Planejamento de Trajetória de Múltiplos Robôs Terrestres Autônomos em Ambientes Dinâmicos

Johnathan Fercher da Rosa*, Paulo Fernando Ferreira Rosa
Instituto Militar de Engenharia, Praça General Tibúrcio, 80, 22290-270, Praia Vermelha, Rio de Janeiro, RJ, Brasil.
***johnathanfercher22@gmail.com**

RESUMO: Este artigo demonstra uma abordagem para tratar o problema de planejamento de trajetória de múltiplos robôs em ambientes dinâmicos. Nesse trabalho, buscou-se tratar o problema de maneira a possibilitar o uso de diferentes tipos de robôs em ambientes com presença de obstáculos móveis. Devido ao dinamismo do problema, foram utilizadas técnicas de baixo custo computacional para uma execução em tempo real e navegação segura. Por último, o artigo busca resolver o problema em ambientes reais. A solução proposta encontra caminhos viáveis e guia robôs ao longo dos mesmos, boa parte das colisões são evitadas durante o percurso, porém algumas ainda ocorrem.

PALAVRAS-CHAVE: Planejamento de Trajetória. Múltiplos Robôs. Autônomos. Tempo Real. Robôs Cooperativos.

ABSTRACT: This paper presents an approach to treat the problem of multiple robots path planning in dynamic environments. In this work, we deal with the problem in a way to enable the use of different robot types in environments with mobile obstacles. Due to the dynamic nature of problem, we used a low coast techniques to enable real-time execution and secure navigation. Finally, this work tries to resolve the problem in real environments. The solution proposed finds viable paths and guides robots along them, a good part of possible collisions are avoided during the course, but some collisions still happen.

KEYWORDS: Path Planning. Multiple Robots. Autonomous. Real-time. Cooperating Robots.

1. Introdução

A robótica vem evoluindo muito nas últimas duas décadas. O que antes somente era utilizado em fábricas, nos dias de hoje está ganhando presença no cotidiano. Isso ocorre, pois a evolução computacional está possibilitando que tarefas antes impossíveis de serem resolvidas com computadores comuns, hoje em dia sejam passíveis de resolução. Com isso, a robótica móvel também vem se tornando cada vez mais comum, tanto em ambientes industriais quanto no dia a dia.

Esse artigo trata de um dos problemas mais primordiais que todo robô móvel deve resolver: planejamento e controle de trajetória. Para que um robô seja capaz de resolver qualquer problema de mais alto nível que requeira movimentação, é necessário que o mesmo consiga definir uma rota e percorrê-la.

O ato de planejar, segundo *LaValle* [6], pode ser definido, de maneira genérica, como encontrar uma sequência de ações que levem o robô de um estado inicial a um estado final. Planejar trajetórias segue a mesma base, porém com algumas modificações que incluem definições geométricas. Como citado em [7], o problema de planejamento de trajetória pode ser definido, de maneira que, dado um espaço de trabalho limitado e que pode possuir áreas não-convexas, é necessário encontrar uma sequência de pontos que conectem duas localizações no espaço. Na **figura 1**, encontra-se uma representação do problema, as localizações que devem ser conectadas estão representadas pelos pontos q_{init} e q_{goal} e o caminho encontrado é representado por uma sequência de pontos, conectados por segmentos de retas. Note que, o caminho evita contato com obstáculos no ambiente, representados por áreas pretas dentro do espaço.

O objeto de estudo desse trabalho busca resolver uma especialização do problema básico de planejamento de trajetória. Foram levadas em consideração a utilização de múltiplos robôs em ambientes com presença de obstáculos móveis, e a possibilidade de execução no mundo real.

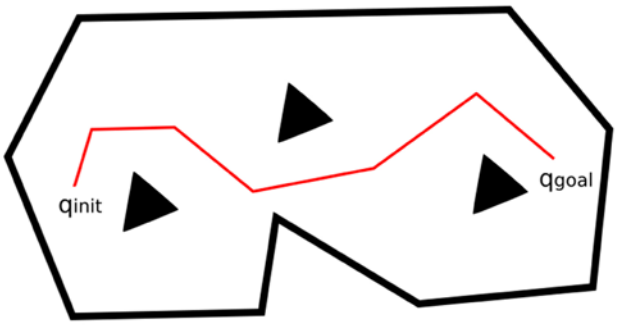

Fig. 1 – Espaço de trabalho limitado com obstáculos fixos. O caminho encontrado é representado por uma sequência de pontos conectados por segmentos de retas.

Foram adicionadas essas premissas devido à abrangência de aplicações que poderiam usufruir dessa solução. A solução descrita aqui também possibilita a utilização de diferentes tipos de robôs, como robôs diferenciais e omnidirecionais. Na **figura 2**, encontra-se dois exemplos de plataformas que foram utilizadas: (a) dois robôs diferenciais da equipe de futebol de robôs (Liga *VSS*), e (b) um robô omnidirecional da equipe de futebol de robôs (Liga *SSL*).

