LEONARDO SILVEIRA PAIVA

ANÁLISE E GERAÇÃO DE CAMINHO MÍNIMO APLICADO À NAVEGAÇÃO DE VEÍCULOS TERRESTRES

LAVRAS - MG

2017

LEONARDO SILVEIRA PAIVA

ANÁLISE E GERAÇÃO DE CAMINHO MÍNIMO APLICADO À NAVEGAÇÃO DE VEÍCULOS TERRESTRES

Tese apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Engenharia Agrícola, área de concentração: Instrumentação, para a obtenção do Título de Doutor.

Prof. Dr. Giovanni Francisco Rabelo
Orientador

LAVRAS - MG

2017

Ficha catalográfica elaborada pelo Sistema de Geração de Ficha Catalográfica da Biblioteca Universitária da UFLA, com dados informados pelo(a) próprio(a) autor(a).

Paiva, Leonardo Silveira.
Análise e geração de caminho mínimo aplicado à navegação de veículos terrestres / Leonardo Silveira Paiva. - 2017.
141 p.

Orientador: Giovanni Francisco Rabelo.
Tese (doutorado) - Universidade Federal de Lavras, 2017.
Bibliografia.

1. Robótica. 2. Agricultura. 3. Otimização. I. Rabelo, Giovanni Francisco. II. Título.

LEONARDO SILVEIRA PAIVA

ANÁLISE E GERAÇÃO DE CAMINHO MÍNIMO APLICADO À NAVEGAÇÃO DE VEÍCULOS TERRESTRES

ANALYSIS AND GENERATION OF MINIMUM PATHWAY APPLIED TO NAVIGATION OF LAND VEHICLES

Tese apresentada à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Engenharia Agrícola, área de concentração em Instrumentação, para a obtenção do Título de Doutor.

APROVADA em 10 de Abril de 2017.

Prof. Dr.

Prof. Dr. Giovanni Francisco Rabelo
Orientador

LAVRAS - MG

2017

AGRADECIMENTOS

O presente trabalho não é apenas resultado de esforço individual. Surgiu como resultado de contribuições de várias instituições e entidades individuais ao longo da formação profissional, acadêmica e social.

Deste modo, agradeço a todos que de uma forma ou de outra contribuiram para a minha formação e para a realização deste trabalho, especialmente à Giovanni Francisco Rabelo, Giovani Bernardes Vitor, Guillaume Bresson, Amar Ramdane Cherif, Victor Etgens e Arthur de Miranda Neto, pela dedicação, confiança, apoio e disponibilidade ao longo desta jornada.

Aos meus pais, Geraldo e Eulina (in memorian), que se enveredaram ao suor diário para me proporcionar conhecimento vitalício. Aos meus irmãos, Luiz e Edivânio, que juntos me deram apoio durante vários momentos difíceis da vida familiar, mas sempre permanecemos unidos, valorizando e enriquecendo o laço que nos une e que nos torna fortes e vencedores. Às cunhadas e sobrinhos que completam minha amada família.

A todos os meus amigos que se fizeram família ao longo dos anos. Um agradecimento especial a Júlio Cesar, Vânia, Amanda, João Marcos, Daniel Bernadino, Walclee, Alisson, Juliano e Daniel Resende.

Agradeço especialmente a Laurent Garnier que, durante todo o tempo que estive longe de amigos e familiares, esteve presente e suprindo toda falta que a distância traz, me possibilitou ver grandes belezas deste mundo e ver a beleza em querer bem e estar bem ao lado de alguém.

Muito obrigado a todos!

RESUMO

O objetivo deste trabalho foi desenvolver um conjunto de rotinas de otimização capazes de gerar um caminho sub-ótimo para um conjunto de plataformas móveis, levando em consideração a menor distância entre dois pontos previamente estabelecidos, e respeitando um conjunto de restrições impostas ao longo do ambiente, bem como as restrições cinemáticas de cada plataforma. Aplico-se um conjunto de algoritmos de otimização que recebeu o mapa de um ambiente estático de maneira off-line e, a partir do tratamento desta imagem, retornou o menor caminho existente entre dois pontos. A trajetória então percorrida pelo robô móvel foi aquela que apresentou a menor distância e que satisfez todas as restrições impostas ao longo do ambiente. Em conjunto com as técnicas de otimização, foi utilizado um estimador de estados, caracterizado como o Filtro de Kalman, para análise e controle da trajetória ao longo de um conjunto de rotas previamente estabelecidas pelo planejador. Para o controle do sistema, foi implementado um controlador do tipo Proporcional, Integral e Derivativo, de forma a minimizar o erro gerado quando a plataforma móvel se afastava do caminho ótimo. Tratando-se de um pacote computacional formado por um conjunto de algoritmos de otimização, foi desenvolvido um módulo didático, onde o usuário pode ter acesso a todas as funcionalidades deste trabalho por meio de uma interface gráfica de fácil manipulação. Tal interface atuou como agente facilitador de ensino e aprendizagem na área da robótica móvel. A técnica explorada neste estudo fez uso de um amplo conjunto de algoritmos de otimização para maior probabilidade de se encontrar um caminho executável como também desenvolveu um conjunto de robôs móveis com diferentes arquiteturas, para validar a metodologia proposta, a qual deve ser aplicável a uma ampla variedade de problemas de controle não linear na agricultura.

Palavras-chave: Robótica. Agricultura. Otimização.

ABSTRACT

The objective of this work was to develop a set of routines that are capable of generating a sub-optimal path for a set of mobile platforms, taking into account the smaller distance between two previously established points, and respecting a set of constraints imposed over the environment, as well as the kinematic constraints of each platform. The study applied to set of optimization algorithms that in an offline way received the map of a static environment, and from the treatment of this image returned the smallest path between two points. The trajectory then traveled by the mobile robot was the one that presented the shortest distance and that satisfied all the restrictions imposed throughout the environment. In conjunction with the optimization techniques, a state estimator, characterized as the Kalman Filter, was used to analyze and control the trajectory along a set of routes previously established by the planner. For control of the system, a Proportional, Integral and Derivative controller was implemented in order to minimize the error generated when the mobile platform moved away from the optimum path. As a computational package formed by a set of optimization algorithms, a didactic module was developed, once the user has access to all the functionalities of this work through a graphical interface of easy manipulation. This interface served as a facilitator of teaching and learning in the area of mobile robotics. The technique explored made use of a large set of optimization algorithms to be able to find an executable path, as well as to develop a set of mobile robots with different architectures to validate the proposed methodology that should be applicable to a wide variety of problems of nonlinear control in agriculture.

Keywords: Robotics. Agriculture. Optimization.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Diante a diversidade de interesses existente entre as áreas da engenharia, uma tecnologia que vem se apresentando como potencial ferramenta de aplicação em sistemas de segurança é a navegação autônoma e semi-autônoma. A navegação semi e/ou autônoma de veículos terrestres em ambientes considerados desconhecidos tem sido impulsionada pelos mais variados objetivos, entre eles a exploração espacial, propósitos militares, atividades domésticas, sistemas de auxilio ao condutor, dentre outras (VICTORINO; RIVES, 2006).

Segundo Niku (2013), os robôs podem ser classificados de acordo com a constituição de sua estrutura, nomeadamente robôs móveis e robôs industriais. Os robôs móveis têm a capacidade de se movimentar em seu ambiente e não estão fixados em um local físico. Em contraste, os robôs industriais geralmente consistem de um braço articulado (manipulador com múltiplos graus de liberdade) e um elemento final que está ligado a uma superfície fixa.

A robótica móvel possui uma área de aplicação muito diversificada e, por possuir um espaço de trabalho maior, comparado ao robô industrial, é utilizada em uma variedade maior de aplicações. Apesar da grande versatilidade de aplicações da robótica móvel, existem algumas desvantagens como restrições de *design* que incluem o peso da plataforma e a curta durabilidade do sistema de alimentação que, na maioria dos casos, é feita por bateria. Este último é um grave problema e que, por consequência, pode interromper uma operação em execução. Por outro lado, os robôs industriais são normalmente fixados a uma determinada posição e não se sujeitam a tais restrições.

A utilização da robótica na agricultura é um processo cuja implantação tem ocorrido de forma lenta, devido a uma série de fatores, dos quais destaca-se a fragilidade das máquinas, tecnologia mecânica dispendiosa, trabalho sob limite da capacidade da máquina, bem como a eficiência do trabalho ainda a ser

melhorada e adaptada às diversas situações (INAMASU et al., 2011). Outro fator que merece destaque é o uso de tecnologias que sejam flexíveis a um ambiente que pode ser alterado, de forma que os robôs possam ser adaptados para mais de uma aplicação, ou que possam prover-se de ferramentas que permitam mais de uma solução para um determinado objetivo.

Para a agricultura, os robôs precisam navegar de forma autônoma em seu ambiente e realizar ações em locais pré-determinados como, por exemplo, na colheita de frutas, pulverização, semeadura, coleta de imagens de uma planta ou fazer medições (INAMASU et al., 2011). As estufas são ambientes mais simples para a movimentação de robôs, pois possuem um ambiente controlado e cuidadosamente projetado para que uma plataforma possa se movimentar sem colisões até ao local desejado. No caso da agricultura ao ar livre, os robôs trabalham recebendo um conjunto de informações que traduzem em um mapa previamente definido de todo o ambiente e dos pontos que devem ser visitados no campo. Segundo Earl, Thomas e Blackmore (2000), quando as trajetórias de uma plataforma móvel são conhecidas, a mesma pode utilizar um posicionamento à base de GPS e um controle de malha fechada para se certificar de que ele permanece na pista. Quando a tarefa é seguir uma trajetória desconhecida, a visão computacional é frequentemente utilizada para permitir que o robô encontre o caminho certo.

A multidisciplinaridade e o avanço das tecnologias empregadas no campo passam a caracterizar novas práticas e oportunidades de avanços na agricultura, e a inserção de agentes autônomos nas atividades agrícolas se torna uma realidade presente a cada dia. Este, por sua vez, deve ser analisado como uma ferramenta facilitadora que irá compor e incrementar a produtividade de todo o sistema agrícola ao qual esteja inserido.

1.1 Motivação e Objetivos

Este projeto tinha como objetivo contribuir para o desenvolvimento de um algoritmo de precisão, aplicado à geração de caminho mínimo a ser utilizado por uma plataforma robótica móvel, com foco de aplicações em maquinários mais robustos tais como, máquinas agrícolas e sistemas de transporte de grandes empreendimentos agrícolas, como fazendas sucroalcooleiras, composta por veículos leves (caminhonetes, carros e motos) e caminhões de transporte de insumos e dos produtos resultantes do processo produtivo.

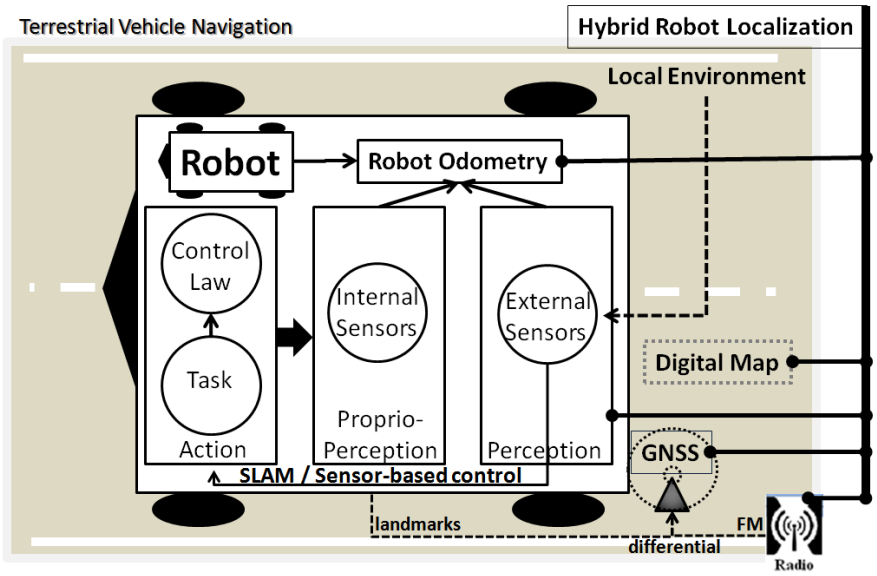
A técnica para o planejamento da rota foi realizada a partir de conceitos de programação não linear off-line, estabelecendo uma ferramenta para a determinação da direção durante a realização do movimento, baseado em um conjunto de algoritmos de otimização. O planejamento de tal trajetória foi realizado em ambiente estático e estruturado, toda a informação sobre os obstáculos era conhecida a priori, e o movimento do robô foi estabelecido a partir desta informação.

Foi também enfatizada a aplicação de algoritmos de otimização para a geração de trajetórias de uma plataforma (veículo/robô) móvel na agricultura, com aplicações diversificadas, que incluem aspetos relativos a agricultura de precisão, tais como as operações de plantio e colheita, ordenhas automáticas, e os sistemas autônomos de aspersão em irrigação. Tanto para a aplicação em sistemas de assistência ao condutor, quanto para a navegação de veículos autônomos, esta proposta de pesquisa interdisciplinar propõe estudos a fim de desenvolver uma técnica de geração de trajetória de forma a capacitar o planejamento de movimentos de uma plataforma móvel, sem a necessidade de interferência humana. Para a geração de trajetórias, foi proposta uma solução baseada no caminho obtido pelo planejador, de forma a gerar o menor trajeto existente entre dois pontos previamente determinados.

O planejador citado anteriormente, refere-se a um conjunto de algoritmos de otimização que irá receber o mapa de uma determinada região estática de forma off-line e, a partir de tal informação, gerar o menor percurso possível entre o ponto de partida e o ponto de chegada. Após o processamento das informações, o caminho final a ser percorrido pelo robô será determinado pelo algoritmo que gera a menor distância existente entre tais pontos, levando em consideração as restrições cinemáticas da plataforma móvel e as restrições impostas pelo ambiente.

Os sistemas de localização precisa que se baseiam em sistema diferencial (de alto custo e restrito em distância) são considerados comuns aos sistemas de geração de trajetórias. Contudo, a localização precisa é uma temática de pesquisa atual e tem valor tecnológico estratégico elevado, como apresentado no diagrama da Figura 1.

Figura 1 – Proposta: Localização Terrestre Híbrida.



Fonte: Victorino (2006).

Neste sentido, este trabalho propôs também um estudo a fim de obter um modelo de localização que poderá prezar a factibilidade por meio da utilização de dados de sensores considerados de baixo custo, como câmeras, radares/sonares, giroscópios/acelerômetros, GNSS, etc. A fusão dos dados destes sensores será realizada por meio do Filtro de Kalman, que se caracteriza como uma abordagem padrão para a integração de dados, mas que requer ajustes cuidadosos, a fim de alcançar resultados de qualidade. Isso cria uma motivação para o uso de um Filtro de Kalman que seja capaz de se adaptar às diferentes circunstâncias e aos sensores.

Nesta perspectiva, foi desenvolvida uma interface computacional para que a utilidade das técnicas implementadas neste estudo se estendam a mais aplicações e sejam utilizados como facilitadores no ensino de robótica. Ao provar a robustez dos algoritmos implementados e a versatilidade de suas aplicações, este módulo de ensino será disponibilizado como ferramenta de ensino para o curso de Engenharia de Controle e Automação da Universidade Federal de Lavras, e para os demais cursos e discentes interessados nesta área de conhecimento.

Contudo, além do desenvolvimento de um banco de dados com diversas técnicas de otimização, foi também construido um conjunto de plataformas móveis que poderão fazer uso das técnicas propostas para a realização de um determinado objetivo. Tais plataformas serão apresentadas na parte final deste trabalho, juntamente com os detalhes da estrutura física, modelo, módulos de programação e respectivas variáveis de entrada e saída, e o ambiente onde ocorreram as respectivas utilizações.

1.2 Justificativa

Para uma plataforma autônoma seguir um caminho desejado, deve estar equipada com três tipos de sistemas caracterizados como sistemas de controle,

sistemas de navegação e sistemas de orientação (SASIADEK; HARTANA, 2000). O controle é utilizado para determinar os comandos que devem ser enviados aos atuadores do veículo para uma determinada ação desejada (velocidade, aceleração, etc.). O sistema de navegação é utilizado para fornecer a estimativa de posição e de velocidade do veículo, enquanto que os sistemas de orientação são utilizados para determinar a trajetória ideal a uma determinada posição e velocidade desejada.

Para o problema de posicionamento, geralmente são utilizados sensores internos como, por exemplo, os encoders, giroscópios, acelerômetros e bússolas, cuja finalidade é medir as variáveis físicas no veículo e sensores externos como sonar, radares, entre outros que medem a relação existente entre o veículo e o ambiente. Ambos os sensores possuem vantagens e desvantagens. Por curto período de tempo, as medições efetuadas usando sensores internos são mais precisas, entretanto, para longos períodos, tais medições são afetadas por erros que produzem divergências da ordem de quilômetros entre a posição real e a estimada. Tratando-se de sensores externos, estes não produzem as mesmas divergências citadas anteriormente, contudo, as suas medições nem sempre estão disponíveis durante o percurso total realizado (SANTINI; NICOSIA; NANNI, 1997).

Para obter um resultado ideal e de forma a combinar as vantagens de se utilizar mais de um tipo de sensoriamento, é necessário o uso de uma técnica que combine todas as informações disponíveis. O método normalmente utilizado é a fusão de todos os dados, que visam à combinação de informações provindas de múltiplas fontes de medição. Na literatura cientifica, este termo é conhecido como Fusão de Dados (*Data Fusion*), Fusão de Sensores (*Sensor Fusion*), Integração de Múltiplos Sensores (*Multi-sensor Integration*), Fusão de (Dados de) Múltiplos Sensores (*Multi-sensor* (*Data*) Fusion).

A fusão sensorial tem por objetivo determinar um conjunto de propriedades frequentemente representadas por um vetor de estados, cujas informações sensoriais são combinadas para produzir informações mais precisas. Uma das técnicas continuamente utilizada para a fusão sensorial entre as medições inerciais e outros tipos de sensores é o Filtro de Kalman. O Filtro de Kalman é um observador que estima os estados de um sistema dinâmico linear ou não linear, no qual os estados e as medidas são frequentemente contaminadas por ruídos.

Todo o desenvolvimento tecnológico citado anteriormente trouxe maior produtividade e eficiência econômica ao setor agrosilvopastoril. Inicialmente, a geração de tal eficiência agrícola deu-se por meio da mecanização de campos que gradativamente foram aumentando as suas dimensões, e cuja tendência atual é a substituição das máquinas grandes e pesadas por tecnologias baseadas nas informações que podem propiciar as operações autônomas viáveis e confiáveis em campo (EARL; THOMAS; BLACKMORE, 2000).

Compreende-se a necessidade dessas informações como definido por Hackenharr, Hachenharr e Abreu (2015, p. 125):

> Os agricultores têm necessidade de colher informações sobre a cultura e o solo, seu estado antes e durante a estação de crescimento, como por exemplo: robôs batedores, que podem viajar para um local pré-determinado, retirar uma amostra do solo para identificar os níveis de umidade usando um penetrômetro de cone, tipo de sonda que é inserida no solo para medir a compactação, e utilizar uma sonda elétrica para medir o pH.

Utilizando-se dessa afirmação como justificativa do trabalho, a movimentação de um robô agrícola para um local pré-determinado é realizada levando em consideração todas as restrições existentes ao longo desse caminho, como a presença de obstáculos e restrições físicas da plataforma móvel. Sendo assim, é fundamental o posicionamento preciso no local onde serão colhidas tais informações a um custo mínimo de deslocamento entre a posição inicial e o

ponto de interesse, fazendo-se assim, necessário o uso de um planejador do caminho a ser percorrido.

Este mesmo autor afirma que um fator limitante da presença de robôs em zonas agrícolas corresponde ao consumo de energia que interfere na operação. Tais plataformas podem ser alimentadas por fontes disponíveis no local, como luz solar ou energias renováveis. Entretanto, a otimização de um percurso é fundamental para que um robô possa realizar uma ação pré-determinada a um custo mínimo, o que pode ser alcançado com o planejamento correto do caminho a ser percorrido.

Uma plataforma móvel agrícola possui uma estrutura física desenvolvida de acordo com as características do ambiente, podendo variar de robôs de grande porte utilizados em ambientes livres, como tratores autônomos, a pequenos veículos inteligentes utilizados para aplicações específicas. Segundo Chen (2012), robôs exteriores requerem sistemas de navegação sofisticados para explorar um ambiente desestruturado ou não protegido e, em grande parte, tal exploração diz respeito à navegação e ao planejamento de trajetórias para evitar obstáculos.

A Embrapa, juntamente com o núcleo de pesquisa de Engenharia Mecânica da Universidade de São Paulo, estão em fase de desenvolvimento de uma plataforma robótica móvel para a aquisição de dados em agricultura de precisão, cujo objetivo é capacitar tal plataforma a se locomover em ambientes típicos de regiões agrícolas com a finalidade de coleta de dados. Tal locomoção é fundamental para o sucesso do projeto e o estudo do planejamento de trajetória, e é feito com base em algoritmos de otimização obtidos como função da estrutura física do robô e as restrições do ambiente onde o mesmo irá se locomover (TANGERINO et al., 2011).

Grandes empresas do setor agrícola, tal como fazendas sucroalcooleiras e a indústria de base florestal, possuem frotas com ampla quantidade de veículos

em suas estruturas, compostas por máquinas agrícolas (tratores diversos, colhedoras, etc), caminhões para transporte da produção e de insumos, caminhonetes e motocicletas empregadas na segurança e nos deslocamentos diversos, entre outros sistemas móveis que dão suporte a todo processo produtivo. Estes veículos trafegam em locais caracterizados por grandes dimensões de terra. Neste sentido, sistemas de localização e planejamentos de trajetórias podem ser de grande utilidade no rastreamento, manejo e logística dos sistemas de transporte empregados nestes empreendimentos agrícolas.

No Brasil, a temática de veículos terrestres em ambientes externos é desenvolvida, em muitos casos, de forma descentralizada, em universidades e centros de pesquisa. Os projetos que se destacam atualmente são os seguintes:(a) Projeto CaRINA do Laboratório de Robótica Móvel (LRM) da USP em São Carlos/SP, (b) Projeto VERO (BUENO et al., 2009); (c) Projeto CADU (UNIVERSIDADE FEDERAL DE MINAS GERAIS - UFMG, 2010); e (d) Laboratório de Mobilidade Terrestre (LMT) inaugurado em 2014, com um conceito inovador em veículos inteligentes - automação, comunicação e eficiência energética - no contexto dos Sistemas de Transporte Inteligentes e sediado na Universidade Federal de Lavras.

1.3 Principais Contribuições

O foco principal deste trabalho consiste na resolução de problemas relacionados à geração de um caminho que possa ser executado por uma plataforma móvel sujeita às restrições cinemáticas e com a presença de diferentes tipos e geometria de obstáculos ao longo do ambiente a ser percorrido. Diferentemente do que se encontra nos trabalhos relacionados a essa temática, tal plataforma fará uso de um conjunto de algoritmos de otimização que irá gerar uma solução ao problema proposto e receberá como melhor solução, o caminho

que representa a menor distância existente entre dois pontos previamente estabelecidos.

Os algoritmos utilizados nestes trabalhos são amplamente difundidos na comunidade acadêmica sobre a geração de trajetória para robótica móvel. Fazendo uso de um grande conjunto destes algoritmos, a probabilidade de encontrar um caminho executável é extremamente elevada para diversos cenários e, especificamente, no que tangeu à validação do projeto, pois sempre foi possível gerar um caminho executável para todos os cenários criados.

Apesar de todos os algoritmos implementados fazerem uso das técnicas de programação não-linear, nem todos utilizam o mesmo método de varredura para determinar a solução ótima. Por exemplo, os Algoritmos Genéticos utilizam regras de transição probabilísticas e não determinísticas, enquanto que o Algoritmo A Estrela utiliza uma combinação de aproximações heurísticas em um grafo de um vértice inicial até um vértice final (SUGIHARA; SUZUKI, 1996). Devido a tal fato e dependendo da complexidade do mapa a ser estudado, alguns algoritmos podem ficar presos em mínimos locais e, consequentemente, não retornar um resultado satisfatório. Com base neste projeto, a solução proposta foi a criação de uma base de dados com diversos métodos de pesquisa, a qual irá garantir alta taxa de probabilidade de encontrar um caminho entre dois referenciais.

Além da base de dados compostas pelos algoritmos de otimização, este trabalho também propõe a criação de um módulo facilitador para que diferentes usuários possam fazer uso das técnicas implementadas. Este módulo caracteriza-se como uma interface gráfica, de fácil acesso, intuitiva e com funções que permitem uma manipulação global de todos os códigos e funções desenvolvidas para o objetivo proposto.

Poderá também fazer-se uso de diferentes tipos de plataformas móveis que visam validar a robustez, bem como a versatilidade do uso deste projeto à

diferentes plataformas móveis utilizadas pelos discentes do curso de Engenharia de Controle e Automação da Universidade Federal de Lavras em suas pesquisas.

Ao utilizar-se de um conjunto de diferentes técnicas de otimização de trajetórias, este trabalho poderá também contribuir com um acervo bibliográfico em que se faz um relato técnico e comparativo de várias técnicas, além de utilizar-se apenas de uma técnica especifica para a resolução do mesmo tipo de problemas, como referido na literatura. Esta pesquisa visa sanar esta deficiência, apresentando uma visão global das técnicas e metodologias para a problemática que envolve a geração de trajetórias.

1.4 Organização da tese

Esta tese está estruturada em seis capítulos e dois anexos.

No primeiro capítulo apresenta-se a introdução, abordado o contexto da pesquisa, além da motivação, objetivos e as principais contribuições que levaram à realização do trabalho. Faz-se também a descrição detalhada dos outros capítulos e dos anexos que constituem a tese.

O capítulo 2 corresponde à Revisão Bibliográfica, onde se apresentada a introdução, os principais conceitos associados à problemática da pesquisa e as principais técnicas de otimização utilizadas para a obtenção de um caminho mínimo. O objetivo é enquadrar o leitor ao contexto da pesquisa e à importância de se utilizar critérios ótimos no que se refere à trajetória de uma plataforma móvel.

No capítulo 3, Material e Métodos, faz-se uma descrição detalhada dos procedimentos utilizados para a validação da metodologia proposta. Neste capítulo, é apresentada a plataforma móvel e os cenários utilizados para validar as técnicas de otimização em análise.

No capítulo 4 apresenta-se os resultados obtidos para os algoritmos de otimização implementados na pesquisa. Os resultados são expostos de forma

comparativa, detalhando o que apresentou melhor performance segundo um critério estabelecido, incluíndo uma conclusão que analisa de forma global a robustez e a aplicabilidade das técnicas avaliadas.

No capítulo 5, Interface Gráfica, é apresentado o ambiente computacional desenvolvido em MatLab para a simulação das técnicas de otimização de caminho mínimo utilizadas no estudo. É também analisado o ambiente desenvolvido, apresentando-se os principais módulos utilização para o desenvolvimento da interface.

No capítulo 6, Conclusões e Perspectivas de Trabalhos Futuros, são tecidas as principais conclusões, bem como as perspectivas para trabalhos futuros.

Para finalizar, é apresentado no Anexo A uma base teórica referente aos filtros gaussianos, bem como o Filtro de Kalman, cuja ferramenta foi utilizada neste projeto como um estimador de estados para a localização e controle da plataforma móvel. O anexo B refere-se às diferentes plataformas móveis que fizeram uso da metodologia apresentada neste estudo, bem como uma descrição física de cada plataforma e o trajeto executado.

2 REVISÃO BIBLIOGRÁFICA

2.1 Planejamentos de trajetória

O planejamento de um caminho pode ser entendido como uma questão puramente geométrica, uma vez que implica a geração de um caminho geométrico sem uma lei temporal especificada. Por outro lado, o planejamento de uma trajetória atribui uma lei temporal ao caminho geométrico (RAJA; PUGAZHENTHI, 2012). O planejamento do caminho é certamente um dos tópicos mais interessantes da robótica avançada, caracterizada por um alto grau de autonomia de robôs para aplicações em ambientes hostis (espaço, submarino, nuclear, militar, etc).

