Impacto da Mobilidade sobre o RA-TDMA

Daniel Leal
up201305517@fe.up.pt

Rodrigo Passos
up201303434@fe.up.pt

***Abstract*— Neste artigo é verificado o impacto da mobilidade de um grupo de robots que comunicam entre si numa rede ad-hoc, em que cada um representa um nó. A informação da posição de cada robot é comunicada aos robôs que estão no seu alcance até que toda a rede saiba a posição de cada um dos nós. Serão testados diferentes parâmetros de mobilidade, como velocidade e alcance, para verificar o impacto que esses têm na disseminação da rede. Os resultados mostram que variando o alcance do robot é possível uma obter consistência na rede.**

Keywords*— *TDMA, partilha informação, distância, rede ad hoc, disseminação

I. INTRODUÇÃO

A coordenação e cooperação entre robôs é, nos dias de hoje, fundamental em sistemas distribuídos, uma vez que decorrem diversos benefícios do paralelismo entre unidades móveis. A título de exemplo, os carros autónomos podem, num futuro próximo, através de um sistema baseado ao que é desenvolvido neste trabalho, ter um controlo automático de velocidade e travagem, consoante a informação que tiverem ao seu redor (na sua rede). Na verdade, relativamente à coordenação entre robôs móveis a distância e a partilha de informação têm um papel relevante. Nesse sentido, este trabalho tem como principal intuito a medição de distâncias entre unidades móveis de robôs e a partilha de informação e conhecimento entre eles.

II. DEFINIÇÃO DO PROBLEMA

Neste trabalho sobre o Impacto da Mobilidade sobre o RA-TDMA foram apresentados inúmeros desafios. Na verdade, para este projeto são necessários um conjunto de robôs que devem começar com uma posição aleatória, bidimensional e terem um padrão de movimento bem definido. Como já foi referido, o fator distância é importante para a coordenação entre robôs. Assim, cada robô deve estabelecer uma comunicação com outro, caso se encontrem no raio de alcance um do outro. Além disso, a partilha de informação é um objetivo crucial no desenvolvimento do trabalho, uma vez que cada robô deve conhecer a rede à sua volta. Por fim, o principal desafio deste trabalho consiste em perceber e analisar quanto tempo é que a rede demora a convergir sempre que existem novas alterações e atualizações (variações no alcance, raio e a velocidade).

III. DESCRIÇÃO DA SOLUÇÃO

Numa fase inicial e de gestão do projeto, o grupo decidiu tomar algumas decisões em relação ao modo como seria realizado este trabalho. Na sequência dessa reflexão, definiu-se que teríamos um conjunto de 10 robôs e que cada um deles teria um padrão circular de movimento numa área de 10x10 metros, que na simulação criada no computador, foi tido em conta que essa área seria de 100x100 pixéis, de forma a criar uma interface gráfica onde fosse possível verificar a movimentação dos robôs. De seguida, determinou-se que haveria um algoritmo de verificação de alcance do robô. Por fim, a fase de partilha de informação: cada robô cria um vetor com as ligações estabelecidas (“0” e “1”) consoante o conhecimento de um robô da rede/vizinhança e depois de todos os robôs conhecerem a sua vizinhança é criada uma matriz com as ligações estabelecidas de todos os nós da rede, de forma a se saber a posição absoluta de todos os robôs. A partir daqui, será explicado com maior detalhe a abordagem e as soluções implementadas.

A. *Classe Robô*

A classe Robot é uma *thread* que possui as variadas propriedades de um robô para o seu movimento circular, que é um elemento fundamental no progresso deste projeto. As principais características desta classe são:

- ID – número identificador do robô;
- X – Posição do robô relativamente ao eixo xx;
- Y – Posição do robô relativamente ao eixo yy;
- Raio – raio de movimento de cada um dos robôs.

É nesta classe que são feitos os cálculos para a execução do padrão de movimento do robô. Esta operação é feita com um período de 100ms, período esse que corresponde à periocidade da atualização da *thread*. O robô irá descrever um círculo baseado nos parâmetros que são indicados por nós no momento da criação da *thread*, como o seu centro (baseado nas coordenadas x e y) e raio. A velocidade linear com que o robô descreve o circulo é definida inicialmente e pode ser alterada, tornando o funcionamento da simulação do robô o mais real possível. A velocidade pode ser alterada para ser possível observar qual o seu impacto na disseminação da rede. O cálculo da posição atual do robô é efetuado com base na posição anterior do mesmo somada ao ângulo descrito no período de 100ms. Na figura seguinte é possível ver a interface que foi criada para representar os robots numa área quadrada. O círculo interior representa o robô que se está a mover em círculo, como foi explicado, e o círculo exterior representa o alcance desse mesmo robô. De seguida irá ser explicado para que serve esse alcance.