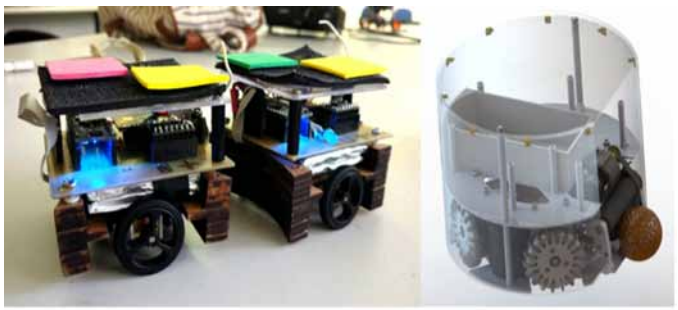

Fig. 2 – Plataformas utilizadas: (a) Liga VSS e (b) Liga SSL.

É necessário atenção em detalhes de cinemática quando se busca utilizar robôs no mundo real. Muitas soluções de planejamento de trajetória buscam encontrar caminhos para um determinado tipo de robô, o que torna a solução muito específica, visto que, diferentes tipos de estruturas de movimentação possuem diferentes restrições e benefícios. Por exemplo, como pode ser visto em [2], robôs omnidirecionais possuem vantagens de manobrabilidade, pois conseguem movimentar-se em qualquer direção, a qualquer momento. Robôs diferenciais possuem menor manobrabilidade, porém, conseguem conservar melhor a energia. Para que uma solução possa ser utilizada por diferentes tipos de robôs, é necessário que sejam levadas em consideração tais características.

1.1 Motivação

Qualquer aplicação que funcione a céu aberto em ambientes reais deve considerar que o ambiente possui incertezas, como obstáculos móveis com movimentações nem sempre previsíveis. Um exemplo de aplicação que está começando a ser utilizada em ambientes como esse, são os carros autônomos. Rodovias são ambientes que possuem muitos obstáculos fixos e móveis, como: sinais, pedestres e carros no ambiente. Um acontecimento que retrata bem a importância de se preocupar com tais incertezas ocorreu recentemente. Em 2018, um carro autônomo se envolveu em um acidente com um pedestre que tentava atravessar fora da faixa [15]. Esse ocorrido representa bem a necessidade de soluções que considerem a existência de variáveis incontroláveis no ambiente.

Outra questão tratada por esse trabalho, é a utilização de múltiplos robôs, que trazem benefícios, como a divisão de trabalho e a possibilidade de resolução de problemas que não são passíveis de serem resolvidos por um único robô. Um exemplo onde múltiplos robôs apresentam um melhor desempenho, que somente um único robô, é a organização dos armazéns de uma empresa de varejo [1]. A empresa vem investindo em robôs para automatizar o processo de localização, busca e embalagem de produtos, o que possibilita que sejam enviados de forma mais eficiente um grande número de produtos. Já em problemas onde existe a necessidade de cooperação, como as categorias de futebol de robôs, a utilização de múltiplos robôs é necessária.

1.2 Organização do trabalho

Esse trabalho está organizado em mais 5 seções. Na seção 2 são apresentados os trabalhos relacionados ao

objeto de estudo desse artigo. Na seção 3 é apresentada a formulação do problema. Na seção 4 é apresentada a metodologia empregada para resolução. Na seção 5 são apresentados os experimentos e os resultados obtidos. Por último, na seção 6 são apresentados a conclusão e trabalhos futuros.

2. Revisão de Literatura

Existem muitos artigos que buscam resolver problemas parecidos com o objeto de estudo desse trabalho. Porém, nesse artigo é buscada a solução para um problema mais complexo, onde os requisitos de utilização de múltiplos robôs em ambientes reais com presença de obstáculos móveis faz com que não seja possível, simplesmente, utilizar os trabalhos relacionados sem adaptação.

2.1 Planejamento de trajetória de múltiplos robôs aplicados a liga SSL

Em [3], é abordada a construção de um time de futebol de robôs para a categoria de futebol de robôs *Small Size League* (*SSL*). Nessa categoria, dois times de seis robôs omnidirecionais se enfrentam. Os robôs possuem capacidade de dominar a bola, tocar, realizar chute baixo e chute alto. Tal problema é conhecido por prover um ambiente de alta complexidade, pois trata de cooperação de múltiplos robôs, execução em tempo real, no ambiente com obstáculos dinâmicos. Nesse trabalho é utilizada uma adaptação do algoritmo de planejamento de trajetória *RRT* (*Rapidly-Exploring Random Trees*) [8]. O algoritmo denominado *BK-BGT* (*Behavioral Kinodynamic Balanced Growth Trees*) une a funcionalidade do *RRT* de encontrar caminhos, decisões de comportamento de estratégias de jogo, e limitações cinemáticas e dinâmicas das plataformas utilizadas.

Diferente do apresentado em [3], nesse trabalho não existe preocupação com estratégias de futebol e existe a necessidade de trabalhar com diferentes tipos de robôs. Devido à utilização de decisões de jogo e restrições da plataforma utilizada serem levadas em conta no planejamento, a solução apresentada nesse trabalho relacionado não é passível de utilização sem adaptação.