A temática que se refere ao planejamento de trajetória é utilizada em diversos campos científicos como robótica, inteligência artificial e teoria de controle. Em relação à robótica, o planejamento de trajetórias preocupa-se em como deslocar uma plataforma móvel de um ponto para outro. Com os avanços dos estudos referentes a este área, o planejamento desse caminho pode incluir outras variáveis como incerteza, comunicação de robôs múltiplos, ou dinâmica da plataforma (LAVALLE, 2006).

O planejamento de trajetórias no estudo da robótica móvel está focado no desenvolvimento de algoritmos que geram movimentos executáveis e que possam ser processados por modelos geométricos simples ou mais complicados (LIU; SUN; ZHU, 2010). O foco principal deste trabalho está no desenvolvimento de tais algoritmos e na obtenção de uma solução de forma a possibilitar o movimento do robô ao longo da solução encontrada, levando em consideração as suas restrições mecânicas e cinemáticas. Além disso, este estudo aborda a automação de um sistema mecânico que possui sensores, atuadores e capacidades de computação e processamento de informações.

Ao realizar a navegação em um determinado ambiente, uma tarefa comum a ser realizada pelo robô é a construção de um mapa que possa descrever tal ambiente e determinar, de forma precisa, a sua localização dentro deste mapa. Tais tarefas estão intrinsicamente relacionadas ao problema designado SLAM (*Simultaneous Localization And Mapping*) (VALÊNCIA; ANDRADE-CETTO; PORTA, 2011). Este problema é conhecido e atualmente há um grande número de soluções códigos abertos abordando essa temática, como por exemplo HectorSLAM e OpenSLAM.

A navegação do robô em um ambiente depende do conhecimento desse ambiente pelo robô. Se o robô não tem qualquer descrição do ambiente, ele usa seus sensores para "sentir" esse ambiente. Com base nesses dados, o robô pode navegar de um lugar para outro. Tal abordagem é chamada de navegação reativa, porque o robô só reage ao estímulo dos sensores (LAVALLE, 2006). No entanto, se o robô tiver um mapa preciso sobre o ambiente alvo, ele pode fazer uso de muitas técnicas para navegar de maneira ideal. Isso é chamado de planejamento de caminho ou navegação global. Neste trabalho, o problema do planejamento do caminho é descrito e vários métodos de planejamento do caminho são avaliados.

A problemática que envolve o planejamento de trajetória de um robô caracteriza-se em determinar um caminho seguro e eficiente para o deslocamento do robô, dado um local de início, um local de destino e um conjunto de obstáculos distribuídos ao longo do espaço de trabalho. Além do problema fundamental, este trabalhado inclui uma maneira de otimizar o plano, ou seja, minimizar o tempo necessário ou a distância a ser percorrida, como desccrito por Ramakrishnan e Zein-Sabatto (2001), Sadati e Taheri (2002) e Wu, Chen e Lee (1996).

Segundo Raja e Pugazhenthi (2012), os métodos mais populares utilizados para a resolução deste tipo de problema são os algoritmos baseados

em gráfico de visibilidade e campo potencial. No entanto, o primeiro carece de flexibilidade e este último é propenso a sofrer dificuldades com os mínimos locais (ALEXOPOULOS; GRIFFIN, 1992; BORENSTEIN; KOREN, 1989; CHEN; LIU, 1997). As pesquisas de Del, Medrano e Martin (2002) e Zarate et al. (2002) apresentam bons resultados, obtidos por meio da rede neural e do Algoritmo Genético, mostrando, assim, grande eficiência na navegação robótica. Entretanto, tais métodos de planejamento baseados em rede neural sempre estabelecem o modelo de rede neural para um robô da posição inicial à posição meta e implica alto custo computacional. Deste modo, como o Algoritmo Genético (AG), cuja busca é baseada nos princípios da genética natural e seleção natural (GOLDBERG, 1989), fornece-se uma pesquisa robusta em espaços complexos, sendo computacionalmente menos pesado do que outros algoritmos de pesquisa. O AG pesquisa a solução a partir de uma população de pontos e é menos provável que seja preso em um ótimo local. Muitos resultados na literatura mostram a boa aplicação do algoritmo genético no planejamento do caminho para plataformas móveis Khoogar e Parker (1991), Noboru e Hideo (1997) e Ram, Arkin e Boone (1994).

De forma geral, a navegação autônoma assume um ambiente conhecido bem como os obstáculos e restrições presentes no mesmo. Os algoritmos utilizados neste estudo devem incluir um planejamento global de forma a planejar o caminho do robô entre os obstáculos conhecidos, bem como o planejamento do caminho local para evitar obstáculos em tempo real (RAMAKRISHNAN; ZEIN-SABATTO, 2001). No âmbito deste trabalho, pode-se afirmar que o problema concerne em planejar um caminho ótimo em um ambiente estático onde as posições de obstáculos são conhecidas a priori. Este planejamento, também designado planejamento estático, tem uma ampla gama de aplicações conforme referido por Wang, Soh e Wang (2002).

Sobre a temática da robótica móvel, o conceito de navegação consiste em direcionar uma plataforma móvel em um espaço de trabalho durante um determinado intervalo de tempo. Esse caminho deve ser perfeitamente executável de forma a levar o robô de uma posição e orientação inicial, para uma posição e orientação final. Essa tarefa é primordial e deve ser realizada envolvendo diversos subproblemas tais como a localização, planejamento de uma trajetória admissível, restrições cinemáticas da plataforma móvel que irá executar tal atividade e a sua execução (LIU; SUN; ZHU, 2010).

Segundo Russell e Norvig (1995), o planejamento de trajetória pode ser subdividido em três etapas: (i) a primeira consiste na definição do espaço de configuração que visa determinar o estado atual do robô definido pela posição, orientação e os ângulos de articulação; (ii) a segunda etapa consiste em determinar um conjunto de estados válidos para a busca de uma solução. Uma das técnicas mais utilizada é a decomposição celular; e (iii) a terceira etapa consiste na busca do espaço de configurações, ou seja, a utilização de um algoritmo que possa determinar a melhor trajetória existente entre dois pontos, de forma a levar o robô a uma posição e orientação desejada. Tendo como base a citação referenciada, este trabalho aborda, principalmente, o subproblema do planejamento de caminhos e a geração de trajetórias dentro de um conjunto de espaço de trabalho, levando em consideração o resultado de um conjunto de algoritmos de otimização que se utiliza de diferentes técnicas para a geração dessa trajetória.

2.2 Otimização de problemas referentes a geração de trajetórias

A teoria da decisão é um conceito aplicado a diversas áreas de conhecimento, e com efeito em muitos problemas de engenharia. O objetivo dessa metodologia é determinar a melhor solução absoluta que corresponde ao mínimo (ou máximo) de uma função objetivo definida, que tende a descrever o

desempenho total do sistema, satisfazendo um conjunto de restrições. Em Duckett e Nehmzow (2002) e Thrun e Liu (2005), o problema de geração de trajetórias de uma plataforma foi tratado como um problema de otimização global, cuja meta era encontrar a menor distância dentro de um determinado espaço.

As técnicas de otimização possuem utilidades em uma grande classe de problemas que não são resolvidos adequadamente por métodos de programação linear. Tais problemas pertencem à classe de funções não convexas e em grande parte apresentam múltiplos ótimos locais, invalidando o uso de técnicas tradicionais. Assim, tais modelos são difíceis de serem resolvidos, e as regras ou estratégias de solução padrão não podem ser aplicadas nestes casos.

A solução de problemas de otimização utilizando métodos determinísticos exige que requisitos de otimização sejam satisfeitos. Esses algoritmos devem reconhecer soluções pontuais e verificar se atendem às necessidades de diferentes condições de otimalidade. Tais restrições impostas pelos métodos determinísticos têm estimulado o desenvolvimento de métodos probabilísticos que não fazem nenhuma imposição sobre a função objetivo, desde que esta seja mensurável. Esses métodos fazem pesquisa exploratória ao longo da região admissível da função objetivo e, em certos casos, transportam uma grande carga computacional para fazer tal pesquisa no espaço de soluções viáveis (LAVALLE, 2006).

Quando se trata de soluções de problemas de programação não-linear, normalmente, aceita-se como solução do problema uma solução local, visto que é inviável obter o ótimo global. De acordo com Barbera et al. (2005), este assunto já foi explorado na literatura, embora não haja algoritmos eficientes para a solução global. Essa afirmação baseia-se no fato de que a busca de um ótimo global é fortemente dependente do processamento dos computadores, além da

localização de um ponto inicial que gere menores interações possíveis na busca da melhor solução.

Para cada classe de um problema de otimização, métodos específicos foram desenvolvidos e uma técnica em geral tem sido pesquisada para estender a uma classe maior de problemas (BARBERA, 2005). Atualmente, estudos sobre estes modelos são baseados em derivadas (*deterministic problems*), probabilística (*drive search*), heurística (*path ant, simulated annealing*) e híbridos.

Algoritmos inteligentes são baseados em modelos estocásticos e são confiáveis para orientar a busca para a solução ótima de forma mais eficiente, mesmo quando a função objetivo apresenta uma topologia mais complexa. A pesquisa ideal do ponto é feita por processos evolutivos na escolha de elementos elegíveis no espaço de soluções.

Técnicas usando probabilidade para estimar a solução de problemas complexos são datadas do século XVIII, com a agulha de Buffon - Georges Louis Leclers (1707-1788). O algoritmo *Simulated Annealing*, que foi inspirado por fenômenos físicos de cristalização e solidificação de materiais, foi proposto em 1953 por N. Metropolis. Em 1975, Jhon Holland propôs o algoritmo genético com base em processos evolutivos naturais de Darwin.

O estudo de algoritmos para resolver problemas relacionados a veículos autônomos alcançou grande projeção na comunidade acadêmica ao longo dos últimos 20 anos. As soluções com os melhores resultados foram obtidas ao abordar o problema com base em técnicas probabilísticas, de modo que todas as fontes de incerteza envolvidas no processo eram analisadas. Segundo Thrun (2002), tais algoritmos são baseados no teorema de Bayes.

2.3 Algoritmos de otimização para geração de caminho mínimo

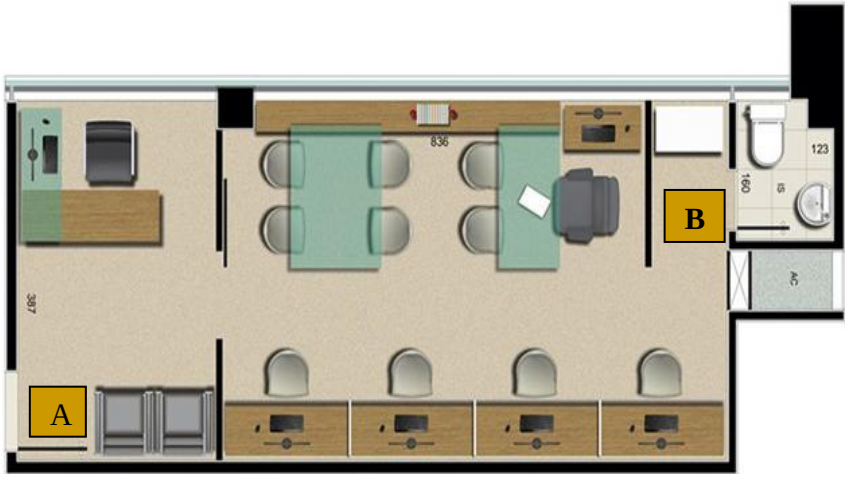
Observa-se nas últimas décadas uma crescente evolução nas pesquisas referentes à navegação robótica, relacionada diretamente com a qualidade de vida que tal tecnologia pode proporcionar (CHOSET, 2001; GERAERTS; OVERMARS, 2002). A principal problemática que envolve a geração de trajetórias reside em como encontrar um caminho entre o início e a meta num determinado cenário, abrangendo toda a área de um espaço de configuração, ou patrulhando uma área de interesse (AGMON; KRAUS; KAMINKA, 2008). Vários são os benefícios gerados pela automação de robôs, o que permite que a sociedade otimize a produção de insumos em um determinado intervalo de tempo, quando comparado ao trabalho manual. Tal automação permite o uso de robôs em diversos segmentos, dentre eles os aéreos, terrestres e aquáticos. Isso leva a uma necessidade de desenvolver algoritmos de navegação de robôs bem sucedidos e confiáveis.

Nesta seção será descrito um conjunto de algoritmos baseados em heurísticas desenvolvidas para abordar os desafios associados ao planejamento de trajetórias de plataformas móveis, seja em cenários construídos para validação especifica, seja para cenários do mundo real. Cada um dos algoritmos apresentados foi verificado em um conjunto de situações descritas em mapas com diferentes configurações que visam avaliar a robustez do método utilizado. Tais algoritmos são de conhecimento da comunidade acadêmica e as bibliografias referentes ao seu formalismo matemático podem ser de fácil acesso. Neste sentido, a sua descrição neste estudo é feita com base numa introdução, análise dos pontos fortes, fraquezas e diretrizes para a escolher de um algoritmo particular em relação a outro. Embora esses algoritmos tenham sido documentados e descritos individualmente, uma análise comparativa de todos os métodos utilizados neste trabalho encontra-se indisponível na literatura.

O planejamento consiste em encontrar uma sequência de ações que transforma algum estado inicial em algum estado de meta desejado. Segundo Ferguson, Likhachev e Stentz (2005), os estados do planejamento do caminho são localizações de agentes e transições existentes entre estados que representam ações que o agente pode tomar, cada uma das quais com um custo associado. Um caminho é ótimo se a soma de seus custos de transição for mínima em todos os caminhos possíveis que vão da posição inicial à posição final. Um algoritmo de planejamento é considerado completo se sempre for capaz de encontrar um caminho em tempo finito nas situações em que existe e, caso contrário, infere uma informação. Da mesma forma, um algoritmo de planejamento é considerado ótimo se sempre for capaz de encontrar um caminho ideal, respeitando as restrições impostas.

Várias abordagens são apresentadas na literatura com o objetivo de determinar caminhos, dada alguma representação do ambiente. Em geral, as duas técnicas mais populares são os algoritmos determinísticos, baseados em heurística (HART; NILSSON; RAFAEL, 1968; NILSSON, 1980) e os algoritmos randomizados ou probabilísticos (KAVRAKI et al., 1996; LAVALLE; KUFFNER, 1999, 2001). Neste estudo, cada algoritmo será descrito e uma trajetória baseada no cenário representado pela Figura 2 será apresentada ao final de cada descrição.

Figura 2 - Cenário modelo dado com ponto inicial e final para a geração de um caminho mínimo.



Fonte: Glaybson Parreiras (2017)

2.3.1 A estrela (A*)

Quando se descreve os problemas que envolvem a busca entre dois pontos de interesse, o melhor resultado é determinado pelo caminho que envolve o menor custo segundo uma função objetivo pré-definida. Neste sentindo, um dos algoritmos mais difundidos na comunidade acadêmica e amplamente utilizado para a solução de problemas que envolvem a determinação de uma trajetória foi descrito pela primeira vez, em 1968, por Peter Hart, Nils Nilsson, e Bertram Raphael (NOSRATI; KARIMI; HASANVAND, 2012). Inicialmente, ele foi designado “algoritmo A” e, logo após o uso, este algoritmo com uma heurística apropriada atingiu um comportamento ótimo, e passou a ser conhecido por “A estrela” ou abreviadamente A*.

O algoritmo A* foi desenvolvido para superar o custo computacional e o tempo de processamento incorrido quando se utiliza grafos em problemas de busca. Este algoritmo baseia-se na utilização de uma heurística, definida para

guiar o processo de busca (DUCHOŇ et al., 2014). Tal heurística estima o custo de deslocamento da posição de origem à posição de destino, minimizando o tamanho da árvore de pesquisa, o que tende a acelerar todo o processo exploratório. Adicionando-se a este fato, o algoritmo acompanha o custo necessário para alcançar cada posição, definido por g_n. Sendo assim, o valor estimado da solução de menor custo passando por n caminhos pode ser expresso por $f_n = g_n + h_n$.

Outra característica relevante que eleva a qualidade do A*, é o fato dele utilizar o conceito de listas abertas e fechadas para controlar a visita realizada em cada posição (NOSRATI; KARIMI; HASANVAND, 2012). Isto significa que o algoritmo mantém os registros de todas as posições que foram alcançadas nas listas abertas, mas que ainda não foram visitadas e expandidas, enquanto que, na lista fechada, o algoritmo mantém todos os registros de posições que já foram previamente visitadas e expandidas. Devido a este fato, pode-se afirmar que, em uma busca exploratória, desde que haja tempo e memória ilimitada, o algoritmo sempre tende a encontrar o menor caminho, caso exista, mesmo que a função heurística seja inexata e admissível.

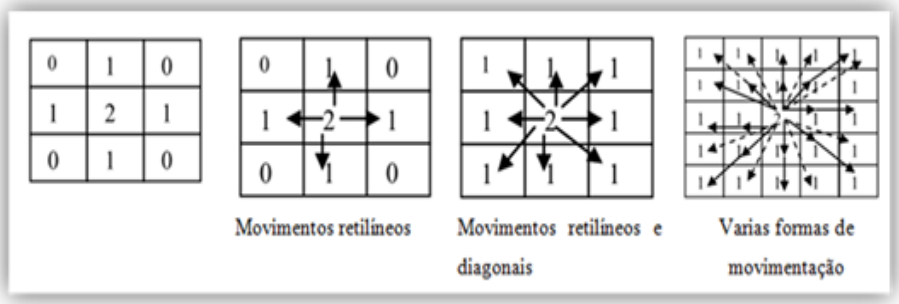
Para iniciar a resolução de problemas utilizado o algoritmo A*, é necessário primeiramente iniciar a modelagem do respectivo problema como um algoritmo de pesquisa de gráfico padrão. O algoritmo A* baseia-se nesse gráfico como uma entrada, e explora, uma a uma, todas as regiões, encontrando o caminho mais curto do ponto inicial para todos os pontos das regiões exploradas, sendo tendencioso para as regiões mais próximas do destino especificado pela função heurística.

O algoritmo A* implementado neste estudo funciona em espaços discretos, o que faz necessário discretizar o ambiente em que a plataforma irá realizar a exploração e tornar este mapa do ambiente uma matriz de pixels

discretos. No entanto, de forma que o algoritmo em questão ocupe um peso computacional reduzido em um mapa de alta resolução, esta foi, portanto, reduzida para obter melhores resultados com um tempo de computação aceitável.

O mapa discretizado pode ser caracterizado com um gráfico que consiste em vértices e arestas. Cada pixel do mapa de resolução reduzida é tomado como um vértice, e cada vértice tem um número de conexões que atuam como arestas, desde que a conexão seja livre de colisão. Para qualquer posição geral da plataforma móvel, as conexões possíveis são dadas pela Figura 3, em que a posição assumida pelo robô é marcada como '2' e todos os movimentos possíveis são indicados por 1, enquanto que os movimentos impossíveis são indicados por 0.

Figura 3 - Possíveis conexões para o algoritmo A Estrela.

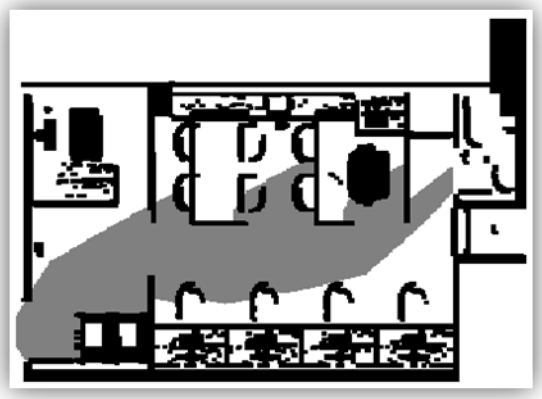


Fonte: Kala (2014).

A função de custo implementada para o algoritmo A * inclui os pesos das arestas, que são definidas como a distância euclidiana entre os pontos de conexão, e a função heurística que denota a proximidade do ponto de origem ao ponto de destino. Ambas as funções são definidas em arquivos separados e podem ser alteradas de acordo com a necessidade do usuário. O conjunto de

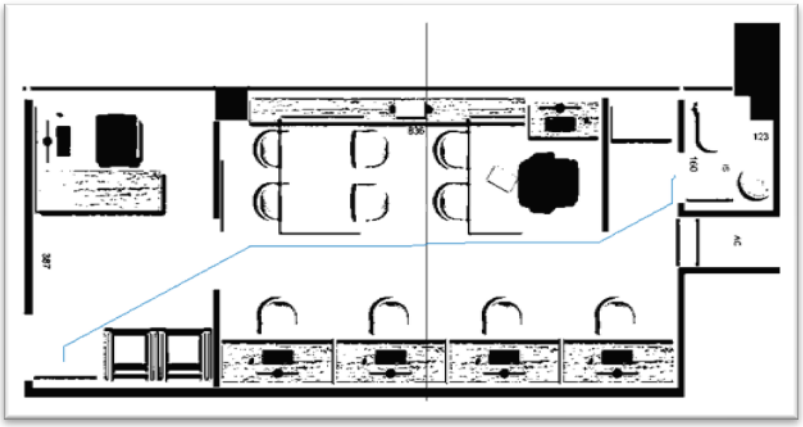
imagens apresentadas nas Figuras 4 e 5 ilustra o método de busca do algoritmo A* bem como o caminho referente à menor distância existente entre os dois pontos previamente definidos.

Figura 4 - Busca inicial realizada pelo algoritmo A estrela para o cenário teste.



Fonte: Glaybson Parreiras (2017)

Figura 5 - Resultado obtido pelo algoritmo A estrela para o cenário teste.



Fonte: Glaybson Parreiras (2017)

2.3.2 Mapas de rotas probabilístico

Dentre os algoritmos utilizados para a definição do caminho existente entre dois pontos, um dos que se caracteriza como mais eficiente e que apresenta bons resultados em um tempo computacional reduzido é definido como Mapa de Rotas Probabilístico (MRP). Este algoritmo pode calcular caminhos sem colisões para qualquer tipo de plataforma robótica que deva se mover entre obstáculos estacionários. No entanto, o MRP é normalmente utilizado para resolver os problemas de planejamento de trajetória para robôs com vários graus de liberdade em ambientes estáticos (KAVRAKI et al., 1996), embora existam algoritmos baseados no MRP que são utilizados em ambientes dinâmicos.

O MRP é constituído de duas fases, a fase de aprendizado e a de questionamento. Na fase de aprendizado, o robô gera um conhecimento sobre a região que deve ser explorada, e este conhecimento é armazenado em um grafo. Após a geração do mapa de rotas, o algoritmo faz um pós-processamento (expansão) com o objetivo de aumentar a conectividade do mapa de rotas nas regiões mais difíceis do espaço de trabalho (KAVRAKI et al., 2002). Utilizando-se de uma abordagem probabilística, as rotas são definidas de forma incremental. O método identifica heuristicamente as regiões difíceis no espaço livre e gera configurações adicionais do robô, de forma a aumentar a conectividade nesses locais. O objetivo final é fazer com que a maior carga de processamento fique na etapa de aprendizado sobre o ambiente, e que a fase de questionamento seja praticamente instantânea.

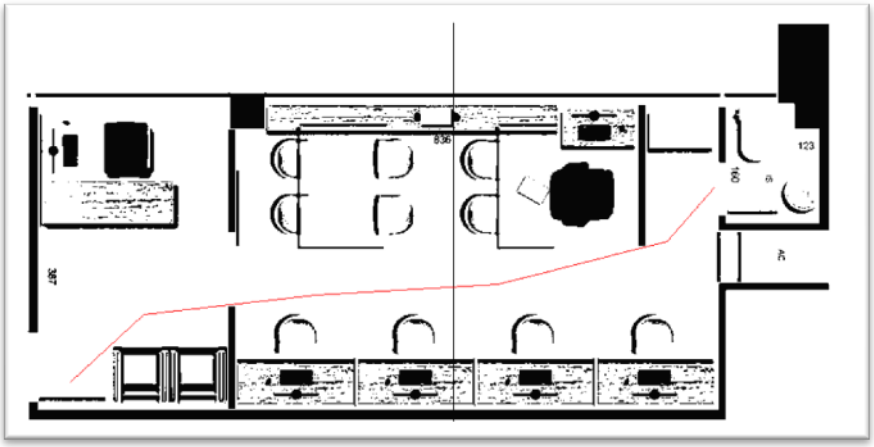
A definição da rota a ser percorrida é aleatoriamente definida em um pequeno gráfico em toda a área de trabalho. Todos os vértices e arestas do gráfico devem estar livres de colisões, de modo que um robô possa usar o mesmo gráfico para o planejamento de movimentos. O MRP seleciona um número de pontos aleatórios no espaço de trabalho como os vértices. Para qualificar-se como vértice, um ponto selecionado aleatoriamente não deve estar

dentro de obstáculos. Para qualificar o resultado do algoritmo, tanto melhor seria à medida que aumentasse o número de vértices, desde que não gerasse uma perda computacional muito grande. Então, o algoritmo tenta conectar todos os pares de vértices selecionados aleatoriamente. Se quaisquer dos vértices pode ser conectados por uma linha reta, essa linha é adicionada como uma aresta (KAVRAKI et al., 2002). Este conceito é ilustrado nas Figuras 6 e 7.

Figura 6 - Estágios 1 e 2 para a geração do caminho mínimo no cenário teste, utilizando o algoritmo MRP.

Fonte: Glaybson Parreiras (2017)

Figura 7 - Resultado obtido utilizando o algoritmo MRP para o cenário teste.



Fonte: Glaybson Parreiras (2017)

Assim como o algoritmo A* citado anteriormente, os pesos das arestas são descritas como distâncias euclidianas existentes entre os pontos de conexão, e a função heurística (denotando a proximidade do objetivo) é definida como a distância euclidiana do ponto de origem ao ponto de destino. Ambas as funções são definidas em arquivos separados e podem ser alteradas de acordo com a necessidade do usuário.

2.3.3 Campo Potencial

A navegação baseada em campo potencial é uma técnica na qual se assumem obstáculos que carregam cargas elétricas, e o campo potencial escalar resultante é usado para representar o espaço livre. As colisões entre os obstáculos e o robô são evitadas por uma força repulsiva existente entre eles, dada pelo gradiente negativo do campo potencial. As funções potenciais foram independentemente investigadas em Kore e Borenstein (1991). De acordo com Weijun, Rui e Chongchong (2010), as funções potenciais são usadas para evitar obstáculos, mas o uso mais comum da abordagem em campo potencial é para o

planejamento de trajetória local. O trabalho realizado por Krogh e Thorpe (1986) usou uma função de custo potencial para projetar um caminho ótimo para um robô circular em duas dimensões. Neste algoritmo, todos os obstáculos repelem a plataforma móvel com uma magnitude inversamente proporcional à distância, dado como objetivo a atração de tal plataforma móvel ao ponto de interesse. O potencial resultante, que representa os componentes de atração e repulsão, é medido e utilizado para a movimentação do robô.

Dentre as principais motivações que levam a implementação deste algoritmo, pode-se destacar a representação do espaço para que o planejamento de uma trajetória possa ser feito em nível global. Um campo potencial contínuo dá uma boa indicação sobre as distâncias e formas dos obstáculos, de modo que, as mudanças necessárias na posição e na orientação da plataforma possam ser feitas de uma maneira suave e contínua. Ao detectar colisões, a complexidade combinatória de detecção de interseção realizada com representações geométricas é evitada (GE; CUI, 2002).

O algoritmo implementado neste trabalho contém dois módulos, um planejador global e um planejador local. O planejador global usa uma descrição global do espaço livre, dada por uma rede dos vales potenciais mínimos, e seleciona um caminho candidato que provavelmente será livre de colisões. Por sua vez, o planejador local modifica o caminho do candidato para evitar colisões e otimiza localmente o comprimento do caminho e a suavidade dos movimentos. Dependendo da complexidade do mapa a ser explorado, o algoritmo pode não convergir a uma solução, dado o fato de que ele pode ficar "preso" em uma região local.

A função de custo implementada é usada para indicar o comprimento de um percurso a ser executado. Um caminho é dado como uma sequência de nós e bordas conectados. O comprimento e o custo de uma aresta corresponde à distância euclidiana existente entre os nós das extremidades da borda. O custo de

um nó mede a dificuldade de colocar o robô no local do nó e é medido em termos da distância até o obstáculo mais próximo.