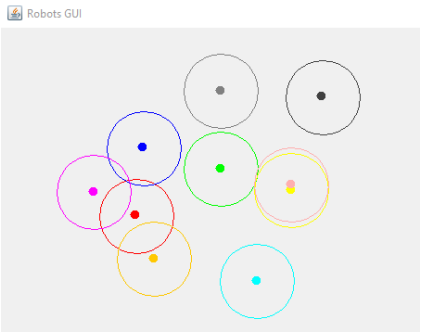

Figura 1: Representação dos robôs

B. *Algoritmo de Alcance*

A distância é essencial para a coordenação entre robôs. Daí que este algoritmo de alcance seja imprescindível para o projeto. Na realidade este algoritmo é utilizado para o robô descobrir quem são os seus vizinhos até um determinado alcance. Assim, este método consiste numa verificação contínua se algum dos robôs está dentro do nosso campo de "visão". Caso haja um novo robô dentro do alcance, então a matriz deve ser atualizada, algo explicado no ponto seguinte.

C. *Protocolo UDP*

Para criar a rede *ad hoc* foi considerado que cada nó era um robô, e que cada robô seria um cliente, e que todos os clientes comunicariam entre si e enviariam informação para um servidor que era responsável por fazer a gestão global do sistema. Para estabelecer todas essas comunicações foi utilizado o protocolo UDP. Apesar de o protocolo UDP não ter um mecanismo de verificação de entrega de mensagens, para o nosso caso, que queríamos um fluxo de dados em tempo real para propagar as mensagens na rede o mais rápido possível, este protocolo é uma boa escolha porque é mais rápido o protocolo TCP. Caso alguma mensagem não seja entregue, o pior que pode acontecer é a rede demorar mais tempo a chegar a um consenso, isto é, todos terem a mesma informação, e como esses casos serão esporádicos, a rapidez do protocolo compensa essa desvantagem.

1. Cliente

Cada cliente, que é representado pela classe robô que foi explicada anteriormente, ou seja, cada cliente é uma *thread* que simula o movimento do robô, faz *broadcast* da sua posição absoluta para todos os outros clientes, sempre a mesma é atualizada. Sempre que é feito um *broadcast* cada nó da rede irá então correr o algoritmo de alcance para verificar quais os nós (vizinhos) que se encontram no alcance. Sempre que esta verificação é feita é criado um *socket* para cada um dos nós que se encontra dentro do alcance, estabelecendo assim um canal de comunicação entre nós que se encontram na vizinhança. É também atualizado o vetor das ligações estabelecidas com outros nós, que reflete quais os *sockets* que estão ativos ou não entre o robô daquele nó e todos os outros pertencentes à rede. Como todos os clientes fazem isto à cadência de atualização da sua posição, é sempre possível saber quais as ligações estabelecidas por qualquer robô, e assim ter conhecimento da posição relativa dos robôs na rede.

O vetor das ligações estabelecidas é enviado para o servidor que faz monitorização da rede.

2. Servidor

O servidor de monitorização da rede tem um mecanismo de *ping* aos clientes, que é nada mais nada menos que uma mensagem predefinida enviada em modo *broadcast*. O período de envio desta mensagem pode ser definido pelo utilizador, sendo que nós usamos períodos como 100ms e 200ms. Sempre que os clientes recebem esta mensagem, que é facilmente identificada pelo seu conteúdo (uma *string* a dizer *ping*), irão enviar os seus vetores. Como a comunicação implementada é assíncrona, ou seja, quando cada cliente recebe o *ping* irá enviar imediatamente a mensagem com o seu vetor, foi necessário criar um mecanismo de receção de mensagens do lado do servidor. Assim, é sempre verificado qual é o ID do Robô que enviou a mensagem e é feita uma ordenação das mensagens para nunca haver problemas com a informação recebida.

Assim, após receber todos os vetores, o servidor irá criar uma matriz global da vizinhança de cada nó da rede, e irá fazer *broadcast* duma mensagem que contém essa matriz para todos os robôs terem sempre a matriz global atualizada e saberem a posição de todos os nós da rede.