2.2 Campos potenciais aplicados à condução de robôs em ambientes dinâmicos

Em [4], é abordado o problema de planejamento de trajetória de um único robô em ambientes altamente dinâmicos. Neste trabalho, existe preocupação em levar um robô de uma localização inicial para uma localização final, evitando colisões.

O ambiente tratado é simulado e possui obstáculos fixos e móveis, com cerca de 300 obstáculos móveis que executam movimentos circulares e lineares. A abordagem do autor busca separar o problema em etapas; inicialmente, é encontrado um caminho que desvie de obstáculos fixos e posteriormente o robô é guiado ao longo do caminho utilizando a técnica de *APF* (Campos Potenciais Artificiais) [9]. O autor não fornece definição clara de qual algoritmo de planejamento de trajetória utiliza e cita os algoritmos *PRM* (*Probabilistc Roadmap*), *EST* (*Expansive-Spaces Trees*) e *RRT* como possibilidades para resolução da primeira etapa.

Apesar de resolver o problema proposto em [4], a mesma abordagem não seria aplicável no objeto de estudo desse artigo, pois o mesmo, não leva em consideração as restrições cinemáticas e dinâmicas dos robôs, além da movimentação dos obstáculos ser previsível.

2.3 Planejamento de trajetória para múltiplos robôs: uma abordagem dupla

Em [5] é abordado o problema de planejamento de trajetória de múltiplos robôs em ambiente simulado, onde existem obstáculos fixos e móveis. A abordagem utilizada define o problema como uma questão de otimização, onde é definido que os caminhos encontrados devem ser suavizados e devem evitar colisões.

A otimização definida, busca calcular todos os caminhos dos robôs durante toda a execução, isto é, são levados em consideração a movimentação de todos os objetos durante todos os passos de simulação. Para que isso seja possível, o trabalho [5] parte da premissa que todos os obstáculos possuem movimentações conhecidas; assim, resolve-se todo o problema em

uma única etapa *offline*.

A premissa de considerar que é possível saber a localização de todos os obstáculos no ambiente durante todo momento, faz com que a solução proposta seja limitada e não atenda aos requisitos propostos neste trabalho.

3. Formulação do problema

Seja um espaço euclidiano W, denominado espaço de trabalho e representado por $\mathbb{R}^2$, que pode possuir qualquer configuração espacial, podendo conter não-convexidades. Seja a pose que um robô pode ocupar em W representada por $p=[xy\,\theta]$. Seja uma trajetória contínua que ligue uma pose inicial p_i e uma pose final p_f representada por uma sequência de poses $P = \{p_1,p_2,...p_n\}$. Seja um robô representado por $u = [\,p\,r\,]^T$ onde r é o raio do círculo que circunscreve o robô no espaço, um obstáculo representado por $o = [\,x\,y\,r\,]^T$ e um estado do espaço de trabalho representado por $E = \{R,O\}$, onde $U = \{u_1,u_2,...u_n\}$ é um conjunto de robôs controláveis e $O = \{o_1,o_2,...o_n\}$ é um conjunto de obstáculos. Assumindo que a forma de U e O são conhecidas e que O possui obstáculos fixos e móveis. O problema pode ser definido como:

Para cada $u_n \in U$, encontrar uma trajetória P_n, que ligue à pose atual P_a a pose final p_f, de maneira a evitar contato com O e com os demais robôs em U.

Na **figura 3**, encontra-se um exemplo do ambiente onde o problema é tratado. Na imagem: (i) os limites do espaço de trabalho são representados pelos segmentos de reta aos extremos, (ii) obstáculos fixos são representados por quadrados pretos, (iii) obstáculos dinâmicos são representados por círculos em cinza e (iv) os robôs são representados pelos quadrados nas cores vermelho, verde e azul (com suas respectivas áreas de objetivo e caminhos da mesma cor).

A existência de obstáculos com movimentação imprevisível acrescenta dificuldade à resolução do problema, além de impossibilitar o uso de algoritmos que resolvam o problema de forma totalmente *offline*, isto é, calculem todos os passos de solução antes de iniciar a resolução.

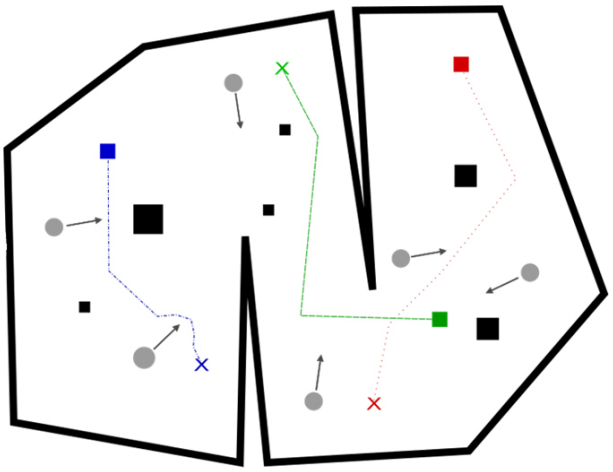

Fig. 3 – Exemplo de ambiente onde o problema é tratado. A área de atuação é limitada e existem obstáculos fixos e móveis.