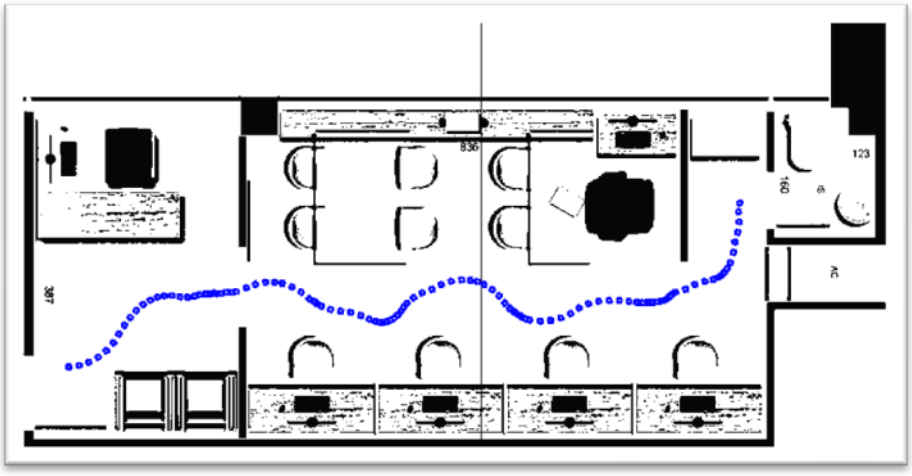
O trabalho realizado por Bin-Qiang, Ming-Fu e Yi (2011) apresenta um método de navegação, que combina o conceito de obstáculo virtual com um método baseado em campo potencial para manobrar robôs cilíndricos móveis em ambientes desconhecidos. A simulação por computador e as experiências de seu método ilustram o desempenho aceito e a capacidade de resolver o problema de mínimos locais relacionado com o método de campo potencial. Uma nova função potencial é proposta por Ge (2002) fazendo consideração em ambientes dinâmicos que contêm obstáculos e metas em movimento. O Campo Potencial Artificial Evolutivo (EAPF) para o planejamento do caminho robótico em tempo real tem sido proposto por Vadakkepat, Tan e Ming-Liang (2000), cuja combinação entre o algoritmo genético e o método de campo de potencial artificial é introduzida para derivar funções de campo potencial ótimas.

Inspirado no trabalho desenvolvido por Gue, Gao e Cui (2013), o algoritmo desenvolvido neste trabalho, pode ser resumido nas seguintes etapas:

a) O espaço livre é representado por um gráfico que consiste em um número finito de nós e arestas, correspondentes a pontos e bordas ao longo dos vales potenciais mínimos;
b) A cada nó é atribuído um custo, dependendo da largura de espaço livre no nó;
c) Um caminho candidato é encontrado com base na minimização da distância entre dois pontos e possibilidade de colisões;
d) Se um caminho sem colisões não puder ser encontrado, o algoritmo é interrompido, indicando a impossibilidade de geração de uma rota;
e) Caso uma solução seja determinada, informa-se a trajetória a ser realizada.

A Figura 8 ilustra o resultado obtido pelo algoritmo Campo Potencial, explicitando o menor caminho existente entre dois pontos previamente definidos.

Figura 8 - Resultado obtido pelo algoritmo Campo Potencial para o cenário teste.



Fonte: Glaybson Parreiras (2017)

2.3.4 Árvores aleatórias de rápida exploração

Durante as duas últimas décadas, diversos algoritmos para a resolução de problemas que envolvem a geração de um caminho a ser percorrido por uma plataforma robótica foram desenvolvidos e apresentados com bons resultados. O MRP e o Campo Potencial são tidos como os mais populares dentre estas classes de algoritmos. Entretanto, surgiram novas necessidades relacionadas aos problemas da robótica móvel, e com isso a necessidade de um algoritmo para o planejamento de trajetórias que levassem em consideração o Planejamento kinodinâmico, ou seja, o planejamento de movimentos que envolvem a dinâmica da plataforma móvel (velocidades e acelerações) se fez necessário (LAVALLE, 1998). O algoritmo "Árvores Aleatórias de Rápida Exploração", também

conhecido como RRT, do inglês *Rapidly-Exploring Random Trees*, foi desenvolvido, substituindo os conceitos relacionados ao espaço de configurações por conceitos relacionados ao espaço de estados (STENTZ, 1994).

O algoritmo RRT tende a realizar uma busca de soluções em espaços de grandes dimensões que possuem restrições provenientes da existência de obstáculos e restrições advindas de equações diferenciais referentes à dinâmica do sistema. Uma das grandes vantagens desse tipo de planejamento é o de resolver a problemática referente ao planejamento de rotas e o problema de planejamento de trajetórias, considerando, assim, as velocidades e as acelerações da plataforma móvel. Além disso, no planejamento kinodinâmico, é considerado o sinal de controle que faz o robô deslocar-se de uma configuração para outra (LAVALLE, 1998).

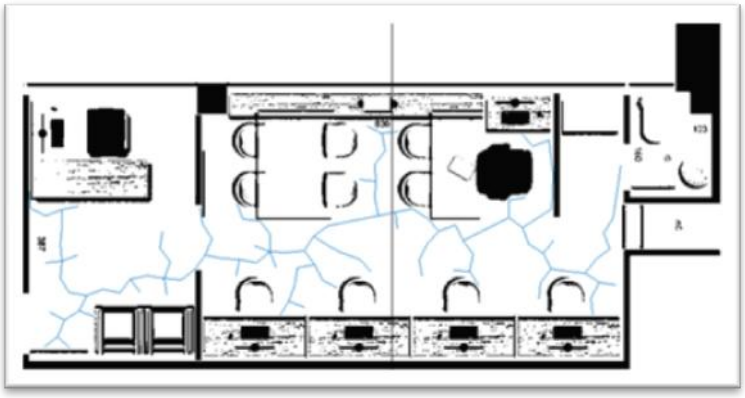
A maior motivação para o desenvolvimento deste tipo de algoritmos é que uma geração de rota puramente cinemática pode levar a uma rota não executável pelo robô, o que pode ser evitado utilizando esta classe de algoritmos.

LaValle (1998) destacou as seguintes vantagens referentes ao algoritmo RRT: 1) a expansão de um RRT é fortemente tendenciosa para porções inexploradas do espaço de estado; 2) a distribuição de vértices em um RRT se aproxima da distribuição de amostragem, levando a um comportamento consistente; 3) um RRT é probabilisticamente completo em condições muito gerais; 4) o algoritmo RRT é relativamente simples, o que facilita a análise de desempenho; 5) um RRT permanece sempre conectado, mesmo que o número de arestas seja mínimo; 6) um RRT pode ser considerado como um módulo de planejamento de trajetórias, que pode ser adaptado e incorporado em uma ampla variedade de sistemas de planejamento; 7) algoritmos de planejamento de trajetória inteira podem ser construídos sem requerer a capacidade de orientar o

sistema entre dois estados prescritos, o que amplia consideravelmente a aplicabilidade do RRT.

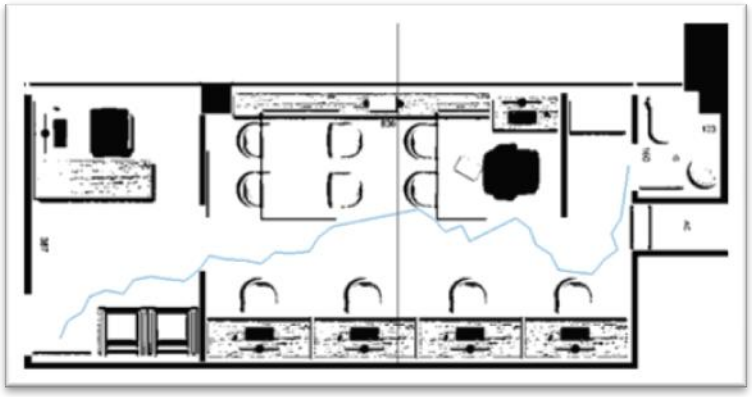
Neste trabalho, a implementação do RRT foi feita de tal forma que o algoritmo expandisse o seu espaço de busca em uma espécie de árvore onde cada nó da árvore é um ponto no espaço de trabalho. A área explorada pelo algoritmo é a área ocupada pela árvore. Inicialmente, o algoritmo começou com uma árvore que tinha fonte como o único nó. Em cada iteração, a árvore foi expandida selecionando um estado aleatório e expandindo a árvore para esse estado. A expansão foi feita estendendo o nó mais próximo na árvore para o estado aleatório selecionado por um pequeno passo. O algoritmo foi executado até que alguma expansão levasse a árvore próxima o suficiente ao ponto de destino. A expansão da árvore foi realizada em direção ao ponto de destino. As Figuras 9 e 10 ilustram o conceito de árvore descrito acima.

Figura 9 - Exploração inicial realizada pelo algoritmo RRT para o cenário teste.



Fonte: Glaybson Parreiras (2017)

Figura 10 - Resultado obtido utilizando o algoritmo RRT para o cenário teste.

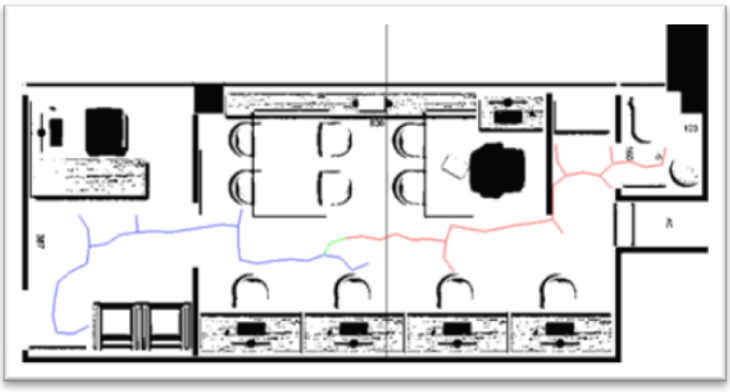


Fonte: Glaybson Parreiras (2017)

2.3.5 Árvores aleatórias de rápida exploração bidirecionais

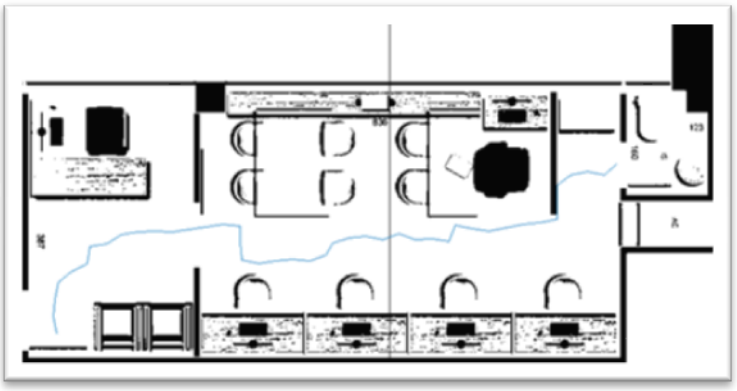
Assim como o RRT, o bidirecional RRT também explora o espaço de pesquisa usando o conceito de árvores. No entanto, ao invés de uma árvore indo da origem ao destino, há a presença de duas árvores, a primeira que inicia a partir da fonte e cresce em direção à meta e a segunda que inicia a partir da meta e cresce em direção à fonte. Quando ambas as árvores se encontram, uma solução é encontrada. As Figuras 11 e 12 ilustram o conceito de duas árvores descrito a cima.

Figura 11 - Exploração inicial realizada pelo algoritmo bidirecional RRT para o cenário teste.



Fonte: Glaybson Parreiras (2017)

Figura 12 - Resultado obtido pelo algoritmo bidirecional RRT para o cenário teste.



Fonte: Glaybson Parreiras (2017)

A versão bidirecional do algoritmo RRT divide a carga computacional entre a exploração do espaço de estados a ser percorrido pela plataforma robótica e crescimento das árvores, uma em direção a outra, até o momento em que ambas são conectadas e a solução é encontrada (RAJA; PUGAZHENTHI, 2012). Esta versão é adequada para casos em que a dinâmica não é levada em

consideração, ou ainda nos casos em que ela é considerada em uma etapa posterior ao processo de planejamento de rotas.

2.3.6 Algoritmos Genéticos

Há muito tempo que a natureza tem sido objeto de inspiração para as leis que regem o dia-a-dia e que foram formuladas com base na observação do meio onde se vive. Como, por exemplo, existem as leis da termodinâmica e as leis de Newton. Como exemplos de objetos inspirados na natureza, pode-se citar: os aviões (pássaros), sonares (morcegos), coletes a prova de balas (teias de aranha), velcro (plantas). Na mesma filosofia de criação, surgiram os Algoritmos Genéticos como uma área de pesquisa que se insere no ramo das técnicas de Inteligência Artificial (IA), cuja base para a resolução dos problemas fundamenta-se nas teorias desenvolvidas por Charles Darwin, a respeito da evolução natural de espécies. Existem atualmente inúmeras áreas de pesquisa que fazem parte da computação natural e, além dos algoritmos genéticos, também podem ser citadas a Lógica Nebulosa, a Inteligência de Enxame e as Redes Neurais Artificiais.

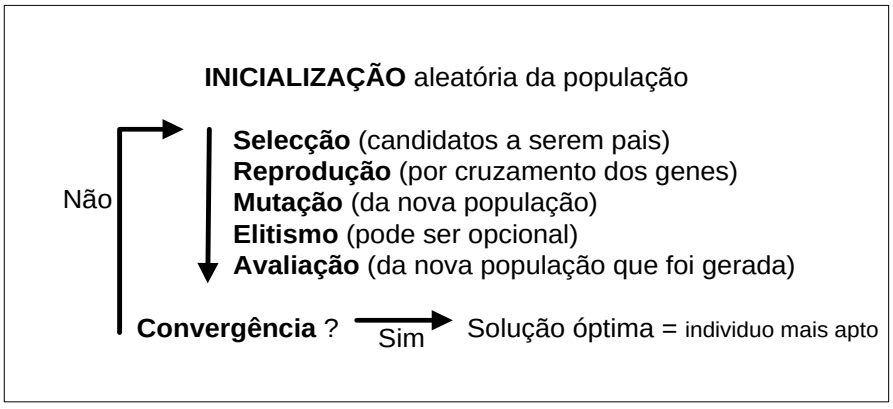
O conceito dos AGs foi introduzido por John Holland e pelos seus colegas da Universidade de Michigan, nos Estados Unidos, entre os anos de 1960 e 1970 (MITCHELL, 1999). Atualmente, a sua popularidade ganha espaço com inúmeros trabalhos e aplicações, principalmente, em projetos de otimização desenvolvidos ao longo dos últimos anos. Apesar de os AGs serem aplicados a uma série de problemas que, até então, não eram facilmente resolvidos com técnicas tradicionais de otimização, eles possuem certas limitações que, de acordo com Cecília Reis (2009)[1], podem ser destacadas em: 1) Dificuldades em definir corretamente a representação para o problema; 2) Pode ocorrer convergência prematura; 3) O problema da escolha dos diversos parâmetros:

[1] Cecília Reis. Apontamentos da disciplina algoritmos genéticos (Instituto Superior de Engenharia do Porto, primeiro semestre 2009). ALGEN: algoritmos genéticos.

tamanho da população, taxa de cruzamento, taxa de mutação, método de seleção; 4) Não é bom para a identificação dos ótimos locais; 5) Não existe uma condição de fim eficaz; 6) Requerem grande número de avaliações da função objetivo;

Inspirado na maneira de como o Darwinismo explica o processo de evolução das espécies, Holland (1975), no seu livro "*Adaptation in Natural and Artificial Systems*" decompôs o funcionamento dos AGs, como uma abstração da evolução biológica, cujas etapas se compreendem em inicialização, avaliação, seleção, cruzamento, mutação, atualização e finalização, conforme apresentado na Figura 13.

Figura 13 - Estrutura de funcionamento de um AG tradicional.



Fonte: Do Autor (2015).

O que um AG faz é, basicamente, criar uma população de possíveis respostas para o problema a ser tratado para depois, submetê-la ao processo de evolução, constituído pelas seguintes etapas:

a) **Avaliação**: avalia-se a aptidão das soluções (indivíduos da população) para que se estabeleça o quão bem elas respondem ao problema proposto.

b) **Seleção**: indivíduos são selecionados para a reprodução. A probabilidade de uma dada solução *i* ser selecionada é proporcional à sua aptidão.
c) **Cruzamento**: os pais são cruzados com uma determinada probabilidade para gerar os descendentes. Se nenhum cruzamento for realizado, a prole é a cópia exata dos pais.
d) **Mutação**: características dos indivíduos resultantes do processo de reprodução são alteradas, acrescentando assim variedade a população.
e) **Atualização**: os indivíduos criados nesta geração são inseridos na população.
f) **Finalização**: verifica-se que as condições de encerramento da evolução foram atingidas, retornando para a etapa de avaliação em caso negativo, e encerrando a execução em caso positivo.

A partir do AG proposto por Holland, várias inovações ocorreram no decorrer dos anos, entre elas os vários tipos de cruzamento, seleção, técnicas de convergência e outras técnicas determinísticas para o código dos GAs. Tão importante como estas inovações é a própria evolução do Algoritmo Genético no sentido de suprimir as falhas a partir das suas derivações.

Algoritmo Genético Híbrido (AGH), também conhecido como Algoritmo Memético (AM), é o algoritmo desenvolvido com o intuito de conjugar as potencialidades do genético com procura local. Sabe-se que os AGs são eficazes no que diz respeito à procura global, mas apresentam deficiência quando se trata de refinar as soluções encontradas. Isto acontece porque, na fase final das interações, os indivíduos da população compartilham praticamente o mesmo código genético, e qualquer melhoria que venha a acontecer nesta fase

deve-se ao operador de mutação que, além de atuar com uma baixa probabilidade, age de forma aleatória.

Vários são os motivos que levam ao estudo e desenvolvimento do AGH, por exemplo, a grande variedade de aplicações que obtiveram sucesso ao utilizarem versões híbridas desse algoritmo para resolver determinados tipos de problemas. Dentre as aplicações, cita-se o bom resultado obtido por Davis (1992), Mitchell, (1999) e Ochi (2009), ao utilizar métodos de busca local em algoritmos genéticos, de forma a melhorar o resultado por meio do refinamento da solução. O AG pode atingir a região perto de um ponto ótimo, porém, pode necessitar de muitas avaliações da função para atingir esta convergência (MITCHELL, 1999).

Devido à característica dos AGs serem resistentes a ficar presos em ótimos locais e falharem a refinação das soluções encontradas, seguiu-se o conselho de "Hibridizar sempre que possível" (DAVIS, 1992, p. 57). Segundo Dias, Captivo e Climaco (2005), o sucesso dos algoritmos meméticos pode ser explicado como sendo uma consequência direta das diferentes abordagens que este assunto incorpora.

Neste trabalho, utilizou-se um algoritmo de otimização padrão. Utilizando-se os conceitos apresentado do AG, a maior dificuldade não se dá na forma como se inserem os parâmetros do algoritmo, mas sim na forma em como modelar o sistema e fornecer tais informações para o correto funcionamento dos parâmetros do algoritmo. A modelagem do problema se inicia com a especificação da função objetivo juntamente com as suas restrições. O algoritmo implementado segue os seguintes passos:

a) Especificar um conjunto de pontos fixos ao longo do caminho livre do mapa fornecido;

b) A fim de obter um caminho livre de restrições, inicia-se da origem e faz-se a conexão destes pontos por meio de uma linha reta, que caracteriza a menor distância entre os pontos;
c) O primeiro ponto é conectado ao segundo ponto por uma linha reta, e assim por diante. No final, o último ponto é ligado à meta.
d) A função objetivo é o comprimento desse caminho;
e) Uma penalidade pesada é adicionada se qualquer parte do caminho estiver dentro de um determinado obstáculo;
f) As localizações de cada um deste número de pontos fixos (posições dos eixos X e Y) são as variáveis de otimização;
g) Todos os pontos obtidos em ambos os eixos formam o indivíduo genético utilizado para a otimização.

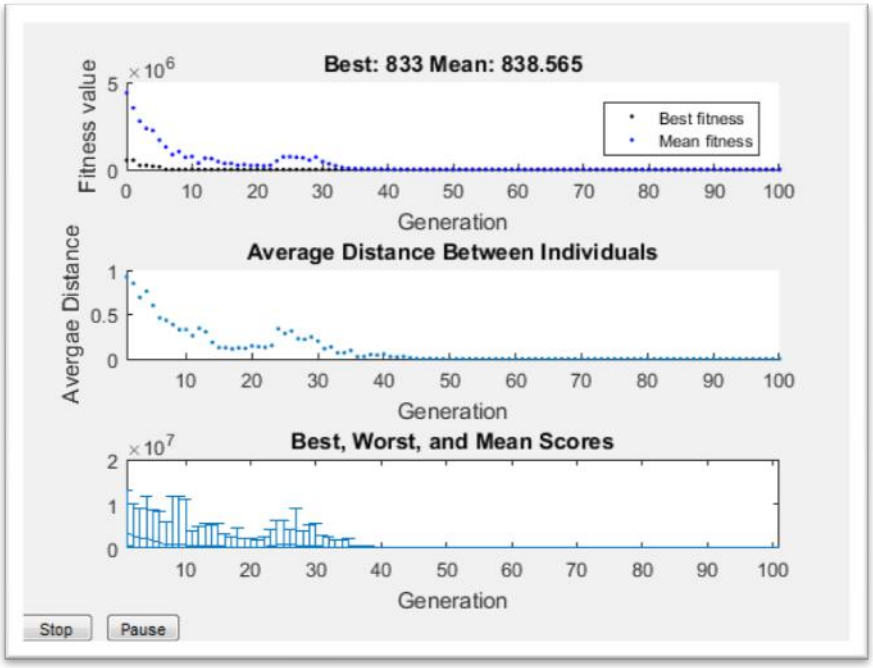
A função custo implementada leva em consideração apenas a distância do percurso para mover um robô de um local inicial para um local final. O cromossomo com a função de menor custo é dito ter a melhor solução e cada cromossomo será lido para obter a coordenada final de cada caminho possível. Se a coordenada final corresponder ao destino desejado do robô, o valor da distância diferencial será zero, o que, por sua vez, produzirá um valor de função de custo mais baixo. A distância diferencial é uma métrica usada para identificar a diferença existente entre a localização final produzida por cada cromossomo e a localização final desejada. Se um caminho sugerido colidir com um obstáculo, a função de custo aumentará drasticamente, portanto, a melhor solução será dita como aquela que possuir a menor distância diferencial, o menor número de mudanças de direção e o menor número de colisões com obstáculos.

Cada ponto, ao representar uma localização ao longo do caminho, marca um ponto de contorno que a plataforma móvel deve realizar, e o número total de pontos é um parâmetro de algoritmo e deve ser igual ao número máximo de

voltas que um robô deve fazer ao longo do mapa fornecido. Caso a carga computacional seja uma restrição de programação, um grande número de pontos ao longo do caminho seria inviável.

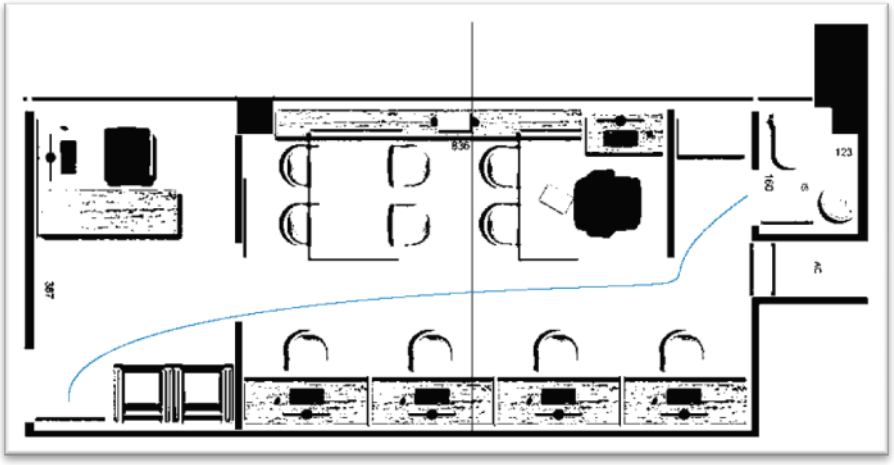
As figuras 14 e 15 representam o caminho obtido utilizando-se do AG desenvolvido e a evolução do algoritmo ao longo de todo o processo de busca.

Figura 14 - Evolução do AG para a geração do caminho mínimo no cenário teste.



Fonte: Do Autor (2017).

Figura 15 - Resultado Obtido pelo AG para o cenário teste.



Fonte: Glaybson Parreiras (2017)

2.4 Filtros Gaussianos

A maioria das abordagens para a resolução de problemas inerentes a plataformas autônomas (localização, mapeamento, determinação de trajetória, posição, etc) são classificados em duas categorias principais, a saber: Filtros e Algoritmos de otimização (Grisetti; STACHNISS; BURGARD, 2005). A abordagem feita com o uso de filtros, baseia-se na estimação de estado em que o vetor de estado inclui a pose do robô e todas as posições de marca terrestres, que são usados para representar o mapa. Este vetor é atualizado à medida que o robô recebe novas medições do ambiente, e tais medidas servem para integrar o estado atual do robô, produzindo uma melhor estimativa do seu posicionamento. Pode-se citar como exemplos para estes métodos os filtros de Kalman, filtros de partículas e filtros de informação. Todos estes métodos de filtragem são diferentes instâncias do filtro Bayesiano correspondente à estrutura matemática que permite modelar a convicção sobre o mundo e melhorar essa crença de forma incremental, através de medições realizadas no mundo físico.

Em contraste com as abordagens de filtragem, o processo de otimização visa suavizar toda a trajetória do robô com base em todo o conjunto de medições recolhido ao longo do tempo. Devido a este fato, a resolução de tal abordagem é usualmente resolvida usando técnicas de mínimos quadrados para a minimização do erro. A grande vantagem dessas abordagens é que a otimização pode ser revista e retificada, desde que o conjunto completo de dados seja mantido durante todo o processo de mapeamento.

É essencial que um robô (veículo) tenha capacidades de auto-localização e saber se referenciar em torno do ambiente durante a navegação. O robô deve estar equipado com um conjunto de medidas obtidas a partir de odometria e sensores exteroceptivos para resolver uma determinada tarefa. A odometria é utilizada para a localização, enquanto que as medições do sensor exteroceptivos

são utilizados para o mapeamento (BEGUM et al., 2006). Assim, o mesmo conjunto de medições é utilizado para resolver tanto o problema de trajetória, localização, quanto o de mapeamento.

No entanto, um algoritmo eficiente deve tratar as seguintes questões: (i) medidas de erros e de incertezas decorrentes de vários tipos de sensores; (ii) o problema da correspondência entre duas medições diferentes do mesmo lugar, tomado em dois pontos diferentes de tempo; (iii) o problema de fechamento de loop que ocorre quando a correspondência é realizada num ambiente cíclico, dado que o robô precisa reconhecer que visitou o mesmo local mais cedo e interpretar as medições do sensor corretamente, além da complexidade de tempo, dado que o algoritmo deve ter a capacidade de processamento de dados de sensores em tempo real. As subseções a seguir apresentam as principais abordagens para resolver o problema de localização e mapeamento de trajetórias. A primeira análise foi baseada nos principais filtros utilizados, seguido dos principais algoritmos desenvolvidos para a resolução deste problema.

2.4.1 Abordagens Probabilísticas

A partir da definição clássica para os problemas relacionados a navegação autônoma, desenvolveu-se várias classes de algoritmos probabilísticos. Neste trabalho serão detalhados os principais algoritmos que servem como base para a grande parte da variação que surgiu com o desenvolvimento do problema. Ao longo do texto são citadas as principais referências, inclusive a análise matemática sobre os mesmos assuntos.

2.4.2 O Filtro de Bayes

O algoritmo mais geral utilizado para determinar a probabilidade de cálculo é dado pelo algoritmo do filtro de Bayes. Este é um algoritmo recursivo probabilístico que estima uma determinada variável, dependendo de outras

variáveis, por exemplo, determinar o valor de uma variável que não pode ser medida diretamente (a posição de um veículo), mas que pode fornecer informações sobre outras variáveis (medições do sensor, movimentos executados e mapa do meio ambiente).

