UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE AUTOMAÇÃO E SISTEMAS

Filipe Lopes de Barros Correia

CONTROLE DE FORMAÇÃO COM SEGUIMENTO DE REFERÊNCIA PARA GRUPOS DE VEÍCULOS AUTÔNOMOS UTILIZANDO CONSENSO E RHC

Florianópolis – SC
2014

Filipe Lopes de Barros Correia

CONTROLE DE FORMAÇÃO COM SEGUIMENTO DE REFERÊNCIA PARA GRUPOS DE VEÍCULOS AUTÔNOMOS UTILIZANDO CONSENSO E RHC

Dissertação submetida ao Programa de Pós-Graduação em Engenharia de Automação e Sistemas para obtenção do grau de "Mestre em Engenharia de Automação e Sistemas".

Orientador: Prof. Dr. Ubirajara Franco Moreno, PPGEAS – UFSC.

Florianópolis – SC
2014

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Correia, Filipe Lopes de Barros
Controle de Formação com Seguimento de Referência para Grupos de Veículos Autônomos Utilizando Consenso e RHC / Filipe Lopes de Barros Correia ; orientador, Ubirajara Franco Moreno - Florianópolis, SC, 2014.
79 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia de Automação e Sistemas.

Inclui referências

1. Engenharia de Automação e Sistemas. 2. Veículos autônomos. 3. Controle cooperativo. 4. Controle de formação. 5. Teoria de consenso. I. Moreno, Ubirajara Franco. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Automação e Sistemas. III. Título.

Filipe Lopes de Barros Correia

CONTROLE DE FORMAÇÃO COM SEGUIMENTO DE REFERÊNCIA PARA GRUPOS DE VEÍCULOS AUTÔNOMOS UTILIZANDO CONSENSO E RHC

Esta dissertação foi julgada aprovada para a obtenção do grau de "Mestre em Engenharia de Automação e Sistemas" e aceita em sua forma final pelo Programa de Pós-Graduação em Engenharia de Automação e Sistemas.

Florianópolis – SC, 5 de Setembro de 2014.

Prof. Dr. Rômulo Silva de Oliveira
Coordenador do Programa de Pós-Graduação em Engenharia de Automação e Sistemas

---

Prof. Dr. Ubirajara Franco Moreno
PPGEAS – UFSC
Orientador

Banca Examinadora:

Prof. Dr. Ubirajara Franco Moreno
PPGEAS – UFSC
Presidente

Prof. Dr. Marcelo R. Petry
UFSC

Prof. Dr. Daniel Ferreira Coutinho
PPGEAS – UFSC

Prof. Dr. Eugênio de Bona Castelan Neto
PPGEAS – UFSC

Agradecimentos

Agradeço:

a Deus, pela vida e pelas oportunidades;

aos meus pais, Carlos e Luzia, pelo exemplo e apoio;

ao Prof. Ubirajara, pela orientação no desenvolvimento do trabalho;

à UFSC e à Capes, pelos recursos sem os quais o trabalhado não poderia ter sido feito;

por fim, a todos que contribuíram positivamente para a realização deste trabalho.

Resumo

O movimento em formação e o seguimento de uma trajetória de referência são alguns dos problemas que tipicamente compõem uma tarefa para um grupo de veículos autônomos que cooperam entre si. Neste trabalho, ambos os problemas são abordados e uma solução é proposta utilizando-se RHC (*Receding Horizon Control*) e o conceito de estruturas virtuais de forma descentralizada. O método proposto consiste em um problema de otimização quadrática que realiza o consenso a respeito do centro da formação e gera uma trajetória para que o robô que a executa alcance sua posição desejada. Para o controle de veículos com restrições não-holonômicas, um controle de baixo nível é utilizado para fazer o robô seguir a trajetória gerada. Para a avaliação e teste do método proposto, um simulador, capaz de interagir com robôs reais através do ROS (*Robot Operating System*), foi desenvolvido e dois conjuntos de testes foram feitos. O primeiro consistiu de simulações com diferentes valores para os parâmetros da função objetivo, de forma a avaliar a eficácia do método e sua dependência nos parâmetros. O segundo consistiu de testes de realidade aumentada, em que veículos reais e virtuais integravam o grupo. Os resultados são analisados e comprovam a eficácia do método, mas ainda são necessários estudos sobre os critérios de convergência e estabilidade.

Palavras-chave: Veículos autônomos, controle cooperativo, controle de formação, teoria de consenso.

Abstract

Moving in formation and tracking a reference trajectory are common problems when composing tasks for a group of cooperating autonomous vehicles. In this work, both problems are treated and a solution is proposed using RHC (Receding Horizon Control) and the concept of virtual structures in a decentralized manner. The proposed method consists of a quadratic optimization problem which is responsible for the consensus about the position of the formation center and the trajectory generation so that the robot can get to its desired position. For the control of non-holonomic vehicles, a low-level controller scheme is used to make the robot follow the generated trajectory. For the evaluation of the proposed method, a simulator capable of interaction with real robots through the use of ROS (Robot Operating System) was developed and two sets of trials were done. The first consisted of simulations with differing values of the objective function's parameters, so as to assess the effectiveness of the method and its dependency on the parameters' values. the second set consisted of tests with augmented reality, in which real and virtual robots integrated the group. The results are analised and show the effectiveness of the method, but the convergence and stability criteria need more study.

Keywords: Autonomous vehicles, cooperative control, formation control, consensus theory.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

Lista de Símbolos

Sumário

Capítulo 1

Introdução

O desenvolvimento de tecnologias de robótica, automação e instrumentação, com destaque para a miniaturização de sistemas eletromecânicos, tem tornado possível a utilização de veículos autônomos, também chamados aqui de robôs, para a realização de tarefas cada vez mais complexas. Muitas dessas atividades podem ser realizadas mais eficientemente por grupos de robôs pequenos que cooperam entre si, em contraste com a utilização de um único grande robô. Como exemplo, podemos destacar: *monitoramento, vigilância ou exploração* de uma região, *busca e resgate* de pessoas perdidas, e *aplicações militares*. A Figura 1.1 mostra esquematicamente uma tarefa de busca e resgate, em que um grupo de veículos, inicialmente dispersos, assume uma formação que otimiza a região de sensoreamento e percorre uma região em busca de uma pessoa perdida, por exemplo, um náufrago.

Nesses cenários, observa-se a necessidade dos robôs concordarem sobre determinadas informações, ou seja, essas informações, em suas instanciações nos vários veículos, devem ser coerentes entre si para que possa haver articulação eficaz das ações individuais.

Espera-se que os veículos integrantes de um grupo sejam autônomos mas que ajam de maneira articulada para atingir o objetivo comum. Para isso, pode-se empregar estratégias de cooperação e coordenção, área de pesquisa cuja atividade vem aumentando nas duas últimas décadas:

- *cooperação* – quando seus integrantes possuem um objetivo comum;
- *coordenação* – quando seus integrantes interagem entre si para melhorar a tomada de decisão.

Revisões dos principais temas e trabalhos sobre cooperação e coordenação em grupos de veículos são feitas por Murray; Veres et al.; Lucasa

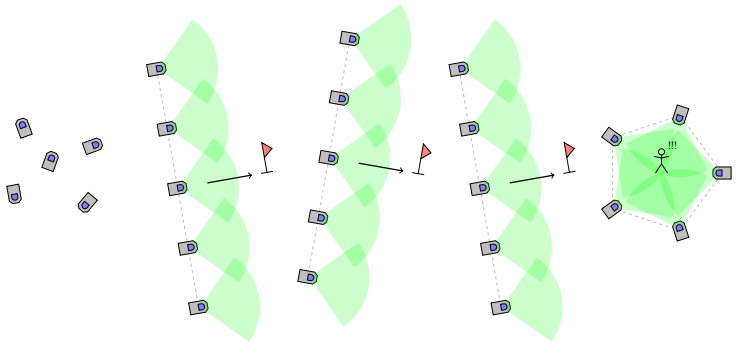


Figura 1.1: Exemplo de cooperação e coordenação em uma tarefa de busca e resgate.

et al.; Cao et al.[1, 2, 3, 4].

Em várias situações, pode não haver possibilidade de comunicação com uma central de gerência ou controle do grupo e então o problema deve ser resolvido localmente pelo robô. Assim, as soluções desenvolvidas podem ser classificadas como *centralizadas* ou *descentralizadas*. Na primeira, os sinais de controle são calculados por uma central de controle, ou "nave mãe", que o comunica a cada veículo individual. Na segunda, cada veículo calcula seu próprio sinal de controle baseado na informação, completa ou incompleta, sobre a situação do grupo e do objetivo que possui. Em ambos os casos a quantidade de comunicação entre os robôs móveis geralmente deve ser minimizada devido a necessidade de economia energética.

Tanto o fluxo como a troca de informação são fatores importantes para o sucesso na coordenação entre os integrantes do grupo. Os dispositivos de comunicação instalados nos veículos naturalmente têm limitações no alcance da transmissão, o que obriga os veículos a manterem uma distância mínima para que possa ocorrer comunicação entre os mesmos. Outro fator importante relacionado à partilha de informação é a possibilidade de falhas na comunicação, que podem acontecer por consequência da alta demanda de dados no canal, por faltas pontuais e passageiras devido à presença de obstáculos no ambiente ou em razão de falhas temporárias nos próprios dispositivos de comunicação.

Figura 1.2: Estratégias usadas em coordenação de grupos de robôs para controle de formação. Adaptado de [5].

1.1. Controle de Formação

O movimento em formação, no qual há controle das posições relativas dos robôs no grupo, e o seguimento de uma trajetória de referência são alguns dos problemas que tipicamente compõem as tarefas para um grupo de veículos autônomos cooperantes. Por exemplo, a cobertura de uma região durante uma atividade de monitoramento dependerá, entre outros fatores, da posição relativa entre os veículos enquanto se movimentam nessa região segundo uma determinada trajetória. Na Figura 1.2 são destacadas as principais abordagens utilizadas para solução desses problemas. Em geral, essas abordagens são usadas em conjunto para obtenção de um resultado final mais robusto. Uma ampla revisão da literatura sobre o controle de formação com seguimento de trajetória para grupos de veículos é feita por Zhang e Mehrjerdi[5]. Outra revisão, em que o foco é veículos aéreos é feita por Scharf et al.[6]. Uma revisão mais geral, sobre movimento de grupos de veículos, é feita por Navarro e Matía[7].

Na estratégia *siga o líder*, um robô é definido como líder e os outros como seguidores. Os seguidores recebem informações do líder e seu comportamento, sua posição por exemplo, é definido em relação a ele. Uma desvantagem desse método é que todo o sistema falha se o líder falhar. A abordagem por *estrutura virtual*, desenvolvida por Lewis e Tan[8], é normalmente utilizada no problema de controle de forma-

ção quando se deseja uma formação rígida e predeterminada. Nela, a formação é descrita como uma lista de posições relativas a um ponto, o centro da formação, que pode ser considerado um líder virtual. Nas técnicas que utilizam *consenso*, as informações possuídas pelos integrantes do grupo são combinadas de acordo com algum algorítmo de forma a se obter coerência entre as informações que cada veículo possui do problema. Na abordagem utilizando *controle inteligente*, o comportamento de cada veículo é modelado para diversas situações possíveis e o seu sinal de controle resulta de uma combinação desses possíveis comportamentos. A técnica de *campos potenciais* é utilizada geralmente no problema de agrupamento em bando, em que os indivíduos são muito parecidos e devem simplesmente manter-se agrupados e com uma mesma velocidade, ou quando se deseja uma formação flexível e não predeterminada.

Os métodos desenvolvidos com base nessas estratégias têm seu embasamento principalmente na teoria de grafos, na teoria de Lyapunov para sistemas dinâmicos e nas técnicas de otimização.

1.2. Abordagem

Para se obter uma solução mais robusta a falhas de comunicação, em que os veículos teriam maior autonomia, uma solução descentralizada e que considere que cada veículo tem conhecimento limitado da situação atual apresenta vantagem. Nesse contexto, a abordagem utilizando consenso vem a ser bastante útil para garantir a coerência das informações entre os robôs.

Em um típico problema de consenso, tem-se vários indivíduos que se comunicam trocando informações e processando-as de acordo com algum algoritmo. Ou seja, ao longo das iterações, os veículos recebem informações de outros e executam um algoritmo para atualizar a sua, de forma que ela se torne coerente com a dos outros. Essa comunicação pode ser modelada através de um grafo direcionado em que os nós representam indivíduos e as arestas representam canais de comunicação.

Uma introdução sobre o problema de consenso é feita por Ren et al.[9], enquanto revisões sobre o tema podem ser lidas em Ren et al.; Olfati-Saber et al.[10, 11]. As condições de convergência para vários casos são analizadas em Ren e Beard[12].

Ordoñez et al.[13] desenvolveram uma estratégia baseada em consenso utilizando RHC (*Receding Horizon Control*) para a solução do problema de *rendez-vous*, no qual os robôs devem concordar sobre um ponto de encontro e alcançá-lo ao mesmo tempo, considerando que

o conhecimento que cada um deles possui é limitado, assim como a possibilidade de perda de comunicação, que também estaria sujeita a falhas. O problema de *rendez-vous* também é tratado por Ferrari-Trecate et al.[14], que considera dinâmicas veículares de primeira e segunda ordem, e por Gowal e Martinoli[15], que considera veículos com dinâmicas não-holonômicas.

A técnica de controle chamada RHC consiste em se obter o valor da variável de controle a partir da otimização de uma função de custo na variável controlada e na de controle, tendo em vista um modelo do sistema e a previsão de seu comportamento para um determinado número de iterações futuras, o *horizonte de predição*. A cada iteração, essa otimização é realizada e uma lista de entradas de controle futuras, das quais apenas a primeira é de fato aplicada, é obtida. Uma introdução aos aspectos teóricos e práticos das técnicas de RHC mais comuns pode ser encontrada no livro de Camacho e Alba[16]. Revisões a respeito de RHC aplicado a sistemas distribuídos são feitas por Camponogara et al.; Scattolini; Christofides et al.[17, 18, 19].

O conceito de estruturas virtuais foi utilizado por Dunbar e Murray[20] e por Keviczky et al.[21], que elaboraram métodos também baseados em RHC para fazer um grupo de robôs seguirem uma trajetória enquanto em formação de forma descentralizada. No primeiro, os veículos comunicam suas trajetórias ótimas a todos os outros, mas apenas alguns têm acesso à trajetória de referência, e a otimização é feita na forma de uma programação não-linear. Já no segundo, cada robô realiza a otimização, na forma de uma programação mista linear-inteira, para si e para seus vizinhos conhecidos; o controlador é dividido em um alto-nível (o RHC) e um baixo-nível (controle de velocidade e posição do veículo) para lidar com o problema da não-holonomia da dinâmica veícular; e a topologia de comunicação é considerada fixa e predeterminada.

Apesar de a ideia de estrutura virtual ter sido originalmente elaborada para um controle central, Ren e Sorensen[22] desenvolveram um método que consiste em descentralizá-la, de forma que cada robô tenha uma instância dela, ou seja, da posição do centro da formação, e usar um algoritmo de consenso para que os robôs concordem sobre sua localização. Nesse trabalho, o método é aplicado a um grupo de veículos para seguimento de trajetória enquanto em formação, numa situação em que apenas alguns veículos recebem informação da referência. O método pode ser aplicado a topologias de comunicação fixas ou variáveis.

Assim, enquanto o método de Ordoñez et al.[13] foi aplicado

ao problema de *rendez-vous*, percebe-se que ele poderia ser utilizado também para o controle de formação utilizando-se da ideia de estrutura virtual.

1.3. Objetivos

O objetivo deste trabalho de mestrado é o desenvolvimento de um método de controle de formação com seguimento de trajetória para um grupo de robôs móveis cooperantes, considerando canais de comunicação sujeitos a incertezas, tais como atraso de comunicação e falha temporária, utilizando RHC.

1.3.1. Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Incrementar o método desenvolvido por Ordoñez et al.[13] para resolver o problema de controle de formação com seguimento de trajetória;
- Adaptar o algoritmo para o uso com robôs com restrições não-holonômicas;
- Abordar outros aspectos do problema para permitir a utilização do método em um cenário real, por exemplo: a alocação dinâmica de posição na formação, a estratégia para evitar colisão entre e perda de comunicação com os integrantes do grupo;
- Avaliar o método proposto através de simulações e experimentos com veículos terrestres reais.

1.4. Organização

O restante deste trabalho tem a seguinte estrutura:

- No *Capítulo 2* é exposto o método de Ordoñez et al.[13], além dos conceitos necessários da teoria de grafo e RHC;
- O *Capítulo 3* contém o desenvolvimento do método proposto nesta dissertação, assim como os conceitos fundamentais para seu entendimento;
- No *Capítulo 4* é descrito o simulador desenvolvido para a avaliação do método proposto;
- No *Capítulo 5* são apresentados os resultados das simulações para escolha dos parâmetros necessários ao método e dos experimentos com veículos terrestres reais;

- Por fim, no *Capítulo 6* são feitos alguns comentários sobre o método desenvolvido e as principais possibilidades de aprimoramento são listadas.

Capítulo 2

Coordenação Utilizando Consenso

Para o desenvolvimento de um método de coordenação descentralizada, garantir que os veículos possuam informações compatíveis entre si é de primeira importância. Este capítulo apresenta uma introdução sobre a estratégia de consenso, com destaque para a formulação feita por Ordoñez et al.[13].

2.1. Consenso

Em um típico problema de consenso, tem-se vários veículos autônomos que se comunicam trocando informações e processando-as de acordo com algum algoritmo. Ou seja, ao longo das iterações, os veículos recebem informações de outros e executam um algoritmo para atualizar as suas, de forma que ela se torne coerente com as dos outros.

A estrutura de comunicação pode ser modelada através de um *grafo orientado* em que os nós representam indivíduos, os veículos, e as arestas representam os canais de comunicação. Um grafo orientado é um grafo no qual um nó estar conectado com outro não implica que o outro está conectado com o um. Matricialmente, um grafo pode ser representado pela *matriz de adjacência*:

$$A = \left[a_{ij}\right], \qquad a_{ij} \geq 0,$$

uma matriz $n \times n$, com n igual ao número de nós no grafo; i e j representam os índices dos nós, $a_{ij} > 0$ indica que a *aresta* (i, j) conectando o nó j ao i existe, enquanto $a_{ij} = 0$ indica que não existe. No contexto desta dissertação, $a_{ij} > 0$ indica que o indivíduo i recebe informação de j.