Também existe o problema do crescimento computacional. Por exemplo, o algoritmo clássico de planejamento de trajetória *RRT* tem sua complexidade aumentada, o que originalmente possuiria complexidade da ordem de $O\,(K^2 + KD)$, onde K é o número máximo de interações para resolver o problema e D é a complexidade do algoritmo de detecção de colisões. Lidando com múltiplos robôs, a complexidade cresce para $O\,(R\,(K^2 + KD))$ e lidando com a existência de obstáculos móveis, a complexidade de D também cresce. O que limita mais rapidamente a quantidade de robôs que podem ser utilizados, pois é mais fácil extrapolar a quantidade de computação que um computador pode realizar. No caso das Ligas *VSS* e *SSL*, a quantidade de robôs de um time é limitada a 3 e 6 respectivamente, assim o custo computacional para encontrar caminhos nessas categorias é menor.

4. Metodologia

A abordagem utilizada nesse trabalho busca dividir a resolução em duas etapas: uma de processamento *offline* e outra de processamento *online*. Na etapa de processamento *offline* é realizado o planejamento de trajetória dos robôs e na etapa de processamento *online* é realizado o controle de trajetória dos robôs ao longo dos trajetos.

4.1 Etapa de processamento offline

A tarefa a ser realizada nessa etapa pode ser definida como: para cada $u_n \in U$, encontrar uma trajetória P_n que ligue a pose atual do robô p_a e a pose final p_f, de maneira a evitar contato com os obstáculos fixos de O. O algoritmo utilizado foi o *RRT*, pois o mesmo é conhecido por encontrar caminhos nos mais diversos ambientes. Para a implementação foi utilizado a biblioteca de planejamento de trajetória *OMPL* (*Open Motion Planning Library*) [14]. As trajetórias calculadas nessa etapa não buscam evitar colisões com obstáculos móveis e nem entre robôs. A motivação desta decisão é evitar o aumento da complexidade e a presença de obstáculos móveis.

4.2 Etapa de processamento *online*

Essa etapa é iniciada após todos os robôs em U possuírem caminhos definidos a serem percorridos. O processamento nessa etapa é realizado de forma retroativa, ou seja, durante toda a tarefa de percorrer os caminhos. Para que os robôs consigam seguir as trajetórias, é utilizada a técnica *APF*, como apresentada em [9]. A técnica se baseia nos campos potenciais, onde na presença de objetos com cargas elétricas positivas e negativas, é criado um campo de força, onde os objetos de cargas opostas se atraem e objetos de mesma carga se afastam.

Na técnica é definido que robôs e obstáculos possuem a mesma carga e poses objetivo possuem carga inversa. Dessa forma, os robôs evitam contato entre si e com obstáculos, ao mesmo tempo que se direcionam para as poses objetivo. Na **figura 4**, encontra-se um exemplo de funcionamento da técnica. Um robô é representado por um círculo azul, um obstáculo é representado por um círculo cinza e o caminho calculado na etapa *offline* é representando por retas azuis conectadas, que partem do robô para a pose final representada por um **X**. A força de repulsão gerada pela proximidade do obstáculo é representada por um círculo vermelho e a força atrativa de uma pose é representada por um círculo verde. Na **figura 4**, a força resultante $\vec{F}$ que representa a tendência de movimento é gerada pela soma das forças repulsivas $\vec{Fr}$ e força atrativa $\vec{Fa}$. Note que, quanto mais próximo de um obstáculo mais forte é a força repulsiva, e quanto mais próximo a uma pose objetivo menor é a força de atração; dessa forma os robôs tendem a evitar se aproximar de obstáculos e se aproximar de forma segura de poses objetivo.

Como cada robô do conjunto U possui uma trajetória $P = \{p_1 p_2 ... p_n\}$ que parte de sua pose inicial p_1 para sua pose final p_n, foi criado um algoritmo que altera as poses objetivo de forma a guiar os robôs pelo caminho. Como os robôs possuem poses iniciais p_1 ; inicialmente suas poses objetivos são p_2, dessa forma, todo calculo é feito de modo a levar os robôs de suas poses iniciais para seus respectivos p_2. Após se aproximarem de p_2, a pose objetivo é incrementada para p_3. O algoritmo segue dessa forma até que o robô alcance p_n. Na **figura 5**,

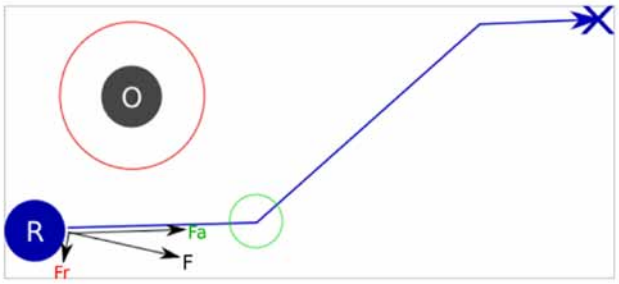

Fig. 4 – Exemplo de funcionado da técnica APF. Na imagem, o vetor Fr representa a força repulsiva gerada por um obstáculo, Fa representa a força atrativa para uma pose e F representa a força resultante, que dita a tendência de movimento.