A ideia é que o algoritmo possa calcular a posição no tempo t, conhecendo a posição no momento t-1, o movimento executado no instante t e o sensor de medição no mesmo instante de tempo. Para isso, o algoritmo consiste em duas etapas: uma de predição e outra de correção.

a) A etapa de predição utiliza um modelo de transição de estado para calcular uma previsão do novo estado, dado o estado anterior e as entradas de controle.
b) A atualização de correção ou medição corrige esta previsão usando as medições e um modelo de sensor.

A Tabela 1 apresenta a regra de atualização em um único passo do algoritmo do filtro de Bayes. Esta regra de atualização é aplicada de forma recursiva para calcular a crença ($\mathrm{bel}(x_t)$) a partir da crença ($\mathrm{bel}(x_{t-1})$), calculada anteriormente (THRUN, 2002).

Tabela 1 - The general algorithm for Bayes filtering.

Algorithm Bayes_filter$(bel(x_{t-1}), u_t, z_t$

for all x_t **do**

$$\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) d_x$$

$$bel(x_t) = \eta\, p(z_t|x_t) \overline{bel}(x_t)$$

endfor

return $bel(x_t)$

Fonte: Do Autor (2017).

Na linha 3, ele processa o controle u_t através do cálculo de uma crença sobre o estado x_t baseada na crença anterior sobre o estado (x_{t-1}) e o controle u_t. Em particular, a crença $\overline{bel}(x_t)$ que o robô atribui para indicar o estado x_t é obtida pela integral do produto de duas distribuições: o valor anteriormente atribuído a x_{t-1}, e a probabilidade de que o controle u_t induz uma transição de x_{t-1} para x_t.

Na linha 4, o algoritmo de filtro de Bayes multiplica a crença $bel(x_t)$ pela probabilidade de que podem ter sido observado o valor z_t na medição. Ele faz isso para cada estado posterior hipotético x_t. O fator η é um normalizador que garante a integral sobre o valor posterior ser igual a 1.

Todo o cálculo pode ser continuado de forma recursiva quando o passo posterior de tempo t torna-se prioritário para estimar o novo posterior em t + 1. A fim de iniciar a recursão, o primeiro valor de P(x0) tem de ser conhecido. Para além destas condições, os modelos de transição de estado e medições devem ser fornecidos.

2.4.3 Filtros Gaussianos

O filtro Gaussiano é uma generalização do filtro de Bayes, onde todas as distribuições de probabilidade são gaussianas. Segundo Thrun (2002), filtros de Gauss constituem as primeiras implementações do filtro de Bayes para espaços contínuos. As técnicas de Gauss compartilham a ideia básica de que as crenças são representadas por distribuições normais multivariadas, dadas pela Equação (1).

$$p(x) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right\} \quad (1)$$

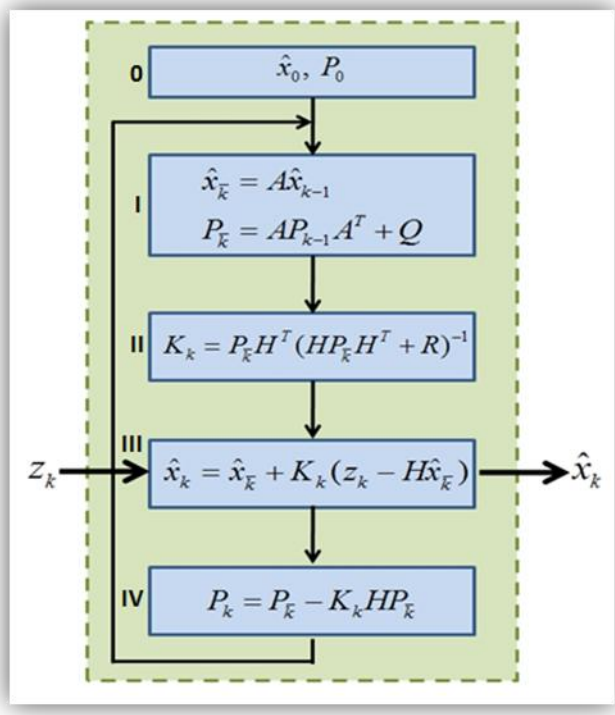
A variável $\mathbf{x}$ é caracterizada por dois conjuntos de parâmetros: A média μ e a covariância $\sum$. A média μ é um vetor que possui a mesma dimensionalidade como o estado $\mathbf{x}$.

As próximas seções descrevem dois algoritmos Gaussianos básicos. A maioria dos autores descrevem estes algoritmos com base em abordagens matemática e suas derivadas (Thrun; Liu, 2005). Os algoritmos para os filtros Gaussianos já foram estabelecidos de forma clara, e as seções seguintes descrevem os principais tópicos a serem destacados na análise desses filtros.

2.4.4 O Filtro de Kalman

O Filtro de Kalman é um filtro gaussiano com função de transição de estado e medida linear, e pode ser aplicado quando o comportamento do sistema for linear e contínuo. A Figura 16 representa todo o processo computacional para o Filtro de Kalman sob a forma de um diagrama esquemático. Ao analisar a estrutura mostrada na Figura 16 pode-se ver que existe apenas uma entrada (medição $\mathbf{z_k}$) e retorna uma única saída (estimativa, $\hat{\mathbf{x}}_\mathbf{k}$), além de que todo o processo interno é realizado em quatro passos.

Figura 16 – O algoritmo do Filtro de Kalman.



Fonte: Do Autor (2016).

O primeiro passo é referente à predição. Nota-se que os valores obtidos para as variáveis $\hat{x}_k^-$, P_k^-são usados no passo seguinte (III e IV), respectivamente, e os valores para A e Q são definidos pelo modelo do sistema. O sobrescrito "-" significa valor previsto.

A segunda etapa é referente o cálculo do ganho de Kalman, K_k. A variável P_k^- calculada na etapa anterior é utilizada neste processo. Os valores de H e R são determinados fora do Filtro de Kalman, de acordo com as

características do modelo do sistema. O valor de ganho de Kalman é utilizado como ponderação no passo seguinte que calcula uma estimativa.

Na etapa III, uma estimativa é calculada a partir de uma medição dada como entrada. O objetivo deste passo é calcular a estimativa, que é o produto final do Filtro de Kalman. Nesta fórmula. z_k significa medição e $\hat{x}_k^-$ significa previsão, e esta variável é necessária devido a recursividade do filtro.

A última etapa, especificada por IV é usada para calcular a covariância do erro, que indica o grau de precisão da estimativa. Em outras palavras, a covariância de erro indica a diferença existente entre a estimativa de Filtro de Kalman e o verdadeiro valor, mas desconhecido. Esse valor pode se entender como grau de precisão da estimativa, porque a maioria das decisões em confiar ou descartar a estimativa calculada na etapa anterior é feita com base na análise da covariância de erro.

Na Figura 1, foi descrito o Filtro de Kalman em quatro etapas mas, o mesmo filtro pode ser divido em duas partes de acordo com o seu significado. O primeiro é um processo predição, onde a estimativa $\hat{x}_k$ poderá variar se o tempo variar de t_k para t_{k+1}, e o segundo é um processo de estimação, quando estão a estimativa ($\hat{x}_k$) e o erro de covariância (P_k). A partir desta perspectiva, a função do Filtro de Kalman pode ser resumida como:

a) Prever o estado $\hat{x}_k^-$ e erro covariância P_k^- para o próximo ponto do tempo, com base no modelo do sistema (A e Q).
b) Compensar a diferença existente entre a medição e previsão, e calcular a nova estimativa, que é o resultado final do Filtro de Kalman: $\widehat{x_k}$, P_k.
c) Efetuar um loop através de dois passos acima descritos.

2.4.5 Filtro de Kalman Extendido

Dadas as inúmeras aplicações do Filtro de Kalman, muitas pesquisas relacionadas ao assunto ainda estão em fase de desenvolvimento e adaptações do algoritmo, e estão sendo aplicadas para sistemas não-lineares. O Filtro de Kalman foi inicialmente desenvolvido para sistemas lineares, o que é uma grande limitação, dado que a maioria dos sistemas em torno de uma realidade não são lineares.

A proliferação do método desenvolvido para a aplicação do Filtro de Kalman em sistemas não-lineares deu-se de forma rápida após sua implementação, sendo caracterizado como Filtro de Kalman Estendido, ou "Extended Kalman Filter" (EKF). Antes de descrever o EKF é importante prestar atenção para o conceito do termo "linearização do Filtro de Kalman", visto que há uma ligeira diferença de modelos de sistema entre o Filtro de Kalman e o Filtro de Kalman Estendido.

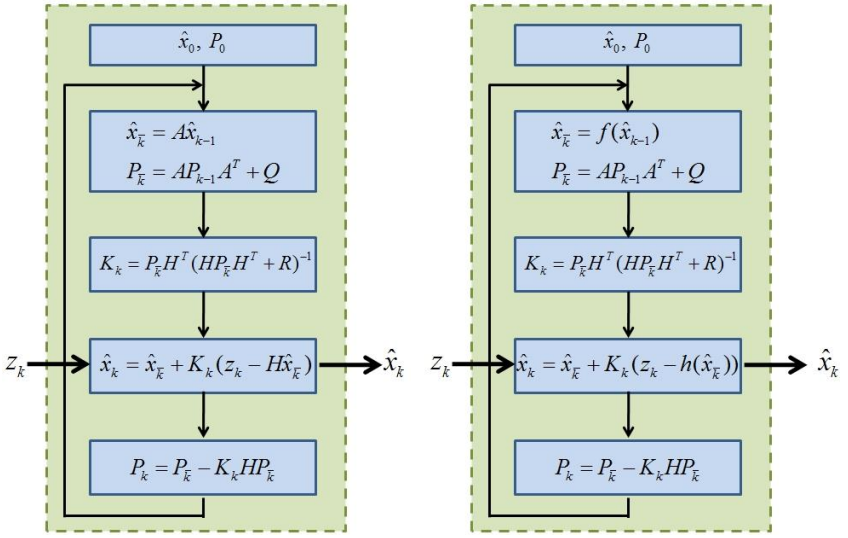
Quando se efetua a aplicação de um Filtro de Kalman a um sistema não linear, a atenção é em particular sobre a gama de funcionamento do sistema, porque um modelo linearizado é próximo do sistema real apenas em torno do ponto onde ocorre a linearização. Fora deste intervalo, a validade do modelo não pode ser garantida. Em termos gerais, pode-se dizer que o sucesso e o fracasso do Filtro de Kalman extendido é totalmente dependente do modelo linearizado de um sistema não-linear.

A análise sobre o EKF será feita comparando o Filtro de Kalman estendido com o Filtro de Kalman linear, tal como ilustra a Figura 17, que é um diagrama de fluxo esquemático dos dois filtros, onde se pode observar a ligeira diferença na expressão em cada passo. Observa-se que a principal diferença comparando o modelo não-linear com o modelo linear é que a equação matricial linear foi alterada para uma função não-linear da seguinte forma:

$$Ax_x \Rightarrow f(x_x)$$

$$Hx_x \Rightarrow h(x_x)$$

Figura 17 – Comparação entre o algoritmo do Filtro de Kalman e do Filtro de Kalman Extendido.



Fonte: Do Autor (2016).

O local onde são utilizados $\mathbf{Ax_x}$ e $\mathbf{Hx_x}$ na equação do modelo de algoritmo linear são substituídos pela equação que descreve o modelo não-linear. Pode-se dizer que, se o sistema sofrer alterações, a equação que descreve o modelo de sistema também deveria ser alterada. Todas as equações restantes são idênticas.

Em relação à matriz do sistema A e H, tais matrizes devem ser dadas a partir do modelo de sistema. O Filtro de Kalman extendido deriva um modelo por linearização do modelo não-linear, utilizando a técnica:

$$A \equiv \frac{\partial f}{\partial x}|_{\hat{x}_k}, \qquad H \equiv \frac{\partial h}{\partial x}|_{\hat{x}_k} \tag{2}$$

Onde $\frac{\partial}{\partial_x}$ significa que a função correspondente está sendo parcialmente diferenciada em relação a x. Por exemplo, $\frac{\partial_f}{\partial_x}$ significa que a função f está sendo diferenciada em relação à x. Se ambos f e x estão na forma de vetor, o resultado tornar-se uma matriz Jacobiana, e $\frac{\partial_f}{\partial_x}|_{\hat{x}_k}$ é o valor obtido aplicando x = $\hat{x}_k$, na matrix $\frac{\partial_f}{\partial_x}$.

É importante saber que um EKF não utiliza um valor predefinido, mas a estimativa anterior ($\hat{x}_k$) como um ponto de referência para a linearização. Desta forma, o modelo linear é obtido com base na estimativa anterior, que se presume ser o valor mais próximo ao estado real do sistema.

Resumindo toda a análise sobre o EKF, pode-se dizer que o processo apresentado na Figura 4 é praticamente idêntico. Diferenças são observadas em relação às equações de cada etapa, dado que o Filtro de Kalman estendido utiliza equações do modelo não-linear $f(x_x)$ e $h(x_x)$ no lugar de Ax_x e Hx_x no modelo linear, e as matrizes A e H são os jacobianos do modelo não-linear, onde os valores nas matrizes jacobianas são obtidos pela aplicação da estimativa anterior.

2.4.6 Filtro de Kalman Unscented

Para introduzir o conceito acerca do Filtro de Kalman Unscented (FKU), pode-se dizer que enquanto o EKF resolve o problema da não-linearidade com a expansão gradual do algoritmo linear, o uso do FKU toma uma abordagem única que elimina o processo de linearização. Devido a este fato, o Filtro de Kalman Unscedent é livre do problema de divergência causado pela linearização do modelo obtidos pela matriz Jacobiana.

O quadro principal do Filtro de Kalman, onde é descrito o processo de predição das variáveis de estado com base no modelo do sistema e ajuste da previsão com a medição para obter a estimativa final, ainda é válido para FKU. O chamado "unscented" é a base deste algoritmo e a principal diferença em relação ao Filtro de Kalman Extendido. Sua ideia geral é selecionar o chamado ponto sigma da distribuição original (Gaussiana) e propagar este ponto através da função não-linear, a fim de calcular a média e a covariância da matriz Gaussiana propagada.

Se a função F (x) for não linear e se tornar extremamente difícil obter a média e covariância da variável transformada analiticamente, uma solução possível para este problema é utilizar a simulação de Monte Carlo, onde aplica-se um número suficiente de amostras de x. Contudo, este método requer uma elevada carga computacional e pode ser complicado para uma simulação em tempo real. A transformação "unscented" é similar ao conceito da simulação de Monte Carlo, onde se escolhe aleatoriamente um número de amostras. Este método tem como objetivo obter uma delicada seleção de ponderação para cada amostra, o que pode reduzir o peso computacional e o número de amostras quando comparado com a simulação de Monte Carlo.

Visto que o FKU não desempenhará qualquer outro papel neste trabalho, detalhes sobre o algoritmo e a derivada matemática da transformação "unscented" serão omitidos no escopo do trabalho. Contudo, informações adicionais podem ser obtidas em Julier, Uhlmann e Durrant-Whyte (1997) e Merwe e Wan (2004) e Thrun, Burgard e Fox (2005).

2.4.7 Conclusões acerca dos Filtros Gaussianos

O Filtro de Kalman tem sido utilizado para um número de aplicações em que o modelo de transição de estado pode ser uma distribuição de Gauss com representação nominal linear. Isso significa que o estado posterior $\mathbf{x}_{t+1}$ deve ser

uma função linear da x_t, além de um ruído Gaussiano. Tal filtro pode ser utilizado em uma série de aplicações como um seguimento de trajetórias de objetos, mapeamento e localização, etc.

A principal vantagem da abordagem do Filtro de Kalman é que a estimativa de estado subsequente no mapa pode ser feita on-line. Esta forma incremental torna-se bastante interessante dentro de suas limitações. A principal limitação do Filtro de Kalman refere-se a ambientes dinâmicos. Sabe-se que para aplicações com o uso deste filtro, como por exemplo o posicionamento de um robô (veículo) em um mapa depende linearmente do posicionamento anterior aplicado. Entretanto, o posicionamento, por exemplo, é geralmente descrito por funções trigonométricas não lineares.

O algoritmo para o Filtro de Kalman Estendido é um representação da expansão do Filtro de Kalman para sistemas não-lineares. O procedimento geral do algoritmo não sofreu mudanças significativas, entretanto, não se pode fazer tal afirmação quando se analisa o método computacional. Primeiro, um modelo linear foi substituído por um modelo não-linear e o Jacobiana do modelo não linear tem-se utilizado em lugar das matrizes do sistema.

De acordo com Moutarlier e Chatila (1989), o Filtro de Kalman extendido foi um dos primeiros algoritmos de filtragem utilizados para resolver o problema de localização e mapeamento de uma plaforma, uma vez que a posição e as marcações terrestres podem ser mantidas em um único vetor. No entanto, quando novas áreas são exploradas, o vetor de estado vai crescendo ao longo do tempo e mais pontos de referências são adicionadas ao mapa. Uma vez que o tempo de execução do filtro é uma função quadrática em relação ao tamanho do vetor de estado, a escalabilidade é uma questão importante a ser analisada, podendo limitar o uso deste tipo de filtro. Soluções alternativas foram implementadas para a resolução de tal problemática como, por exemplo, o uso de uma matriz de informação, em vez da matriz de covariância (Thrun;

BURGARD; FOX, 2004), ou mesmo o uso de sub-mapa ou técnicas de dividir o problema em vários problemas menores a serem abordados separadamente (Bailey, 2002; Bosse et al., 2004; Guivant et al., 2004).

Os erros inerentes do Filtro de Kalman Estendido são devidos à linearização realizada para calcular a média e covariância de uma variável aleatória, a qual deve passar por uma transformação não-linear. O Filtro de Kalman Unscented evita tais erros usando um conjunto de "amostras" determinísticas para calcular a média e covariância, devido ao fato de o Filtro de Kalman Unscented evitar o problema de divergência causada pela linearização do modelo obtidos por matrizes Jacobianas.

Deste modo, surge o seguinte questionamento: em se tratando de um sistema não linear, qual filtro deveria ser utilizado? A resposta para esta questão é que não há um padrão claro para este tipo de questionamento, no entanto que diversos autores divergem em relação a resposta para tal pergunta. Assim, a principal análise deve ser feita em relação ao sistema que se deseja controlar. Em geral, se Jacobiana da função puder ser facilmente obtida, é preferível utilizar o Filtro de Kalman Estendido, considerando o tempo de cálculo necessário. Se este não for o caso e se for difícil obter um Jacobiana ou existir uma possibilidade para a discrepância, o uso do FKU é preferível.

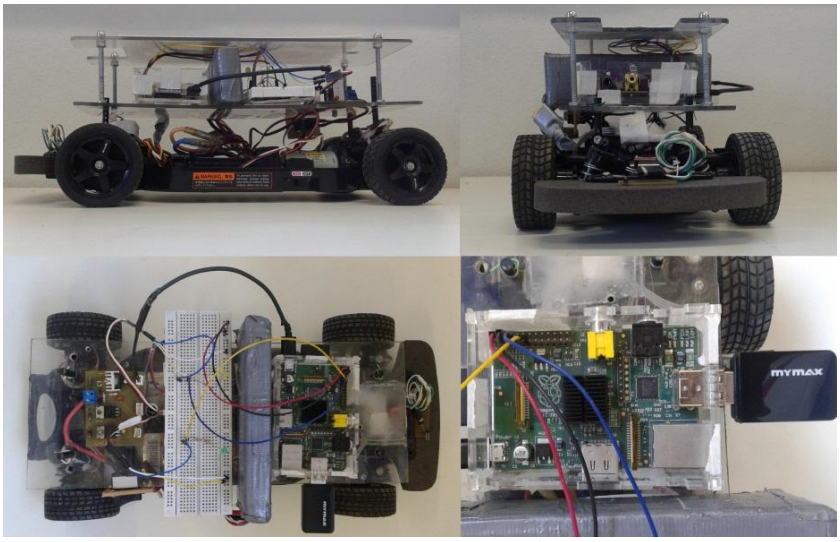
3 MATERIAL E MÉTODOS

No âmbito deste trabalho, foi projetado e construído um robô móvel autônomo, que pode ser usado como uma plataforma para várias aplicações e, por consequência, efetuar a validação dos algoritmos desenvolvidos. O projeto é dividido em três áreas principais: (i) detecção do ambiente por meio de uma câmera devidamente instalada na parte superior do local onde o robô irá trafegar, (ii) planejamento do caminho entre dois pontos previamente definido, e (iii) controle do robô. O componente de planejamento de trajetórias é realizado no âmbito global, e sendo assim, exige que o ambiente seja completamente conhecido e estático. Nesta abordagem, o algoritmo gera um caminho completo do ponto de partida para o ponto de destino, antes que o robô comece o seu movimento.

3.1 Descrição da plataforma utilizada

A plataforma desenvolvida consiste em um veículo terrestre em escala. O chassi do veículo é fabricado pela empresa Tamiya. Possui 42 cm de comprimento, 18 cm de largura, tracionamento diferencial nas quatro rodas por um motor de corrente contínua, e direcionamento pela orientação das duas rodas dianteiras atuado por um servo motor. O diâmetro das rodas é de 33 mm. O esterçamento dessa plataforma atua no máximo até aproximadamente 30° e a sua velocidade mínima é de cerca de 2,3 m/s, com um peso de 2138,9g do sistema completo. A Figura 18 ilustra o modelo da plataforma móvel desenvolvido.

Figura 18 - Visões lateral, frontal e de topo do protótipo usado e o Raspberry Pi embarcado.



Fonte: Do Autor (2017).

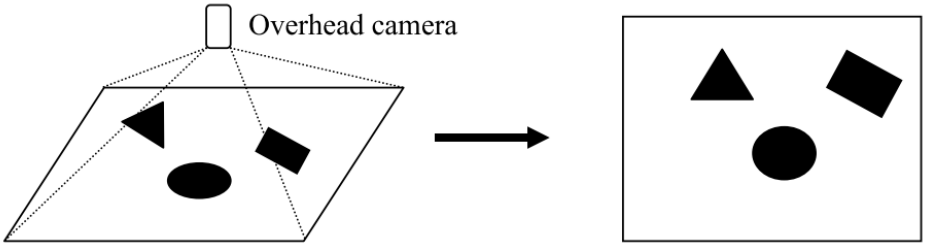
Os sistemas embarcados na plataforma são constituídos pelos seguintes componentes:

a) Um Raspberry Pi;
b) Uma bateria de 12V e uma placa de regulação de tensão (que regula a saída para 5V) de alimentação do Raspberry;
c) Uma bateria de 7.4V para acionar o motor de tração;
d) Um protoboard para a implementação do PWM da velocidade e esterçamento.

3.2 Descrição do ambiente para a validação do projeto

A validação do trabalho realizado inicia com a definição do local, onde pode ser instalada uma câmera sob a área trafegável da plataforma, conforme esquematizado na Figura 19.

Figura 19 - Ambiente para a geração de testes e validação da metodologia proposta



Fonte: Do Autor (2017).

Depois de calibrada, a câmara irá gerar uma imagem que pode ser usada para criar um mapa que servirá de base para a geração do caminho a ser percorrido pelo robô, como ilustra a Figura 17. Esta é uma implementação simplista dos cenários da vida real em que várias câmeras são usadas para capturar diferentes partes do espaço de trabalho inteiro, e suas saídas são fundidas para criar um mapa geral usado pelos algoritmos de planejamento de trajetória.

Utilizando-se da mesma câmera, foi implementado um sistema de predição da localização da plataforma baseado em Filtro de Kalman, onde a câmera é utilizada para capturar a localização do robô no início do planejamento, bem como quando o robô se move, resolvendo o problema da localização do robô. Outra condição é que para o início do planejamento, os pontos definidos como inicial e final devem ser explicitamente fornecidos. O objetivo nesta fase atual é apenas planejar um caminho para o robô. A partir desta fase, quando a plataforma começar a se movimentar, é iniciado o controle de movimentos e a correta localização da plataforma. O trabalho foi desenvolvido no Laboratório de Mecânica e Automação do Departamento de Engenharia da Universidade Federal de Lavras. A Figura 20 ilustra o cenário

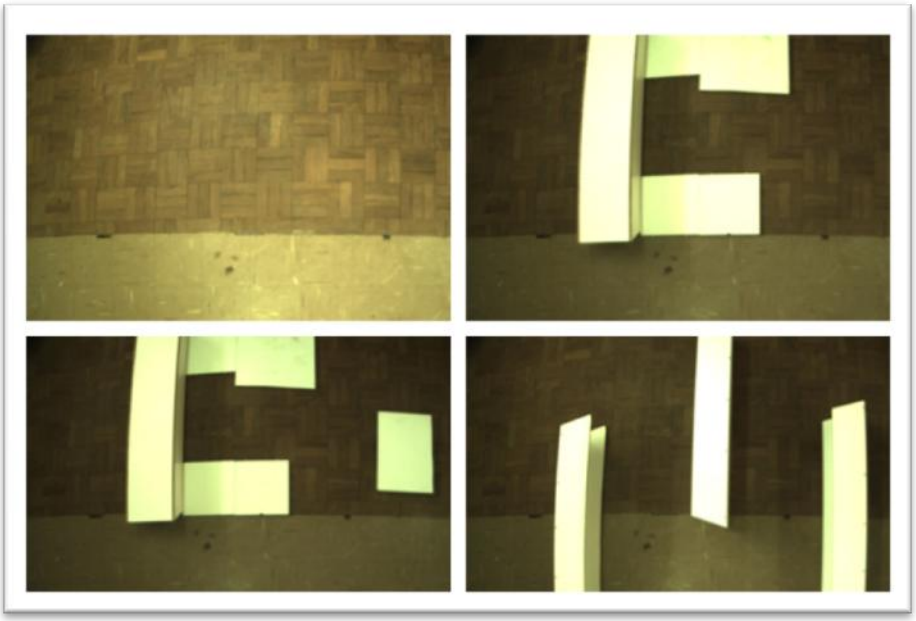
criado para a validação da pesquisa. A câmera utilizada é do modelo " FL3-U3-20E4C-C", a cores, com resolução de 1600x1200 pixels e sensor do tipo CMOS.

Figura 20 - Local para a realização dos testes e validação da metodologia proposta.

Fonte: Do Autor (2017).

Assumindo que não haja obstáculos na área de navegação, o caminho mais curto entre o ponto inicial e o ponto final seria uma linha reta. Deste modo, a plataforma irá prosseguir ao longo deste caminho até que um obstáculo seja detectado. Caso seja detectada a presença de obstáculos, a plataforma móvel tenderá a evitá-los e continuar a navegar em direção ao ponto final especificado. Diversos cenários foram criados para a validação deste trabalho, e parte deles são apresentados no conjunto de imagens da Figura 21.

Figura 21 - Cenários criados para a validação da metodologia proposta.



Fonte: Do Autor (2017).

Os dados provenientes da validação da metodologia proposta foram obtidos por meio da framework ROS (*Robot Operating System*), uma plataforma com grande aceitação no meio acadêmico e científico e utilizado em diversos cursos de robótica em universidades reconhecidas como a Stanford University, Tokyo University e Cornell University (ROBOT OPERATING SYSTEM - ROS, 2016). A maior vantagem de se empregar o ROS é a facilidade com que os blocos podem ser integrados, pois ele conta com uma interface bem definida para a comunicação entre os componentes, além de ser Open Source, o que permite a inserção de incertezas quantitativas que podem ser parâmetros para a validação do modelo proposto.

O Matlab foi escolhido como ambiente de implementação do algoritmo proposto, com a possibilidade de integração ao ambiente Simulink, visto que a

modelagem de todo o sistema pode ser integrada por meio do diagrama de blocos para uma análise mais didática da metodologia proposta.

4 RESULTADOS E DISCUSSÃO

Neste capítulo são apresentados os resultados de simulações realizadas nos diferentes cenários criados para a validação desta metodologia, bem como o melhor caminho obtido em cada trajeto. No contexto deste trabalho, considerou-se como melhor resultado o menor caminho obtido entre dois pontos previamente estabelecidos. Por se tratar de uma técnica de otimização off-line, o tempo computacional levado para a geração deste caminho não foi avaliado como requisito de melhor solução. Ao final de cada cenário, são tecidas as principais conclusões sobre os resultados obtidos.