Na Figura 2.1 está representado um grafo orientado com quatro nós. As setas indicam a orientação da conexão, o que no presente con-

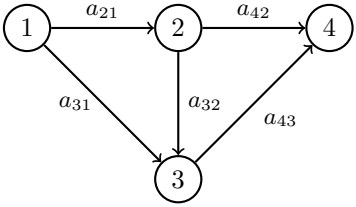


Figura 2.1: Exemplo de grafo orientado.

texto de grupos de veículos significa o sentido do fluxo de informação. Nesse caso, teríamos, por exemplo, a seguinte matriz de adjacência:

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

Note-se que não é necessário que todos os a_{ij} não nulos tenham o mesmo valor. Valores diferentes indicam ponderações diferentes das informações recebidas de cada fonte.

O algoritmo de atualização da informação pode ser entendido, de maneira geral, como uma forma de combinar os diversos valores da variável sobre a qual se deseja que haja concordância entre os robôs do grupo, de forma que cada um obtenha um novo valor que será mais coerente, mais próximo ao de todos os outros integrantes do grupo. Assim, ao longo das iterações, as informações originalmente divergentes convirgirão para o mesmo valor.

Seja $x(t)$ a variável sobre a qual se deseja o consenso, sendo t o tempo. Ela pode ser um escalar real, mas também pode ser um vetor de numeros reais (neste caso, o consenso significa o consenso sobre cada componente individualmente). Pode-se representar então o objetivo do consenso matematicamente como sendo fazer que:

$$\lim_{t\to\infty} \|x_j(t) - x_i(t)\| = 0, \qquad \forall i, j \in 1, ..., n.$$

Na formulação mais comum do algoritmo de consenso, em tempo contínuo, a *dinâmica da informação* é modelada (ou projetada) como linear de primeira ordem, sendo dada pela fórmula seguinte [9]:

$$\dot{x}_i = v_i(t) = \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}(x_j(t) - x_i(t)), \qquad \forall i \in 1, ..., n, \tag{2.1}$$

em que $v = \dot{x}$ é a variação no tempo da variável sobre a qual se deseja o consenso[1]. O funcionamento do algoritmo de consenso consiste portanto em o integrante do grupo atualizar sua informação com esse algoritmo. A implementação é, naturalmente, feita em tempo discreto, podendo-se usar então a aproximação de Euler:

$$x_{i,k+1} = x_{i,k} + hv_{i,k+1}, \qquad \forall i \in 1, ..., n,$$

em que $v_{i,k+1}$ é calculado como em (2.1), k é o índice do instante de tempo discreto e h é o período de atualização do algoritmo.

O comportamento do grupo pode ser descrito utilizando-se do Laplaciano da matriz de adjacência, definido por:

$$L = [l_{ij}], \qquad l_{ij} = \begin{cases} \sum_{\substack{q=1 \\ q \neq i}}^{n} a_{iq}, & \text{se } i = j, \\ -a_{ij}, & \text{se } i \neq j, \end{cases}$$

uma matriz $n \times n$. Então a dinâmica do sistema pode ser expressa como:

$$v = -L\ x.$$

em que x e v são compostos pelos x_i e v_i dos integrantes do grupo.

Para uma topologia de comunicação estática, pode-se mostrar que os nós que contribuem para o resultado final do consenso são aqueles que enviam informação, direta ou indiretamente, para todos os outros. Deve haver portanto pelo menos um tal nó. Usando a terminologia da teoria de grafos, esse fato pode ser expresso como: a condição fundamental para que o consenso possa ser atingido é a existência de pelo menos uma *árvore de extensão orientada*. Uma árvore de extensão orientada é um grafo orientado em que cada nó tem exatamente um nó pai, com excessão do nó raiz, que não tem pai e que possui um caminho orientado para todos os outros nós. Um caminho orientado é uma sequência de conexões orientadas, por exemplo, na Figura 2.1 $\{(2,1),(3,2),(4,3)\}$[2] é um caminho orientado, e o nó 1 está na raíz da árvore. Para o caso de uma topologia de comunicação variante no tempo, basta que pelo menos uma árvore exista "com frequência suficiente" [12, 9]. Ou seja, em cada iteração pode não existir nenhuma árvore, mas ao longo das iterações, deve existir uma árvore efetiva.

No contexto de robótica cooperativa, para o problema de *rendezvous* por exemplo, as variáveis em (2.1) podem ter os seguintes significados: v seria a velocidade do robô e x a sua pose, composta pelas

[1] Para simplificar a notação, omite-se a partir daqui a indicação (t) de variável no tempo.

[2] Notação: aqui, o par ordenado (a, b) representa a aresta que sai de b para a.

posições lineares e angulares, sendo n a quantidade de robôs no grupo[3]. A condição para a possibilidade do consenso pode então ter a seguinte interpretação: deve haver pelo menos um veículo (ou uma outra fonte, como uma central de controle) do qual todos os outros recebam informação, *direta* ou *indiretamente*. Para o caso em que há possibilidade de falhas de comunicação, o consenso será alcançado contanto que as falhas de comunicação sejam temporárias, havendo portanto troca de informação suficiente entre os veículos. Então, o consenso ser alcançado significaria que os robôs têm todos um valor bem parecido da pose do ponto de encontro.

2.2. RHC

O controle preditivo baseado em modelo, RHC, é uma técnica de controle bastante empregada. Ele consiste em: utilizando-se de um modelo do sistema, montar-se um problema de otimização, em geral na forma de uma programação quadrática, cuja solução fornece uma sequência de entradas de controle futuras.

A função objetivo em geral é projetada para minimizar o esforço de controle e o erro do sistema considerando as próximas p iterações, o horizonte de predição. Ou seja, partindo-se da informação sobre o estado atual do sistema, e baseado no modelo que dele se tem, otimiza-se, de acordo com alguma métrica dada na função objetivo, a previsão do comportamento do sistema para p instantes futuros, obtendo-se o sinal de controle necessário para conduzi-lo, na ausência de novas pertubações, para o estado desejado, por exemplo, de menor erro no seguimento de um valor de referência, de forma ótima.

Da sequência de entradas de controle obtida, apenas a primeira é, em geral, aplicada ao sistema controlado, e, a cada iteração, o problema é novamente resolvido, obtendo-se uma nova sequência e aplicando-se apenas o primeiro valor.

Esse método de controle possui diversas vantagens. Para o presente caso, de controle de robôs, pode-se destacar algumas:

- Facilidade para incorporar as limitações dos veículos, como, por exemplo, a velocidade máxima;
- Flexibilidade nos tipos de dinâmicas possíveis;
- Facilidade para incorporar valores futuros da referência;

[3]Note-se, neste caso, que considera-se então a dinâmica dos veículos igual à da informação, apesar de isso não ser necessário, pois se poderia, por exemplo, usar dois níveis de controle: o consenso da informação, a pose desejada; e o seguimento da informação por parte do robô, ou seja, controle do robô.

- Facilidade para lidar com atrasos e tempo morto.

Por outro lado, o método apresenta a desvantagem de depender de um algorítmo para solução do problema de otimização, que, para o caso de ser um programa não-convexo, frequentemente é computacionalmente custoso. Entretanto, normalmente o problema é projetado para que seja uma otimização quadrática com restrições lineares, ou seja, convexo, para os quais existem algoritmos bastante eficientes [23]. Em outros casos, a otimização é realizada previamente para diversos casos, resultando em um controlador com parâmetros tabelados (*gain scheduling controller*).

2.3. Consenso e RHC

No trabalho de Ordoñez et al.[13], o algoritmo básico de controle por consenso (2.1) foi reformulado como um problema de RHC e aplicado para a solução do problema de *rendez-vous*.

Nesse método, a velocidade de cada veículo é obtida pela solução do seguinte problema de otimização[4]:

$$\begin{aligned}
\min_{v_{i,k}} J_i = &\sum_{\substack{j=1 \\ j\neq i}}^{n+1}\sum_{k=1}^{p} \|x_{i,k} - x_{j,k}\|^2_{a_{ij}} + \sum_{k=1}^{p} \|\Delta v_{i,k}\|^2_{\lambda_i}, \qquad \forall i \in 1,...,n, \\
\text{s.t. } & x_{i,k} = x_{i,k-1} + h v_{i,k}, \qquad \forall k = 1,...,p, \\
& x_{j,k} = x_{j,0}, \qquad \forall j \neq i, \qquad \forall k = 1,...,p, \\
& \Delta v_{i,k} = v_{i,k} - v_{i,k-1}, \qquad \forall k = 1,...,p, \\
& v_{i,min} \leq v_{i,k} \leq v_{i,max}, \qquad \forall k = 1,...,p,
\end{aligned} \tag{2.2}$$

em que p é o horizonte de predição, x, variável de consenso, representa a pose, v representa a velocidade, o índice i indica que o elemento refere-se ao próprio veículo, enquanto o j, que refere-se aos vizinhos ou à referência (caso em que $j = n + 1$) e k indica o instante de predição, sendo $k = 0$ o valor conhecido no início da iteração; λ é um fator de ponderação. Note-se que, como neste caso x é um vetor, o problema de consenso é resolvido separadamente para cada um de seus componentes.

Assim, nessa formulação, uma sequência de velocidades desejadas futuras é prevista para o próprio robô, enquanto se considera todos os outros e a referência como estando parados.

[4]Notação: $\|a\|^2_b \equiv a^\top b a$, em que a é um vetor e b é um escalar ou matriz quadrada de dimensões compatíveis com a.

Em (2.2), o primeiro termo da função objetivo realiza o consenso da posição de *rendez-vous*, enquanto o segundo penaliza a variação na velocidade, suavizando a trajetória. A primeira restrição refere-se à dinâmica da trajetória desejada para o veículo; a segunda representa uma aproximação da posição dos outros veículos: como suas velocidades não são conhecidas, são consideradas nulas; a última restrição refere-se à inerente limitação na velocidade de um veículo real.

Esse método permite a solução do problema de *rendez-vous* de forma descentralizada e com topologia variante no tempo (comunicação entre robôs sujeita a falhas temporárias). Em relação à estabilidade e convergência, vale a mesma condição do algoritmo de consenso básico, (2.1).

Deve-se notar que essa formulação supõe que cada robô possui um sistema de controle local que permita a correspondência entre sua velocidade real e v e, consequentemente, sua pose real e x. Não há separação entre a variável representando a pose de encontro desejada e a representando a pose atual do robô.

O algoritmo de consenso básico, (2.1), e este, (2.2), comparam-se da seguinte maneira: ambos esto sujeitos ao mesmo critério de convergência, que depende unicamente do grafo de comunicação, ou seja, de quais termos da matriz de adjacência são nulos e quais não são, mas o resultado do consenso será diferente, pois o efeito do controle preditivo é alterar o peso dos canais de comunicação durante a computação do sinal de controle. Ou seja, o resultado de (2.2) seria similar ao resultado de (2.1) com uma matriz de adjacência em que os pesos fossem diferentes. Isso acontece porque em (2.1) a velocidade gerada é uma espécie de média ponderada dos erros, enquanto que em (2.2) são considerados também estados futuros de forma que a velocidade gerada é resultante nao apenas da média ponderada dos erros atuais, mas considera também erros futuros previstos. O efeito é uma ponderação diferente dos erros atuais.

2.3.1. Forma Matricial Final

Para a utilização desse método em um programa de otimização incorporado a um controlador, pode utilizar-se sua formulação matri-

cial. Nesse caso, a dinâmica da trajetória pode ser escrita como:

$$\underbrace{\begin{bmatrix} x_{i,1} \\ x_{i,2} \\ \vdots \\ x_{i,p} \end{bmatrix}}_{X_i} = \underbrace{\begin{bmatrix} x_{i,0} \\ x_{i,0} \\ \vdots \\ x_{i,0} \end{bmatrix}}_{X_{i,0}} + hT \underbrace{\begin{bmatrix} v_{i,1} \\ v_{i,2} \\ \vdots \\ v_{i,p} \end{bmatrix}}_{V_i}, \qquad X_{j,0} = \underbrace{\begin{bmatrix} x_{j,0} \\ x_{j,0} \\ \vdots \\ x_{j,0} \end{bmatrix}}_{X_{j,0}},$$

e a variação do sinal de controle de i pode ser escrita como:

$$\Delta V_i = \begin{bmatrix} \Delta v_{i,1} \\ \Delta v_{i,2} \\ \vdots \\ \Delta v_{i,p} \end{bmatrix} = V_\Delta \underbrace{\begin{bmatrix} v_{i,1} \\ v_{i,2} \\ \vdots \\ v_{i,p} \end{bmatrix}}_{V_i} - \underbrace{\begin{bmatrix} v_{i,0} \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{V_{i,0}},$$

onde:

$$T = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \qquad V_\Delta = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ -1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}. \tag{2.3}$$

são matrizes $p \times p$ constantes.

Substituindo-se a função objetivo, obtém-se então:

$$J_i = \sum_{\substack{j=1 \\ j \neq i}}^{n+1} \|X_{i,0} + hTV_i - X_{j,0}\|^2_{a_{ij}} + \|V_\Delta V_i - V_{i,0}\|^2_{\lambda_i}.$$

Finalmente, pondo-se V_i em evidência e retirando-se o termo constante, obtém-se a seguinte função objetivo, adequada ao formato padrão para programas de otimização:

$$J_i = \frac{1}{2} V_i^\top H_i V_i + f_i V_i,$$

onde:

$$H_i = \sum_{\substack{j=1 \\ j \neq i}}^{n+1} T^\top h a_{ij} h T + V_\Delta^\top \lambda_i V_\Delta,$$

e

$$f_i = \sum_{\substack{j=1 \\ j \neq i}}^{n+1} (X_{i,0}^\top - X_{j,0}^\top) a_{ij} h T - V_{i,0}^\top \lambda_i V_\Delta.$$

Como comentado anteriormente, o problema de consenso é resolvido para cada componente (ou seja, coordenada da pose) de x, já que não há, na dinâmica considerada, nenhum acoplamento entre elas. A princípio pode-se então resolver uma otimização para cada coordenada. Entretanto, o acoplamento poderia ocorrer nas restrições. Então, junta-se as funções objetivo em uma. Por exemplo, supondo que o veículo se mova em um plano e que representa sua pose possua duas componentes:

$$Z_i = \begin{bmatrix} V_i^x & V_i^y \end{bmatrix}^\top,$$

$$J_i = \frac{1}{2} Z_i^\top \begin{bmatrix} H_i & 0 \\ 0 & H_i \end{bmatrix} Z_i + \begin{bmatrix} f_i^x & f_i^y \end{bmatrix} Z_i.$$

em que os sobrescritos x e y indicam as respectivas componentes. Note-se que apenas o termo vetorial, f_i, precisa ser calculado para cada componente enquanto o valor de H_i não é influenciado pelos valores das componentes.

2.4. Exemplo

Para auxiliar o entendimento, a seguir há um exemplo da aplicação do algoritmo.

Considere-se um grupo com três veículos omnidirecionais idênticos em um espaço plano no qual a pose é descrita por três coordenadas e com a seguinte condição inicial:

$$\begin{aligned} x_1 &= \begin{bmatrix} -7 & -12 & 0.5 \end{bmatrix}, & v_1 &= \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}, \\ x_2 &= \begin{bmatrix} 5 & 11 & 1.5 \end{bmatrix}, & v_2 &= \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}, \\ x_3 &= \begin{bmatrix} -13 & 4.5 & -1 \end{bmatrix}, & v_3 &= \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}, \end{aligned}$$

e com as seguintes características:

$$\begin{aligned} v_{max} &= \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}, \\ v_{min} &= \begin{bmatrix} -1 & -1 & -1 \end{bmatrix}, \end{aligned}$$

e com os seguintes parâmetros da função objetivo:

$$p = 2, \qquad h = 0.5, \qquad \lambda = 1,$$

Figura 2.2: Grafo de comunicação correspontente à matriz em (2.4).

e matriz de adjacência:

$$A = \begin{bmatrix} 0 & 10 & 0 & 100 \\ 10 & 0 & 10 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{2.4}$$

em que a quarta linha/coluna refere-se à comunicação da/para a referência, e cujo grafo correspondente está representado na Figura 2.2. Considere-se também que a posição desejada para o encontro seja a origem do sistema de coordenadas:

$$x_4 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

Considere-se ainda que o algoritmo será executado pelo veículo 1 no início de uma iteração.

Montando-se as matrizes da função objetivo como descrito na Seção 2.3.1, obtem-se:

$$H_1 = \begin{bmatrix} 57 & 26.5 \\ 26.5 & 28.5 \end{bmatrix},$$

$$\begin{aligned} f_1^x &= \begin{bmatrix} -820 & -410 \end{bmatrix}, \\ f_1^y &= \begin{bmatrix} -1430 & -715 \end{bmatrix}, \\ f_1^\theta &= \begin{bmatrix} 40 & 20 \end{bmatrix}. \end{aligned}$$

Então, construindo-se apenas uma otimização, obtem-se um valor de -3251 para a função objetivo. As velocidades previstas são[5]:

$$V_1 = \begin{bmatrix} 1 & 1 & -0.6614 \\ 1 & 1 & -0.08674 \end{bmatrix},$$

[5]Aqui, V_1 está exibido com três colunas por questão de clareza. Na forma real, as três colunas estão empilhadas.