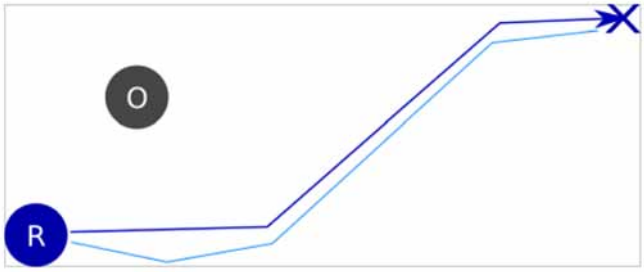

Fig. 5 – Exemplo de trajetória ao seguir um caminho utilizando a técnica APF.

A técnica *APF* nesse caso é completamente geométrica; ou seja, não é levada em consideração a maneira como os robôs vão seguir a força resultante. Para isso é utilizado um modelo cinemático dos robôs.

4.3 Controle de velocidades

Essa etapa ocorre concomitante ao processamento *online*. A cada iteração após possuir a força/torque resultante de cada robô, é necessário converter essa informação para velocidade das rodas. Para isso, é utilizado um modelo cinemático de movimentação daquele tipo de robô. Cada plataforma possui modelos diferentes, os robôs da Liga *VSS* são diferenciais e os robôs da Liga *SSL* são omnidirecionais. Para alternar entre diferentes tipos de robôs, basta utilizar diferentes modelos.

Para o controle de robôs diferenciais, foi utilizado o modelo apresentado em [11]. Na **figura 6**, encontra-se a vista superior de uma plataforma diferencial, onde w representa uma velocidade angular, v uma velocidade linear, B é o comprimento do eixo do robô e θ é a orientação do robô.

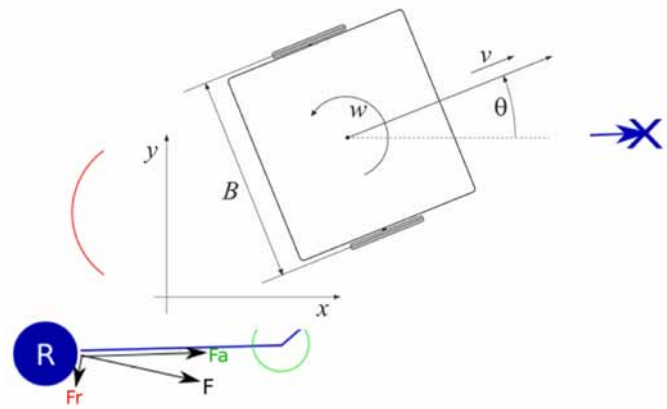

Fig. 6 – Vista superior de um robô diferencial com parâmetros utilizados pelo modelo cinemático.

Para controlar um robô com essa configuração, existe a necessidade de encontrar v e w, que façam com que o robô se direcione como a força resultante $\vec{F}$ indica. Com esse propósito é utilizada a **equação 1**, onde $\dot{p}$ está relacionado com $\vec{F}$.

$$\dot{p}=\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix}=\begin{bmatrix}\cos(\theta) & 0\\ sen(\theta) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}v\\ w\end{bmatrix} \quad (1)$$

Os valores de v e w são utilizados nas **equações 2 e 3** para obter as velocidades das rodas direita v_R e esquerda v_L.

$$v_R = v + \frac{w_B}{2} \quad (2)$$

$$v_L = v - \frac{w_B}{2} \quad (3)$$

O modelo cinemático é uma aproximação do modelo físico; com isso, os valores obtidos estão sujeitos a erros e variações. Para corrigir esse problema é utilizada a técnica de controle *PID* (*Proportional, Integral and Derivative*) [10], que foi calibrado a partir de experimentações.

5. Experimentos

Foram realizados experimentos em três plataformas simuladas e em uma plataforma em ambiente real. Inicialmente foram feitas simulações em uma plataforma criada, a fim de validar a viabilidade computacional do algoritmo. Em um segundo momento, a solução foi adaptada para se comunicar com o simulador *grSim* [12] e foram realizadas simulações em um ambiente que considerava as restrições cinemáticas de robôs omnidirecionais. Por fim, foram realizados experimentos no simulador da plataforma *VSS-SDK* [13] e em ambientes reais, ambos com robôs diferenciais.

5.1 Simulações sem restrições cinemáticas

Na primeira fase da pesquisa, foi criado um simulador de planejamento de controle de trajetórias 2D. As simulações no simulador não consideravam nenhum tipo de restrições cinemáticas. O principal intuito era validar o crescimento computacional e as chances de resolução de caminhos. Para possibilitar a utilização de múltiplos robôs, a solução deveria ser leve. Na **figura 7**, encontra-se um exemplo do ambiente de testes criado.

Na **figura 7** são representados os obstáculos móveis como círculos pretos, os caminhos calculados como sequências de pontos ligados por segmentos de retas,

e a trajetória dos robôs ao seguir esses caminhos são desenhados por cima com cores mais vivas. Note que a trajetória representada pela cor vermelha tem o desenho de outra trajetória vermelha mais clara por cima.

