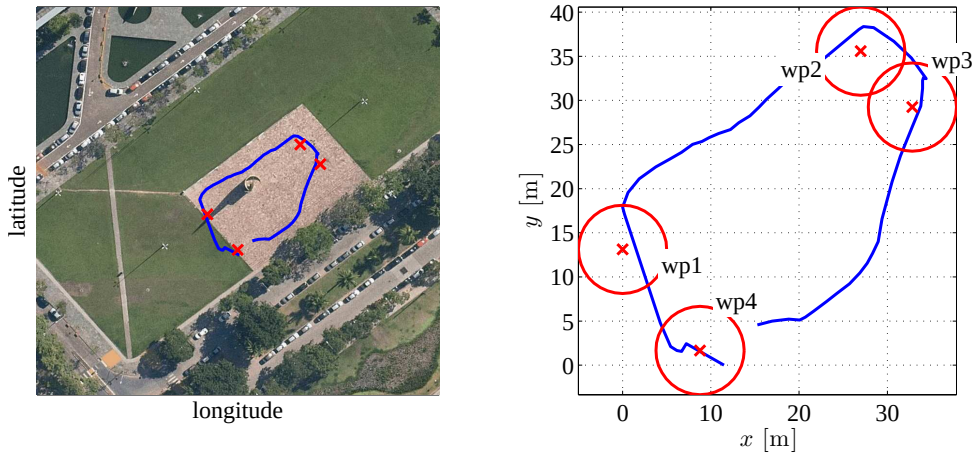


Figura 6: Teste 1: caminho percorrido pelo robô entre 4 alvos, terreno plano (fonte *Google Maps*).

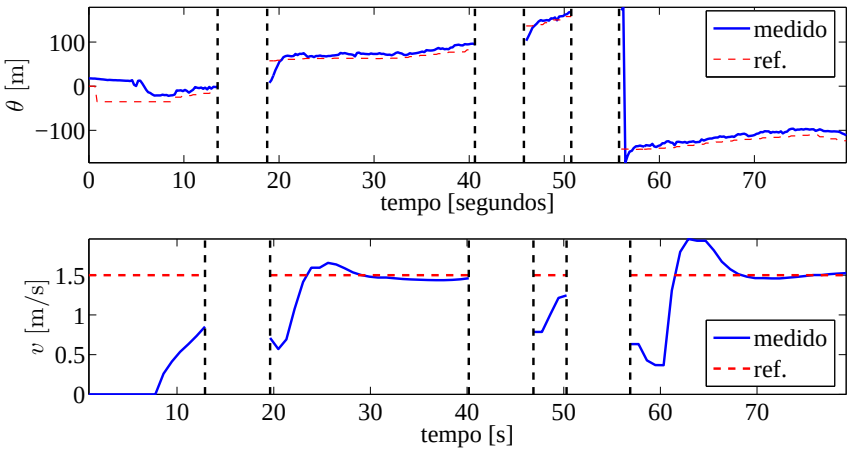


Figura 7: Controle de orientação e velocidade para o primeiro teste.

robot, *Int. Multi-Conference on Systems, Signals & Devices*, IEEE, pp. 1–7.

Anderson, K. and Gaston, K. J. (2013). Lightweight unmanned aerial vehicles will revolutionize spatial ecology, *Frontiers in Ecology and the Environment* **11**(3): 138–146.

Android (2017). último acesso: 06 nov. 2017.

Ayers, P. (1994). Environmental damage from tracked vehicle operation, *Journal of Terramechanics* **31**(3): 173–183.

De Luca, A., Oriolo, G. and Samson, C. (1998). Feedback control of a nonholonomic car-like robot, *Robot motion planning and control* pp. 171–253.