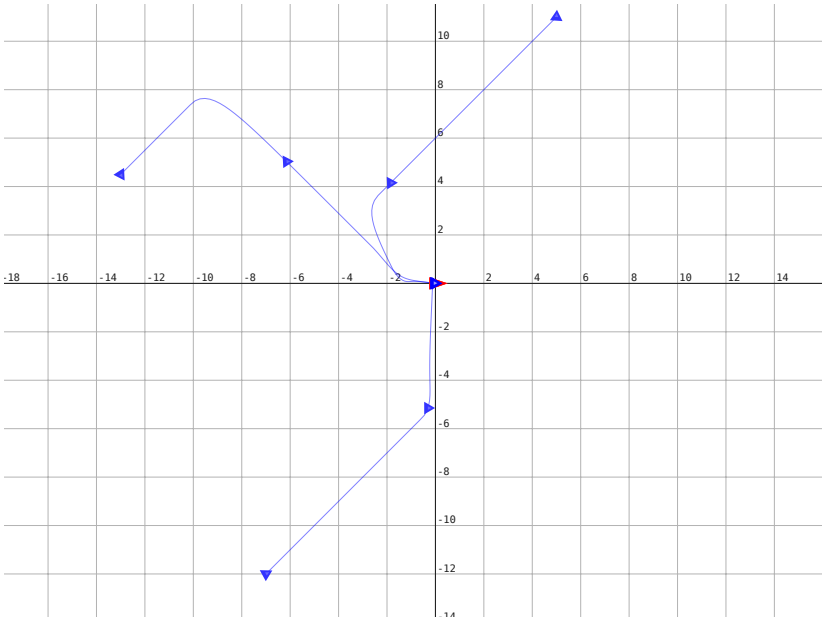


Figura 2.3: Exemplo de aplicação do algoritmo de Ordoñez et al.[13].

em que cada coluna esta relacionada a uma coordenada. De acordo com o método de controle preditivo usado, a primeira linha pode ser usada então como comando de velocidade do veículo omnidirecional.

Esse exemplo corresponde a execução do método de consenso para *rendez-vous* proposto por Ordoñez et al.[13] pelo primeiro veículo do grupo, na primeira iteração da simulação cujo resultado pode ser visto na Figura 2.3, que exibe a trajetória resultante dos veículos ao longo do tempo, durante o qual o algoritmo de consenso vai sendo executado por cada um. Nela, os veículos são representados por triângulos azuis em três instantes da simulação (início, meio e fim), e a referência, pelo triângulo vermelho. As linhas azuis representam os caminhos percorridos pelos robôs. Pode-se observar que os veículos se aproximam um do outro, a cada iteração, de acordo com a matriz de adjacência, por exemplo, o veículo 3 (cuja posição inicial está mais à esquerda da origem), por só receber mensagens do veículo 2 (cuja posição inicial está mais acima da origem), vai ao encontro de 2, enquanto o 2 vai ao encontro da posição “média” entre 1 e 3.

Neste algoritmo de consenso, a dinâmica da informação é considerada coincidente com a dinâmica dos veículos. Ou seja, a pose do veículo é considerada como sendo a ideia (informação) que ele tem sobre o ponto de encontro. Obtendo através do algoritmo o seu comando de velocidade, o veículo irá se aproximar dos outros, e assim, ao longo do tempo, os veículos se aproximaram e o consenso será atingido. Note-se que, evidentemente, não deseja-se a colisão dos veículos no ponto de encontro, entretanto, essa questão constitui outra a parte do problema de consenso. No capítulo seguinte, a proposta deste trabalho será apresentada. Nela, a pose do veículo não será mais a variável de consenso e então o consenso ser atingido não significará os robôs convergirem para o mesmo ponto, mas ainda se considerará ambas as dinâmicas como lineares de primeira ordem.

Capítulo 3

Controle de Formação com Seguimento de Referência

Neste capítulo, a contribuição desta dissertação é apresentada. Ela consiste em adaptar o método de Ordoñez et al.[13], descrito no capítulo anterior, para resolver não apenas o problema de *rendez-vous*, mas também o seguimento de trajetória com controle de formação.

3.1. Adaptação do Método de Ordoñez

O algoritmo de consenso desenvolvido por Ordoñez et al.[13] pode ser adaptado para o controle de formação. O consenso agora é a respeito do centro da formação, não mais do ponto de encontro, e o controle do veículo visa posicioná-lo em seu respectivo lugar na formação. Portanto, é necessário separar a variável que representa a posição sobre a qual ocorre o consenso, o centro da formação, e a que representa a posição atual do robô. Para isso, utiliza-se a ideia de estrutura virtual.

3.1.1. Estrutura Virtual

Uma estrutura virtual descreve uma formação a partir de seu centro (x^c), que funciona como um veículo virtual líder, e uma lista de vetores (x^f), que indicam as posições, relativas ao centro, a serem ocupadas pelos veículos reais. Considera-se que todos os veículos conhecem *a priori* essa lista de posições relativas e, particularmente, qual posição está alocada para ele ($x_i^d = x_i^f + x^c$). Note-se que o conhecimento de qual posição está alocada para ele é um problema a parte. Pode-se considerar que ela é conhecida desde o início pelos robôs do grupo ou que esse problema é resolvido ao longo do tempo, similarmente ao pro-

blema de consenso, porém antes, já que é uma informação necessária para solução do problema de consenso.

Na Figura 3.1 está representado um exemplo de estrutura virtual com três posições. Cada posição deve ser ocupada por um veículo real. Por exemplo: x_1 deve se mover para x_1^d.

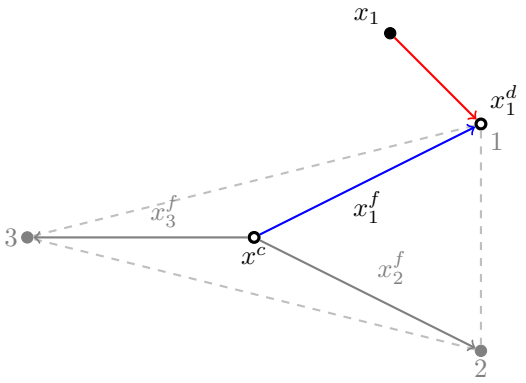


Figura 3.1: Exemplo de formação descrita por estrutura virtual.

No algoritmo proposto, faz-se uma descentralização da estrutura virtual. Cada robô passa a possuir, portanto, uma instanciação dela, como em Ren e Sorensen[22], isto é, uma versão (instância) do centro da formação. Assim, no exemplo dado, $x_1^d = x_1^f + x_1^c$. Essa instância será atualizada pelo próprio robô utilizando a solução do RHC, de forma a convergir para o mesmo valor das instâncias dos outros veículos e da referência.

3.1.2. Função Objetivo

O RHC deve então ser reformulado separando-se a variável que representa a posição do robô e a que representa a variável de consenso, que é agora a posição do centro da formação. Essas duas variáveis se relacionam através da estrutura virtual como explicado na Seção 3.1.1. Considera-se, além disso, o mesmo modelo linear de primeira ordem, tanto para a dinâmica da informação, como para a cinemática do veículo, como em Ren e Sorensen[22].

O RHC é responsável então por dois aspectos do controle:

- realizar o consenso sobre a posição *e* velocidade do centro da formação;
- gerar uma trajetória para o veículo.

Como se deseja que o centro da formação e, consequentemente, o próprio veículo sigam uma referência variante no tempo, adiciona-se termos nas respectivas velocidades. A razão é que a informação das posições dos outros que cada veículo possui estará sempre um pouco atrasada em relação às posições reais, pois os veículos se movem e há atraso na transmissão de informação, de forma que ele precisará estimar as posições futuras para tentar compensar esse atraso.

Percebendo-se, então, que se pode tratar a referência e as instâncias do centro de formação *como se fossem* veículos, obtém-se a seguinte função objetivo para o RHC, em que as variáveis de busca são as velocidades (do próprio veículo e da sua instância do centro da formação):

$$\begin{aligned} J_i = & \sum_{\substack{j=1 \\ j\neq i}}^{n+1} \sum_{k=1}^{p} \left(\beta_i^v \|v_{i,k}^c - v_{j,k}^c\|_{a_{ij}}^2 + \beta_i^x \|x_{i,k}^c - x_{j,k}^c\|_{a_{ij}}^2 \right) \\ & + \sum_{k=1}^{p} \left(\gamma_i^v \|v_{i,k} - v_{i,k}^d\|_{\alpha_i}^2 + \gamma_i^x \|x_{i,k} - x_{i,k}^d\|_{\alpha_i}^2 \right) \\ & + \sum_{k=1}^{p} \|\Delta v_{i,k}\|_{\lambda_i}^2 + \sum_{k=1}^{p} \|\Delta v_{i,k}^c\|_{\lambda_i^c}^2, \qquad \forall i \in 1, ..., n, \end{aligned} \tag{3.1}$$

em que α_i, β^v, γ^v, β^x, γ^x são fatores de ponderação dos termos, cujos sobrescritos v e x indicam se correspondem a termos de velocidade ou de posição. O primeiro termo (somatório duplo) dessa equação refere-se ao consenso sobre o centro da formação e inclui o termo da referência ($j = n + 1$). O segundo é o termo para o controle do próprio veículo. Os dois últimos referem-se à penalização na variação das velocidades, como em (2.2).

As restrições para o problema de minimização incluem as mesmas mostradas em (2.2): referentes à dinâmica dos veículos, e limitação das velocidades. São adicionadas as restrições referentes à dinâmica do centro da formação e, opcionalmente, referentes às distâncias mínimas e máximas entre os veículos. As restrições são discutidas mais à frente.

3.1.3. Forma Matricial

Neste trabalho é considerado o caso de veículos terrestres, cuja pose possui três componentes, duas posições lineares (ou simplesmente posição) e uma posição angular (ou orientação), (x, y, θ). O mesmo acontece com a pose do centro da formação. Portanto, é necessário

transformar o valor da posição relativa na formação antes de efetivamente usá-la para montar a função objetivo. Seja então:

$$\mathrm{Rot}(\cdot) = \begin{bmatrix} \cos(\cdot) & -\sin(\cdot) & 0 \\ \sin(\cdot) & \cos(\cdot) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

a matriz que representa a transformação (rotação das coordenadas lineares) necessária. Obtem-se o valor da posição relativa utilizada com a fórmula:

$$x^f_{i,0} = x^f_{i,\hat{0}}\mathrm{Rot}(\theta^c_{i,0}),$$

em que o índice $\hat{0}$ indica uma informação conhecida por todos igualmente.

Considerando-se todas as dinâmicas como lineares, pode-se reescrever a função objetivo em formato matricial fazendo-se:

$$X_{i,0} = \begin{bmatrix} x_{i,0} \\ \vdots \\ x_{i,0} \end{bmatrix}, \qquad X^c_{i,0} = \begin{bmatrix} x^c_{i,0} \\ \vdots \\ x^c_{i,0} \end{bmatrix}, \qquad X^f_{i,0} = \begin{bmatrix} x^f_{i,0} \\ \vdots \\ x^f_{i,0} \end{bmatrix},$$

$$V_{i,0} = \begin{bmatrix} v_{i,0} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad V^c_{i,0} = \begin{bmatrix} v^c_{i,0} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad V^c_{j,0} = \begin{bmatrix} v^c_{j,0} \\ \vdots \\ v^c_{j,0} \end{bmatrix}$$

vetores de comprimento p, em que o índice 0 indica o valor da variável no início de cada iteração (início do horizonte de predição do RHC), o sobrescrito f indica que o termo é uma posição relativa desejada na formação. Então:

$$\begin{aligned} X_i &= X_{i,0} + hTV_i, & V^c_j &= V^c_{j,0}, \\ X^c_i &= X^c_{i,0} + hTV^c_i, & V^d_i &= V^c_i, \\ X^c_j &= X^c_{j,0} + hTV^c_{j,0}, & \Delta V_i &= V_\Delta V_i - V_{i,0}, \\ X^d_i &= X^c_i + X^f_{i,0}, & \Delta V^c_i &= V_\Delta V^c_i - V^c_{i,0}, \end{aligned}$$

com T e V_Δ definidos em (2.3).

Assim, da mesma forma como na Seção 2.3.1, pode-se definir as

matrizes H_i e f_i[1]:

$$\begin{aligned} H_i = &\begin{bmatrix} 0 & 0 \\ 0 & \sum_j^{n+1} a_{ij} \end{bmatrix} \otimes A_i \\ &+ \begin{bmatrix} \alpha_i & -\alpha_i \\ -\alpha_i & \alpha_i \end{bmatrix} \otimes B_i \\ &+ \begin{bmatrix} \lambda_i & 0 \\ 0 & \lambda_i^c \end{bmatrix} \otimes {V_\Delta}^\top V_\Delta, \end{aligned} \tag{3.2}$$

$$\begin{aligned} f_i = &\begin{bmatrix} 0 & 1 \end{bmatrix} \otimes \sum_j^{n+1} a_{ij} C_{ij} \\ &+ \begin{bmatrix} \alpha_i & -\alpha_i \end{bmatrix} \otimes D_i \\ &+ \begin{bmatrix} {V_{i,0}}^\top \lambda_i V_\Delta & {V_{i,0}^c}^\top \lambda_i^c V_\Delta \end{bmatrix}, \end{aligned} \tag{3.3}$$

em que:

$$\begin{aligned} A_i &= \beta_i^v I_p + \beta_i^x T^\top h^2 T, \\ B_i &= \gamma_i^v I_p + \gamma_i^x T^\top h^2 T, \\ C_{ij} &= \beta_i^x (X_{i,0}^c - X_{j,0}^c)^\top hT - \beta_i^v {V_{j,0}^c}^\top, \\ D_i &= \gamma_i^x (X_{i,0} - X_{i,0}^c - X_{i,0}^f)^\top hT. \end{aligned}$$

Deve-se notar que a montagem da função objetivo é feita *por componente*. Assim, pode-se construir finalmente o problema de otimização para o controle de cada veículo computando-se (3.3) para cada coordenada:

$$Z_i = \begin{bmatrix} V_i^x & V_i^{cx} & V_i^y & V_i^{cy} & V_i^\theta & V_i^{c\theta} \end{bmatrix}^\top,$$

$$J_i = \frac{1}{2} Z_i^\top \begin{bmatrix} H_i & 0 & 0 \\ 0 & H_i & 0 \\ 0 & 0 & H_i \end{bmatrix} Z_i + \begin{bmatrix} f_i^x & f_i^y & f_i^\theta \end{bmatrix} Z_i,$$

em que os sobrescritos x, y e θ indicam as coordenadas usadas na construção dos termos. Note-se que os valores na coordenada θ não estão acoplado aos das outras, x e y, então a otimização relativa a essa coordenada poderia ser feita separadamente. Note-se também que apenas o termo f_i precisa ser calculado para cada coordenada. Os valores nas coordenadas x e y estão acopladas nas restrições, descritas a seguir.

[1]Notação: $A \otimes B \equiv [a_{ij} B]$, em que A e B são matrizes arbitrárias, é o produto de Kronecker.

Dois tipos de restrições são relevantes para o problema. Os robôs possuem limites em suas velocidades e além disso não devem se aproximar ou distanciar demais a ponto de colidirem ou perderem comunicação. Para se obter um problema de otimização quadrática com restrições lineares, foram elaboradas versões lineares dessas limitações.

3.1.4. Restrição nas velocidades dos veículos

O primeiro conjunto de restrições trata do limite máximo e mínimo para a velocidade do veículo em cada coordenada. Assim, por exemplo:

$$v^x_{i,min} \leq v^x_{i,k} \leq v^x_{i,max}, \qquad \forall k \in 1, ..., p.$$

Como as instâncias dos centros da formação são objetos virtuais, suas velocidades não precisam de limites.

Na forma matricial, obtém-se portanto:

$$\begin{bmatrix} I_p & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_p & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_p & 0 \\ -I_p & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -I_p & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -I_p & 0 \end{bmatrix} Z_i + \begin{bmatrix} -v^x_{i,min} \\ -v^y_{i,min} \\ -v^\theta_{i,min} \\ v^x_{i,max} \\ v^y_{i,max} \\ v^\theta_{i,max} \end{bmatrix} \geq 0.$$

em que I_p é a matriz identidade $p \times p$.

3.1.5. Restrição nas distâncias entre os veículos

O segundo conjunto de restrições trata das distâncias mínimas e máximas entre os robôs. É essa a restrição que acopla os valores das coordenadas x e y.

Os veículos devem manter uma distância mínima entre si para evitar colisões. Também devem manter uma distância máxima entre si para evitar a perda de comunicação com seus vizinhos. Como se trata de uma distância no plano, essa restrição considera apenas os termos nas coordenadas lineares (x, y).

Sejam os raios das zonas de segurança de cada veículo dados por $r_{i,sec}$ e os raios que definem o alcance da transmissão de informação de cada robô dados por $r_{i,com}$. Supondo que cada robô conhece esses raios dos seus vizinhos, tem-se que:

$$\underbrace{r_{i,sec} + r_{j,sec} + \delta_{ij,min}}_{r_{ij,min}} \leq \|d_{ij,k}\| \leq \underbrace{r_{j,com} - \delta_{ij,max}}_{r_{ij,max}},$$

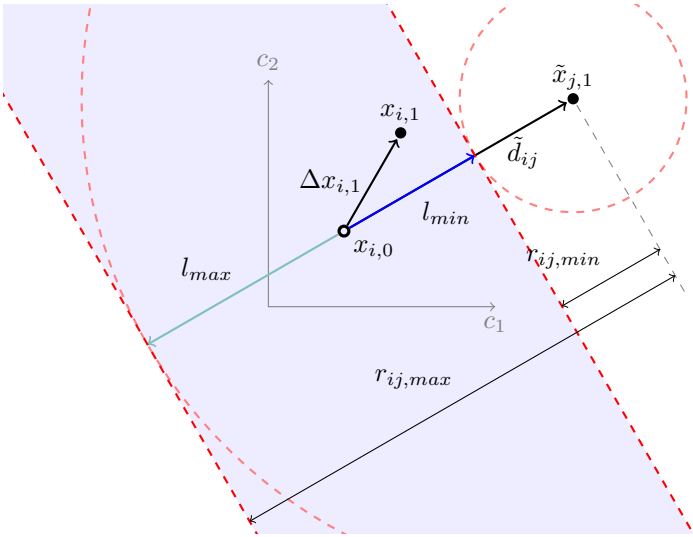


Figura 3.2: Aproximação linear das restrições nas distâncias entre os robôs.

em que d_{ij} é a distância entre os veículos e os termos $\delta_{ij,max} > 0$ e $\delta_{ij,min} > 0$ correspondem a folgas eventualmente necessárias para compensar a incerteza na posição e velocidade de j.