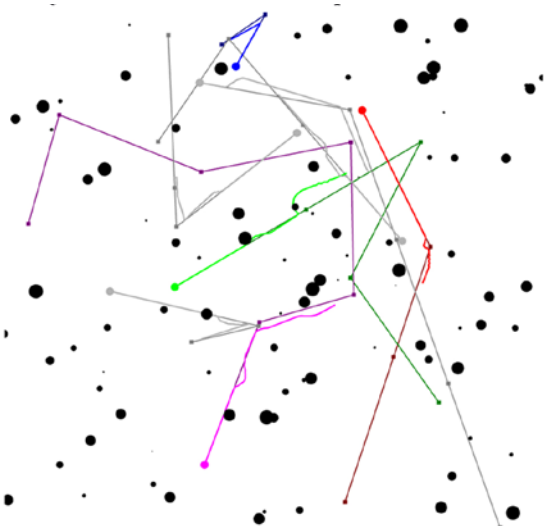

Fig. 7 – Ambiente de testes inicial. Obstáculos móveis representados por círculos pretos e caminhos representados por sequência de pontos ligados por retas.

A primeira representa o caminho criado na etapa *offline* e a segunda representa a trajetória de um robô sendo controlado ao longo do caminho na etapa *online*.

Foram realizados 5000 experimentos com esse simulador. As simulações foram separadas em 5 casos de acordo com a quantidade de robôs, que variavam de 5 a 10. Para cada caso foram criadas 1000 situações, variando as posições iniciais e finais dos robôs, a quantidade de obstáculos, de 1 a 99 por simulação, e o diâmetro desses obstáculos. O espaço de trabalho possuía 1000x1000pixels e os obstáculos possuem movimentação aleatória.

A solução utilizando o algoritmo *RRT* para processamento *offline* e *APF* para controle de trajetória obteve os resultados presentes na **tabela 1**. Na tabela, os casos foram separados de acordo com o número de robôs. Os dados computados levam em consideração todos os estados das simulações, desde o início do planejamento *offline* de todos os robôs até o tempo em que o último robô leva para chegar a seu ponto objetivo final. São comparadas também a taxa média de sucesso de resolução de caminhos e a taxa de colisões.

Tab. 1– Resultados obtidos no ambiente de simulação sem restrições cinemáticas.

Quantidade de Robôs	Tempo de Execução	Porcentagem de Sucesso	Taxa de Colisões
5	0.096s	96.4%	0.30
6	0.121s	97.6%	0.46
7	0.165s	97.6%	0.62
8	0.188s	95.6%	0.76
9	0.221s	97.2%	0.87
10	0.265s	94.1%	1.00

A abordagem não reduz totalmente o número de colisões; porém, essas colisões ocorrem de forma esporádica. Analisando a **tabela 1**, no caso de controle de 10 robôs, ocorreu em média 1 colisão no controle de todos os robôs ao percorrerem os caminhos. Por fim, a solução utilizada em média conseguiu resolver 96.88% dos experimentos realizados, tendo sido executada de forma rápida.

5.2 Simulações no grSim

Para validar o funcionamento da resolução com robôs que possuem limitações quanto à movimentação, foi utilizado o simulador de futebol de robôs da categoria *Small Size League* (*SSL*), o *grSim* [12]. Na **figura 8**, encontra-se o simulador.

Fig. 8 – Simulador grSim. No simulador, existem dois times de seis robôs com cores principais azul e amarelo.

Foram realizadas 100 simulações utilizando o grSim. Em cada experimento, os obstáculos se moviam de

forma aleatória e os robôs possuíam poses iniciais e poses finais diferentes. Devido a limitações do ambiente, os experimentos somente possuíam 6 robôs e 6 obstáculos móveis. Na **tabela 2**, encontram-se os resultados obtidos. Em 98% dos casos, todos os caminhos foram resolvidos e, em média, ocorriam 0.52 colisão na resolução dos 6 caminhos.

Tab. 2 – Resultados obtidos nas simulações no grSim.

Quantidade de Robôs	Porcentagem de Sucesso	Taxa de Colisões
6	98.0%	0.52

Comparando a porcentagem de caminhos resolvidos nos experimentos no grSim com 6 robôs (98%) e a porcentagem obtida nas simulações, também com 6 robôs, na plataforma onde os robôs não possuíam restrições de cinemática (97.6%), fica claro que o algoritmo *RRT* mantém uma boa taxa de resolução de caminhos. A taxa de colisões ocorridas levemente maior no grSim (0.52) em comparação à plataforma simplificada (0.46), demonstra um comportamento razoável, visto que os robôs no grSim são modelados de forma a simular restrições dinâmicas de plataformas reais; isto é: os robôs possuem velocidade e aceleração máximas bem definidas, e frenagem e capacidade de curva em altas velocidades limitadas.