O servidor também é responsável por efetuar a gestão de resultados, verificando qual é o tempo que a rede demorou a convergir, isto é, quanto tempo demorou desde que houve uma alteração na rede até todos os nós saberem dessa alteração. Também verifica qual o tempo máximo que a rede se manteve estável, isto é, sem ter havido uma alteração, entre outros dados importantes.

D. *Partilha de informação*

Como já foi dito nos pontos anteriores, cada vez que há uma mudança na topologia da rede (movimento dos robôs), cada robô deve enviar uma mensagem em modo *broadcast* para os outros robôs a indicar a sua posição absoluta. A mensagem enviada tem o seguinte formato:

ID Robô	Sequence Number	Posição x	Posição y

Figura 2: Vetor da posição absoluta

A mensagem contém um parâmetro que é o "Sequence Number". Este número é incrementado por todos os clientes sempre que há uma atualização da posição do robô, permitindo ao cliente que recebe a mensagem saber se a mesma é atual ou se já é uma mensagem antiga, comparando com o seu "Sequence Number". Se já for uma mensagem antiga a mesma não será tida em conta e o cliente irá esperar

por uma nova mensagem para atualizar o seu vetor de ligações.

Cada vez que um robô recebe as posições dos outros robôs da rede irá correr o algoritmo de alcance, e, se na sua vizinhança encontrar um novo robô, o seu vetor das ligações (*sockets*) será atualizado com o valor "1" na coluna referente ao robô que encontra (i.e. caso seja encontrado o robô 2, deve ser atualizada a coluna 2). Na imagem seguinte é possível ver como é que o vetor é constituído.

	Robot 1	**…**	**Robot n**
ID Robot	Ligação On (1)/ Off (0)	Ligação On (1)/ Off (0)	Ligação On (1)/ Off (0)

Figura 3: Vetor de ligações estabelecidas.

Este vetor será enviado por mensagem para o servidor, como já foi dito, e de seguida o servidor irá atualizar a matriz global de acordo com as modificações que ocorreram nas ligações. A matriz tem o seguinte formato:

	Robot 1	**…**	**Robot n**
Robot 1	Ligação On (1)/ Off (0)	Ligação On (1)/ Off (0)	Ligação On (1)/ Off (0)
…	Ligação On (1)/ Off (0)	Ligação On (1)/ Off (0)	Ligação On (1)/ Off (0)
Robot n	Ligação On (1)/ Off (0)	Ligação On (1)/ Off (0)	Ligação On (1)/ Off (0)

Figura 4: Matriz de ligações estabelecidas

IV. RESULTADOS

Nesta parte do nosso relatório são analisados os diversos impactos que a mobilidade dos robôs tem na disseminação da rede. Assim, foram modificados alguns parâmetros dos robôs como alcance, velocidade e posição para verificar o impacto que os mesmos causavam em vários indicadores da performance da rede.

O tempo máximo convergência corresponde ao tempo máximo que os nós demoram a reconhecer uma alteração na rede. O tempo máximo de consistência corresponde ao tempo que a rede permanece sem ser feita uma alteração à matriz mesmo com alterações na topologia da rede. A percentagem de consistência corresponde ao número de vezes que a rede permaneceu igual (não houve alterações na vizinhança) em função do número total de vezes que houve alteração na posição dos robôs. Esta percentagem também dá para avaliar o número de inconsistências na rede. De salientar que os resultados apresentados nos três gráficos seguintes utilizaram posições iniciais dos robôs pré-definidas e não aleatórias para ser possível observar o impacto de diferentes variáveis. Para cada resultado apresentado foram feitas 500 movimentações dos robôs na rede, para os resultados serem fiáveis.

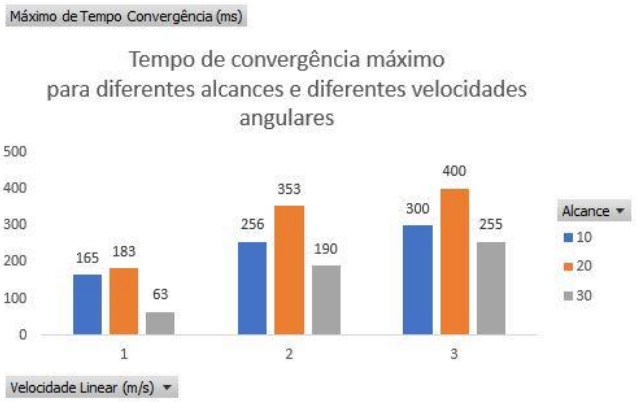

Figura 5: Tempo de convergência máximo para diferentes alcances e velocidades lineares