Da forma como está escrita, essa é uma restrição quadrática. Na Figura 3.2, vê-se esquematicamente a aproximação linear que pode ser obtida para o primeiro instante da predição ($k = 1$). Nela, a região azul corresponde à permitida.

Como $x_{j,1}$ não é conhecido, utiliza-se uma aproximação $\tilde{x}_{j,1} = x_{j,0}$, de forma que $\tilde{d}_{ij} = \tilde{x}_{j,1} - x_{i,0}$. A distância percorrida pelo veículo i deve respeitar os raios mínimos e máximos, então, utilizando-se a projeção escalar[2] do deslocamento $\Delta x_{i,1} = h v_{i,1}$ nas direções indicadas pelos $l_{min/max}$, obtém-se:

$$-\frac{h}{\|\tilde{d}_{ij}\|}\begin{bmatrix}\tilde{d}^x_{ij} & \tilde{d}^y_{ij}\end{bmatrix}\begin{bmatrix}v^x_{i,1}\\ v^y_{i,1}\end{bmatrix} + \|\tilde{d}_{ij}\| - r_{min} \geq 0,$$
$$\frac{h}{\|\tilde{d}_{ij}\|}\begin{bmatrix}\tilde{d}^x_{ij} & \tilde{d}^y_{ij}\end{bmatrix}\begin{bmatrix}v^x_{i,1}\\ v^y_{i,1}\end{bmatrix} + r_{max} - \|\tilde{d}_{ij}\| \geq 0.$$

Se o passo temporal (h) for suficientemente pequeno, a aproxi-

[2]Projeção escalar de um vetor A na direção de um outro B: $A_B = A \cdot \frac{B}{\|B\|}$.

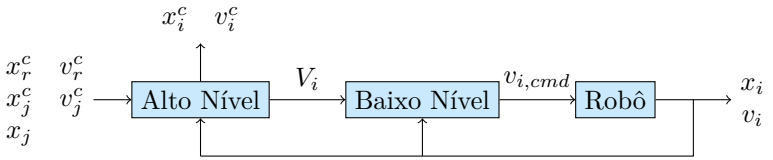


Figura 3.3: Estrutura de Controle.

mação será aceitável. Além disso, a aplicação dessas restrições em uma determinada iteração para um determinado robô é relevante apenas se houver possibilidade de serem violadas: se os robôs estiverem próximos ou distantes o suficiente. Assim, nem sempre são aplicadas.

3.2. Estrutura do Controlador

Pretende-se que o método aqui desenvolvido seja aplicado a robôs com restrições não-holonômicas, em particular, veículos diferenciais. É então necessária a divisão do controle do veículo em dois níveis:

- *Alto Nível* – composto pelo RHC, responsável por realizar o consenso e gerar a trajetória desejada para o veículo, em que se abstrai seu caráter não-holonômico;
- *Baixo Nível* – responsável por seguir a trajetória gerada, lidando com a não-holonomia do robô para que a aproximação da trajetória gerada no alto nível seja aceitável.

A Figura 3.3 mostra esquematicamente a estrutura do controle. $v_{i,cmd}$ denota um comando de velocidade adequado ao sistema de movimentação do robô. Note-se que a posição dos vizinhos é utilizada no controle de alto nível apenas nas restrições, mas não na função objetivo, e sua velocidade não é utilizada. Na função objetivo utiliza-se a posição da instância do centro de formação e sua respectiva velocidade.

Aqui, é necessário uma nota a respeito da natureza da referência. Ela pode ser uma referência global, transmitida por um líder ou uma central de controle para alguns dos robôs do grupo, mas também pode ser um valor local, único para cada veículo, saído de algum método de planejamento de trajetória executado descentralizadamente. Neste caso, se cada robô do grupo projetou uma trajetória, a trajetória final seria resultante do consenso sobre essas trajetórias de referência, contanto que os valores gerados por cada veículo nao sejam excessivamente diferentes.

3.3. Exemplo

Para auxiliar o entendimento, nesta seção é apresentado um rápido exemplo da aplicação do algorítmo.

Considere-se um grupo com três veículos omnidirecionais idênticos, com a seguinte condição inicial:

$$
\begin{aligned}
x_1 &= \begin{bmatrix} 0 & -1 & 0.5 \end{bmatrix}, & v_1 &= \begin{bmatrix} 0.5 & -0.1 & 0.1 \end{bmatrix}, \\
x_2 &= \begin{bmatrix} 0 & 1 & 1.5 \end{bmatrix}, & v_2 &= \begin{bmatrix} -0.2 & -0.2 & -0.2 \end{bmatrix}, \\
x_3 &= \begin{bmatrix} -1 & 0 & -1 \end{bmatrix}, & v_3 &= \begin{bmatrix} 0.3 & -0.4 & 0.3 \end{bmatrix}, \\
x_1^c &= \begin{bmatrix} 1 & 1 & 0.5 \end{bmatrix}, & v_1^c &= \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}, \\
x_2^c &= \begin{bmatrix} -1 & -1 & -0.2 \end{bmatrix}, & v_2^c &= \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}, \\
x_3^c &= \begin{bmatrix} 1 & 0 & 0.3 \end{bmatrix}, & v_3^c &= \begin{bmatrix} 0 & 0 & 0 \end{bmatrix},
\end{aligned}
$$

com as seguintes características:

$$
\begin{aligned}
v_{max} &= \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}, \\
v_{min} &= \begin{bmatrix} -1 & -1 & -1 \end{bmatrix}, \\
\delta_{min} = \delta_{max} &= 1.5,
\end{aligned}
$$

e com os seguintes parâmetros da função objetivo:

$$
\begin{aligned}
p &= 2, & h &= 0.5, \\
\alpha_i &= 1, & a_{ij} &= \begin{cases} 10, & \text{se } j \in 1, ..., n, \\ 100, & \text{se } j = n, \end{cases} \\
\beta^x &= 4, & \gamma^x &= 2, \\
\beta^v &= 1, & \gamma^v &= 1, \\
\lambda &= 1, & \lambda^v &= 1,
\end{aligned}
$$

e matriz de adjacência:

$$
A = \begin{bmatrix} 0 & 10 & 0 & 100 \\ 10 & 0 & 10 & 0 \\ 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{3.4}
$$

em que a quarta linha (coluna) refere-se à comunicação da (para) a referência, e cujo grafo correspondente está representado na Figura 3.4.

Figura 3.4: Grafo de comunicação correspontente à matriz em (3.4).

Considere-se também uma formação desejada descrita pelos vetores, já com suas respectivas alocações:

$$x_1^f = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix},$$
$$x_2^f = \begin{bmatrix} -1.5 & -1.5 & 0 \end{bmatrix},$$
$$x_3^f = \begin{bmatrix} -1.5 & 1.5 & 0 \end{bmatrix},$$

e uma referência com os valores:

$$x_4^c = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix},$$
$$v_4^c = \begin{bmatrix} 0.5 & 0 & 0.0754 \end{bmatrix}.$$

Considere ainda que o algoritmo será executado pelo veículo 1 no início de uma iteração.

Montando-se as matrizes como descrito na Subseção 3.1.3, obtem-se:

$$T = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \qquad \Delta V = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix},$$
$$A_1 = \begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix}, \qquad B_1 = \begin{bmatrix} 2 & 0,5 \\ 0.5 & 1.5 \end{bmatrix},$$
$$C_{12} = \begin{bmatrix} 8 & 4 \\ 8 & 4 \\ 2.8 & 1.4 \end{bmatrix}, \qquad C_{14} = \begin{bmatrix} 2 & 0.5 \\ 4 & 2 \\ 1.698 & 0.7738 \end{bmatrix},$$
$$D_1 = \begin{bmatrix} -0.8055 & -0.4028 \\ 0.07102 & 0.03551 \end{bmatrix},$$

notando-se que cada linha de C e D corresponde a uma coordenada e cada coluna a uma mensagem recebida, no caso, do veículo 2, vizinho, e da referência. Finalmente, obtem-se a função objetivo:

$$H_1 = \begin{bmatrix} 4 & -0.5 & -2 & -0.5 \\ -0.5 & 2.5 & -0.5 & -1.5 \\ -2 & -0.5 & 334 & 109.5 \\ -0.5 & -1.5 & 109.5 & 222.5 \end{bmatrix},$$

$$
\begin{aligned}
f_1^x &= \begin{bmatrix} -0.3055 & -0.4028 & 280.8 & 90.4 \end{bmatrix}, \\
f_1^y &= \begin{bmatrix} -0.02898 & 0.03551 & 479.9 & 240 \end{bmatrix}, \\
f_1^\theta &= \begin{bmatrix} 0.1 & 0 & 197.8 & 91.38 \end{bmatrix}.
\end{aligned}
$$

Como o veículo só recebeu mensagem de um outro vizinho e este estava perto o suficiente, é considerada também a restrição na distância, relativa ao raio de segurança. Calculando-se como explicado anteriormente, obtem-se a seguinte restrição:

$$
\begin{bmatrix} 0 & 0 & 0 & 0 & -0.5 & 0 & \dots & 0 \end{bmatrix}_{1\times 12} Z_1 + 0.8 \geq 0.
$$

Construindo-se uma otimização englobando as três coordenadas, obtem-se o valor de -543.7 para a função objetivo. As velocidades previstas são[3]:

$$
Z_1 = \begin{bmatrix} -0.3547 & -0.784 & -0.35 \\ -0.0738 & -0.6985 & -0.2658 \\ -0.8458 & -1.296 & -0.5477 \\ 0.008651 & -0.4471 & -0.1437 \end{bmatrix},
$$

em que cada coluna está relacionada a uma coordenada, as duas primeiras linhas às velocidades previstas para o veículo, enquanto as duas últimas, às velocidades previstas para o centro da formação. De acordo com o método de controle preditivo usado, a primeira linha pode ser usada como comando de velocidade do veículo omnidirecional, enquanto a terceira linha será usada para atualizar o valor da posição da instância do centro da formação.

Esse exemplo corresponde à execução do método proposto pelo primeiro veículo do grupo, na primeira iteração da simulação cujo resultado pode ser visto na Figura 3.5, que exibe a trajetória dos veículos. Nela, os veículos são representados por triângulos azuis, a referência, pelo triângulo vermelho, e as posições desejadas na formação, por triângulos verdes. A posição da referência coincide com uma das posições na formação. Já as linhas representam os respectivos caminhos percorridos, sendo as pretas referentes às instâncias dos centros de formação. Pode-se observar que os veículos entram em formação e seguem a referência, ainda que com algum erro.

[3] Aqui, Z_1 está exibido com três colunas por questão de clareza. Na forma real, as três colunas estão empilhadas.

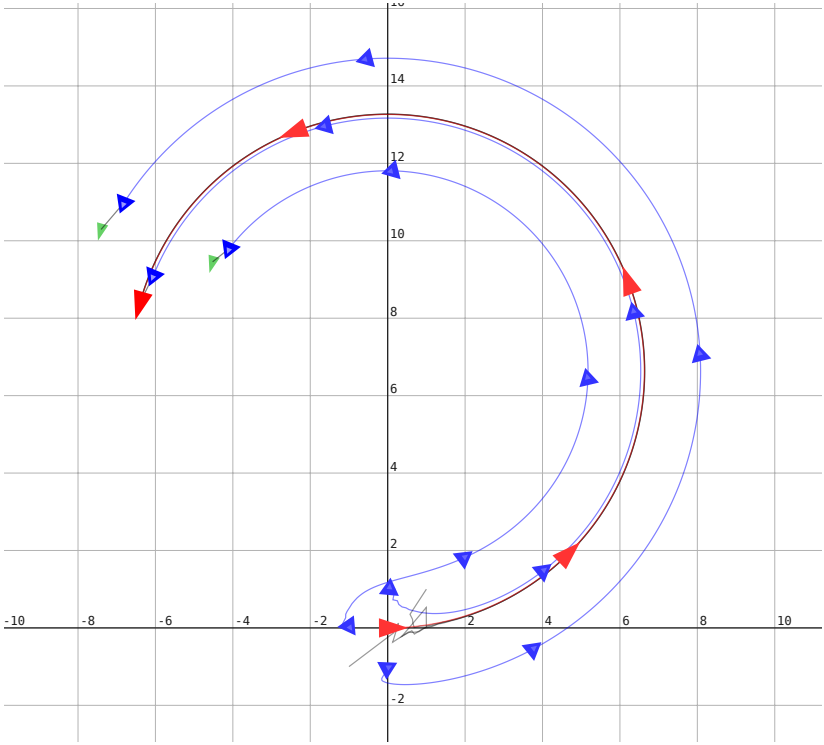


Figura 3.5: Exemplo de aplicação do algoritmo proposto.

3.4. Alocação de Posição na Formação

No método desenvolvido para controle de formação, é necessário que cada robô conheça qual posição na formação está alocada para si e para os outros.

O problema de alocação de posição pode ser formulado como uma otimização linear com restrições lineares a ser resolvido de forma centralizada da sequinte maneira: existem n veículos e portanto n posições na formação, e a cada veículo deve ser alocada uma posição de forma a diminuir o total de deslocamento necessário aos veículos para que entrem na formação. Seja w_{ij} a distância entre o robô i e a posição na formação j e u_{ij} uma variável binária que indica se o veículo i está alocado à posição j. Então:

$$\begin{aligned} \min_{u_{ij}} J &= \sum_{i=1}^{n}\sum_{j=1}^{n} u_{ij}w_{ij}, \\ \text{s.t.} \sum_{i=1}^{n} u_{ij} &\leq 1, \qquad \forall j \in 1,...,n, \\ u_{ij} &\in \{0,1\}, \qquad \forall i,j \in 1,...,n, \end{aligned}$$

que pode ser reescrito como:

$$\begin{aligned} \min_{u_{ij}} J &= \sum_{i=1}^{n}\sum_{j=1}^{n} u_{ij}w_{ij}, \\ \text{s.t.} \sum_{i=1}^{n} u_{ij} &= 1, \qquad \forall j \in 1,...,n, \\ \sum_{j=1}^{n} u_{ij} &= 1, \qquad \forall i \in 1,...,n, \\ u_{ij} &\geq 0, \qquad \forall i,j \in 1,...,n. \end{aligned}$$

Esse problema é conhecido como *assignment problem*. Um dos métodos para resolve-lo é o algoritmo de Kuhn-Munkres, que tem desempenho polinominal $O(n^3)$ [24].

Capítulo 4

Simulador Numérico

Neste capítulo, o simulador desenvolvido para realização das simulações numéricas, avaliação e uso do método proposto é apresentado.

O simulador opera em dois modos: o de *simulação*, em que realiza a simulação propriamente dita, mostrando graficamente a evolução dos robôs e outras variáveis, e o de *replay*, em que carrega o arquivo de saída gerado por uma simulação e exibe a trajetória dos robôs.

4.1. Simulador

O simulador foi desenvolvido utilizando a linguagem *C++*.

A interface gráfica foi feita utilizando-se as bibliotecas *GTK+*[1], para interface com o usuário e *Cairo*[2], para construção dos gráficos vetoriais para exibição da simulação.

O simulador suporta robôs simulados, simulando sua cinemática, e robôs reais (realidade aumentada), enviando comandos de velocidade e recebendo sua posição e velocidade. A comunicação com os robôs reais foi feita utilizando-se o ROS (*Robot Operating System*)[3], que é uma plataforma de código aberto para desenvolvimento de programas para robôs, tendo como base um sistema de gerenciamento de comunicação entre subprogramas.

Para o RHC, foi utilizado o algoritmo para solução de otimização quadrática com restrições lineares desenvolvido por Goldfarb e Idnani[25], que é um método dual baseado em conjunto ativo. A implementação utilizada foi adaptada a partir da versão de Gaspero[26].

A versão utilizada do algoritmo hungariano para alocação de

[1] <http://www.gtk.org/>
[2] <http://cairographics.org/>
[3] <http://www.ros.org/>

posição na formação foi adaptada para a linguagem *C++* a partir da versão de Pilgrim[27].

Uma simulação é configurada principalmente através de um arquivo de entrada, no qual são especificadas opções da simulação, valores dos parâmetros, as condições iniciais dos veículos e suas características. A referência pode ser, adicionalmente, configurada pela interface gráfica. O Apêndice A contém um exemplo de arquivo de entrada.

4.1.1. Arquitetura

A implementação do simulador é dividida em três níveis: *simulacional*, *de comunicação com o ROS* e *gráfico*.

O nível simulacional corresponde a implementação das funções que realizam a simulação propriamente dita: o algoritmo de alocação, o algoritmo de consenso de Ordoñez et al.[13] e o proposto, os controladores de baixo nível, os integradores da dinâmica dos veículos. E além dessas, funções auxiliares como: as funções de carregamento de um arquivo de entrada, configuração da simulação e realização de uma iteração, entre outras. Essa funcionalidade é implementada na classe `Sim`. Essa classe pode ser utilizada para compor um simulador com interface pela linha de comando. O esquema de uma possível tal implementação é descrito na subseção seguinte.

O nível gráfico corresponde à implementação de todas as funções necessárias para a interface gráfica com o usuário, entre elas, a construção do painel do simulador e da janela com a cena correspondente à simulação. Essa funcionalidade é implementada na classe `Gui`, que possui uma instância da classe `Sim` para ter acesso à funcionalidade de simulação. A programação da interface gráfica segue o modelo de programação orientada a eventos. Na implementação feita, usa-se um temporizador para gerar periodicamente o evento no qual uma iteração da simulação é realizada.

A funcionalidade de comunicação com os veículos reais através do ROS é feita na forma de funções externas, chamadas pela classe `Sim`, implementadas numa biblioteca estática. Isso é feito para separar o desenvolvimento do simulador propriamente dito do desenvolvimento da interação com os robôs reais.

O ROS consiste basicamente de um *framework* com funções para realizar comunicação entre nós em uma rede. No caso da presente dissertação, nos experimentos com veículos reais, cada um deles pertence a um nó, e o computador em que o simulador é executado pertence a outro. Durante a simulação, todos os cálculos são feitos pelo simulador.