5.3 Simulações no VSS-Simulator

Após a validação utilizando robôs omnidirecionais, buscou-se validar com robôs com movimentações diferentes e com mais limitações de movimento. Foi utilizada a plataforma *VSS-Simulator* [13] da categoria de futebol de robôs *VSS*, para trabalhar com robôs diferenciais. Na **figura 9** encontra-se o simulador, o mesmo possui um ambiente não-convexo e limitado com 2 times de 3 robôs diferenciais cada. Foi realizada uma adaptação no mesmo para possibilitar o desenho de caminhos. Na imagem, os caminhos possuem a mesma cor das cores secundárias de cada robô, por exemplo, o robô verde possui o caminho marcado com a cor verde.

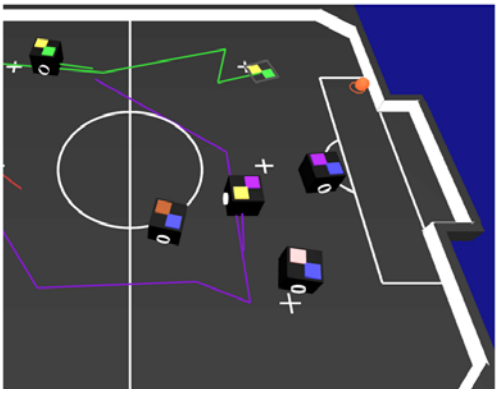

Fig. 9 – Plataforma VSS-Simulator. Existem 2 times de robôs diferenciais, para cada robô controlado são desenhados caminhos.

Foram realizados 40 experimentos utilizando o *VSS-Simulator*. Os experimentos foram divididos em duas situações: na primeira os robôs adversários não se moviam e na segunda os robôs se moviam de forma aleatória. Cada experimento tinha duração de 1 minuto de execução. Inicialmente foram geradas poses finais aleatórias para que os robôs seguissem; após isso, eram calculados os caminhos utilizando *RRT* e os robôs começavam a percorrer os caminhos utilizando *APF*. Após um robô chegar a sua pose final, era calculado uma nova pose para o mesmo.

Com a finalidade de validação da diminuição de colisões, foi criado uma simplificação da abordagem que utiliza somente o campo atrativo. Na **tabela 3**, encontram-se os resultados obtidos nas simulações em ambientes estáticos.

Tab. 3 – Resultados em ambientes estáticos simulados.

	Resol.	C. Robôs	C. Obstác.
RRT com APF	56.2	1.7	0.6
Somente Campo Atrativo	50.2	27.7	23.8

Comparando os resultados da abordagem do *RRT* com *APF* em relação à solução somente com campo atrativo, fica claro que existe benefício em calcular caminhos defensivos e segui-los de forma defensiva. Além de reduzir em 93.8% o número de colisões entre robôs e em 97.4% o número de colisões com obstáculos, também existem ganhos de 11.9% de caminhos resolvidos.

Os ganhos ficam cada vez mais evidentes quando se busca trabalhar em ambientes dinâmicos. Na **tabela 4** encontram-se os resultados obtidos em ambientes dinâmicos.

Tab. 4 – Resultados em ambientes dinâmicos simulados.

	Resol.	C. Robôs	C. Obstác.
RRT com APF	57.3	2.8	4.9
Somente Campo Atrativo	30	39.6	44.6

Enquanto a resolução de caminhos de nossa abordagem aumenta em média 1.9% em ambientes dinâmicos, a resolução sem foco em segurança cai em média 40.2%. Em relação à quantidade de colisões, a solução sem foco em segurança sofre em média 1.414% mais colisões entre robôs e 910.2% mais colisões com obstáculos no ambiente.

5.4 Experimentos em ambiente real

Os últimos experimentos foram realizados em ambiente real e serviam para validação em plataformas robóticas reais. Para reduzir a complexidade desse tipo de experimento, foi utilizada a plataforma de futebol de robôs *IEEE Very Small Size* (*VSS*) do Laboratório de Sistemas Inteligentes e Robótica (SIRLab) da Faculdade de Educação Tecnólogica do Estado do Rio de Janeiro (FAETERJ), campus Petrópolis.

Foi utilizada a plataforma *opensource VSS-SDK* para realizar os experimentos em ambiente real; pois além de estar integrado com o *VSS-Simulator,* o mesmo provê um sistema de visão computacional, que foi utilizado para obter a localização dos robôs e obstáculos no campo. Na **figura 10**, encontra-se o ambiente de experimento. O campo possui 130x150cm, os gols possuem 10 cm de profundidade, os robôs possuem raio máximo de 8 cm e os obstáculos são representados por padrões de cores azuis na base do campo.

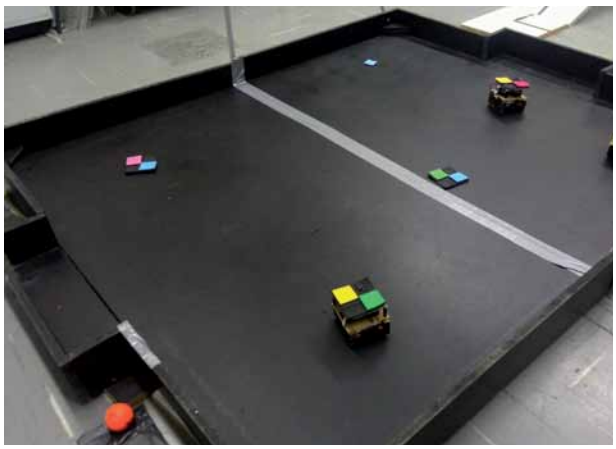

Fig. 10 – Campo da categoria de futebol de robôs VSS, com robôs e obstáculos.