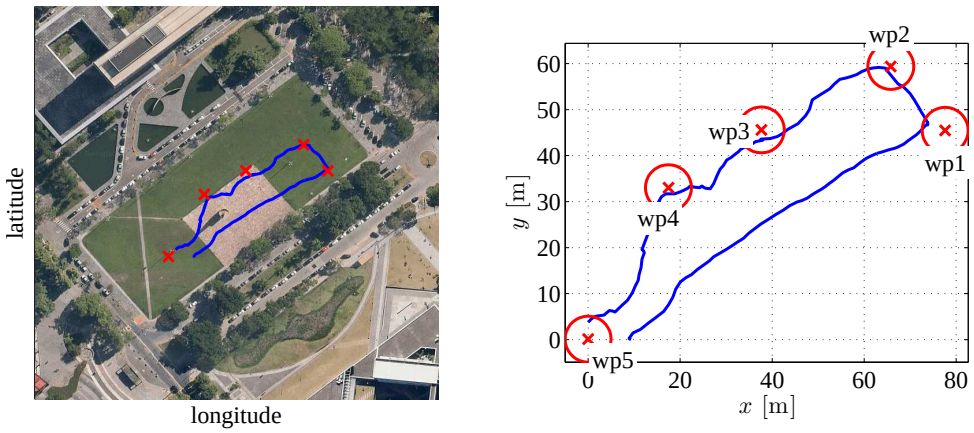


Figura 8: Teste 2: caminho percorrido pelo robô entre 5 alvos, terreno inclinado (fonte *Google Maps*).

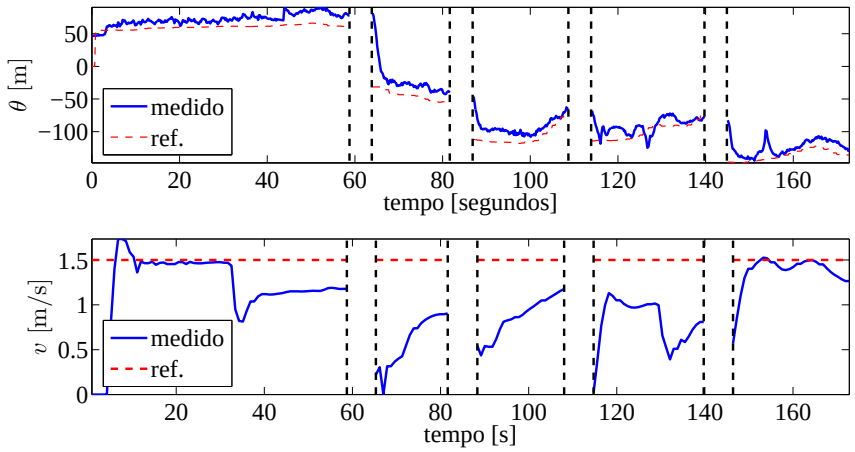


Figura 9: Controle de orientação e velocidade para o segundo teste.

Gobor, Z., Lammers, P. S. and Martinov, M. (2013). Development of a mechatronic intra-row weeding system with rotational hoeing tools: Theoretical approach and simulation, *Computers and electronics in agriculture* **98**: 166–174.

Grocholsky, B., Keller, J., Kumar, V. and Pappas, G. (2006). Cooperative air and ground surveillance, *IEEE Robotics & Automation Magazine* **13**(3): 16–25.

Martinez, D., Teixidó, M., Font, D., Moreno, J., Tresanchez, M., Marco, S. and Palacín, J. (2014). Ambient intelligence application based on environmental measurements performed with an assistant mobile robot, *Sensors* **14**(4): 6045–6055.

Paneque-Gálvez, J., McCall, M. K., Napoletano, B. M., Wich, S. A. and Koh, L. P. (2014). Small drones for community-based forest monitoring: An assessment of their feasibility and potential in tropical areas, *Forests* **5**(6): 1481–1507.

Valente, J., Sanz, D., Barrientos, A., Cerro, J. d., Ribeiro, Á. and Rossi, C. (2011). An air-ground wireless sensor network for crop monitoring, *Sensors* **11**(6): 6088–6108.