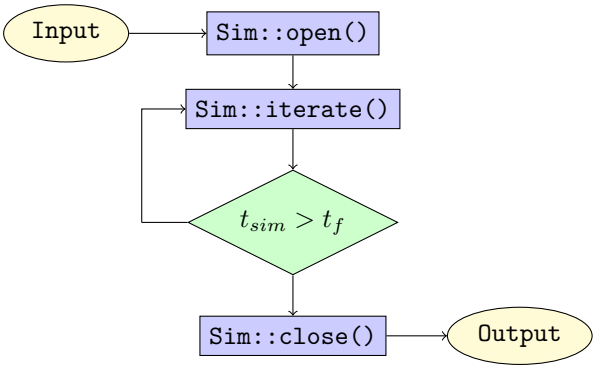


Figura 4.1: Fluxograma de um simulador para linha de comando.

Os comandos de velocidade de baixo nível são enviados através do ROS para os veículos reais, que por sua vez comunicam de volta suas odometrias (pose relativa à pose inicial e velocidade em coordenadas locais), que, ao serem recebidas, são convertidas para o sistema de coordenadas usado no simulador. Considera-se que qualquer filtragem de eventuais erros da odometria é feita pelo próprio robô.

4.1.2. `Sim::iterate()`

Utilizando-se a classe `Sim`, um simulador que funciona pela linha de comando pode ser feito de acordo com o fluxograma representado na Figura 4.1. O programa recebe como argumento o nome do arquivo de entrada, configura a simulação, realiza as iterações e termina. Como saída, é criado um arquivo com os dados da simulação.

A função `Sim::open()` é responsável por ler o arquivo de entrada e ajustar as opções e valores iniciais das variáveis do simulador, enquanto a `Sim::close()` é responsável por terminar o programa.

A função `Sim::iterate()` realiza a simulação propriamente dita, executando uma iteração do controle de baixo nível e, com um período maior, do controle de alto nível.

Ela funciona seguindo os passos abaixo, descritos para o caso em que é utilizado o algoritmo proposto nesta dissertação:

1) *Atualizar posição dos robôs reais.*
Caso existam.

2) *Corrigir o valor das variáveis angulares.*
Para evitar problemas devido ao domínio de uma variável angular ser restrito, os valores de variáveis angulares são ajustados

para um intervalo adequado.

3) *Calcular o valor do erro.*
A medida de erro será descrita no capítulo seguinte.

4) *Escrever no arquivo de saída.*
Adiciona uma linha no arquivo de saída contendo as informações relevantes sobre a simulação, como poses e tempo.

5) *Atualizar o controle de alto nível.*
Essa ação não é realizada em todas as iterações, mas apenas a cada tantas iterações quanto especificadas no arquivo de entrada. Isso se dá com a seguinte sequência de cálculos:

- **a)** *Atualização da matriz de adjacência.*
  Caso especificado que possa haver perda de comunicação.
- **b)** *Alocação das posições desejadas.*
  Caso haja realocação dinâmica, pois esta também pode ser estática.
- **c)** *Execução do algoritmo de consenso proposto.*
  Executado por cada veículo.
- **d)** *Atualização das instâncias do centro da formação.*
- **e)** *Atualização do controle de baixo nível.*

6) *Computar o sinal de controle de baixo nível.*
E enviá-lo para os robôs reais, caso existam.

7) *Integrar a dinâmica dos veículos.*
Apenas para os robôs simulados.

8) *Incrementar o tempo.*

9) *Atualizar a referência.*

4.2. Interface Gráfica

Na Figura 4.2, pode-se ver o painel principal do simulador. No topo, estão os botões para iniciar uma simulação, `Load`, ou exibir uma simulação salva, `Replay`, os dois modos de operação do programa, além do campo de entrada do arquivo de configuração da simulação. Abaixo do campo de entrada estão duas opções, `External`, que, se ativa, permite a comunicação através do ROS com os robôs reais, e `Output`, que indica se o arquivo de saída da simulação deve ser criado.

Ao se clicar no botão `Load`, a simulação configurada no arquivo de entrada especificado é carregada e os paineis de controle da simulação e de opções gráficas é ativado. Analogamente, quando o botão `Replay` é acionado, o arquivo de saída da simulação especificada é carregado para revisualização da simulação anteriormente feita. Esses dois estados do painel podem ser vistos nas Figuras 4.3 e 4.4, respectivamente.

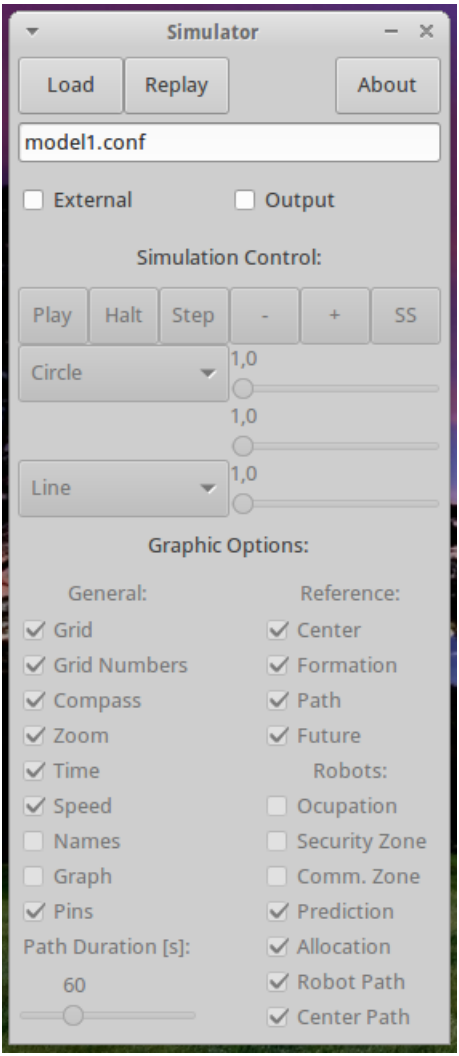


Figura 4.2: Painel de controle do simulador em seu estado inicial.

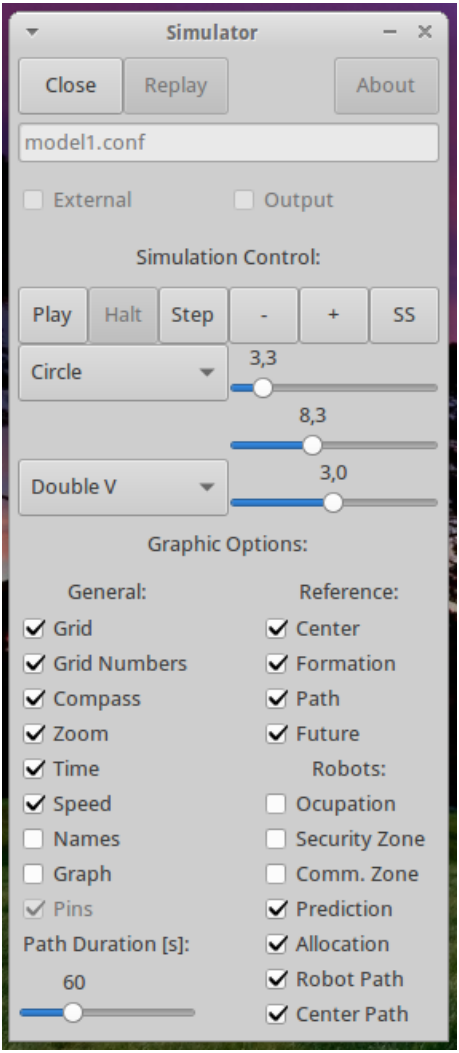


Figura 4.3: Painel de controle do simulador no modo de *simulação*.

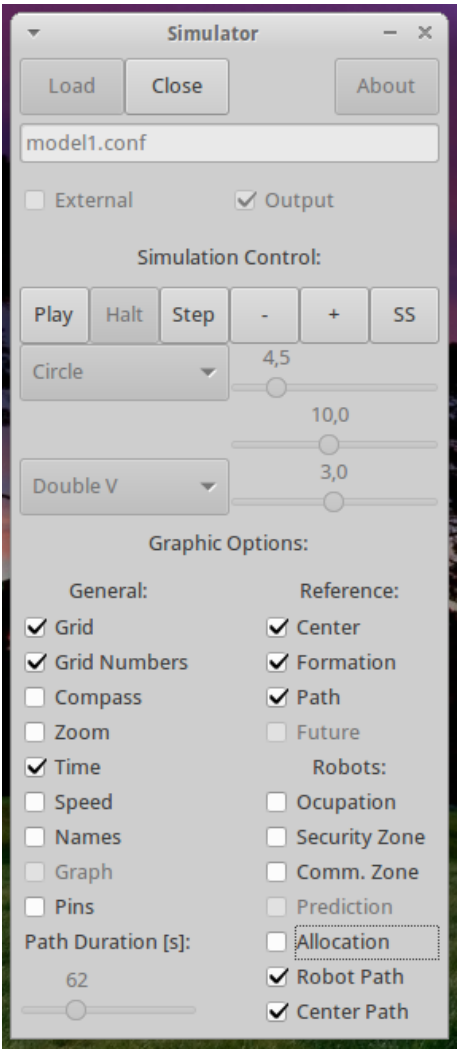


Figura 4.4: Painel de controle do simulador no modo de *replay*.

No painel de controle da simulação, o botão `Play` ativa a simulação ou a pausa, a depender do estado em que esteja. O botão `Halt`, faz com que a referência pare ou continue sua trajetória. O botão `Step`, ativo apenas se a simulação estiver pausada, serve para realizar apenas uma iteração da simulação. Os botões `-` e `+` servem para diminuir ou aumentar a velocidade da simulação. Por motivo evidente, se há robos reais participando da simulação, esses três últimos botões são desativados. O botão `SS` serve para salvar em `pdf` a tela atualmente exibida da simulação.

Os dois *combo boxes* servem para selecionar a trajetória desejada da referência e o tipo de formação. Os três *slides* ao lado servem para ajustar o tamanho e a velocidade da referência, e o tamanho da formação. O ajuste da formação é permitido a qualquer momento, mas o ajuste da referência, apenas no início, em $t = 0$.

A Figura 4.5 mostra o estado dos robôs ao se iniciar uma simulação. Como a referência pode ser ajustada nesse momento, ela é exibida para vários instantes futuros. Nessa figura pode-se ver, em azul, os robôs, em verde, as posições desejadas na formação, e em vermelho a referência. A linhas cinzas ligando os robôs às posições servem para indicar a alocação de posição.

A Figura 4.6 mostra o estado da simulação após algum tempo. Nela se pode ver em azul mais claro as posições previstas pelo algoritmo de controle para cada veículo, além das posições futuras da referência para os instantes de predição usados no algoritmo de controle, e, um tanto ocultadas por estas, as posições das instâncias do centro da formação de cada robô para os instantes de predição. As linhas azuis, pretas e a vermelha mostram os caminhos percorridos durante os últimos segundos da simulação. Já na Figura 4.7, pode-se ver uma tela salva através do clique no botão `SS` para o mesmo instante da simulação, dessa vez exibindo o grafo de comunicação.

A exibição de objetos na cena pode ser controlada pelas opções no painel de opções gráficas, descritas a seguir:

- `Grid`: ativa exibição da grade espacial;
- `Grid Numbers`: ativa a exibição dos valores das coordenadas na grade;
- `Compass`: ativa a exibição da bússola que aponta para a origem dos eixos coordenados;
- `Zoom`: ativa a exibição do valor do *zoom*, `z`;
- `Time`: ativa a exibição do valor do tempo da simulação, `t`;
- `Speed`: ativa a exibição do valor da velocidade da simulação, `s`;
- `Names`: ativa a exibição dos nomes dos robôs e das posições na

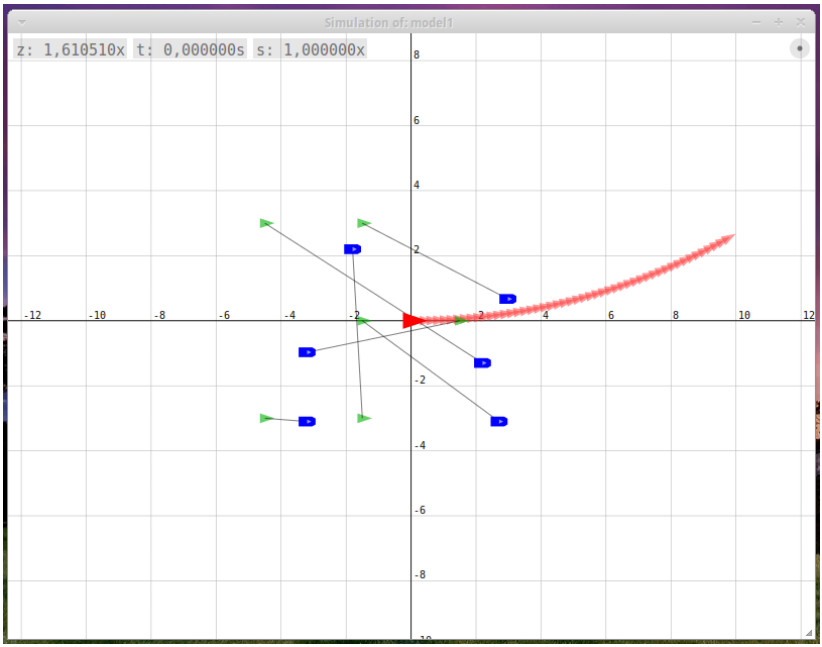


Figura 4.5: Cena do estado inicial em uma nova simulação.

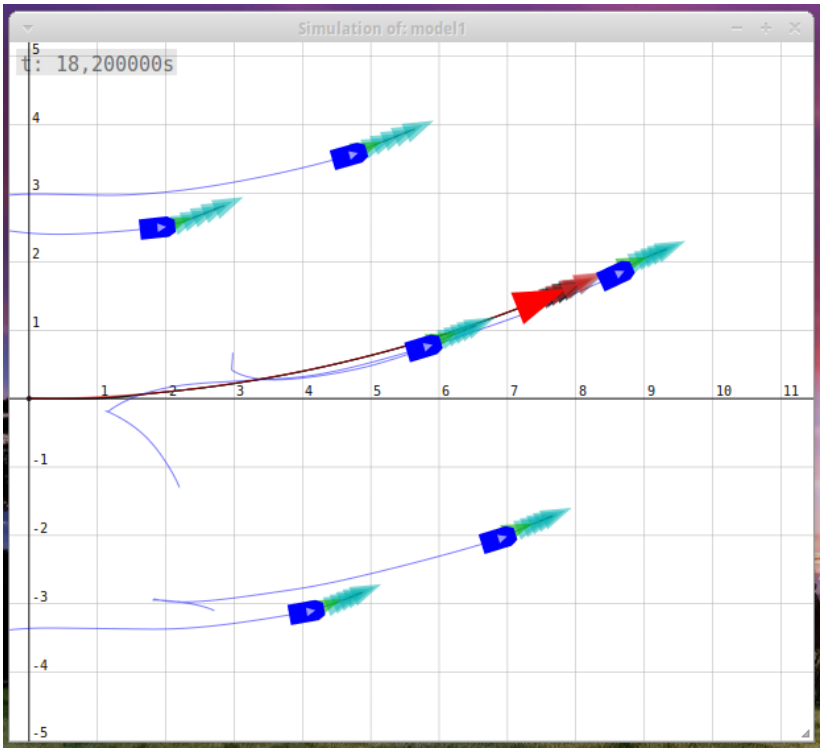


Figura 4.6: Cena exibida pelo simulador durante simulação.

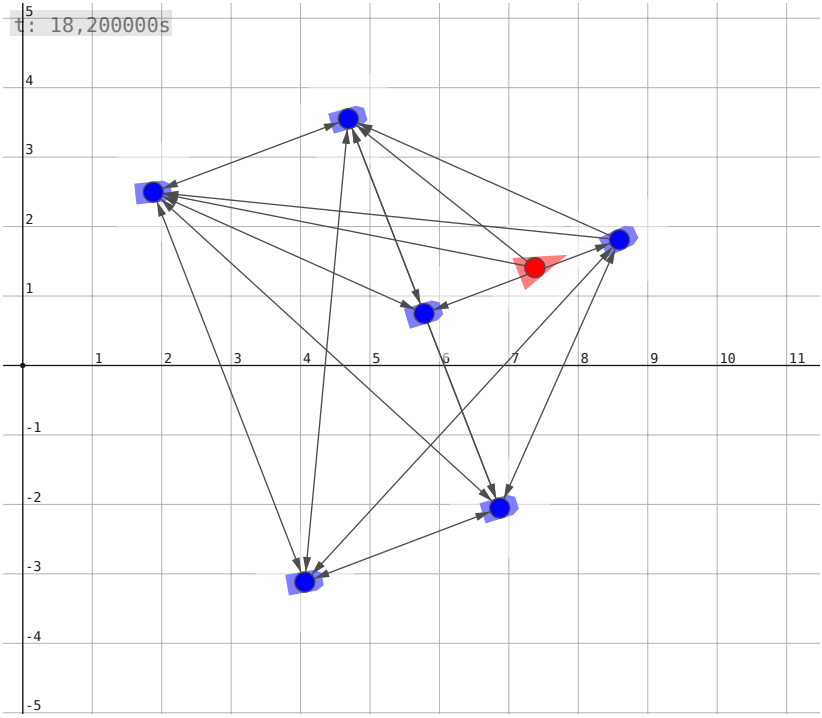


Figura 4.7: Grafo de comunicação exibido no simulador.

formação;
- `Graph`: ativa a exibição do grafo de comunicação;
- `Pins`: ativa exibição dos robôs em instantes selecionados na situação de *replay*;
- `Path Duration`: *slide* para selecionar a quantidade de segundos passados até o qual o caminho percorrido é exibido;
- `Center`: ativa exibição do centro da formação desejado, ou seja, da referência;
- `Formation`: ativa exibição da formação desejada;
- `Path`: ativa exibição do caminho percorrido pela referência;
- `Future`: ativa exibição dos valores futuros da referência dentro do horizonte de predição;
- `Ocupation`: ativa exibição da área circular ocupada pelo robô;
- `Security Zone`: ativa exibição das áreas de segurança dos veículos;
- `Comm. Zone`: ativa exibição das regiões de alcance de comunicação dos veículos;
- `Prediction`: ativa exibição exibição das posições previstas dos robôs;
- `Allocation`: ativa exibição dos traços ligando o robô à sua posição desejada na formação;
- `Robot Path`: ativa exibição dos caminhos percorridos pelos veículos;
- `Center Path`: ativa exibição dos caminhos percorridos pelas instâncias do centro da formação;

A Figura 4.8 mostra a cena inicial de uma simulação cujo arquivo de saída foi carregado ao se clicar no botão `Replay`. Nela se pode ver o caminho percorrido durante a simulação inteira e os robôs e a referência no instante inicial da simulação. O *slide* abaixo da cena serve para selecionar o instante em que os robôs são exibidos. O botão `Pin` serve para fixar a exibição dos robôs no instante atual, de forma a se poder vê-los ao mesmo tempo em diversos momentos da simulação. Até cinco instantes podem ser exibidos ao mesmo tempo. O botão `Clear` serve para apagar os instantes fixados. Já na Figura 4.9, pode-ser ver os veículos dos instantes fixados, inicial e final, além dos veículos em um instante intermediário, com seus nomes exibidos.