Foram realizados 10 experimentos de 1 minuto cada, seguindo os padrões dos experimentos em ambiente simulado. Na **tabela 5** encontram-se os resultados obtidos.

Devido ao limite de apenas 3 robôs da plataforma *VSS*, foram realizados experimentos somente com obstáculos estáticos. Os resultados obtidos foram semelhantes aos

Tab. 5 – Resultados em ambientes estáticos reais.

	Resol.	C. Robôs	C. Obstác.
RRT com APF	55.5	3.5	2.7

resultados em ambiente simulado. Houve uma queda de resolução de apenas 1.2% em média, e ocorrem em média uma colisão entre robôs a cada 15.8 caminhos resolvidos e uma colisão com obstáculos a cada 20.5 caminhos resolvidos.

6. Conclusões e trabalhos futuros

A abordagem utilizada para resolução do problema de planejamento e controle de trajetória de múltiplos robôs em ambientes dinâmicos, demonstrou-se funcional e possibilitou a utilização de diferentes tipos de robôs. A adaptação da abordagem para tratar de robôs omnidirecionais e diferenciais, demonstrou-se simples, bastando a troca do modelo cinemático dos mesmos.

Além de tornar o trajeto dos robôs mais seguro, a abordagem provê um pequeno ganho na resolução do problema, resolvendo caminhos de forma mais rápida. A separação do problema em uma etapa *offline* e uma etapa *online* possibilita que caminhos sejam traçados de forma segura em relação a obstáculos fixos e que o custo computacional seja baixo, possibilitando que computadores comuns consigam controlar um conjunto de robôs.

Contudo, a abordagem pode ser melhorada. Mesmo com todas as medidas para evitar colisões elas ainda ocorrem, pois os obstáculos dinâmicos com movimentação aleatória às vezes forçam a ocorrência de colisões. Para projetos futuros, o refinamento do modelo cinemático deve ser

realizado, extendendo-o para considerar as limitações dinâmicas das plataformas. Também pode ser considerada a utilização de modelos para prever a movimentação dos obstáculos.

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (CAPES) – Código de Financiamento 001.

Referências Bibliográficas

[1] As Amazon Pushes Forward With Robots, Workers Find New Roles. https://www.nytimes.com/2017/09/10/technology/amazon-robots-workers.html
[2] Nourbaskhsh, I. and Siegwart, R. (2004). Introduction to autonomous mobile robots, The MIT Press, Cambridge, Massachusetts, England.
[3] Rodrigues, S. H. (2013). Sistemas Autônomos e Inteligentes para Robôs Cooperativos no Ambiente Small Size League. Dissertação de Mestrado, Instituto Militar de Engenharia, Rio de Janeiro.
[4] Chiang, H.-T., Malone, N., Lesser, K., Oishi, M, and Tapia, L. (2015). Path-guided artificial potential fields with stochastic reachable sets for motion planning in highly dynamic environments, Robotics and Automation (ICRA), 2015 IEEE International Conference on, IEEE, pp. 2347-2354.
[5] Motee, N., Jadbabaie, A. and Pappas, C. (2010). Path planning for multiple robots: An alternative duality approach, American Control Conference (ACC), 2010, IEEE, pp. 1611-1616.
[6] LaValle, S. M. (2006). Path Planning Algorithms. Cambridge University press, 2006.
[7] Latombe, J. -C. (2012). Robot motion planning, Vol.124, Springer Science & Business Media.
[8] LaValle, S. M. (1998). Citeseer. Rapidly-exploring random trees a new tool for path planning.
[9] Goodrich, M. A. (2002). Citeseer . Potential fields tutorial.
[10] Ogata, K. (2003). Modern control engineering, Prentice Hall PRT.
[11] Blazic, S. (2014). On periodic control laws for mobile robots. IEEE Transactions on Industrial Eletronics.
[12] Monajjemi, V. Koochakzadeh, A. Ghidary, S. S. (2011) SpringerLink. grSim – RoboCup Smal Size Robot Soccer Simulator.
[13] VSS-SDK (2016). Um projeto opensource que auxília novas equipes a ingressagem na competição de futebol de robôs IEEE Very Small Size [Soccer]. 05 out. de 2016. https://github.com/SIRLab/VSS-SDK
[14] Sucan, I. A, Moll, M. Kavraki, L. E. (2012). The Open Motion Planning Library. IEEE Robotics & Automation Magazine.
[15] Self-driving Uber kills Arizona woman in first falta crash involving pedestrian. https://www.theguardian.com/technology/2018/mar/19/uber-self-driving-car-kills-woman-arizona-tempe.