4.1 Resultados obtidos para o primeiro cenário

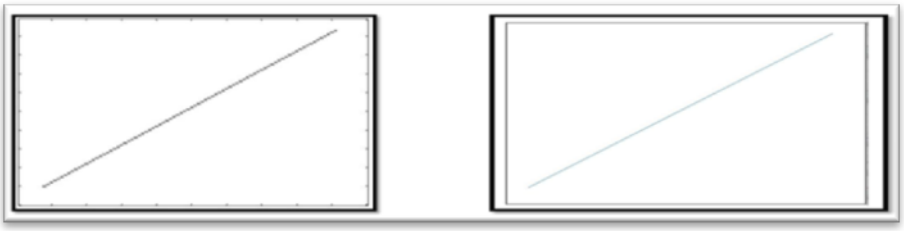
A Figura 22 corresponde ao primeiro cenário utilizado para a validação do trabalho. Trata-se de um ambiente livre de obstáculos, cuja distância entre os pontos A e B, dados como ponto de partida e ponto de chegada, respectivamente, é de 4,5 metros.

Figura 22 - Cenário inicial sem a presença de obstáculos para a geração do menor caminho entre os pontos A e B.

Fonte: Do Autor (2017).

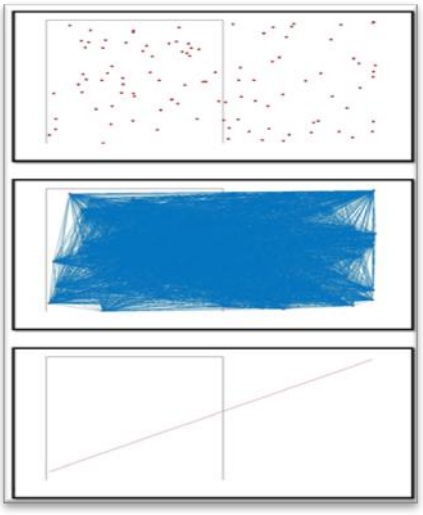
No conjunto de imagens representadas nas Figuras 23 a 29 são detalhados os resultados obtidos pelos algoritmos A Estrela, MRP, Campo Potencial, Árvores Aleatórias de Rápida Exploração, Árvores Aleatórias de Rápida Exploração Bidirecional e Algoritmo Genético, respectivamente, cujos gráficos correspondem ao caminho gerado entre os dois pontos A e B referidos na Figura 22. Por se tratar do mesmo cenário, com as mesmas características para cada algoritmo, as principais considerações sobre as soluções obtidas serão descritas ao final de cada subsecção.

Figura 23 - Resultados para o Algoritmo A Estrela.



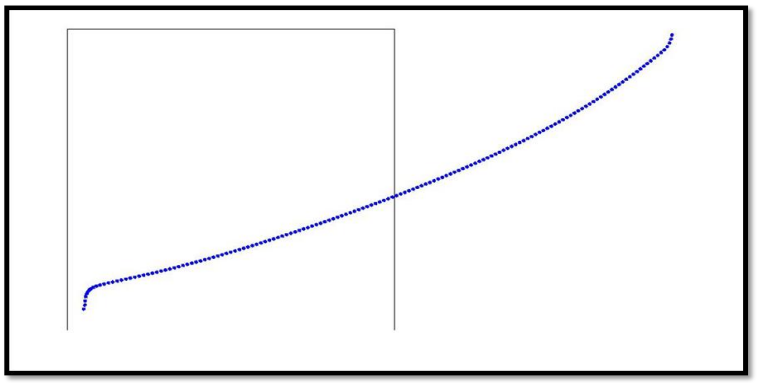
Fonte: Do Autor (2017).

Figura 24 - Resultados para o Algoritmo MRP.



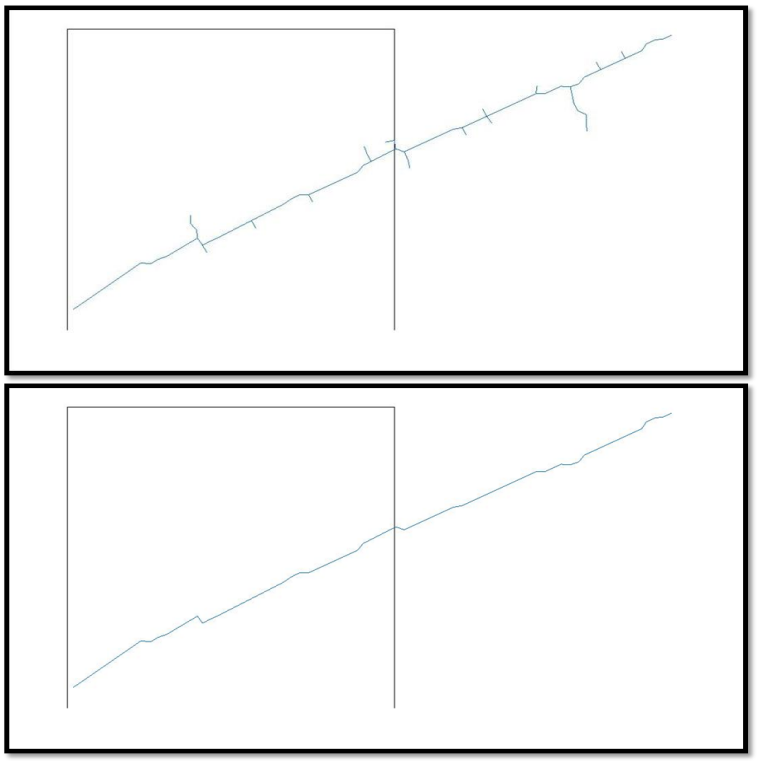
Fonte: Do Autor (2017).

Figura 25 - Resultados para o Algoritmo Campo Potencial.



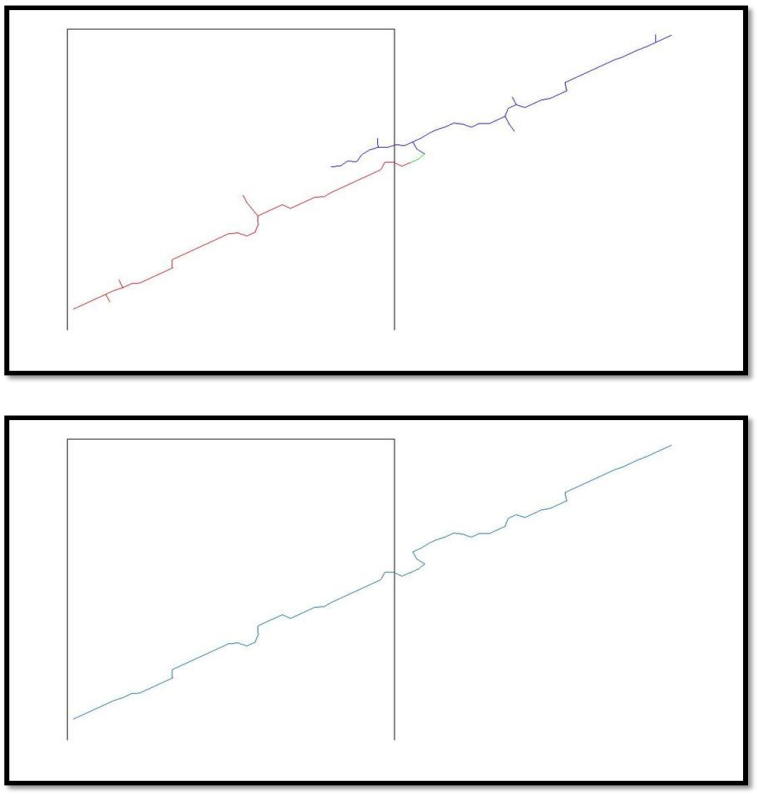
Fonte: Do Autor (2017).

Figura 26 - Resultados para o Algoritmo Árvores Aleatórias de Rápida Exploração.



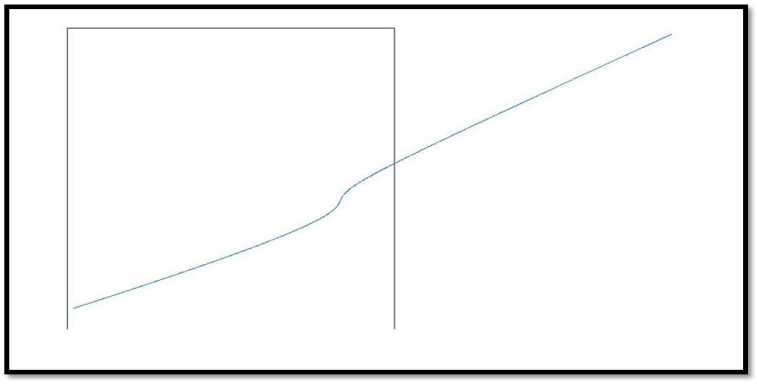
Fonte: Do Autor (2017).

Figura 27 - Resultados para o Algoritmo Árvores Aleatórias de Rápida Exploração Bidirecional.



Fonte: Do Autor (2017).

Figura 28 - Resultados para o Algoritmo Genético.



Fonte: Do Autor (2017).

A Tabela 2 descreve os resultados obtidos para o primeiro cenário criado. Por se tratar de uma situação que envolve a melhor opção de trajetória, sem a presença de obstáculos, todos os algoritmos obtiveram uma solução admissível, cujo melhor resultado foi igualmente dado pelos algoritmos A Estrela e Algoritmos Genéticos.

Tabela 2 - Resultados obtidos para o cenário inicial.

Algoritmo	**Resultados**
A Estrela	Processing time=7.357928e+01 Path Length=1.187939e+03
Mapa de Rotas Probabilísticos	Processing time=1.595909e+01 Path Length=1.474912e+03
Campo Potencial	Processing time=1.084582e+01 Path Length=7.445004e+03
Árvores Aleatórias de Rápida Exploração	Processing time=2.061316e+01 Path Length=1.522946e+03
Árvores Aleatórias de Rápida Exploração Bidirecional	Processing time=3.173043e+01 Path Length=1.630281e+03
Algoritmo Genético	Processing time=2.942958e+01 Path Length=1.187939e+03

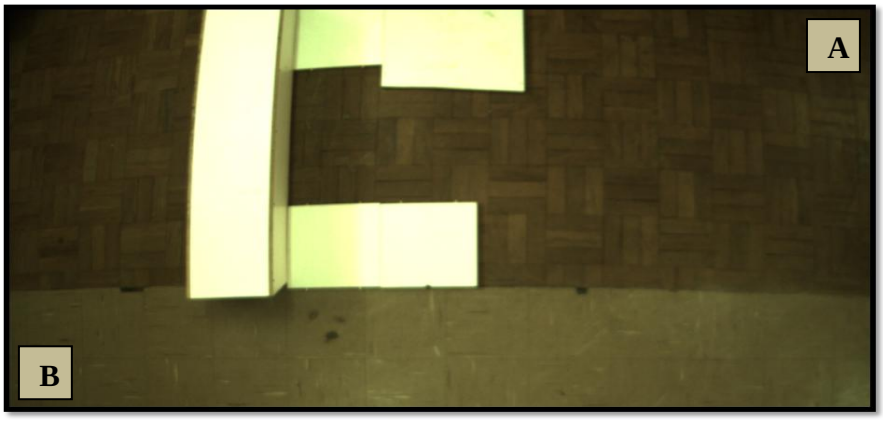
Fonte: Elaborado pelo Autor (2017).

Para este cenário, todos os algoritmos convergiram para uma solução admissível, visto que nenhuma restrição foi imposta ao longo da execução do caminho percorrido. Apesar da simplicidade da trajetória gerada, este cenário valida a condição de uma linha reta ser a menor distância entre dois pontos estabelecidos em um ambiente e, consequentemente, todas as técnicas de otimização implementadas convergiram para o melhor resultado. As demais subseções irão validar a robustez dos algoritmos implementados à medida que novas e diferentes restrições forem inseridas no ambiente de simulação, sendo possível uma análise mais detalhada das soluções geradas por cada técnica de otimização.

4.2 Resultados obtidos para o segundo cenário

Para o segundo cenário, representado pela Figura 29, observa-se a presença de um obstáculo em forma de U cujo objetivo é forçar o algoritmo a gerar um caminho diferente do anterior, e avaliar a possibilidade de algum destes algoritmos ficar "preso" em um local dentro do obstáculo inserido.

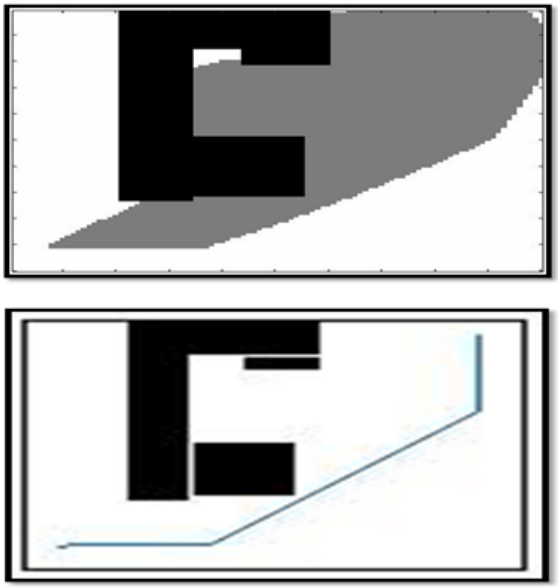
Figura 29 - Segundo cenário com a presença de obstáculos para geração do menor caminho entre os pontos A e B.



Fonte: Do Autor (2017).

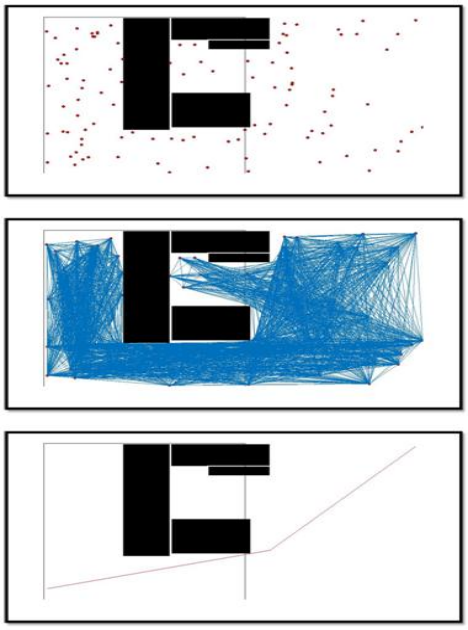
O conjunto de imagens representado pelas Figuras 30 a 35 traduz os resultados obtidos para o segundo cenário, utilizando os algoritmos A Estrela, MRP, Campo Potencial, Árvores Aleatórias de Rápida Exploração, Árvores Aleatórias de Rápida Exploração Bidirecional e Algoritmo Genético, respectivamente.

Figura 30 - Resultados do segundo cenário para o Algoritmo A Estrela.



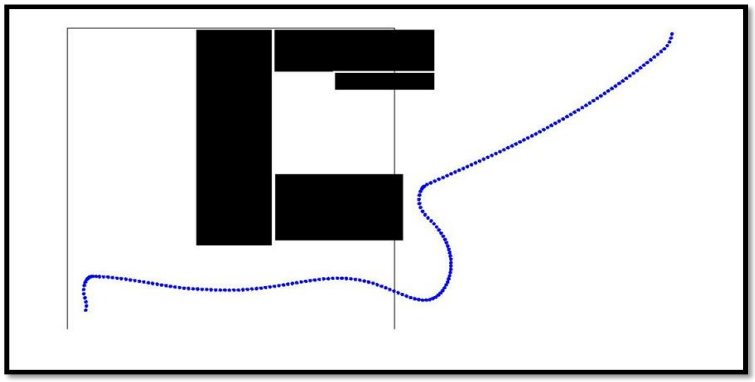
Fonte: Do Autor (2017).

Figura 31- Resultados do segundo cenário para o Algoritmo Mapa de Rotas Probabilístico.



Fonte: Do Autor (2017).

Figura 32 - Resultados do segundo cenário para o Algoritmo Campo Potencial.

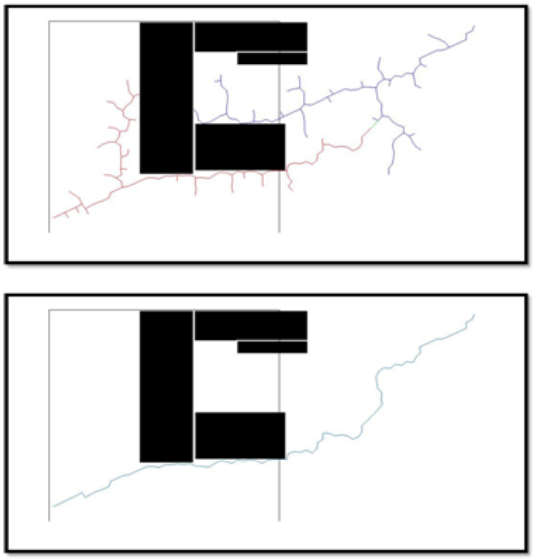


Fonte: Do Autor (2017).

Figura 33 - Resultados do segundo cenário para o Algoritmo Árvores Aleatórias de Rápida Exploração.

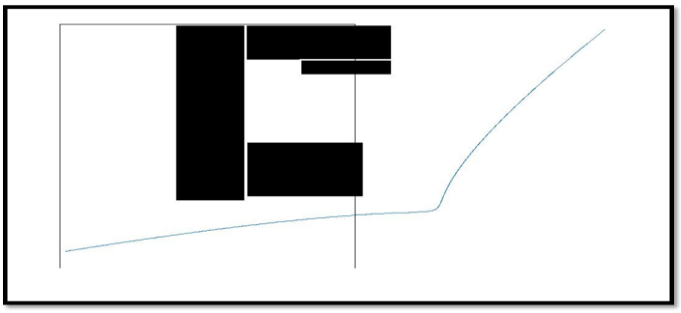
Fonte: Do Autor (2017).

Figura 34 - Resultados do segundo cenário para o Algoritmo Árvores Aleatórias de Rápida Exploração Bidirecional.



Fonte: Do Autor (2017).

Figura 35 - Resultados do segundo cenário para o Algoritmo Genético.



Fonte: Do Autor (2017).

A Tabela 3 apresenta os resultados obtidos para o exemplo em questão.

Tabela 3 - Resultados obtidos para o segundo cenário.

Algoritmo	**Resultados**
A Estrela	Processing time=1.463882e+02 Path Length=1.363675e+03
Mapa de Rotas Probabilísticos	Processing time=4.854505e+01 Path Length=1.557342e+03
Campo Potencial	Processing time=1.528347e+01 Path Length=1.492594e+03
Árvores Aleatórias de Rápida Exploração	Processing time=6.475783e+02 Path Length=1.965046e+03
Árvores Aleatórias de Rápida Exploração Bidirecional	Processing time=2.849483e+01 Path Length=2.167213e+03
Algoritmo Genético	Processing time=1.321480e+02 Path Length=1.410538e+03

Fonte: Elaborada pelo Autor (2017).

O melhor resultado foi dado pelo algoritmo A Estrela, conforme pode ser observado na Tabela 3. Vale ressaltar que uma análise em termos de esforços mecânicos para uma plataforma móvel pode ser feita nesta situação. Em termos cinemáticos, a trajetória gerada pelo algoritmo A estrela envolve duas rotações e

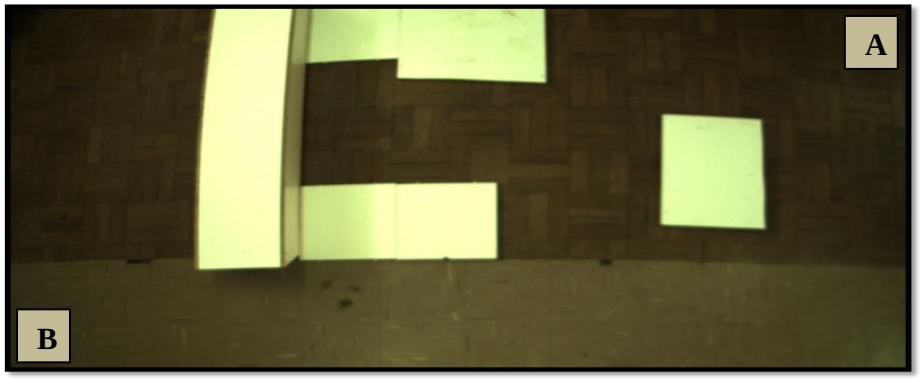
três translações, enquanto a trajetória gerada pelo algoritmo Mapa de Rotas Probabilísticos envolve apenas uma rotação e duas translações entre os pontos de partida e o de chegada, com uma pequena diferença referente ao comprimento do caminho. A mesma situação acontece quando se observa os resultados obtidos utilizado Algoritmos Genéticos com uma diferença ainda menor em termos de comprimento do caminho gerado. Neste sentido vale ressaltar que, dependendo da estrutura física da plataforma móvel, bem como os esforços mecânicos gerados para a movimentação entre os dois pontos A e B, uma vantagem seria o uso de uma otimização multi-objetivo, utilizando como um moderador a distância entre os dois pontos e o menor número de movimentos a ser realizado entre estes.

Outro fato que cabe ressaltar é que todos os algoritmos obtiveram uma solução aceitável, sendo que nenhum deles ficou "preso" em um mínimo local.

4.3 Resultados obtidos para o terceiro cenário

Para o terceiro cenário, utilizou-se da mesma estrutura anterior, com o acréscimo de um novo obstáculo de forma a dificultar ainda mais a obtenção de um caminho entre os dois pontos A e B. A Figura 36 traduz o que foi descrito neste cenário especifico.

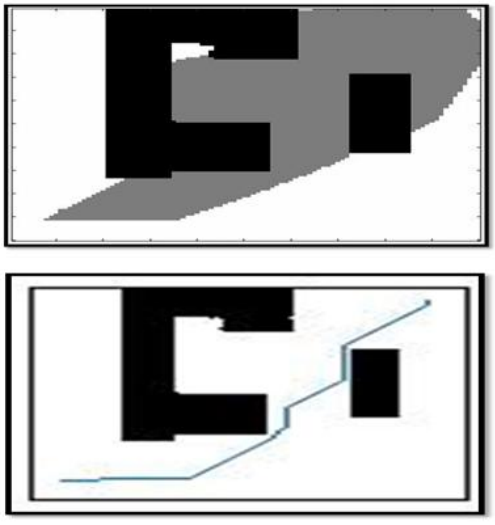
Figura 36 - Terceiro cenário com a presença de obstáculos para a geração do menor caminho entre os pontos A e B.



Fonte: Do Autor (2017).

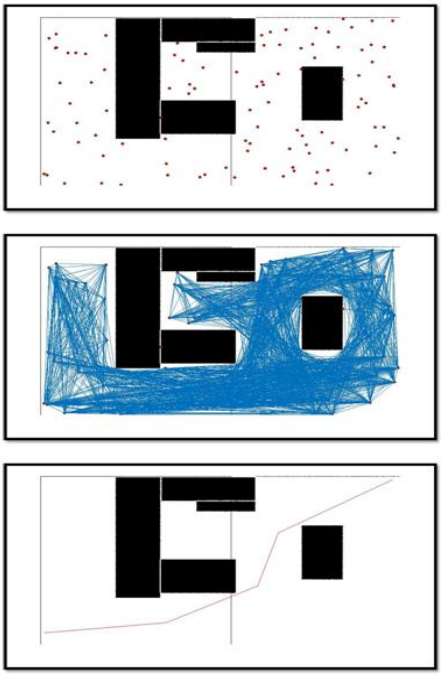
O conjunto de imagens representado pelas Figuras 37 a 42 traduz os resultados obtidos para o terceiro cenário.

Figura 37 - Resultados do terceiro cenário para o algoritmo A Estrela.



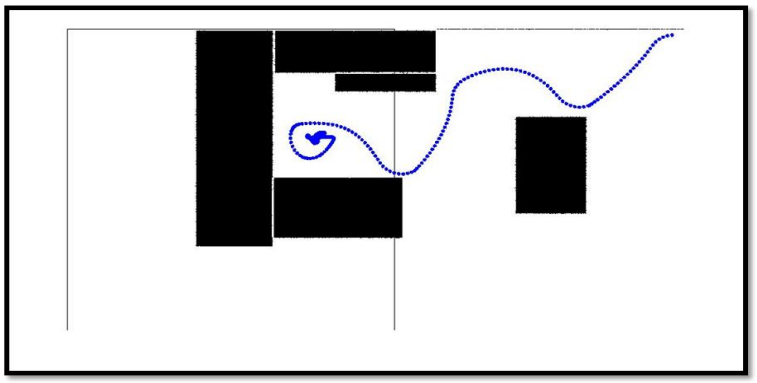
Fonte: Do Autor (2017).

Figura 38 - Resultados do terceiro cenário para o algoritmo Mapa de Rotas Probabilísticos.



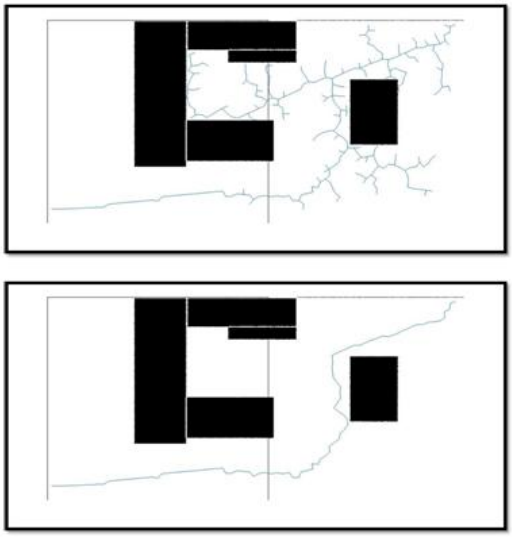
Fonte: Do Autor (2017).

Figura 39 - Resultados do terceiro cenário para o algoritmo Campo Potencial.



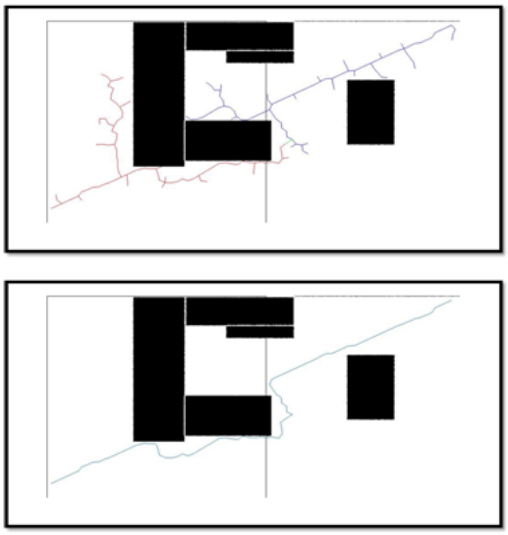
Fonte: Do Autor (2017).

Figura 40 - Resultados do terceiro cenário para o Algoritmo Árvores Aleatórias de Rápida Exploração.



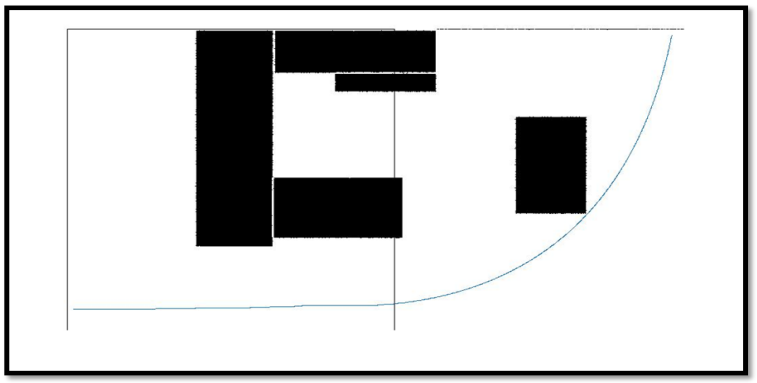
Fonte: Do Autor (2017).

Figura 41 - Resultados do terceiro cenário para o Algoritmo Árvores Aleatórias de Rápida Exploração Bidirecional.



Fonte: Do Autor (2017).

Figura 42 - Resultados do terceiro cenário para o Algoritmo Genético.



Fonte: Do Autor (2017).