No estado de *replay*, apenas as informações salvas no arquivo de saída podem ser exibidas. A princípio poder-se-ia salvar tudo, mas, no estado de implementação do simulador no momento de escrita desta dissertação, são salvos apenas: as posições dos veículos, as posições das instâncias do centro da formação, as posições alocadas na formação em

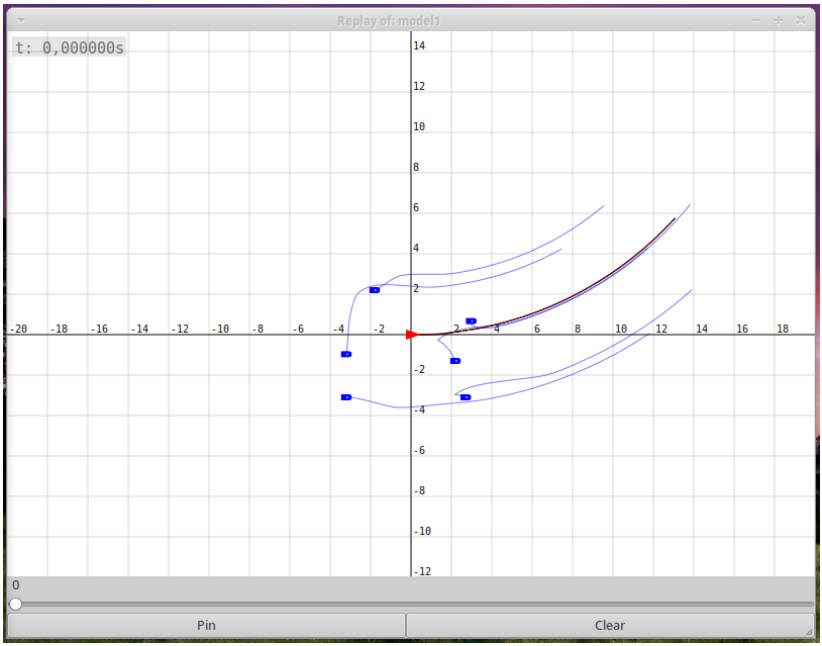


Figura 4.8: Cena inicial exibida no *replay*.

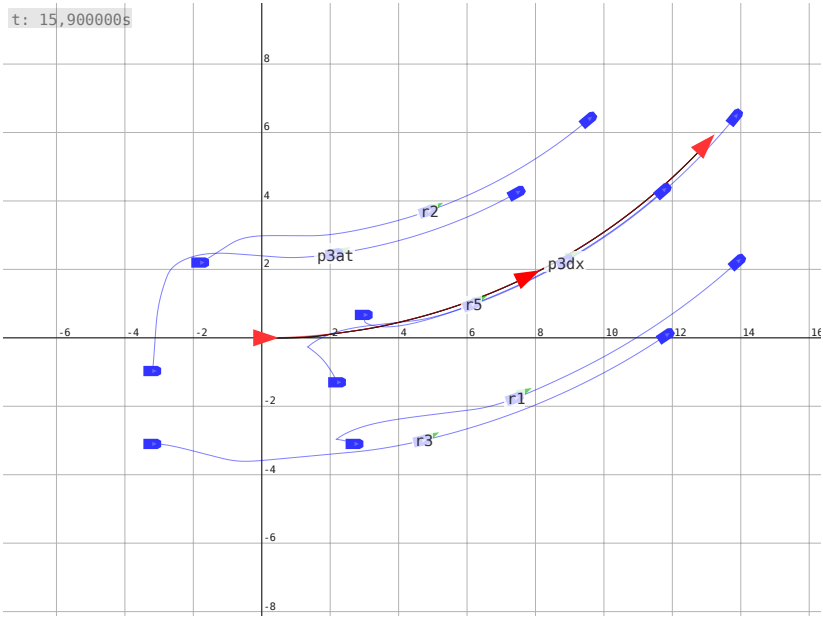


Figura 4.9: Cena do *replay* com instantes “pinados” e exibição dos nomes.

sua forma entendida pelo veículo e em sua forma desejada, as posições da referência, o tempo e o valor da métrica de erro, a ser descrita no próximo capítulo.

Capítulo 5

Resultados

O método proposto foi avaliado através de simulações e de experimentos com robôs reais.

Primeiramente, foi necessario determinar o conjunto de parâmetros da função objetivo do RHC que levassem a um melhor desempenho do método. Em seguida, o simulador foi usado em conjunto com robôs reais, que eram veículos com restrições não-holonômicas.

5.1. Parâmetros da Função Objetivo

Após a verificação de que a implementação do algorítmo e da parte não gráfica do simulador estava funcionando, constatou-se a necessidade de se ajustar os parâmetros da função objetivo e, consequentemente, avaliar como eles afetaria o desempenho do método. Para isso, foi definido um problema e feita uma busca em grade dos parâmetros. Assim, para uma determinada condição inicial dos veículos e uma determinada referência, foi feita uma série de simulações variando-se, de uma para outra, apenas os parâmetros da função objetivo.

No que se segue, para diferenciar o índice da referência do dos vizinhos, usa-se o índice r, que equivale a $j = n + 1$. Por exemplo, a_{ir}, adjacência com a referência, e a_{ij}, adjacência com vizinho.

A busca realizou-se para combinações considerando possíveis os seguintes valores dos parâmetros.

$$\begin{aligned} p &\in \{2, 6, 10\}, \\ \alpha_i, a_{ij}, a_{ir} &\in \{1, 10, 100\}, \\ \beta^x, \gamma^x &\in \{1, 2, 4\}, \\ \beta^v, \gamma^v &\in \{0, 1, 2, 4\}, \\ \lambda, \lambda^c &\in \{0, 1, 10, 100\}, \end{aligned}$$

o que resultou em pouco mais de 180000 simulações. Considerou-se o intervalo de tempo de cada iteração do RHC como constante, $h = 0.5s$.

Além disso, utilizou-se a seguinte fórmula para cálculo do erro em determinado instante da simulação:

$$\begin{aligned} E = &\sum_{i=1}^{n} \sum_{j=i+1}^{n} \left| \|x_i - x_j\| - \|x^f_{i,\hat{0}} - x^f_{j,\hat{0}}\| \right| \\ &+ \sum_{i=1}^{n} \|x_i - x^c_r - x^f_i(\theta^c_r)\| \\ &+ \sum_{i=1}^{n} \|x^c_i - x^c_r\|. \end{aligned} \tag{5.1}$$

Essa fórmula é composta por três termos, que referem-se respectivamente ao erro da formação, ao erro da posição desejada do veículo e ao erro do centro de formação, todos em relação ao desejado, à referência. A métrica para classificação da combinação de parâmetros testada consistia portanto da integração dessa medida de erro para toda a simulação e deveria ser minimizada.

Os parâmetros dos termos em velocidades também podem ser zero para se avaliar a diferença entre a presença e ausência desses termos. A seguinte alteração na construção da função objetivo torna-se então necessária:

$$X^c_j = X^c_{j,0} + \begin{cases} hTV^c_{j,0}, & \text{se } \beta_v \neq 0, \\ 0, & \text{se } \beta_v = 0. \end{cases}$$

A topologia de comunicação usada está representada na Figura 5.1. Note-se que apenas os robôs 1 e 3 recebem informação diretamente da referência, enquanto os 4, 6 e 2 a recebem indiretamente com atraso de uma iteração (através de 1 ou 3), e o 5, de duas iterações (através de 4 e 6).

Nessas simulações, os veículos foram considerados omnidirecionais, com dinâmica de baixo nível linear para seus movimentos, de forma a isolar o RHC da possível influência do controle de baixo nível. Além disso, a alocação de posição foi fixada, não utilizando o algoritmo de alocação; não se considerou as restrições nas distâncias entre os veículos; e utilizou-se uma topologia de comunicação fixa. Os robôs iniciam parados fora de formação e devem entrar em formação enquanto seguem a referência, que tem uma trajetória circular, com módulo da velocidade constante, de forma a permitir a avaliação da aproximação linear na predição da trajetória por cada veículo.

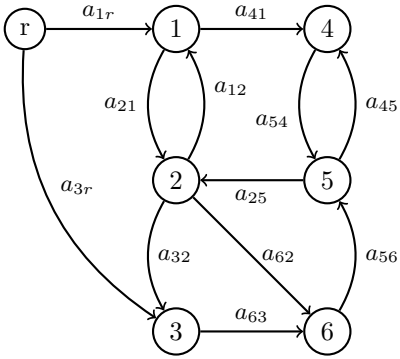


Figura 5.1: Grafo da comunicação.

5.1.1. Avaliação

Através das simulações feitas, constatou-se o funcionamento adequado do algoritmo: os veículos iniciam fora de formação e são capazes de entrar em formação enquanto seguem a referência. E, como esperado, a qualidade desse seguimento depende da escolha dos parâmetros da função objetivo.

A seguir são apresentados os melhores resultados obtidos na busca para os casos mais interessantes:

- *caso A*: a melhor combinação encontrada;
- *caso B*: a melhor sem os termos em velocidade;
- *casos C* e *D*: as melhores com $a_{ir} \in \{1, 10\}$, respectivamente;
- *casos E* a *H*: as melhores com a presença de outros termos em velocidade além de β^v.

Os parâmetros encontrados em cada caso estão mostrados nas Tabela 5.1, assim como o valor da métrica obtido através da utilização de (5.1).

No *caso B*, o erro em regime permanente dos veículos em relação a suas posições desejadas na formação é notavelmente maior (e cresce com a velocidade do centro da formação). Isso acontece porque, como não se consideram as velocidades, cada robô considera, a cada momento, que os outros estão parados, e não se adianta além da posição conhecida deles, que é intrinsecamente atrasada em relação à real. Assim, há nesse caso um atraso no seguimento tanto maior quanto menos se ponderar a referência e quanto mais atrasada for a informação da referência que o veículo recebe, visto que nem todos se comunicam diretamente com ela.

Levando-se em conta a velocidade, o veículo pode adiantar-se

Tabela 5.1: Resultados da busca em grade pelos melhores valores para os parâmetros.

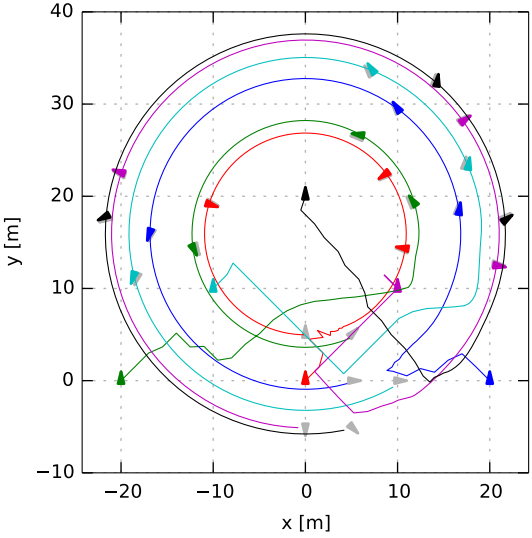
Caso	p	α_i	a_{ij}	a_{ir}	β^x	γ^x	β^v	γ^v	λ	λ^c	E
A	2	1	10	100	4	2	1	0	0	0	3010
B	2-10	1	1	100	4	1	0	0	0	0	7679
C	2	1	1	1	4	1	4	0	0	0	4008
D	2	1	1	10	4	1	1	0	0	0	3081
E	2	1	10	100	4	4	1	1	0	0	3117
F	2	1	10	100	4	1	1	0	0	1	3159
G	2	10	10	100	4	1	1	0	1	0	3538
H	6	10	10	100	4	2	4	0	1	1	3704

para uma posição esperada. Essa compensação só poderá ocorrer perfeitamente no caso de uma trajetória linear, pois o modelo dinâmico considerado na construção da função objetivo é linear. Portanto, há um erro em regime permanente proporcional à não linearidade da trajetória na posição dos veículos. Além disso, esse erro na estimativa da informação futura é o que causa o impacto negativo de um aumento no horizonte de predição na maioria dos casos, pois consiste numa extrapolação que em geral desviará do valor real.

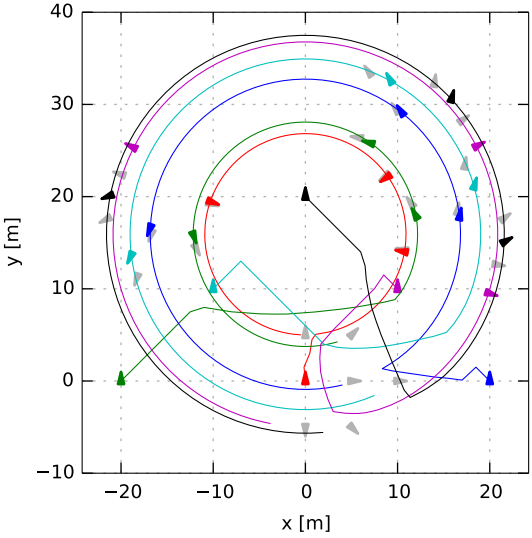
Os resultados mostrados na Tabela 5.1 também indicam um efeito negativo dos termos λ e λ^c, que têm como função suavizar a trajetória do robô e de sua instância do centro, respectivamente. Esses termos representam uma limitação a mais imposta na velocidade, o que explica seu efeito negativo.

Nas Figuras 5.2a e 5.2b são mostrados os comportamentos das posições dos veículos, linhas coloridas, na simulação para os *casos A* e *B* respectivamente. Para três instantes estão destacados os veículos, triângulos coloridos, e a formação desejada, triângulos cinzas, para comparação. Observe-se que, no segundo momento evidenciado, o erro das posições relativo ao centro da formação já atingiu estado estacionário, sendo igual ao do terceiro instante.

Nas Figuras 5.3a e 5.3b são mostradas as evoluções temporais das coordenadas x das instâncias dos centros de formação dos veículos, linhas coloridas, e da referência, linha cinza. O fato de que os robôs 1 e 3, vermelho e azul, recebem informação da referência, enquanto os outros não, reflete-se no "atraso de fase" visível no *caso B*, Figura 5.3b, conforme comentado anteriormente.

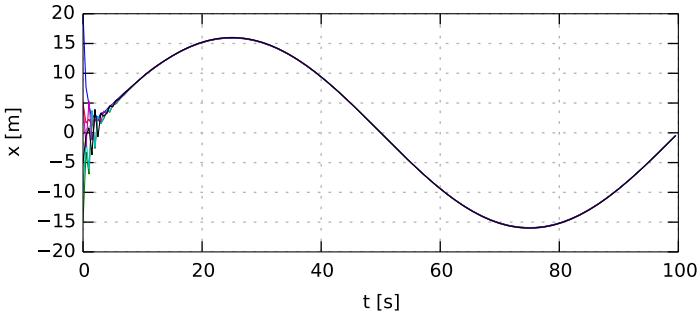


(a) *Caso A.*

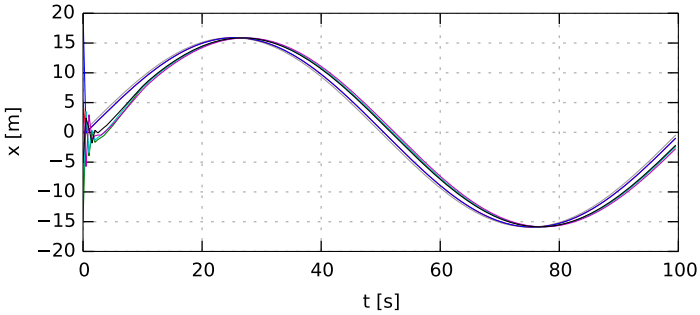


(b) *Caso B.*

Figura 5.2: Comportamento dos veículos para o *caso A* e o *caso B*.



(a) *Caso A.*



(b) *Caso B.*

Figura 5.3: Evolução da coordenada x das instâncias do centro da formação (veículos e referência) para o *caso A* e *caso B*.

5.1.2. Referência com Descontinuidades

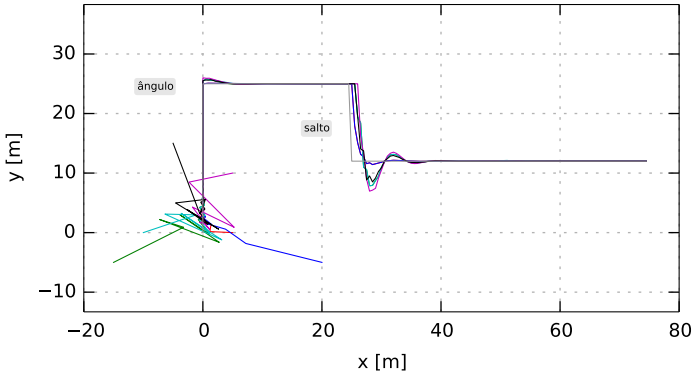
O comportamento do método proposto para o caso de uma referência descontínua também foi simulado. Nessa simulação, foram utilizados os parâmetros do *caso A*.