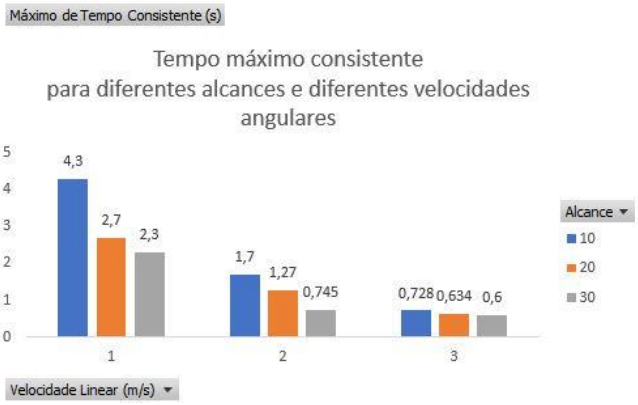

Figura 6: Tempo máximo que a rede está consistente para diferentes alcances e velocidades lineares

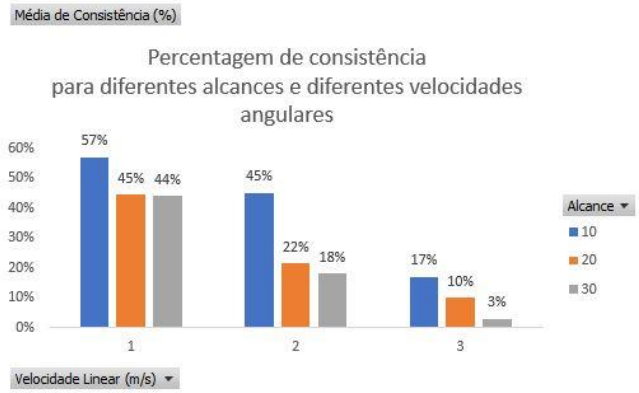

Figura 7: Percentagem de consistência da rede para diferentes alcances e velocidades lineares

Com estes gráficos é possível observar que a consistência da rede diminui com o aumento do alcance e com o aumento da velocidade linear do robô. O aumento da velocidade linear do robô causa alterações mais frequentes na vizinhança dos nós, o que faz com que a rede varie mais vezes e a sua consistência diminua, assim como o tempo máximo que a mesma se mantém consistente.

O aumento do alcance do robô piora a consistência da rede porque haverá certos momentos em que nós mais longínquos serão considerados vizinhança e, no instante seguinte já não serão, o que provoca alterações na matriz das ligações, mas não de forma sistemática, aumentado assim as inconsistências na rede.

V. Análise Crítica e Conclusão

Nesta última secção do relatório deve ser feita uma reflexão sobre os objetivos alcançados e o que devia ser feito para se melhorar o projeto. Na verdade, todas as metas do trabalho foram realizadas com sucesso:

- Robôs com posição bidimensional e com padrões de movimento definidos;
- Comunicação entre robots caso estejam num determinado raio de alcance;
- Matriz que indica as ligações de cada robot;
- Impacto da mobilidade sobre o RA TDMA e análise de resultado.

No entanto, numa fase futura poderiam ser acrescentadas algumas melhorias que podem ser interessantes e pertinentes:

- Controlo básico para evitar obstáculos;
- Forma de controlar a velocidade a que a rede muda;

Através da investigação, estudo e realização deste projeto foi possível não só adquirir uma compreensão da dificuldade associada à implementação dos sistemas distribuídos em tempo real, mas também refletir em relação ao impacto e importância que os sistemas distribuídos têm para a sociedade nos dias de hoje.

A implementação do trabalho foi realizada com sucesso e foi possível recolher resultados e perceber o impacto da mobilidade no RA TDMA para diversos casos, tal como foi possível ver na secção Resultados.

Para finalizar, perante as circunstâncias temporais limitadas, acreditamos que o trabalho apresentado está bastante razoável, dentro dos objetivos definidos, e os dois elementos contribuíram de igual modo para a resolução deste projeto: enquanto que um desenvolveu a parte do servidor e do GUI, o outro desenvolveu a parte dedicada ao cliente (robô).

References

[1] Almeida, L., Oliveira, L., and Santos,F. , “A Loose Synchronisation Protocol for Managing RF Ranging in Mobile Ad-Hoc Networks”.

[2] Oliveira, L.F., “ Communications and Localisation for Cooperating Autonomous Mobile Robots”,20