Observa-se neste caso, ao inserir novos obstáculos de forma a dificultar a obtenção de uma rota entre dois pontos A e B, que nem todos os algoritmos obtiveram resultados satisfatórios. Tal situação é caracterizada pelo algoritmo Campo Potencial, cuja solução não foi possível de ser obtida, pelo fato do mesmo ficar "preso" em um mínimo local, conforme se observa na Figura 39. Esta situação justifica o uso de um conjunto de pacotes computacionais para que a probabilidade de obter uma solução admissível seja maior. Neste sentido, o melhor resultado em termos de distância mínima, bem como linearidade do caminho gerado foi obtido com o algoritmo genético, conforme observado na Tabela 4.

Tabela 4 - Resultados obtidos para o terceiro cenário.

Algoritmo	**Resultados**
A Estrela	Processing time=8.998482e+01 Path Length=1.357817e+03
Mapa de Rotas Probabilísticos	Processing time=4.146222e+01 Path Length=1.575787e+03
Campo Potencial	Preso em um mínimo local
Arvores Aleatórias de Rápida Exploração	Processing time=9.106133e+01 Path Length=1.905714e+03
Arvores Aleatórias de Rápida Exploração Bidirecional	Processing time=2.399893e+01 Path Length=1.782104e+03
Algoritmo Genético	Processing time=1.368066e+02 Path Length=1.357817e+02

Fonte: Elaborada pelo autor (2017).

4.4 R esultados obtidos para o quarto cenário

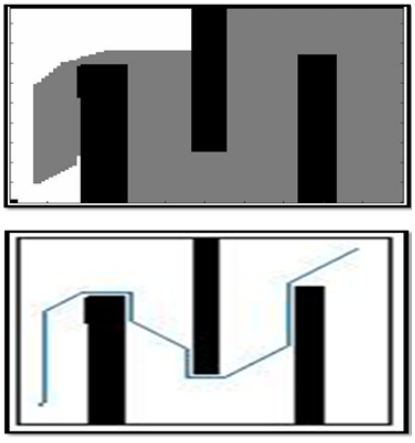
Seguindo a proposta de criação de cenários para validar a robustez dos algoritmos implementados, o cenário seguinte envolve a inclusão de novos obstáculos, cujo caminho a ser gerado nos pontos A e B envolve a necessidade de vários movimentos para que uma plataforma móvel não colida entre os mesmos. Este cenário avalia a necessidade de um pacote computacional que envolve mais de um algoritmo de otimização para aumentar a probabilidade de encontrar uma solução que possa ser executável. A descrição do cenário pode ser visualizada na Figura 43.

Figura 43 - Quarto cenário com a presença de obstáculos para a geração do menor caminho entre os pontos A e B.

Fonte: Do Autor (2017).

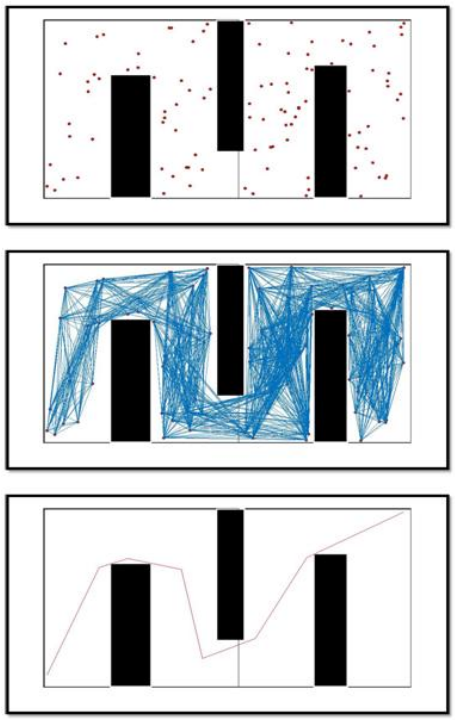
O conjunto de imagens referentes às Figuras 44 a 49, correspondem aos resultados obtidos para o quarto cenário.

Figura 44 - Resultados do quarto cenário para o Algoritmo A Estrela.



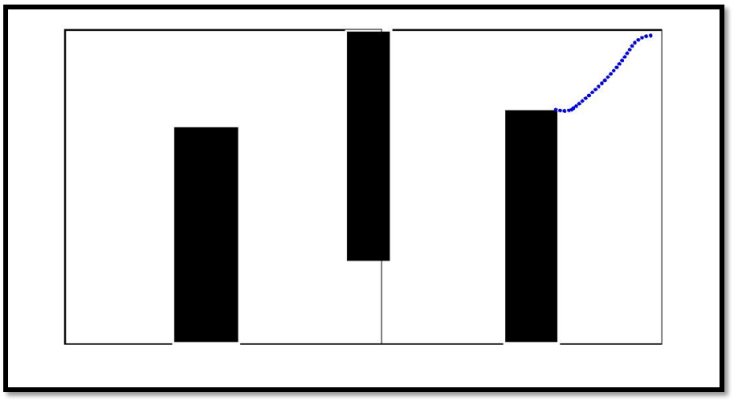
Fonte: Do Autor (2017).

Figura 45 - Resultados do quarto cenário para o Algoritmo Mapa de Rotas Probabilísticos.



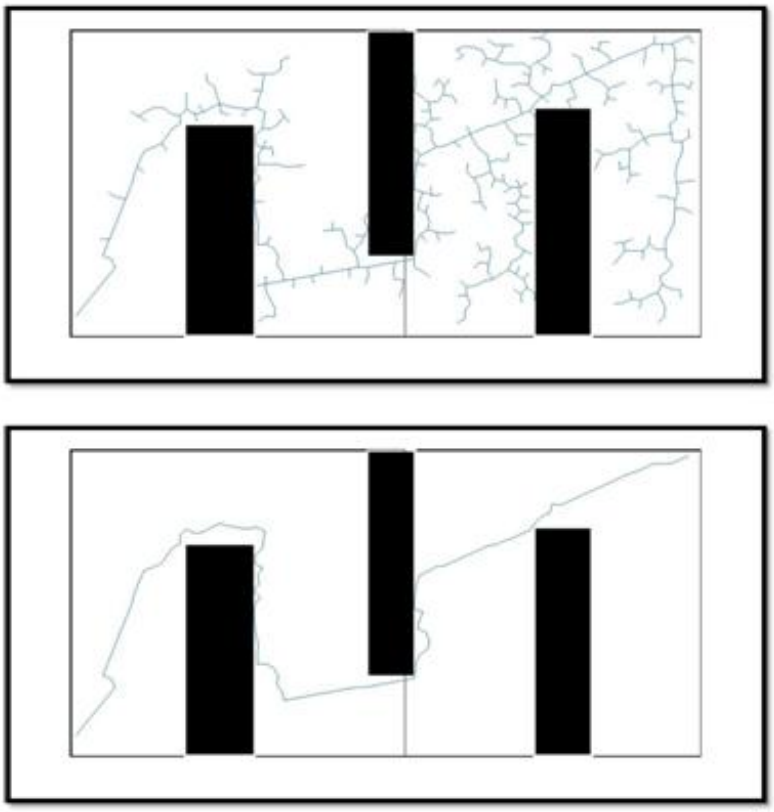
Fonte: Do Autor (2017).

Figura 46 - Resultados do quarto cenário para o Algoritmo Campo Potencial.



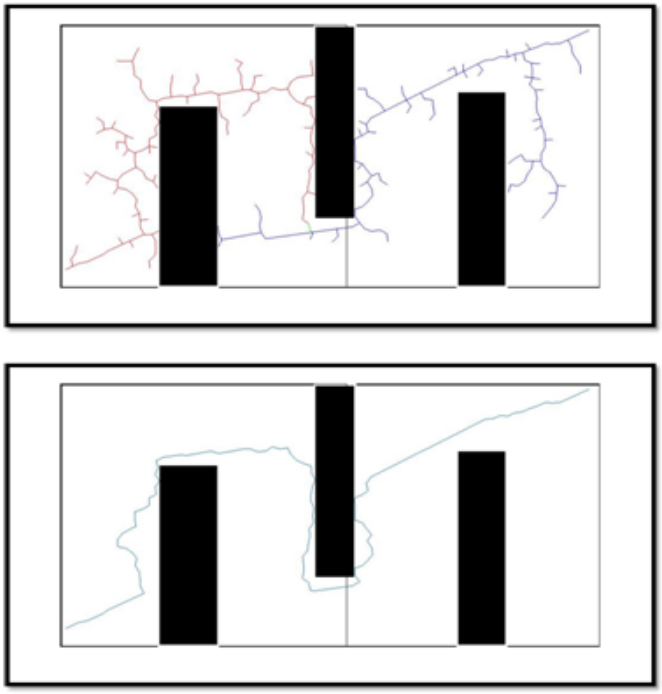
Fonte: Do Autor (2017).

Figura 47 - Resultados do quarto cenário para o Algoritmo Árvores Aleatórias de Rápida Exploração.



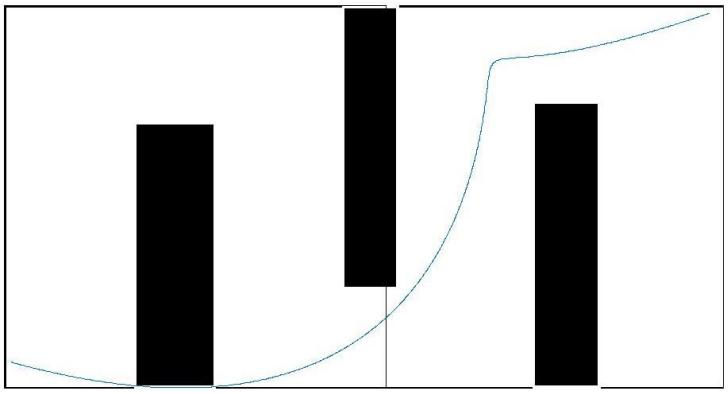
Fonte: Do Autor (2017).

Figura 48 - Resultados do quarto cenário para o Algoritmo Árvores Aleatórias de Rápida Exploração Bidirecional.



Fonte: Do Autor (2017).

Figura 49 - Resultados do quarto cenário para o Algoritmo Genético.



Fonte: Do Autor (2017).

Os resultados obtidos no quarto cenário podem ser observados na Tabela 5. A melhor solução foi obtida com o algoritmo Mapa de Rotas Probabilísticos, seguido do algoritmo A Estrela. Observa-se que neste caso, duas colisões foram reportadas tratando-se do algoritmo "Campo Potencial" e do "Algoritmo Genético".

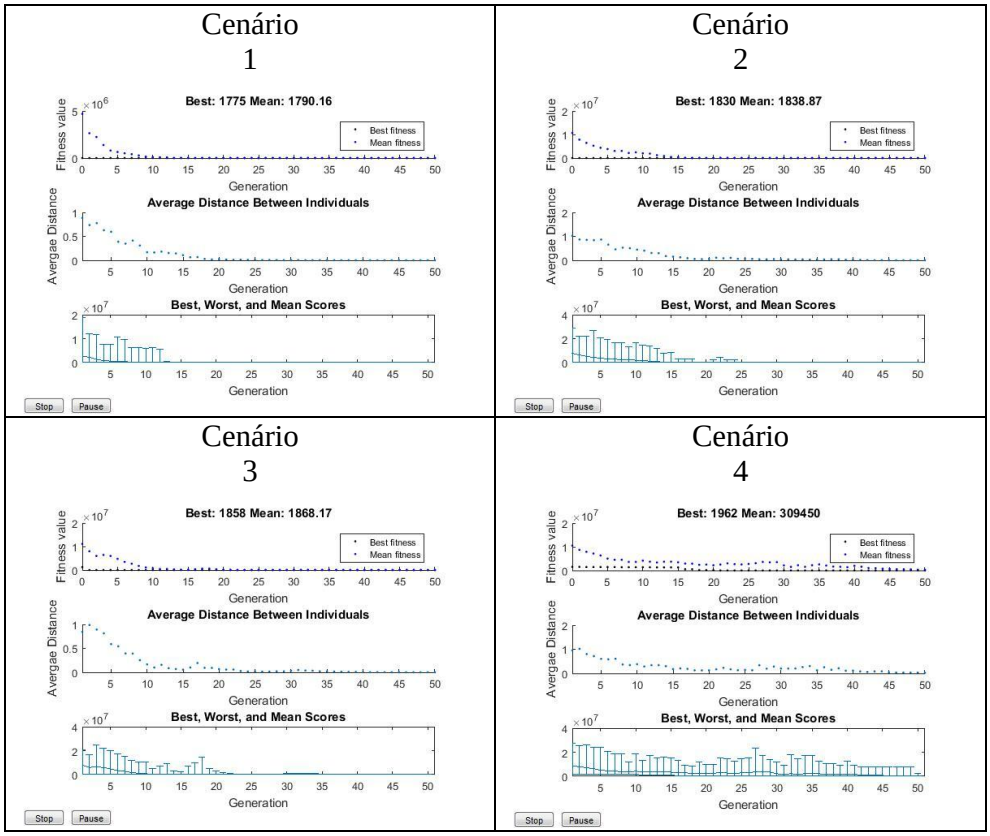
Tabela 5 - Resultados obtidos para o quarto cenário.

Algoritmo	**Resultados**
A Estrela	processing time=9.815020e+01 Path Length=2.222670e+03
Mapa de Rotas Probabilísticos	processing time=3.733397e+01 Path Length=2.150005e+03
Campo Potencial	collission recorded
Arvores Aleatórias de Rápida Exploração	processing time=4.703887e+01 Path Length=2.468824e+03
Arvores Aleatórias de Rápida Exploração Bidirecional	processing time=3.440449e+01 Path Length=2.577821e+03
Algoritmo Genético	collission recorded

Fonte: Elaborada pelo autor (2017).

Em relação ao Algoritmo Genético, uma possível solução para aumentar a probabilidade de uma solução aceitável seria aumentar o número da população e geração, o que aumentaria, por consequência, a carga computacional. Os parâmetros utilizados pelo AG foram uma população de 100 indivíduos e um número de gerações igual a 50, e mantidos iguais ao longo da simulação de todos os cenários, e a evolução do algoritmo pode ser observada na Figura 50.

Figura 50 - Evolução do processo de busca utilizando AG para os quatro cenários de validação.



Fonte: Do Autor (2017).

Observa-se que para o cenário 4, o algoritmo finalizou a simulação utilizando como critério de parada o número de gerações igual a 50. Neste caso, o AG ainda não tinha convergido a um resultado executável e, por isso, uma solução não admissível foi gerada pelo próprio cenário.

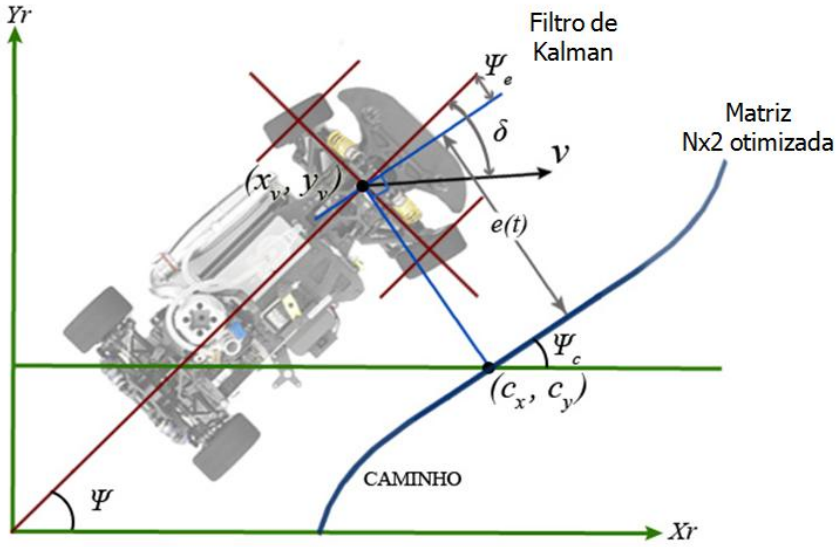
4.5 Considerações finais sobre os resultados obtidos

Face aos resultados obtidos, pode-se afirmar que os algoritmos desenvolvidos neste trabalho apresentam resultados de grande valor para os

propósitos estabelecidos. A justificativa para os cenários criados para a validação dessa metodologia é feita com base nos materiais e recursos existentes para o trabalho proposto. Entretanto, independente do cenário, foi sempre possível obter uma trajetória executável e que pudesse validar a robustez do pacote computacional utilizado pela plataforma móvel, dado um cenário predefinido.

A utilização de um estimador de estados, caracterizado pelo Filtro de Kalman, foi de extrema importância no que diz respeito à realimentação do sistema. Através deste filtro, foi possível rastrear a plataforma móvel e determinar o erro do sistema à medida que esta se desviava do caminho informado. A Figura 51 traduz o erro do sistema, que se caracteriza como a diferença existente entre a matriz que informa o caminho otimizado e matriz de estados informada pelo Filtro de Kalman.

Figura 51 - Erro de posicionamento dado pela diferença das matrizes de estado estimado e caminho mínimo.



Fonte: Do Autor (2017).

Com base no valor do erro, foi possível implementar um controlador do tipo Proporcional, Integral e Derivativo, mais conhecido como controlador PID, para a minimização da diferença de posicionamento da plataforma móvel. Os parâmetros do controlador foram obtidos utilizado os conceitos inerentes a cada ganho do controlador e realizando um ajuste fino de forma a obter a melhor sintonia para o sistema em questão.

Neste sentido, toda a proposta metodológica informada previamente foi validada, e os resultados obtidos utilizando todas as técnicas descritas, foram executados de forma satisfatória pela plataforma móvel construída para o objetivo proposto. Deste modo, a pesquisa irá apresentar novos robôs móveis que fizeram uso dos algoritmos desenvolvidos através de um controle em malha aberta, ou seja, sem a utilização de sensores e que, por consequência, sem a implementação do Filtro de Kalman, conseguiram executar uma trajetória previamente definida pelos algoritmos de otimização desenvolvidos neste trabalho.

5 INTERFACE GRÁFICA PARA A GERAÇÃO DE CAMINHO MÍNIMO

O objetivo desta seção é descrever o ambiente computacional que foi desenvolvido utilizando a ferramenta GUI (*Graphic User Interface*) em MatLab, para a geração do menor caminho entre dois pontos A e B. Esta interface, tem como objetivo utilizar os algoritmos que foram objeto de estudo deste trabalho, bem como a possibilidade de integrar novas funcionalidades, de forma a facilitar ao utilizador a compreensão dos algoritmos desenvolvidos e, consequentemente, a sua aplicação na área de navegação autônoma.

A interface foi desenvolvida com o intuito de criar uma metodologia facilitadora para todos os usuários que, por sua vez, não necessitam de conhecimento avançado em programação ou uso do software MatlLab. Sendo assim, espera-se que esta interface possa contribuir para o ensino e, de uma forma fácil e intuitiva, despertar maior interesse daqueles que venham a aprofundar os conhecimentos nesta área de conhecimento.

5.1 Introdução

As interfaces computacionais para o ensino de engenharia vêm sendo cada vez mais aplicadas como uma forma amigável de aprendizagem. Este recurso é capaz de substituir vários equipamentos eletrônicos e trazer grandes vantagens com mais versatilidade para captar o interesse do aluno. O desenvolvimento de um ambiente interativo de ensino com um propósito específico foi o que levou ao desenvolvimento desta interface no âmbito desta pesquisa.

Apesar do software MatLab ser de conhecimento da maioria dos docentes e discentes do curso de engenharia, esta aplicação foi desenvolvida para evitar o seu uso e, assim, fornecer ao utilizador uma aplicação mais

intuitiva e didática das funções que foram utilizadas no desenvolvimento deste trabalho.

5.2 Visão Geral do Ambiente

O programa é dividido em apenas um módulo, onde o utilizador tem acesso a um conjunto de informações que o auxiliam na utilização da interface, bem como um menu, onde o mesmo pode direcionar a alguma informação de interesse antes de iniciar a aplicação.

A Figura 52 apresenta o primeiro módulo da interface desenvolvida. Designado “LIR”, este módulo é a janela que dá acesso a todas aplicações, bem como a um manual de utilização escrito de forma a facilitar a sua utilização.

Figura 52 - Menu principal da interface gráfica.

Fonte: Do Autor (2017).

Conforme pode ser visualizado na Figura 52, por meio de um único módulo, o utilizador pode ter acesso a todas as funcionalidades da interface, no qual é possível realizar as simulações pretendidas. Caso o utilizador necessite de uma base teórica e dos conceitos referente a cada algoritmo, tal informação é possível de ser acessada através do menu criado para tal intuito. Basta selecionar o algoritmo de interesse que um documento é disponibilizado com a base teórica necessária para a compreensão do algoritmo, conforme pode ser visualizado na Figura 53.

Figura 53 - Menu dos módulos de simulação e código M.File dos índices de desempenho.

Fonte: Do Autor (2017).

Da mesma forma, e conforme pode ser visualizado na Figura 54, é permitido que o usuário tenha acesso a um módulo de ajuda, onde lhe é fornecido um documento contendo informações de como utilizar a interface desenvolvida, como deve ser feita a inserção dos parâmetros de entrada e as características necessárias para o mapa do ambiente a ser inserido na simulação. Caso o mesmo tenha terminado a simulação, é lhe permitido fechar a aplicação através de uma opção "sair" inserida no topo da interface.

Figura 54 - Módulo de acesso às informações de utilização da interface e modulo de saída.

Fonte: Do Autor (2017).

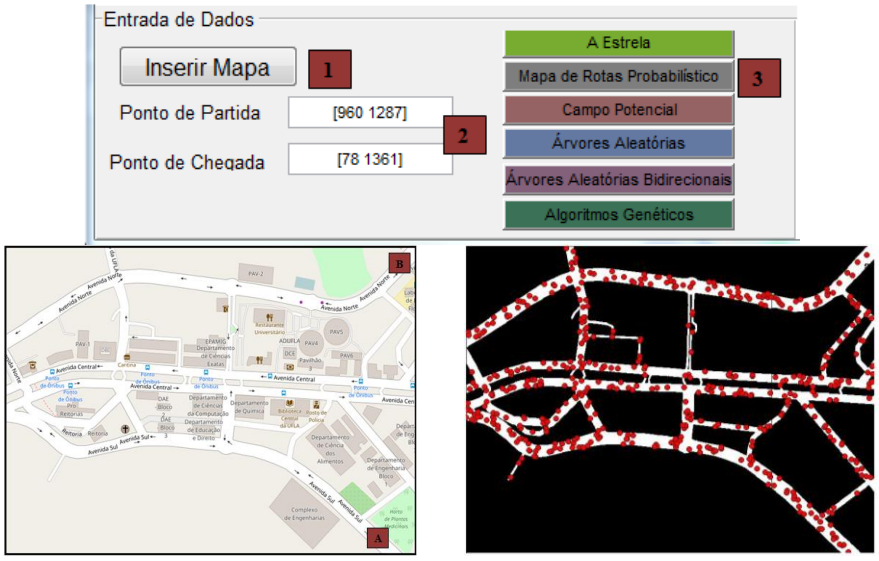
A entrada de dados e inicialização do programa é efetuada inserindo o mapa do ambiente através da opção "Inserir Mapa", juntamente com as coordenadas (x,y) para o ponto de partida, bem como as coordenadas (x,y) para o ponto de chegada. Após a realização do procedimento, deve-se selecionar o algoritmo de interesse. Este procedimento irá iniciar o programa em ambiente Matlab, onde todo o processo de simulação poderá ser acompanhado. Ao final do processo, será retornado o comprimento do caminho gerado, o tempo de simulação, as imagens referentes a tal percurso, bem como todos os vetores que, em conjunto, formam o caminho a ser percorrido pela plataforma móvel. A Figura 55 ilustra o módulo em referência.

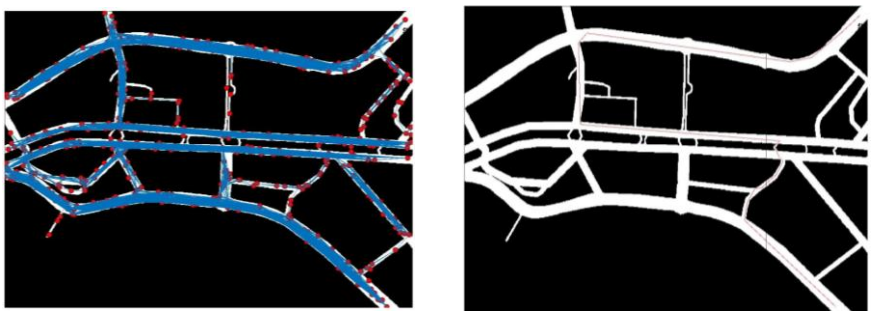
Figura 55 - Módulo de seleção do algoritmo a ser utilizado.

Fonte: Do Autor (2017).

A seguir apresenta-se um conjunto de imagens, ilustrado pela Figura 56, referente à utilização deste módulo para um ambiente de alta complexidade, obtido na avenida principal da Universidade Federal de Lavras. O mapa inserido está em formato bitmap, em uma escala de 1520x1086 e apresenta diversas rotas existentes entre dois pontos. O algoritmo que apresentou melhor resultado foi o Mapa de Rotas Probabilísticos em uma simulação com 500 partículas. Para tal simulação, levou-se em consideração a menor distância existente entre dois pontos especificados, sem considerar questões de orientação e regras de trânsito ao longo da rodovia.

Figura 56 - Resultado obtido para um mapa de alta complexidade via interface gráfica.





Fonte: Do Autor (2017).

5.3 Considerações finais sobre a interface desenvolvida

Em linhas gerais, pode-se concluir que a interface gráfica criada no âmbito deste trabalho mostrou-se de grande utilidade para as finalidades propostas. Numa primeira abordagem teve-se a intenção de criar um ambiente computacional que pudesse atrair a atenção de possíveis interessados no assunto e complementar os conhecimentos que venham a ser adquiridos em geração de trajetórias e caminho mínimo para plataformas móveis.

Sem a necessidade de se ter um conhecimento avançado sobre as funcionalidades da ferramenta MatLab, o utilizador pretendia fornecer os parâmetros referentes à entrada de dados e escolher o algoritmo de otimização. Neste sentido, esta interface mostra-se de grande utilidade, graças a um conjunto de informações que se encontram disponíveis para auxiliar o utilizador a fazer uma correta utilização da interface. Esta interface também pode ser de grande utilidade para usuários que já tenham algum conhecimento sobre o assunto, possibilitando a criação de novas funcionalidades, para além daquelas que foram utilizadas no estudo.

Apesar deste ambiente computacional ter sido criado para um propósito específico, ou seja, para simular os sistemas, é sempre possível que o mesmo possa ser modificado de forma a atender a outros interesses ou, até mesmo, sofrer melhorias na sua concepção. A interface desenvolvida serviu não apenas

como agente factível dos conhecimentos adquiridos ao longo da pesquisa efetuada, como também servirá de elemento enriquecedor a ser explorado em ambiente acadêmico, e até mesmo para futuras dissertações que venham a ser desenvolvidas nesta área de pesquisa.

6 CONCLUSÕES

A tecnologia robótica está desempenhando um papel de grande importância em diversos seguimentos da sociedade. O crescimento acelerado de manipuladores robóticos bem como plataformas robóticas móveis, é objeto de estudo de vários pesquisadores, cujo objetivo é aumentar a eficiência global em diversos tipos de mercado, que vão desde o setor industrial aos afazeres domésticos. Tal condição tem por base o uso eficiente dos recursos existentes, bem como o consumo de energia e outras fontes necessárias ao correto funcionamento desta tecnologia.

Diferentemente da atividade industrial como automotiva, aeroespacial dentre outras, a atividade agrícola gera uma margem de lucro inferior as demais e, devido a tal fato, é essencial que a tecnologia robótica empregada no meio agrícola seja robusta e acessível a todos os usuários. É fato que a prática da agricultura atual possui uma miscigenação entre o tradicional e o moderno, cuja multidisciplinaridade e os avanços tecnológicos passam a envolver as novas atividades e abrem a oportunidade para a inserção de sistemas autônomos no campo.