Nas Figuras 5.4a e 5.4b são mostrados os comportamentos das instâncias do centro da formação e dos veículos, respectivamente, assim como da referência. Aqui, a trajetória de referência apresenta três descontinuidades: de ângulo, em que a trajetória faz uma curva bruscamente, θ_r^c é descontínuo; de salto, em que a posição varia repentinamente, $(x, y)_r^c$ é descontínuo; de formação, em que as posições relativas desejadas, x_i^f, variam. Por clareza, as descontinuidades de ângulo e de salto estão indicadas na Figura 5.4a.

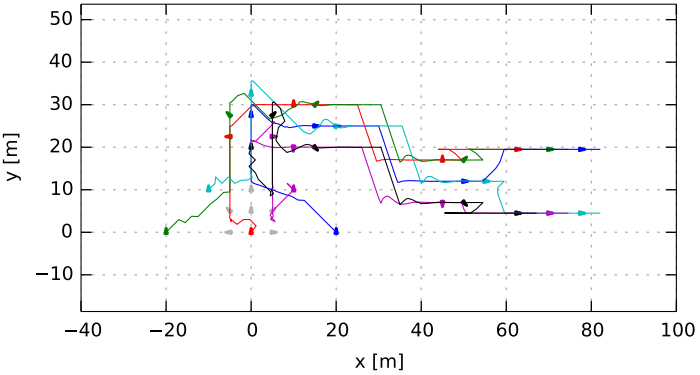
Na Figura 5.4a também pode ser observado o atraso da informação e o efeito da extrapolação linear explicados anteriormente. Os robôs 1 e 3 (vermelho e azul), que recebem informação diretamente da referência, apresentam melhor seguimento da trajetória. Na simulação, a velocidade de referência *não* apresenta descontinuidade no salto, apenas a posição é descontínua, de forma que os robôs 1 e 3 não "tentam se adiantar". Mas para os outros robôs, a velocidade do centro da formação que recebem de vizinhos, apresenta uma variação brusca, o que os faz "tentarem se adiantar", acarretando em maior desvio da trajetória desejada.

Ademais, o erro em regime permanente para o seguimento dos trechos de trajetória linear é zero, como esperado.

Para comparação, os comportamentos para a simulação com os parâmetros do *caso B*, ou seja, sem a comunicação das velocidades, são mostrados na Figura 5.5a e 5.5b. Neste caso, a mudança na formação afeta, ainda que pouco, o centro da formação. Isso é devido ao atraso no seguimento da trajetória.

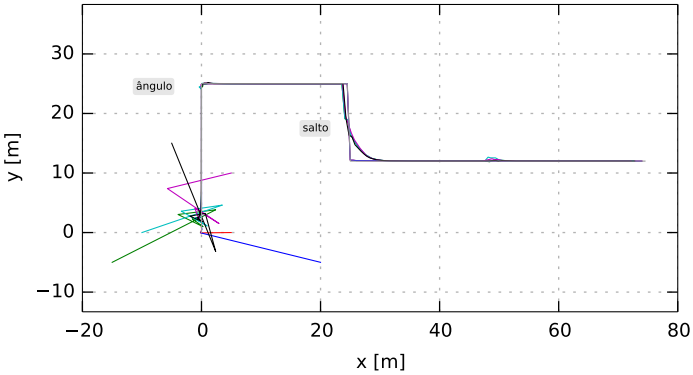


(a) Instâncias do centro da formação.

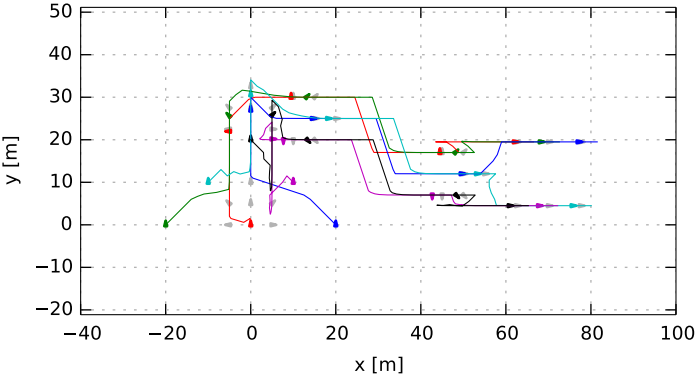


(b) Veículos.

Figura 5.4: Simulação para o caso de trajetória descontínua.



(a) Instâncias do centro da formação.



(b) Veículos.

Figura 5.5: Simulação para o caso de trajetória descontínua sem comunicação de velocidade.

Figura 5.6: Robôs e local dos experimentos

5.2. Experimentos com Robôs Reais

Para avaliar o funcionamento do método proposto em uma situação mais realista, em que haveria troca de mensagens por uma rede real e erros de medição da posição dos veículos, foram feitos dois experimentos em que dois dos seis robôs integrantes do grupo não eram mais simulados, mas sim robôs reais: um em que a trajetória era circular e outro em que era um quadrado.

Os robôs utilizados foram um P3AT[1] e um P3DX[2] da *Pioneer*, ambos diferenciais. O local dos experimentos e os robôs podem ser vistos na Figura 5.6. O método de localização utilizado por eles é baseado em odometria.

Os quatro robôs simulados foram considerados, do ponto de vista do algoritmo, idênticos aos dois reais. Isto é, eram também differenciais com idêntico controlador de baixo nível.

Para adequar o sinal de controle ao formato necessário para controlar um veículo diferencial, que recebe comandos de velocidade linear e velocidade angular, e não uma velocidade para cada coordenada, foi elaborado um controlador de baixo nível simples.

[1] <http://www.mobilerobots.com/ResearchRobots/P3AT.aspx>.
[2] <http://www.mobilerobots.com/ResearchRobots/PioneerP3DX.aspx>.

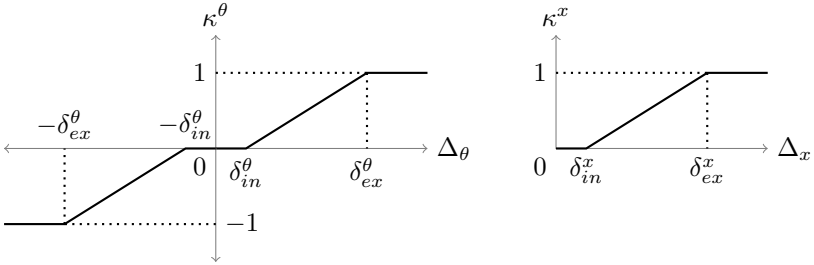


Figura 5.7: Variáveis do controle de posição para o baixo nível.

5.2.1. Controle de Baixo Nível

Enquanto veículos omnidirecionais podem receber comandos de velocidade diretamente nas três coordenadas locais, (x, y, θ), veículos diferenciais recebem comandos para movimento em apenas duas coordenadas locais, (x, θ).

O controle de baixo nível recebe como entrada a trajetória prevista pelo RHC na forma de uma sequência de velocidades. Ele utilizará essas velocidades para obter a posição desejada para o robô ao longo do tempo e guiá-lo até lá.

Para as simulações e experimentos com robôs reais, foi elaborada uma versão simples de um tal controlador. Ele usa a primeira velocidade da sequência e a posição atual do veículo para gerar uma posição desejada. Então, baseado no Δx, distância do veículo até a posição desejada, e no $\Delta\theta$, direção da posição desejada em relação à orientação do veículo, o comando de velocidade é gerado pela fórmula:

$$v^\theta_{cmd} = v^\theta_{cmd,max}\ \kappa^\theta, \qquad \text{e} \qquad v^x_{cmd} = v^x_{cmd,max}\ \kappa^x\ (1 - |\kappa^\theta|),$$

onde κ^θ e κ^x são dados pela fórmula representada graficamente na Figura 5.7. Nela, os $\delta_{in/ex}$ são parâmetros do controlador.

Esse controlador gera um comportamento do tipo *girar e ir em frente*. Os valores dos parâmetros utilizados estão contidos na Tabela 5.2.

5.2.2. Resultado dos Experimentos

Uma amostra do resultado de cada experimento está nas Figuras 5.8 e 5.9. Nelas, o robô *P3AT* está representado em vermelho, enquanto

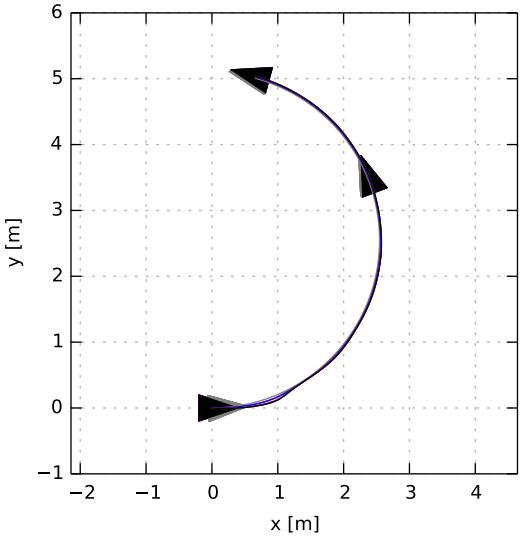
Tabela 5.2: Valores dos parâmetros do controlador de baixo nível usado.

$v^x_{cmd,max}$	$v^\theta_{cmd,max}$	δ^x_{ex}	δ^x_{in}	δ^θ_{ex}	δ^θ_{in}
$0,5m/s$	$0,7rad/s$	$0,5m$	$0,2m$	$0,5rad$	$0,1rad$

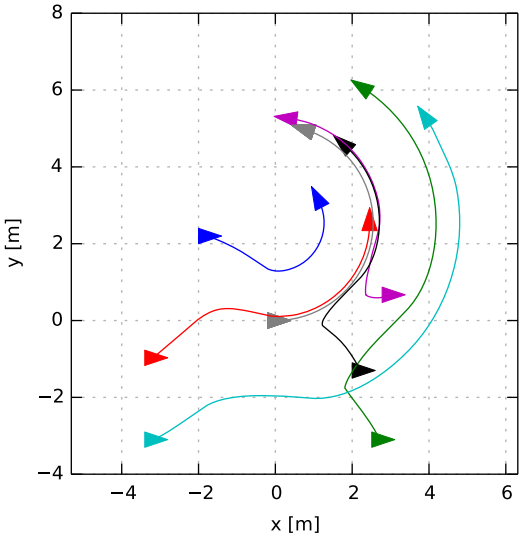
P3DX está representado em rosa. Os outros veículos foram simulados, e a referência está em cinza.

Nesses experimentos pôde-se everiguar o adequado funcionamento do método diante de uma situação em que havia transmissão de mensagens numa rede real, através da rede sem fio local, e veículos reais com incertezas na posição e na velocidade, além de necessidade de um controlador de baixo nível para adequar a trajetória gerada pelo RHC a um sistema não linear, pois os robôs eram veículos diferenciais.

O desempenho no transiente é bastante condicionado pela qualidade do controle de baixo nível, enquanto no estado permanente é condicionado pela qualidade do método de localização dos veículos reais e da estimativa das posições iniciais.

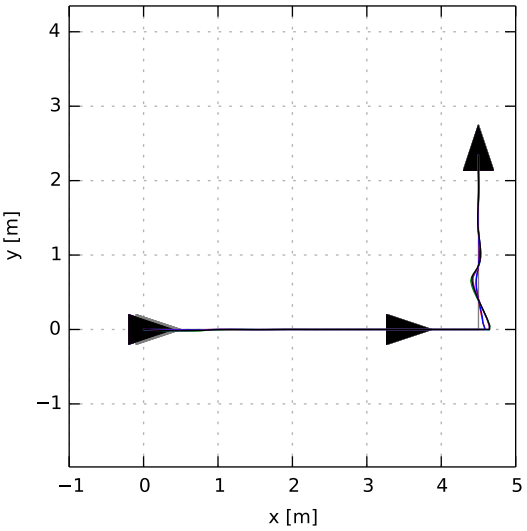


(a) Instâncias do centro da formação.

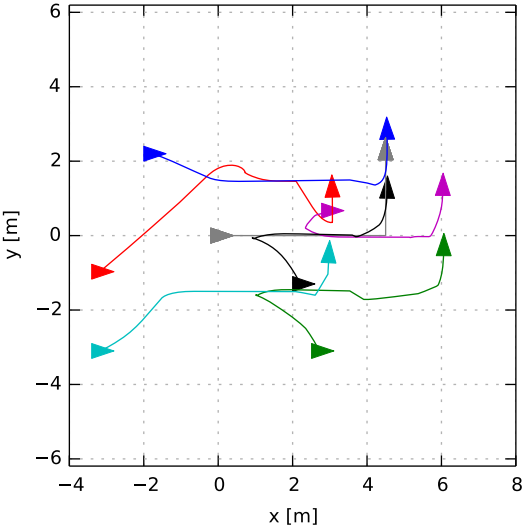


(b) Veículos.

Figura 5.8: Resultado do experimento com trajetória circular.



(a) Instâncias do centro da formação.



(b) Veículos.

Figura 5.9: Resultado do experimento com trajetória quadrada.

Figura 5.10: Grafo da comunicação correspondente a matriz (5.2), que resulta em instabilidade da variável de consenso.

5.3. Convergência

Durante as simulações, pôde-se perceber que o critério de convergência do consenso como descrito na Seção 2.1 não é mais suficiente no método proposto caso $\beta^v \neq 0$.

Por exemplo, substituir a matriz de adjacência usada no exemplo da Seção 3.3 pela seguinte:

$$A = \begin{bmatrix} 0 & 10 & 0 & 100 \\ 10 & 0 & 10 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{5.2}$$

resulta em divergência na variável de consenso, o centro da formação, cujo valor se torna instável, ou seja, oscila com módulo crescente. Na Figura 5.10 está representado o grafo correspondente. Percebe-se que ele satisfaz a condição de convergência do algorítimo de Ordoñez et al.[13].

Uma outra possibilidade para a matriz de adjacência que resulta em instabilidade é:

$$A = \begin{bmatrix} 0 & 10 & 0 & 100 \\ 0 & 0 & 10 & 0 \\ 10 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

A ocorrência da instabilidade é independente dos valores iniciais de posição e velocidade dos robôs ou referência, dependendo apenas da topologia de comunicação.

Caso não seja considerado o termo em velocidade, $\beta^v = 0$, a instabilidade não ocorre, e o critério de convergência citado torna a ser válido. Entretanto, a qualidade do seguimento de trajetória é comprometida, havendo então maior atraso.

Nas Figuras 5.11 e 5.12, vê-se o resultado da simulação para os dois casos, respectivamente: $\beta^v \neq 0$, instável; e $\beta^v = 0$, estável, mas com desemepenho comprometido. Particularmente, na primeira, vê-se

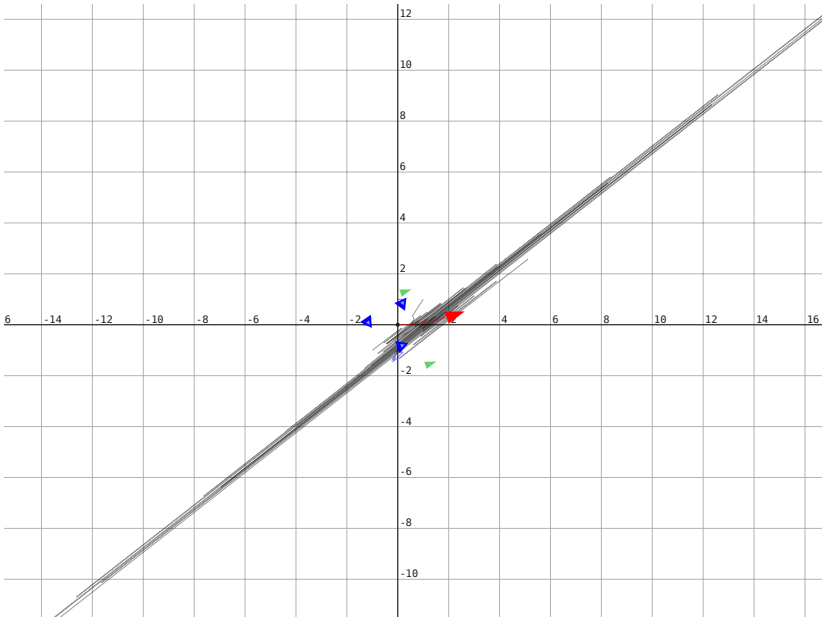


Figura 5.11: Simulação em que ocorre instabilidade.

que as trajetórias das instâncias do centro da formação, representadas pelos traços cinzas, oscila com amplitude exponencialmente crescente.

Outra situação que ameniza o efeito da instabilidade é quando há falhas de comunicação com frequência suficiente. As falhas de comunicação acarretam uma alteração no grafo de comunicação "efetivo" de forma que o consenso pode convergir. Na Figura 5.13 pode-se ver o resultado da mesma simulação, dessa vez com a probabilidade de falha na comunicação entre os veículos 2 e 3 (a_{32} e a_{23}) de 50%.

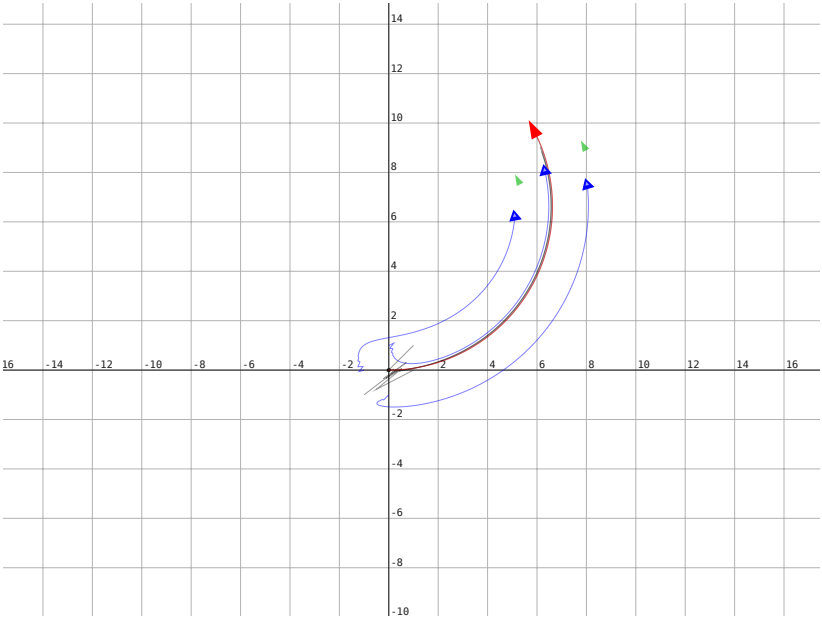


Figura 5.12: Simulação em que se evita a instabilidade através do uso de $\beta^v = 0$.