O impacto que a tecnologia robótica irá causar no processo produtivo rural, ainda é tema de discussão entre diversos pesquisadores. É de conhecimento que o capitalismo e o avanço tecnológico industrial são promotores de uma diminuição significativa do número de trabalhadores rurais. As máquinas alteraram o modo de vida dos fazendeiros e produtores rurais, visto que todos estes, em conjunto com seus empregados, possuem conhecimento sobre como operar um maquinário sofisticado, bem como efetuar as devidas manutenções. Neste sentindo, o trabalho realizado apresenta-se como um agente facilitador e gerador de economia nas atividades rurais e agrícolas, onde o mesmo venha a ser utilizado.

A contribuição primal deste trabalho concerne na análise, implementação e validação de um conjunto de técnicas de otimização utilizadas para a geração de uma trajetória entre dois pontos previamente estabelecidos, com base no menor caminho e diante a presença ou não de obstáculos na trajetória. Tais metodologias foram comparadas em termos de eficiência computacional, soluções numéricas e possibilidades de encontrar uma solução admissível de forma a não ficar presos em mínimos locais. Uma das principais contribuições foi a realização de planejamento e otimização de trajetória em uma única etapa.

Todas as técnicas para o planejamento do caminho mínimo e geração de trajetória discutidas ao longo do trabalho podem ser aplicáveis a todos os tipos de robôs, sejam manipuladores ou móveis. Neste trabalho, utilizou-se a plataforma móvel descrita no capítulo 3 para validação, devido à viabilidade e disponibilidade dos recursos para a sua construção. Este fato possibilitou uma implementação dinâmica realística e implementação de parâmetros cinemáticos, em vez de assumir quaisquer parâmetros robóticos irreais.

O planejamento do caminho é o primeiro passo da solução generalizada de otimização do caminho de plataformas móveis. No Capítulo 2, o planejamento de caminhos foi discutido utilizando um conjunto de algoritmos de otimização, com o intuito de aumentar a probabilidade de encontrar uma solução admissível. Embora bons resultados tenham sido obtidos com o uso do MRP e do algoritmo A estrela, ainda faz-se necessário diferentes técnicas de busca, dada a complexidade do ambiente e das restrições impostas ao longo do percurso.

Outra grande contribuição deste trabalho é a implementação de um planejador de caminho mínimo que envolve a otimização do percurso a ser realizado, levando em consideração as restrições impostas a longo do ambiente, bem como as restrições cinemáticas da plataforma móvel. Ao combinar o

planejamento do caminho e a otimização da trajetória em uma única etapa, a dinâmica do robô foi incorporada em cada etapa do processo cujas restrições foram consideradas em todas as etapas, gerando resultados mais praticáveis.

A validação deste trabalho deu-se com base na utilização dos pacotes computacionais por um conjunto de plataformas móveis desenvolvidas e utilizadas em diferentes cenários, provando a robustez e a versatilidade dos códigos implementados. Apesar das limitações físicas existentes em cada plataforma, foi possível gerar trajetórias executáveis que levassem em consideração as restrições impostas no ambiente, bem como as restrições cinemáticas de cada robô móvel desenvolvido.

A implementação de uma interface gráfica para a manipulação de todos os códigos desenvolvidos mostrou-se ser uma importante ferramenta didática e intuitiva para os usuários menos experientes. O ambiente de simulação utilizado, MatLab, também se mostrou ser uma importante ferramenta, devido à facilidade de implementação dos algoritmos, bem como o desenvolvimento da interface gráfica.

Neste sentido, todos os objetivos do trabalho foram alcançados, e a implementação de cada algoritmo foi verificada em diferentes cenários, validando a robustez do pacote computacional formado pelo conjunto de técnicas de otimização implementadas. Ademais, a metodologia foi além da proposta inicial, visto que diversos cenários e diferentes plataformas fizeram uso dos algoritmos de geração de caminho mínimo e obtiveram resultados satisfatórios em diversas situações.

Contudo, o conhecimento gerado é de fundamental importância para uma satisfação pessoal em alcançar os resultados propostos, bem como produzir um trabalho que contribua para o desenvolvimento intectual dos discentes desta e de outras universidades, com o objetivo de contribuir para uma atividade

agrícola mais sustentável e que otimize os seus lucros com os recursos existentes.

6.1 Perspectivas para Trabalhos Futuros

Neste trabalho discutiu-se o planejamento e otimização de trajetória para robótica móvel usando diferentes técnicas de otimização. A praticidade e a comparação entre diferentes técnicas são discutidas por meio de exemplos numéricos. Além disso, há espaço para mais melhorias em futuros trabalhos, entre as quais se destaca as seguintes:

a) Avaliação de um requisito temporal de forma a obter uma resposta admissível que possa ser utilizada para uma programação on-line;
b) Utilização de uma estrutura híbrida que possa conjugar diferentes técnicas de otimização em um único algoritmo planejador de caminho mínimo. Tal citação deve-se ao fato que cada algoritmo possuir sua particularidade e poder ser ineficiente quando se trata de refinar uma solução; ficar preso em mínimos locais ou, após colisão com um determinado obstáculo, não finalizar o seu critério de pesquisa. Desta forma, ao utilizar uma estrutura hibrida é possível diminuir as limitações existentes em cada algoritmo;
c) Utilização de outras técnicas evolutivas, como por exemplo o uso do PSO (*Particle Swarm Optimizacion*) em comparação com os resultados obtidos pelos algoritmos genéticos em sistemas que representam dinâmica complexa;
d) Utilização de diferentes técnicas de controle na determinação dos melhores parâmetros para o controlador PID implementado. Tal controlador pode ser empregado em sistemas uni ou multi-objetivos, utilizando como referência o valor da sobreelongação ou o tempo de

estabelecimento. Este estudo pode ser feito utilizando as funções objetivo desenvolvidas neste estudo, acrescentando restrições referentes ao desempenho dos parâmetros avaliados;

e) Estudar a viabilidade de outros parâmetros na configuração do AG, como, por exemplo, o uso de funções de penalidades ou, até mesmo, o uso de regras determinísticas para a obtenção dos valores como a taxa de cruzamento, tamanho da população, taxa de mutação e número de gerações;
f) Implementação de uma rotina que se adapte a ambientes dinâmicos e, por consequência, gere respostas a um tempo computacional reduzido e suficiente para uma ação de controle admissível;
g) Desenvolver novas ferramentas de controle na interface criada, de forma a permitir que esta tenha uma maior aplicabilidade em ambientes reais;
h) Fazer uso de sensores internos e externos às plataformas móveis desenvolvidas nesta pesquisa para a realização de um controle realimentado e que minimize os erros obtidos ao longo do percurso realizado.

Neste trabalho, apresentou-se um critério de otimização conjunta para assegurar a solução ótima global ao utilizar uma programação off-line em ambiente estático. Contudo, este é um critério de otimização baseado em heurísticas e muito difundido na comunidade acadêmica, embora não existam provas matemáticas que assegurem a melhor condição de otimização. Embora tenha sido efetuada uma pesquisa profunda sobre a otimização de trajetórias de plataformas móveis, consta que ainda existe um grande atraso em técnicas de otimização para trajetórias on-line. Esta, é uma área de investigação que ainda

requer maior atenção, dadas as necessidades que surgem com o avanço da tecnologia em robótica móvel.

REFERÊNCIAS

AGMON, N.; KRAUS, S.; KAMINKA, G. Multi-robot perimeter patrol in adversarial settings. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2008, Pasadena. **Proceedings…** Pasadena: IEEE, 2008. 1 CD-ROM.

ALEXOPOULOS, C.; GRIFFIN, P. M. Path planning for a mobile robot. **IEEE Transactions on Systems, Man and Cybernetics**, New York, v. 22, n. 2, p. 318-322, 1992.

ARULAMPALAM, M. S. et al. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. **IEEE Transactions on Signal Processing**, New York, v. 50, n. 2, p. 174-188, Feb. 2002.

BAILEY, T. **Mobile robot localisation and mapping in extensive outdoor environments**. 2002. 50 p. Thesis (Ph.D. of Philosophy)-University of Sydney, Sydney, 2002.

BARBERA, T. et al. How task analysis can be used to derive and organize the knowledge for the control of autonomous vehicles. **Robotics and Autonomous Systems**, Amsterdam, v. 49, n. 1/2, p. 67-78, 2005.

BEGUM, M. et al. Integrated fuzzy logic and genetic algorithmic approach for simultaneous localization and mapping for mobile robots. **Applied Soft Computing**, New York, v. 8, n. 1, p. 150-165, Jan. 2006.

BIN-QIANG, Y.; MING-FU, Z.; YI, W. Research of path planning method for mobile robot based on artificial potential field. In: INTERNATIONAL CONFERENCE ON MULTIMEDIA TECHNOLOGY, 2011, Hangzhou. **Proceedings…** Hangzhou, 2011. 1 CD-ROM.

BORENSTEIN, J.; KOREN, Y. Real-time obstacle avoidance for manipulators and mobile robots. **IEEE Transactions on Systems, Man and Cybernetics**, New York, v. 5, n. 19, p. 1179-1187, 1989.

BOSSE, M. et al. Simultaneous localization and map building in large-scale cyclic environments using the atlas framework. **International Journal of Robotics Research**, Cambridge, v. 23, n. 12, p. 1113-1139, 2004.

BUENO, S. et al. Uma plataforma para pesquisa e desenvolvimento em robótica terrestre de exterior. In: SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE, 9., 2009, Brasília, DF. **Anais...** Brasília, DF, 2009. 1 CD-ROM.

CHEN, H. **Design of a controlled environment agricultural plant inspection robot**. 2012. 91 p. Thesis (Master of Science Industrial Engineering)-University of Iowa, Iowa, 2012.

CHEN, L.; LIU, D. Y. An efficient algorithm for finding a collision-free path among poly obstacles. **Journal of Robotics Systems**, Reno, v. 7, n. 1, p. 129-137, 1997.

CHOSET, H. Coverage for robotics: a survey of recent results. **Annals of Mathematics and Artificial Intelligence**, New York, v. 31, n. 1, p. 113-126, Oct. 2001.

DAVIS, L. **Handbook of genetic algorithms**. New York: V. N. Reinhold, 1992. 385 p.

DEL, H. A. R.; MEDRANO, M. N.; MARTIN, D. B. B. A simple approach to robot navigation based on cooperative neural networks. In: IEEE ANNUAL CONFERENCE OF THE INDUSTRIAL ELECTRONICS SOCIETY, 28., 2002, Sevilla. **Proceedings…** Sevilla: IEEE, 2002. p. 2421-2426.

DIAS, J.; CAPTIVO, M. E.; CLIMACO, J. **A hybrid algorithm for dynamic location problems**. Coimbra: Universidade de Coimbra; Lisboa: Universidade de Lisboa, 2005. 49 p.

DOUCET, A. **On sequential simulation-based methods for Bayesian filtering**. Cambridge: University of Cambridge, 1987. 26 p.

DUCHOŇ, F. et al. Path planning with modified a star algorithm for a mobile robot. **Procedia Engineering**, New York, v. 96, p. 59-69, 2014.

DUCKETT, T.; NEHMZOW, U. Mobile robot self-localisation using occupancy histograms and a mixture of Gaussian location hypotheses. **Robotics and Autonomous Systems**, Amsterdam, v. 34, n. 2/3, p. 119-130, 2002.

EARL, R.; THOMAS, G.; BLACKMORE, B. S. The potencial role of GIS in autonomous field operations. **Computers and Eletronics in Agriculture**, New York, v. 25, p. 107-120, 2000.

FERGUSON, D.; LIKHACHEV, M.; STENTZ, A. A guide to heuristic-based path planning. In: INTERNATIONAL WORKSHOP ON PLANNING UNDER UNCERTAINTY FOR AUTONOMOUS SYSTEMS; INTERNATIONAL CONFERENCE ON AUTOMATED PLANNING AND SCHEDULING, 2005, Pittsburgh. **Proceedings…** Pittsburgh: ICAPS, 2005. 1 CD-ROM.

GE, S. S.; CUI, Y. J. Newpotential functions for mobile robot path planning. ***IEEE Transactions on Robotics and Automation***, *New York, v.* 16, n. 5, p. 615-620, 2002.

GERAERTS, R.; OVERMARS, M. A comparative study of probabilistic roadmap planners. In: WORKSHOP ON THE ALGORITHMIC FOUNDATIONS OF ROBOTICS, 2002, Nice. **Proceedings…** Nice, 2002. 1 CD-ROM.

GLAYBSON PARREIRAS. Consultoria Imobiliária. Disponível em: <http://glaybson.com.br/comerciais/comercial-mayfair-offices-salas-comerciais-funcionarios>. Acesso em: 10 jan. 2017.

GOLDBERG, D. **Genetic algorithm in search, optimization, and machine learning**. Addison: Wesley, 1989. 432 p.

GRISETTI, G.; STACHNISS, C.; BURGARD, W. Improving grid-based SLAM with Rao-Blackwellized particle filters by adaptive proposals and selective resampling. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AUTOMATION, 2005, Barcelona. **Proceedings…** Barcelona: ICRA, 2005. p. 2443-2448.

GUE, J.; GAO, Y.; CUI, G. Path planning of mobile robot base on improved potential field. **Information Technology Journal**, Deira, v. 12, p. 2188-2194, July 2013.

GUIVANT, J. et al. Navigation and mapping in large unstructured environments. **The International Journal of Robotics Research**, Cambridge, v. 23, p. 449-472, 2004.

HACKENHARR, N. M.; HACHENHARR, C.; ABREU, Y. V. de. Robótica na agricultura. **Interações**, Campo Grande, v. 16, n. 1, p. 119-129, jan./jun. 2015.

HART, P.; NILSSON, N.; RAFAEL, B. A formal basis for the heuristic determination of minimum cost paths. **IEEE Transactions on Systems, Man and Cybernetics**, New York, v. 4, p. 100-107, 1968.

HOLLAND, J. **Adaptation in natural and artificial systems**. Ann Arbor: University of Michigan Press, 1975. 232 p.

INAMASU, R. Y. et al. **Sistema de informação em elementos de aquisição de dados para ambiente agropecuário**. 2011. Disponível em: <http://ainfo.cnptia.embrapa.br/digital/bitstream/CNPDIA/9807/1/PA10_96.pdf >. Acesso em: 10 out. 2016.

JULIER, S. J.; UHLMANN, J. K.; DURRANT-WHYTE, H. F. A new approach for the nonlinear transformation of means and covariances in linear filters. **IEEE Transactions on Automatic Control**, Minato-Ku, v. 45, n. 3, p. 477-482, Mar. 1997.

KAVRAKI, L. et al. Probabilistic roadmaps for fast path planning in high dimensional configuration spaces. **Robotics and Automation**, New York, v. 12, n. 4, p. 566-580, Aug. 2002.

KAVRAKI, L. et al. Probabilistic roadmaps for path planning in highdimensional configuration spaces. **IEEE Transactions on Robotics and Automation**, New York, v. 12, n. 4, p. 566-580, 1996.

KHOOGAR, A. R.; PARKER, J. K. Obstacle avoidance of redundant manipulators using genetic algorithms. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1991, Sacramento. **Proceedings…** Sacramento: IEEE, 1991. p. 317-320.

KOREN, Y.; BORENSTEIN, J. Potential field methods and their inherent limitations for mobile robot navigation. In: IEEE CONFERENCE ON ROBOTICS AND AUTOMATION, 1991, Sacramento. **Proceedings…** Sacramento: IEEE, 1991. 1 CD-ROM.

KROGH, B. H.; THORPE, C. E. Integrated path planning and dynamic steering control for autonomous vehicles. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1986, San Francisco. **Proceedings…** San Francisco: IEEE, 1986. 1 CD-ROM.

LAVALLE, S. M. **Planning algorithms**. New York: Cambridge University Press, 2006. 844 p.

LAVALLE, S. M. **Rapidly-exploring random trees:** a new tool for path planning. Ames: Iowa State University, 1998. 4 p.

LAVALLE, S.; KUFFNER, J. Randomized kinodynamic planning. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1999, Brisbane. **Proceedings…** Brisbane: ICRA, 1999. p. 3122-3127.

LAVALLE, S.; KUFFNER, J. **Rapidly-exploring random trees:** progress and prospects. 2001. Disponível em: <http://msl.cs.uiuc.edu/~lavalle/papers/LavKuf01.pdf>. Acesso em: 10 out. 2016.

LIU, S.; SUN, D.; ZHU, C. Coordinated motion planning for multiple mobile robots along designed paths with formation requirement. **IEEE/ASME Transactions on Mechatronics**, Piscataway, v. 16, n. 6, p. 1021-1031, Sept. 2010.

MERWE, R. van der; WAN, E. Sigma-point kalman filters for integrated navigation. In: ANNUAL MEETING, INSTITUTE OF NAVIGATION, 60., 2004, Dayton. **Proceedings…** Dayton: ION, 2004. p. 641-654.

MITCHELL, M. **An introduction to genetic algorithms**. London: MIT Press, 1999. 221 p.

MOUTARLIER, P.; CHATILA, R. G. Stochastic multisensory data fusion for mobile robot location and environment modeling. In: INTERNATIONAL SYMPOSIUM ON ROBOTICS RESEARCH, 1989, Tokyo. **Proceedings…** Tokyo, 1989. p. 85-94.

NIKU, S. B. **Introdução à robótica:** análise, controle, aplicações. 2. ed. Rio de Janeiro: LTC, 2013. 402 p.

NILSSON, N. **Principles of artificial intelligence**. Berlin: Springer-Verlag, 1980. 476 p.

NOBORU, N.; HIDEO, T. Path planning of agricultural mobile robot by neural network and genetic algorithm. **Computers and Electronics in Agriculture**, New York, v. 18, p. 187-204, 1997.

NOSRATI, M.; KARIMI, R.; HASANVAND, H. A. Investigation of the *(Star) search algorithms: characteristics, methods and approaches. **World Applied**

Programming, Dadaab, v. 2, n. 4, p. 251-256, 2012.

OCHI, L. S. **Algoritmos genéticos:** origem e evolução. Niterói: Ed. UFF, 2009. Disponível em: <http://www.sbmac.org.br/bol/bol-2/artigos/satoru/satoru.html#geneticos>. Acesso em: 10 dez. 2009.

RAJA, P.; PUGAZHENTHI, S. Optimal path planning of mobile robots: a review. **International Journal of Physical Sciences**, Lagos, v. 7, n. 9, p. 1314-1320, Feb. 2012.

RAM, A.; ARKIN, R.; BOONE, G. Using genetic algorithms to learn reactive control parameters for autonomous robotic navigation. **Adaptive Behavior**, Cambridge, v. 2, n. 3, p. 100-107, 1994.

RAMAKRISHNAN, R.; ZEIN-SABATTO, S. Multiple path planning for A group of mobile robots in A 3D environment using genetic algorithms. In: IEEE SOUTHEASTCON SOUTH CAROLINA, 2001, Kentucky. **Proceedings…** Kentucky, 2001. p. 359-363.

RISTIC, B.; ARULAMPALAM, S.; GORDON, N. **Beyond the Kalman filter:** particle filters for tracking applications. Manukau: Artech, 2004. 318 p. (Artech House Radar Library).

ROBOT OPERATING SYSTEM. Disponível em: <http://www.ros.org/>. Acesso em: 2 ago. 2016.

RUSSELL, S.; NORVIG, P. **Artificial intelligence A modern approach**. Upper Saddle River: Prentice-Hall, 1995. 1152 p.

SADATI, N.; TAHERI, J. Genetic algorithm in robot path planning problem in crisp and fuzzified environments. In: IEEE INTERNATIONAL CONFERENCE ON INDUSTRIAL TECHNOLOGY, 2002, Bangkok. **Proceedings…** Bangkok: IEEE, 2002. p. 175-180.

SANTINI, A.; NICOSIA, S.; NANNI, V. Trajectory estimation and correction for a wheeled mobile robot using heterogeneous sensors and Kalman filter. In: PREPRINTS OF IHE FIFTH IFAC SYINPOSIUNI ON ROBOT CONTROL, 1997, Nantes. **Proceedings…** Nantes, 1997. 1 CD-ROM.

SASIADEK, J. Z.; HARTANA, P. **Sensor data fusion using kalman filter**. Ontario: Carleton University, 2000. 7 p.

STENTZ, A. Optimal and efficient path planning for partially-known environments. In: IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1994, Pittsburgh. **Proceedings…** Pittsburgh: ICRA, 1994. p. 3310-3317.

SUGIHARA, K.; SUZUKI, I. Distributed algorithms for formation of geometric patterns with many mobile robots. **Robotic Systems**, New York, v. 13, p. 127-139, 1996.

SUGIHARA, K.; SMITH, J. Genetic algorithms for adaptive motion planning of an autonomous mobile robot. In: IEEE INTERNATIONAL SYMPOSIUM ON COMPUTATIONAL INTELLIGENCE IN ROBOTICS AND AUTOMATION, 1997, Monterey. **Proceedings…** Monterey: IEEE, 1997. p. 138-146.

TANGERINO, G. et al. Controle de esterçamento de robô agrícola móvel de quatro rodas guiáveis. In: SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE, 10., 2011, São João Del-Rei. **Anais...** São João Del-Rei: SBAI, 2011. 1 CD-ROM.

THRUN, S. **Robotic mapping:** a survey. Pittsburgh: M. Kaufmann, 2002. 28 p.

THRUN, S.; BURGARD, W.; FOX, D. **Probabilistic robotics**. Cambridge: The MIT Press, 2004. 672 p.

THRUN, S.; BURGARD, W.; FOX, D. **Probabilistic robotics**. Cambridge: The MIT Press, 2005. 672 p.

THRUN, S.; LIU, Y. Multi-robot SLAM with sparse extended information filers. In: INTERNATIONAL SYMPOSIUM OF ROBOTICS RESEARCH, 11., 2005, Sienna. **Proceedings…** Sienna: Springer, 2005. 1 CD-ROM.

UNIVERSIDADE FEDERAL DE MINAS GERAIS. Laboratório de Sistemas de Computação e Robótica. **Carro autônomo CADU**. Belo Horizonte, 2010. Disponível em: <http://coro.cpdee.ufmg.br/index.php/fotos-e-videos/45-videos/134-carro-autonomo-cadu-setembro-de-2010>. Acesso em: 10 out. 2016.

VADAKKEPAT, P.; TAN, K. C.; MING-LIANG, W. Evolutionary artificial potential fields and their application in real time robot path planning. In: CONGRESS ON EVOLUTIONARY COMPUTATION, 2000, State College. **Proceedings…** State College, 2000. 1 CD-ROM.

VALENCIA, R.; ANDRADE-CETTO, J.; PORTA, J. M. Path planning in belief space with Pose SLAM. In: IEEE INTERNATIONAL CONFERENCE ROBOTICS AND AUTOMATION, 2011, Pittsburgh. **Proceedings…** Pittsburgh: ICRA, 2011. 1 CD-ROM.

VICTORINO, A. C.; RIVES, P. SLAM with consitent mapping in an hybrid model. In: IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS, 2006, Beijing. **Proceedings…** Beijing, 2006. 1 CD-ROM.

WANG, C.; SOH, Y. C.; WANG, H. A hierarchical genetic algorithm for path planning in A static environment with obstacles. In: IEEE CANADIAN CONFERENCE ON ELECTRICAL AND COMPUTER ENGINEERING, 2002, Ontario. **Proceedings…** Ontario: IEEE, 2002. p. 1652-1657.

WEIJUN, S.; RUI, M.; CHONGCHONG, Y. A study on soccer robot path planning with fuzzy artificial potential field. In: INTERNATIONAL CONFERENCE ON COMPUTING, CONTROL AND INDUSTRIAL ENGINEERING, 2010, Wuhan. **Proceedings…** Wuhan: CCIE, 2010. 1 CD-ROM.

WU, K. H.; CHEN, C. H.; LEE, J. D. Genetic-based adaptive Fuzzy controller for robot path planning. In: IEEE INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS, 5., 1996, New Orleans. **Proceedings…** New Orleans, 1996. p. 1687-1692.

ZARATE, L. E. et al. An artificial neural network structure able to obstacle avoidance behavior used in mobile robots. In: IEEE ANNUAL CONFERENCE OF THE INDUSTRIAL ELECTRONICS SOCIETY, 28., 2002, Sevilla. **Proceedings…** Sevilla: IEEE, 2002. p. 2457-2461.

ANEXO A - PLATAFORMAS MÓVEIS PARA VALIDAÇÃO DA METODOLOGIA PROPOSTA

Seguem abaixo uma descrição das plataformas móveis construídas no âmbito deste trabalho é que se utilizaram do algoritmo, Mapa de Rotas Probabilísticos, para geração e controle de uma trajetória previamente especificada. Tal algoritmo foi o escolhido devido as trajetórias geradas serem caracterizadas em movimentos sucessivos de translações e rotações. Neste sentido fez-se uso dos conceitos de transformações homogêneas combinadas de manipuladores para aplicação em robótica móvel.

Tais transformações consistem em uma série de translações e rotações sucessivas em relação aos eixos dos sistemas de referência fixo ou o movimento dos eixos do referencial atual. Qualquer transformação pode ser resolvida em um conjunto de translações e rotações em uma ordem particular, e utilizando-se dos caminhos gerados pelo algoritmo MRP é possível aplicar tais conceitos às plataformas desenvolvidas.

6.2 Seguidor de Trajetória com Arduino Uno (Mega 328P);

Para a plataforma descrita nesta subseção, tem-se um conjunto de imagens, representados pelas Figuras 1 e 2, que descrevem a validação do modelo implementado para o respectivo robô.

Característica física:

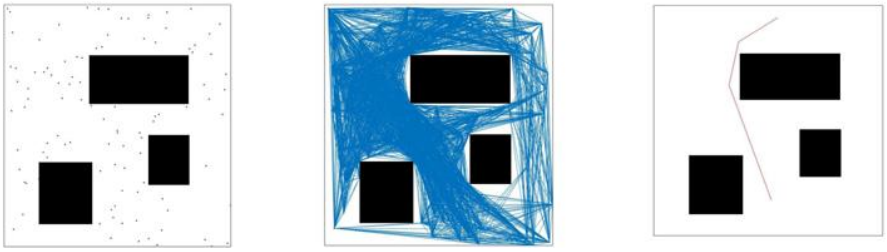
1 Arduino Uno (Mega 328P);
2 baterias (9V);
1 shield com ponte H;
1 plataforma de plástico e isopor;

2 rodas de plástico;

2 motores DC (9V).

Trajetória MRP:

Figura 1 – Trajetória gerada pelo algoritmo MRP para plataforma Arduino Uno.



Fonte: Do Autor (2017).

Modelo e ambiente de simulação:

Figura 2 – Modelo e ambiente de simulação para a plataforma Arduino Uno.

Fonte: Do Autor (2017).

Resultados:

Com o dimensionamento espacial adequado, o robô fez o trajeto com êxito, atendendo o que era esperado (tendo como base a simulação e o caminho ótimo). Porém, o carro não teve precisão ao andar em trajetória retilínea, tendendo a curvar para direita, devido a falta de distribuição de peso adequada e a diferença entre os motores (estrutura interna e acionamento).

6.3 Seguidor de Trajetória com Lego Mindstorm NXT

Para a plataforma descrita nesta subseção, tem-se um conjunto de imagens, representados pelas Figuras 3 e 4, que descrevem a validação do modelo implementado para o respectivo robô.

Característica física:

92 peças de LEGO;

2 servomotores;

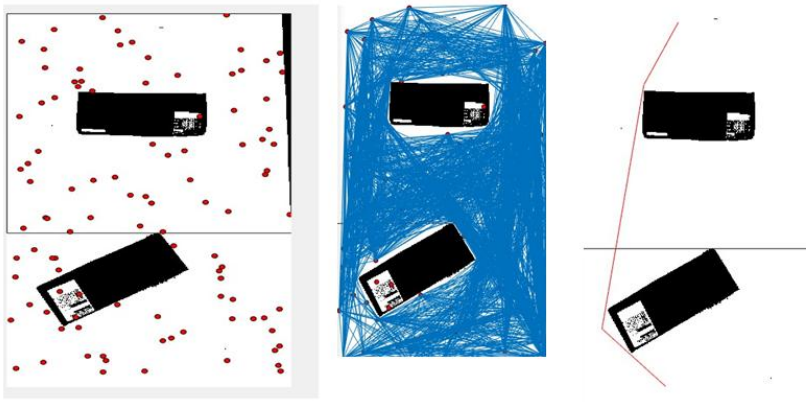
2 cabos seriais;

1 bateria;

1 microcontrolador LEGO Mindstorms NXT.