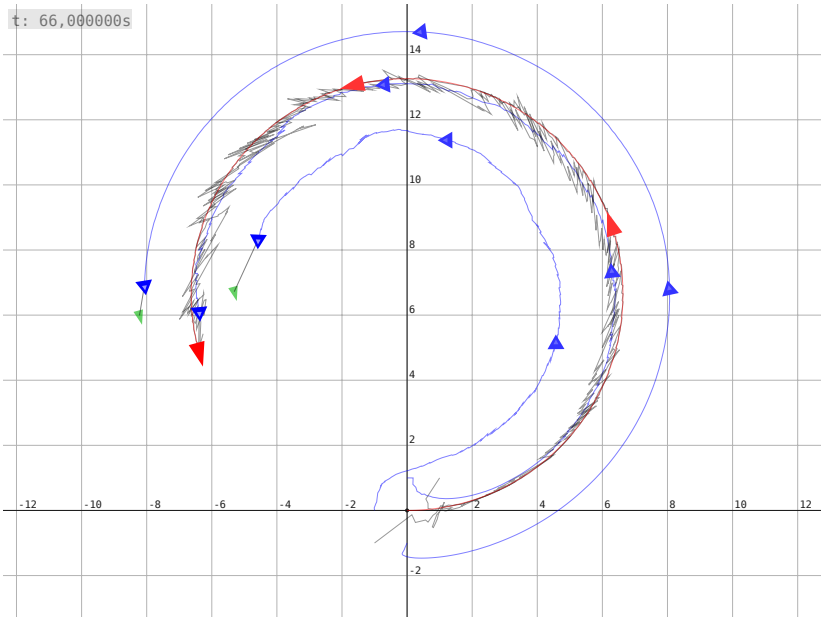


Figura 5.13: Simulação em que o efeito da instabilidade é amenizado pela falha na comunicação.

Capítulo 6

Conclusão

Neste trabalho de mestrado foi estudado o controle de formação e seguimento de trajetória para grupos de veículos.

Particularmente, o método de consenso para *rendez-vous* de Ordoñez et al.[13] foi incrementado para suportar o controle de formação e seguimento de referência através do uso do conceito de estrutura virtual, de maneira similar a Ren e Sorensen[22]. O método proposto foi formulado como um RHC baseado em otimização quadrática com restrições lineares.

O desempenho do método depende da escolha dos parâmetros da função objetivo. Para achar o melhor conjunto desses parâmetros, foi realizada uma busca em grade. Deve-se notar que, entretanto, a escolha dos melhores parâmetros está naturalmente condicionada tanto pela métrica utilizada como pelos estados iniciais dos robôs e trajetória da referência. O comportamento obtido para a variável sobre a qual o consenso ocorre corresponde a uma dinâmica de segunda ordem, e, a depender da escolha dos parâmetros, será sub ou sobre-amortecido.

Mesmo com a escolha adequada de parâmetros o erro em regime permanente para o caso de uma referência variante no tempo não pôde ser anulado, sendo proporcional ao atraso da informação e à qualidade da aproximação linear usada para a previsão da trajetória.

O método apresentado foi aplicado a veículos terrestres. Para isso foi desenvolvido um simulador capaz de se comunicar com robôs reais através do ROS. Foram apresentados os resultados de algumas simulações, considerando veículos omnidirecionais e diferenciais (veículos com restrições não-holonômicas), e topologias de comunicação fixas ou variantes no tempo. Algums experimentos em que dois dos robôs eram reais, nos quais pôde-se comprovar a viabilidade do método, também foram apresentados. Constatou-se também a influência da qualidade

do controle de baixo nível, em especial dos limites nas velocidades dos veículos e do método de localização, na qualidade do comportamento final.

A vantagem do método proposto em relação a outros da literatura que também utilizam RHC está na menor quantidade de informação trocada, apenas a posição e a velocidade da instância do centro da formação e a própria posição, e menor esforço computacional, pois a otimização é realizada apenas para o próprio veículo. No método de Dunbar e Murray[20], cada robô comunica toda sua trajetória prevista. Já no de Keviczky et al.[21], cada robô realiza a otimização para si e para seus vizinhos conhecidos.

Entretanto, o método proposto pode apresentar divergência na variável de consenso e consequentemente falhar em seu objetivo, aspecto que necessita de maiores investigações. Além disso, supõe-se que a execução do algoritmo em cada veículo e a troca de mensagens ocorre de maneira síncrona (a troca de mensagens ocorre "instantaneamente", não se considerando atrasos ou tempo morto).

6.1. Publicações

Durante o desenvolvimento desta dissertação, o seguinte trabalho foi publicado no Congresso Brasileiro de Automática:

- Controle de Veículos Autônomos em Formação com Seguimento de Referência Utilizando Consenso e RHC [28].

6.2. Trabalhos Futuros

Várias são as possibilidades de continuação e aprimoramento do método proposto. Entre elas, destacam-se:

- *Estudar as condições de convergência do algoritmo.*
  Com a adição dos termos em velocidade na função objetivo, a condição de convergência do método de Ordoñez et al.[13] passa a não valer mais. De fato, em algumas simulações, pôde-se observar a instabilidade no consenso sobre a posição do centro da formação. Isso impõe a necessidade do estudo das condições de convergência do método proposto.
- *Descentralizar a alocação de posição na formação.*
  O método utilizado para isso é centralizado, o que diminui a autonomia dos robôs e sabota a proposta de um controle descentralizado por consenso. Deve-se, então, substituí-lo por um método descentralizado, possivelmente baseado em consenso.

- *Reelaborar as restrições* ou *utilizar um método de otimização não-convexa.*
  A aproximação linear das restrições de distâncias cumpre a função de evitar colisão ou distanciamento excessivo dos veículos. Entretanto, particularmente na restrição de proximidade, elas só podem ser aplicadas para o primeiro instante previsto, o que impede a geração de trajetórias que contornem um obstáculo. A reelaboração das restrições para que apareçam como elementos da função objetivo *ou* a formulação do problema como uma otimização não-convexa (com, por exemplo, função objetivo *e* restrições quadráticas) daria ao método a capacidade de gerar trajetórias que contornem obstáculos. Essa segunda opção traria a desvantagem de ser computacionalmente mais custosa, mas, por ser mais geral e portanto mais flexível, teria a vantagen de permitir a inserção de outros elementos no problema. Além disso, poderia-se elaborar restrições relacionadas à não-holonomia dos veículos diferenciais, por exemplo, para que as trajetórias geradas sejam mais compatíveis com as possibilidades de movimento do veículo.
- *Utilizar como modelo uma dinâmica linear de segunda ordem.*
  A dinamica considerada ainda é linear de primeira ordem, em que a variável de controle é a velocidade. Mas, uma vez que também deseja-se o consenso sobre a velocidade, poder-se-ia utilizar a aceleração como variável de controle e consequentemente uma dinamica linear de segunda ordem. Decidiu-se por ater-se à dinâmica de primeira ordem pois assim foi feito por Ren e Sorensen[22], cujo resultado serve de base para este trabalho. Entretanto, há outros trabalhos que utilizam dinâmica de segunda ordem.
- *Utilizar para o controle de baixo nível uma trajetória interpolada.*
  O método proposto utiliza a saída do RHC para gerar um destino estático a cada iteração, que é então utilizado como referência pelo controle de baixo nível. Isso pode resultar em um comportamento que não corresponde ao desejado no movimento do robô. Por exemplo, se ele é rápido o suficiente, irá continuamente "andar e parar" ao invés de ter um movimento suave. Ter uma trajetória desejada e não um destino estático ajudaria a suavizar o movimento do veículo. Entretanto, o próprio método de controle de baixo nível teria de ser alterado.
- *Continuar o desenvolvimento do simulador* e *comparações com outros algoritmos.*

Entre as possibilidades de incremento do simulador, está a inclusão de outros algoritmos de consenso, em particular o de Ren e Sorensen[22], com o qual o método aqui proposto pode ser diretamente comparado. Além disso, vários aprimoramentos na interface gráfica podem ser feitos.

Referências

1 MURRAY, R. M. Recent research in cooperative control of multi-vehicle systems. **ASME Journal of Dynamic Systems, Measurement, and Control**, v. 129, n. 5, p. 571–583, Mai. 2007.

2 VERES, S. M.; MOLNAR, L.; LINCOLN, N. K.; MORICE, C. P. Autonomous vehicle control systems – a review of decision making. **IMechE Journal of Systems and Control**, p. 1–46, Abr. 2010.

3 LUCASA, N. P.; PANDYAA, A. K.; ELLISB, D. Review of multi-robot taxonomy, trends and applications for defense and space. **Proceedings of the SPIE Unmanned Systems Technology XIV**, v. 8387, Abr. 2012.

4 CAO, Y.; YU, W.; REN, W.; CHEN, G. An overview of recent progress in the study of distributed multi-agent coordination. **IEEE Transactions on Industrial Informatics**, v. 9, n. 1, p. 427–438, Fev. 2013.

5 ZHANG, Y.; MEHRJERDI, H. A survey on multiple unmanned vehicles formation control and coordination: normal and fault situations. **International Conference on Unmanned Aircraft Systems**, p. 1087–1096, Mai. 2013.

6 SCHARF, D. P.; HADAEGH, F. Y.; PLOEN, S. R. A survey of spacecraft formation flying guidance and control (Part II): Control. **Proceeding of the American Control Conference**, v. 4, p. 2976–2985, Jun. 2004.

7 NAVARRO, I.; MATíA, F. A survey of collective movement of mobile robots. **International Journal of Advanced Robotic**

Systems, v. 10, n. 73, 2013.

8 LEWIS, M. A.; TAN, K.-H. High precision formation control of mobile robots using virtual structures. **Autonomous Robots**, v. 4, n. 4, p. 387–403, Out. 1997.

9 REN, W.; BEARD, R. W.; ATKINS, E. M. Information consensus in multivehicle cooperative control. **IEEE Control Systems Magazine**, v. 27, n. 2, p. 71–82, Abr. 2007.

10 REN, W.; BEARD, R. W.; ATKINS, E. M. A survey of consensus problems in multi-agent coordination. **Proceedings of the American Control Conference**, v. 4, p. 1859–1864, Jun. 2005.

11 OLFATI-SABER, R.; FAX, J. A.; MURRAY, R. M. Consensus and cooperation in networked multi-agent systems. **Proceedings of the IEEE**, v. 95, n. 1, p. 215 – 233, Jan. 2007.

12 REN, W.; BEARD, R. W. Consensus seeking in multiagent systems under dynamically changing interaction topologies. **IEEE Transactions on Automatic Control**, v. 50, n. 5, p. 655–661, Mai. 2005.

13 ORDOÑEZ, B.; MORENO, U. F.; CERQUEIRA, J.; ALMEIDA, L. Generation of trajectories using predictive control for tracking consensus with sensing. **Procedia Computer Science**, v. 10, p. 1094–1099, Ago. 2012.

14 FERRARI-TRECATE, G.; GALBUSERA, L.; MARCIANDI, M. P. E.; SCATTOLINI, R. Model predictive control schemes for consensus in multi-agent systems with single- and double-integrator dynamics. **IEEE Transactions on Automatic Control**, v. 54, n. 11, p. 2560–2572, Nov. 2009.

15 GOWAL, S.; MARTINOLI, A. Real-Time optimization of trajectories that guarantee the rendezvous of mobile robots. **IEEE/RSJ International Conference on Intelligent Robots and Systems**, p. 3518–3525, Out. 2012.

16 CAMACHO, E. F.; ALBA, C. B. **Model Predictive Control**. 2ª Edição. [S.l.]: Springer, 2007. (Advanced Textbooks in Control and Signal Processing).

17 CAMPONOGARA, E.; JIA, D.; KROGH, B. H.; TALUKDAR, S. Distributed model predictive control. **IEEE Control Systems Magazine**, v. 22, n. 1, p. 44–52, Fev. 2002.

18 SCATTOLINI, R. Architectures for distributed and hierarchical model predictive control – a review. **Journal of Process Control**, v. 19, n. 5, p. 723–731, Mar. 2009.

19 CHRISTOFIDES, P. D.; SCATTOLINI, R.; PENA, D. M. de la; LIU, J. Distributed model predictive control: a tutorial review and future research directions. **Computers and Chemical Engineering**, v. 51, n. 5, p. 21–41, Jun. 2013.

20 DUNBAR, W. B.; MURRAY, R. M. Distributed receding horizon control for multi-vehicle formation stabilization. **Automatica**, v. 42, n. 4, p. 549–558, Abr. 2006.

21 KEVICZKY, T.; BORRELLI, F.; FREGENE, K.; GODBOLE, D.; BALAS, G. J. Decentralized receding horizon control and coordination of autonomous vehicle formations. **IEEE Transactions on Control Systems Technology**, v. 16, n. 1, p. 19–33, Jan. 2008.

22 REN, W.; SORENSEN, N. Distributed coordination architecture for multi-robot formation control. **Robotics and Autonomous Systems**, v. 56, n. 4, p. 324–333, Abr. 2008.

23 BOYD, S.; VANDENBERGHE, L. **Convex Optimization**. New York, NY, EUA: Cambridge University Press, 2004.

24 BURKARD, R.; DELL'AMICO, M.; MARTELLO, S. **Assignment Problems**. [S.l.]: Society for Industrial and Applied Mathematics, 2009.

25 GOLDFARB, D.; IDNANI, A. A numerically stable dual method for solving strictly convex quadratic programs. **Mathematical Programming**, v. 27, n. 1, p. 1–33, 1983.

26 GASPERO, L. D. **QuadProg++**. Programa de Código Aberto. Disponível em: <http://sourceforge.net/projects/quadprog/>, último acesso em Julho de 2014.

27 PILGRIM, R. A. **Munkres' Assignment Algorithm**. CSC 445 Algorithms Lecture Notes, Murray State University. Disponível em:

<http://csclab.murraystate.edu/bob.pilgrim/445/munkres.html>, último acesso em Julho de 2014.

28 CORREIA, F. L. de B.; ORDOÑEZ, B.; CERQUEIRA, J.; ALMEIDA, L.; MORENO, U. F. Controle de Veículos Autônomos em Formação com Seguimento de Referência Utilizando Consenso e RHC. **Congresso Brasileiro de Automática**, 2014.

Apêndice A

Arquivo de Entrada do Simulador

Este apêndice contém um exemplo de arquivo de entrada utilizado no simulador desenvolvido. Este arquivo corresponde ao exemplo da Seção 3.3. Linhas que iniciam com # são comentários.

```
1
2  # simulation input file model
3
4  [simulation-timing]
5  low-level-time-step     = 0.05;
6  high-level-step-number  = 10;
7  integration-step-number = 10;
8  simulation-end-time     = 60;
9
10 [reference-specification]
11 trajectory-id           = 1;
12 trajectory-scale0       = 12.0;
13 trajectory-scale1       = 10.0;
14
15 formation-id            = 5;
16 formation-scale0        = 1.5;
17
18 [simulation-options]
19 use-communication-radius        = 0;
20 use-security-radius             = 1;
21 use-robot-type                  = 0;
22 use-variable-adjacency-matrix   = 0;
23 use-dynamic-allocation          = 1;
24
25 # penalize velocity                 3
26 # penalize velocity variation       2
27 # use last trajectory               1
28 # do nothing (velocities = 0)       0
29 no-communication-behavior       = 3;
30
31 # formation & trajectory            0
32 # rendez-vous                       1
```

```
consensus-algorithm             = 0;

# centralized hungarian alg.        0
allocation-algorithm            = 0;

[consensus-parameters]
prediction-horizon  = 2;
alpha-i0            = 1;
alpha-ij0           = 10;
alpha-ir0           = 100;
beta-x              = 4;
# beta-v = 0 to avoid instability
beta-v              = 1;
gamma-x             = 2;
gamma-v             = 1;
lambda              = 1;
lambda-c            = 1;

[robots]
group-size = 3;

adjacency-r =
1  0  0;

eta-r =
0  0  0;

adjacency-j =
0  1  0
1  0  1
1  0  0;

# this adjacency matrix results in instability
# adjacency-j =
# 0  1  0
# 1  0  1
# 0  1  0;

eta-j =
0  0  0
0  0  0
0  0  0;

xi-0 =
 0.0   -1.0    0.5
 0.0    1.0    1.5
-1.0    0.0   -1.0;

vi-0 =
 0.5   -0.1    0.1
-0.2   -0.2   -0.2
 0.3   -0.4    0.3;
```

```
85
86 xic-0 =
87 1.0    1.0    0.5
88 -1.0   -1.0   -0.2
89 1.0    0.0    0.3;
90
91 vic-0 =
92 0  0  0
93 0  0  0
94 0  0  0;
95
96 v-max =
97 1.0   1.0   1.0
98 1.0   1.0   1.0
99 1.0   1.0   1.0;
100
101 v-min =
102 -1.0   -1.0   -1.0
103 -1.0   -1.0   -1.0
104 -1.0   -1.0   -1.0;
105
106 robot-radii =
107 0.3   0.6   30
108 0.3   0.6   30
109 0.3   0.6   30;
110
111 delta = 1.5;
112
113 low-level-robot-type =
114 # omni linear  0
115 # diff tng_0   1
116 # diff tng_1   2
117 1 0 1;
118
119 low-level-control-parameters =
120 # v^x_max  v^\theta_max  \theta_in  \theta_ex  x_in  x_ex
121       0.5           0.7        0.1   0.5        0.2   0.5
122
123       0.5           0.7        0.1   0.5        0.2   0.5;
124
125 [using-external-robots]
126 external-robots =
127 1  0  0  0  1  0;
128
129 external-names =
130 p3at  p3dx;
```