Trajetória MRP:

Figura 3 – Trajetória gerada pelo algoritmo MRP para plataforma Lego Mindstorm NXT.



Fonte: Do Autor (2017).

Modelo e ambiente de simulação:

Figura 4 – Modelo e ambiente de simulação para a plataforma Lego Mindstorm NXT.

Fonte: Do Autor (2017).

Resultados:

Foi verificado, com base na menor rota encontrada, que os movimentos deveriam ser os seguintes: Rotação de 49° em torno do eixo Z; Translação de 40,86 cm (direção apontada); Rotação de -57,6º em torno do eixo Z; Translação de 118,5 cm (direção apontada); Rotação de -20° em torno do eixo Z; Translação de 33,4 cm (direção apontada).

Com base na sequência de movimentos descrita foram montadas as seguintes matrizes de transição:

$$H_0^1 = \begin{bmatrix} 0.6561 & -0.7547 & 0 & 26.8722 \\ 0.7547 & 0.6561 & 0 & 30.9129 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad H_1^2 = \begin{bmatrix} 0.5358 & 0.8443 & 0 & 63.4955 \\ -0.8443 & 0.5358 & 0 & -100.0529 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H_2^3 = \begin{bmatrix} 0.9397 & 0.3420 & 0 & 31.3857 \\ -0.3420 & 0.9397 & 0 & -11.4235 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H_0^3 = \begin{bmatrix} 0.8780 & 0.4787 & 0 & 173.3644 \\ -0.4787 & 0.8780 & 0 & -2.7953 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Foi necessário calibrar o robô, uma vez que os parâmetros passados na sua programação são em rotações do motor. Os resultados obtidos foram:

220° reais = 1 rotação de um motor com 75% de potência;

17,76 cm reais = 1 rotação dos dois motores com 75% de potência.

6.4 Seguidor de Trajetória com Arduino UNO and Shield

Para a plataforma descrita nesta subseção, tem-se um conjunto de imagens, representados pelas Figuras 5 e 6, que descrevem a validação do modelo implementado para o respectivo robô.

Característica física:

2 motores DC 5V;

Duas rodas fixas e uma roda móvel;

Arduino UNO;

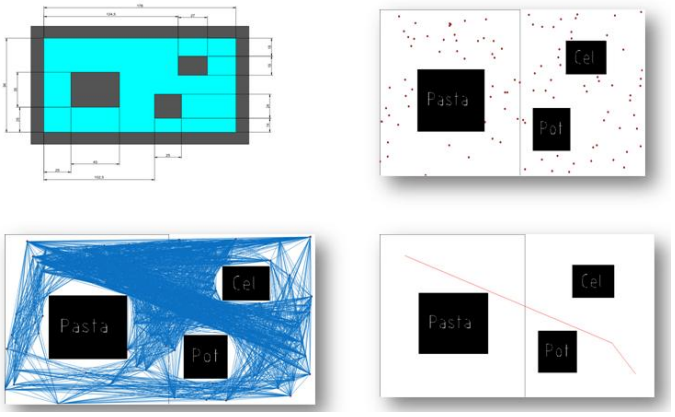
Shield L298 Sparkfun;

Bateria 12V;

Cabos e conexões

Trajetória MRP:

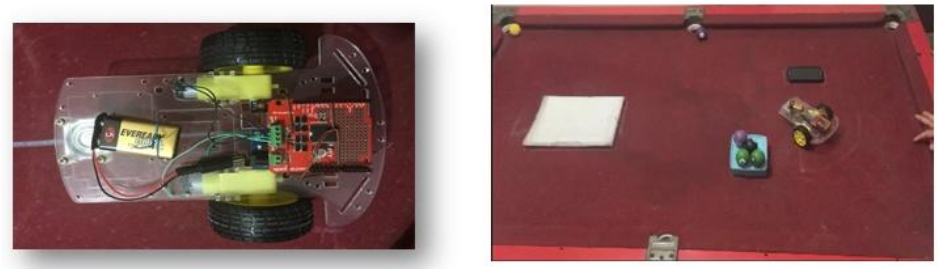
Figura 5 – Trajetória gerada pelo algoritmo MRP para a plataforma Arduino UNO and Shield.



Fonte: Do Autor (2017).

Modelo e ambiente de simulação:

Figure 6 – Modelo e ambiente de simulação para a plataforma Arduino UNO and Shield.



Fonte: Do Autor (2017).

Resultados:

O objetivo principal deste protótipo foi movimentar de um ponto ao outro, sem colidir com nenhum obstáculo, usando a trajetória obtida pelo algoritmo MRP. Resultados de simulações mostraram que a trajetória real traçada pelo robô móvel foi bem aproximada da trajetória ideal estipulada pela planejador. Entretanto, foi possível notar que o robô móvel poderia seguir qualquer trajetória na mesa de sinuca, desde que um novo código, com diferentes tensões aplicadas nas rodas, seguindo uma proporcionalidade de tempo, fosse implementado.

Sendo assim, para o projeto, a trajetória do robô móvel foi controlada por tensões nos motores DC acopladas as rodas, dado possibilidade de controlar a distância percorrida no eixo x, no eixo y, e a respectiva rotação do robô móvel.

6.5 Seguidor de Trajetória "Robô Explorador"

Para a plataforma descrita nesta subseção, tem-se um conjunto de imagens, representados pelas Figuras 7 e 8, que descrevem a validação do modelo implementado para o respectivo robô.

Característica física:

PIC16F628A

Motores - 1 prara tração e 4 servos Hobbico CS-60

Sensores - 2 bumpers, 1 entrada extra, 1 saida por relé

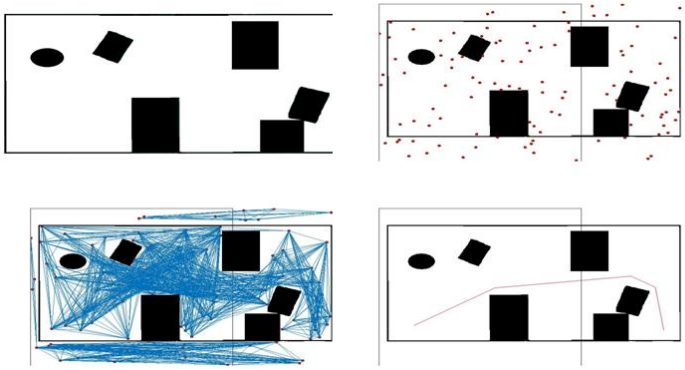
Alimentação- 1 bateria GEL 12V/1.3Ah

Chassi – madeira

Programa - Assembly Microchip (robô) e LOGO (PC)

Trajetória MRP:

Figure 7 – Trajetória gerada pelo algoritmo MRP para a plataforma "Robô Explorador".



Fonte: Do Autor (2017).

Modelo e ambiente de simulação:

Figure 8 – Modelo e ambiente de simulação para a plataforma "Robô Explorador".

Fonte: Do Autor (2017).

Resultados:

A plataforma móvel denominada, Robô Explorador, foi capaz de "explorar" um ambiente conectado através do canal serial de um PC. Todo o controle do robô foi realizado através de um programa desenvolvido, especificamente para este propósito, na linguagem LOGO. Todo o trajeto foi executado conforme especificação do planejador. Entretanto, observou-se a necessidade de um controle em malha fechada, devido a certas imposições que venham a ser inseridas no ambiente e a inclusão de sensores internos e externos ao modelo para um controle adequado do mesmo.

6.6 Seguidor de Trajetória com Arduíno UNO

Para a plataforma descrita nesta subseção, tem-se um conjunto de imagens, representados pelas Figuras 9 e 10, que descrevem a validação do modelo implementado para o respectivo robô.

Característica física:

2 Rodas

2 Motores de corrente contínua.

Chassi de acrílico.

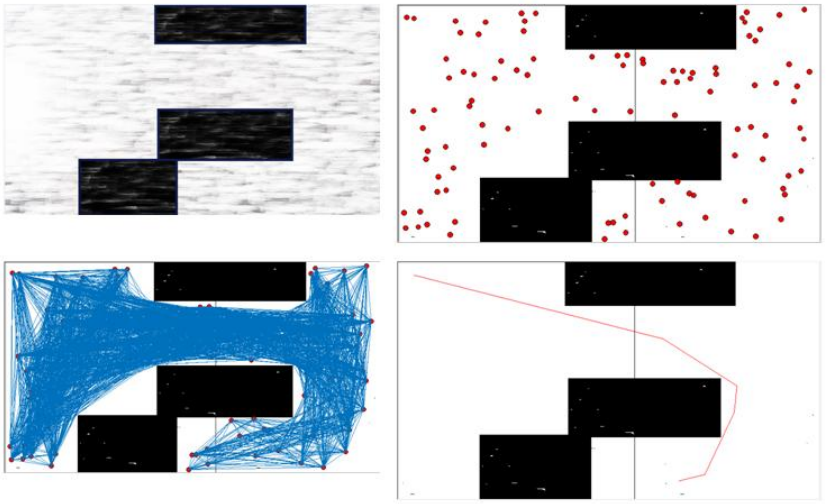
Arduino UNO.

Bateria Recarregável.

4 pilhas AA e suporte de pilha.

Trajetória MRP:

Figure 9 – Trajetória gerada pelo algoritmo MRP para a plataforma Arduíno UNO.



Fonte: Do Autor (2017).

Modelo e ambiente de simulação:

Figura 10 – Modelo e ambiente de simulação para a plataforma Arduíno UNO.

Fonte: Do Autor (2017).

Resultados:

A plataforma desenvolvida obteve um resultado esperado e a trajetória definida foi perfeitamente executável. Entretanto certas considerações podem ser

tecidas após os resultados obtidos por este projeto. A necessidade de um conjunto de sensores para um controle em malha fechada novamente se fez necessário e por isso toda a calibragem do modelo foi realizada em função do tempo de rotação e translação dos motores. A cronometragem foi realizada de forma manual para mensurar a distância percorrida e o ângulo necessário. Em linha reta, o tempo decorrido pelo robô ao percorrer 2 metros é igual a 5,73 segundos.

Desligando um dos motores e deixando o outro motor ligado, calcula-se o tempo para dar uma volta completa, cujos testes realizado em ambos os motores revelaram um valor de: Motor Esquerdo: 360º em 3.5s e Motor Direito: 360º em 3.1s

Todas as medidas foram manuais e realizadas utilizando régua para medição e no caso da imagem bitmap no computador a medição da imagem foi realizada por um software PixelMeter cuja escala utilizada foi de 1 cm na imagem de computador para 10 cm na arena física.

6.7 Controle de posição e velocidade de robô móvel Go-to-Goal com Filtro de Kalman

Diferentemente das plataformas móveis apresentadas no anexo deste documento, cuja proposta ate então era executar uma trajetória pré-definida utilizando o algoritmo MRP, o objetivo da robô Go-to-Goal descrito a seguir é validar o funcionamento do Filtro de Kalman para estimação de estados de um robô móvel, e utilizar os estados estimados para realizar o controle de posição e velocidade de um robô Go-to-Goal por realimentação de estados.

A modelagem do sistema utilizou o conceito de "Identificação Caixa-Preta" onde o sistema é visto como uma caixa fechada à qual aplicou-se uma entrada e em seguida coletou-se uma saída. O sistema é caracterizado como um robô móvel com acionamento diferencial. O acionamento é feito por PWM

(Pulse Width Modulation) regulado por um microcontrolador. Para medição de posição e velocidade, foram acoplados encoders nas rodas traseiras do robô.

Característica física:

04 Motores DC (3 ~ 6V) com redução;

04 Rodas de plástico com revestimento de borracha;

02 Placas de Acrílico (Chassis do Robô);

01 Microcontrolador Arduino Uno (Processador Atmega328P, 16MHz);

02 Encoders angulares (Encoder + Encoding Disk com 20 Furos);

01 Ponte H dupla L298N (5 ~ 35V e 2A de corrente de pico);

01 Power Bank Pineng 10000mAh com duas saídas: (5V – 1.0A) e (5V - 2.1A).

O modelo cinemático de um robô móvel com acionamento diferencial é dado pela matriz:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \frac{r\,\varphi'_1}{2} + \frac{r\,\varphi'_2}{2} \\ 0 \\ \frac{r\,\varphi'_1}{2l} - \frac{r\,\varphi'_2}{2l} \end{bmatrix} \quad (9)$$

Sendo r o raio das rodas e φ a velocidade nas rodas ativas. A velocidade no eixo x do referencial do robô, então, é dada por:

$$x' = \frac{r\,\varphi'_1}{2} + \frac{r\,\varphi'_2}{2} \quad (10)$$

Como o robô em questão possui duas rodas ativas em cada um dos lados, divide-se o resultado por dois para obtenção da velocidade real:

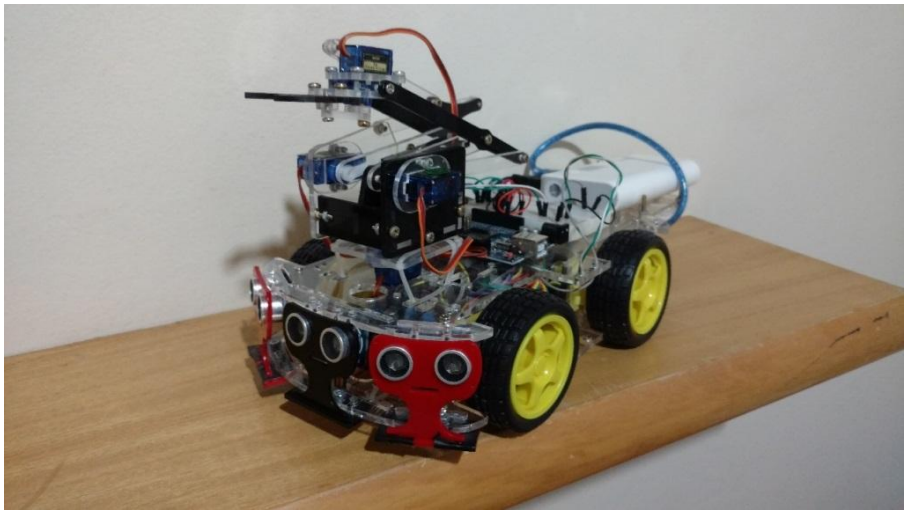
$$x' = \frac{r\,\varphi'_1}{4} + \frac{r\,\varphi'_2}{4} \tag{11}$$

Integrando a velocidade no eixo x, obtém-se a distância percorrida ao longo do tempo:

$$x = \int \left(\frac{r\,\varphi'_1}{4} + \frac{r\,\varphi'_2}{4} \right) dt \tag{12}$$

Seguindo essas equações, é possível determinar a posição do robô móvel a partir da velocidade angular das rodas, visto que o raio é fixo. A Figura 11 apresenta o modelo da plataforma em questão.

Figura 11- Robô móvel utilizado para modelagem e controle de posição e velocidade.



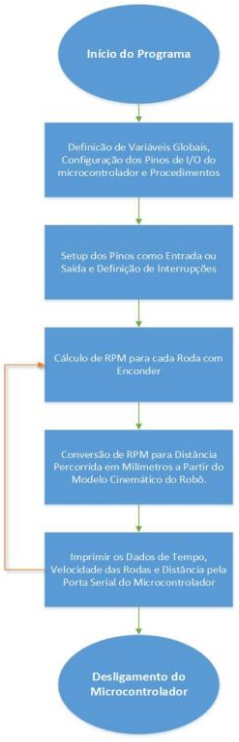
Fonte: Do Autor (2017).

A partir da equação 12 é possível determinar a posição do robô a partir da velocidade de rotação das rodas, mas a velocidade de rotação das rodas é dada a partir da tensão aplicada aos motores. A tensão aplicada é regulada a

partir de um sinal de PWM, então a função de transferência do sistema pode ser encontrada utilizando um sinal PWM como entrada e obtendo a distância percorrida como saída.

Primeiramente implementou-se um código, em linguagem C no microcontrolador, para que fosse possível coletar os dados dos encoders. Os sensores enviam pulsos para o microcontrolador, que conta quantos pulsos foram recebidos. Em seguida o microcontrolador converte o número de pulsos em Rotações Por Minutos (RPM) e então, a partir do modelo cinemático do robô, converte as RPM em milímetros percorridos pelo robô. O funcionamento do código é exibido de forma simplificada no fluxograma da Figura 12:

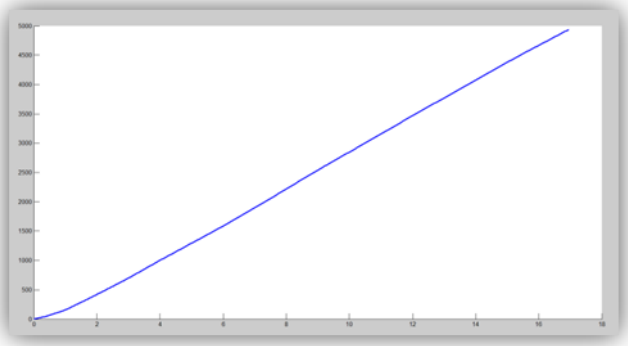
Figura 12 - Fluxograma de Funcionamento do Programa para o Ensaio e Modelagem.



Fonte: Do Autor (2017).

Conectou-se, então, o microcontrolador a um computador, por meio de uma porta USB, e utilizando comunicação serial coletou-se os dados de tempo, RPM e distância percorrida ao longo do tempo, por aproximadamente 15 segundos. A Figura 13 apresenta o gráfico de Posição x Tempo obtido:

Figura 13 - Gráfico de Distância Percorrida ao longo do tempo.



Fonte: Do Autor (2017).

Após a obtenção dos dados da saída, a partir da entrada degrau aplicada, utilizou-se a ferramenta *System Identification Application* no Matlab para obtenção de uma função de transferência com resposta correspondente à do sistema. A função de transferência obtida é dada pela equação abaixo:

$$G = \frac{5.91}{s^2 + 1.554s + 0.001} \tag{13}$$

A partir de uma regressão por diagrama de blocos, obteve-se que as seguintes matrizes de Espaço de Estados:

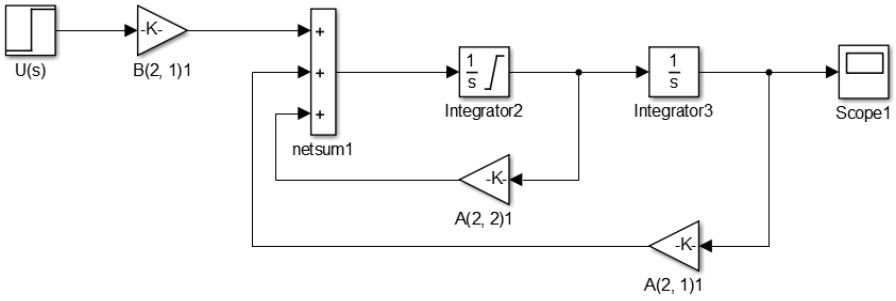
$$A = \begin{bmatrix} 0 & 1 \\ -0.001 & -1.554 \end{bmatrix} \quad (14)$$

$$B = \begin{bmatrix} 0 \\ 5.91 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Nota-se que a representação do sistema está na forma canônica controlável. Para uma melhor representação, o diagrama da Figura 14 apresenta como os estados se relacionam em malha aberta:

Figura 14 - Diagrama de Blocos da malha aberta em Espaço de Estados.



Fonte: Do Autor (2017).

Para a implementação do filtro de Kalman, as matrizes de espaço de estados foram utilizadas, para ditar a dinâmica do sistema, assim como foram definidas as variáveis do filtro. Para uma melhor visualização do funcionamento do filtro, a Figura 15 apresenta implementação em ambiente Matlab:

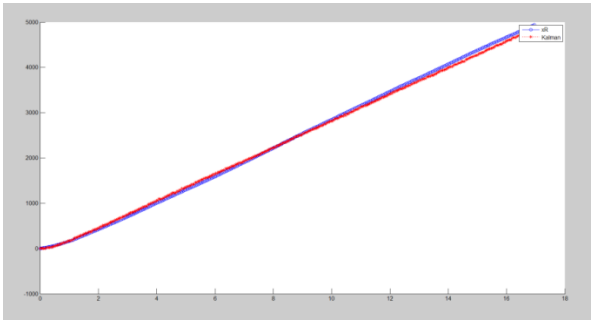
Figura 15 - Implementação do Filtro de Kalman em Matlab.

```
%%
%   ### KALMAN FILTER ###

Q = 350; %Estimate error
R = 45; %Measurement error

n = length(t);
rng default
w = sqrt(Q)*randn(n,1);
v = sqrt(R)*randn(n,1);


y = lsim(Gx, u+w, t);

%y = xR(1:length(t));
yv = y + v;

P = B*Q*B';         % Initial error covariance
x = zeros(2,1);     % Initial condition on the state
ye = zeros(length(t),1);
ycov = zeros(length(t),1);

for k = 1:length(t)

  % Measurement update
  Mn(1:2, k) = (P*C')/(C*P*C' + R);     %Kalman's Gain
  x = x + Mn(1:2, k)*(yv(k)-C*x);   % x[n|n] %Estimate
  P = (eye(2)-Mn(1:2, k)*C)*P;      % P[n|n] %Error in Estimate


  ye(k) = C*x;
  errcov(k) = C*P*C';

  % Time update
  x = A*x + B*u(k);         % x[n+1|n]
  P = A*P*A' + B*Q*B';      % P[n+1|n]

end
```

Fonte: Do Autor (2017).

O resultado apresentado na Figura 16 compara a resposta real dos sensores com a resposta estimada pelo filtro:

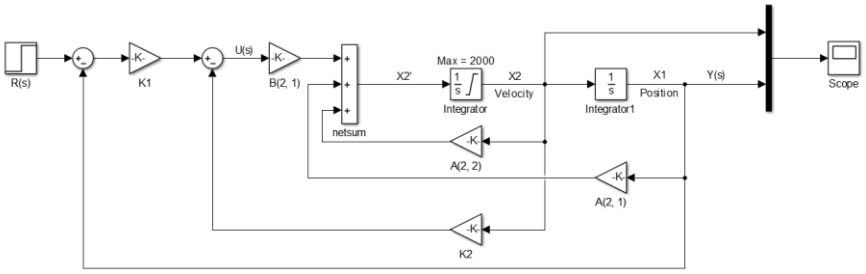
Figura 16 - Comparação entre a saída real do sistema e a saída estimada pelo filtro de Kalman.



Fonte: Do Autor (2017).

Com os resultados obtidos pelo Filtro de Kalman é possível utilizar as estimativas de estados para obter uma melhor resposta do controlador, visto que o filtro tende a diminuir o ruído do sistema. Escolheu-se então um valor de overshoot e um tempo de pico desejados para calcular os polos a serem alcançados em malha-fechada. Para o overshoot igual a 0.00001 e tempo de pico igual a 2 segundos, calculou-se que os polos em malha-fechada deveriam ser: $s_{1,2} = -5.7565 \pm 1.5708i$. Utilizando a função acker() do MATLAB, obteve-se os ganhos do controlador K1 e K2 baseados nos polos calculados: $K1 = 6.0242$ e $K2 = 1.6851$ A partir desses ganhos, desenvolveu-se uma simulação, em ambiente Simulink, da malha-fechada do sistema, como mostra o diagrama da Figura 17:

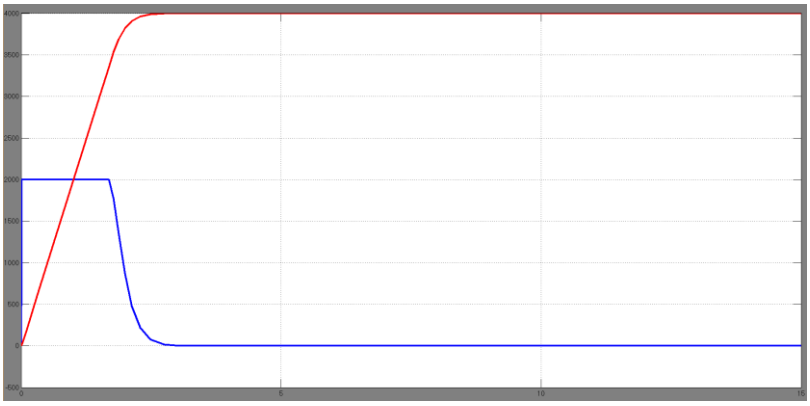
Figura 17 - Diagrama de Blocos representando a Malha-Fechada com realimentação de estados.



Fonte: Do Autor (2017).

Em seguida, definiu-se uma referência de 4000 milímetros a ser alcançada pelo sistema em malha-fechada. A resposta pode ser observada na Figura 18:

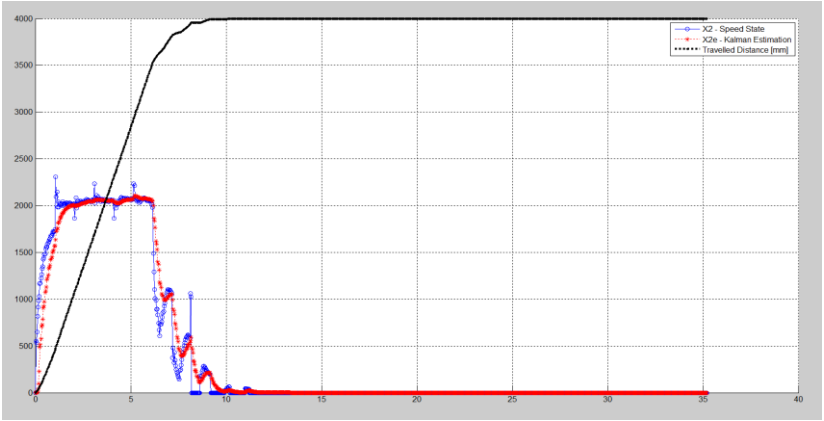
Figura 18 - Reposta em Malha-Fechada simulada.



Fonte: Do Autor (2017).

Após a implementação do código, o robô foi ligado e coletou-se os dados de distância percorrida, velocidade real e velocidade estimada pelo Filtro de Kalman. Com esses dados foi possível construir um gráfico para comparar as respostas reais do sistema com as respostas obtidas durante as simulações. O gráfico em questão pode ser observado na Figura 19.

Figura 19 - Dados coletados do sistema real ao longo do tempo.



Fonte: Do Autor (2017).

Resultados:

É possível observar que o robô se estabilizou na posição esperada. Além disso, o Filtro de Kalman atuou satisfatoriamente, filtrando os dados e acompanhando a dinâmica do sistema, tornando a resposta do controlador ainda mais precisa. O modelo obtido mostrou-se aceitável para a síntese de controladores, uma vez que sua dinâmica era bastante próxima à dinâmica do sistema. O controlador projetado obteve uma excelente resposta em malha-fechada, visto que conseguiu levar o robô à posição desejada o mais rápido possível. O Filtro de Kalman implementado apresentou bons resultados tanto para a estimação de estados, como também para a filtragem do sinal, o que é de grande ajuda para a obtenção de um sistema mais preciso.

Cabe ressaltar que diante dos materiais utilizados, os resultados foram satisfatórios, entretanto melhorias na precisão das medições e no controle da plataforma podem ser obtidas, caso fossem usados encoders nas quatro rodas do robô. Isso possibilitaria a obtenção de uma melhor média dos dados das rodas, melhorando a qualidade do modelo.

6.8 Conclusões acerca das plataformas móveis

Assim como a validação da metodologia proposta neste trabalho, teve como base o uso de uma série de pacotes computacionais, para provar a robustez dos métodos utilizados e garantir a existência de uma solução admissível, independente do mapa e das característica física do ambiente, este trabalho fez uso de um conjunto diversificado de plataformas móveis para validar as técnicas de otimização empregadas. Tais plataformas obtiveram grande êxito no que condiz a execução da trajetória informada pelo algoritmo MRP, bem como o uso da interface gráfica para geração das mesmas.

A construção dos robôs apresentados nesta subseção se deu no âmbito do curso de Engenharia de Controle e Automação da Universidade Federal de Lavras, com o apoio dos discentes do referido curso. As limitações físicas das plataformas que foram apresentadas se deram pela grande falta de material, que cada vez mais vem se tornando um limitante a um ensino de qualidade das universidade publicas brasileiras. Tais necessidades foram sanadas com material de baixo custo e ate mesmo componentes que foram reaproveitados de computadores e impressoras, que se encontram no desfazimento desta Universidade.