UMA ARQUITETURA CENTRALIZADA PARA A NAVEGAÇÃO AUTÔNOMA DE MÚLTIPLOS ROBÔS SPHERO

Renan Taizo Nakashima*, Bruno Campera*, João Paulo Lima Silva de Almeida†, Lucia Valeria Ramos de Arruda*, Flávio Neves Junior*

* *Universidade Tecnológica Federal do Paraná*
Curitiba, Paraná, Brasil

† *Instituto Federal do Paraná*
Jacarezinho, Paraná, Brasil

Emails: renan.taizo@gmail.com, brunoc@alunos.utfpr.edu.br, jplimasa@gmail.com, lvrarruda@utfpr.edu.br, neves@utfpr.edu.br

Abstract— This work proposes a centralized architecture for autonomous navigation of a Multi Robot System (MRS), which uses simple robots that cooperate to reach a common goal, bringing efficiency to the system. The developed MRS consists of Sphero-type mobile robots, and the developed planning and control algorithms allow to circumvent the sensing and communication limitations of such robots. The proposed architecture is validated through two real experiments of search and cooperative collection of targets with well-known positions and of deviation from obstacles with unknown positions. The obtained results demonstrate that the architecture works well to small systems with potential application in swarm robotics.

Keywords— Multi-Robot Systems, Autonomous Navigation, Path Planning, Fuzzy Control

Resumo— Este trabalho propõe uma arquitetura centralizada para a navegação autônoma em um Sistema Multirrobô (SMR), que usa robôs simples, atuando cooperativamente para atingir um objetivo comum, aumentando a eficiência do sistema. O SMR desenvolvido é composto por robôs móveis do tipo Sphero, e os algoritmos desenvolvidos de planejamento e controle permitem contornar as limitações de sensoreamento e comunicação destes robôs. A arquitetura proposta é validada através de dois experimentos reais de busca e coleta cooperativa de alvos com posições conhecidas e de desvio de obstáculos com posições desconhecidas. Os resultados obtidos demonstram que a arquitetura é válida para a aplicação em pequenos sistemas, com potencial aplicação em robótica de enxame.

Palavras-chave— Sistemas Multirrobô, Navegação Autônoma, Planejamento de Trajetória, Controle *Fuzzy*

1 Introdução

Atualmente, a área de pesquisa de navegação autônoma de robôs móveis para sistemas multi-robôs (SMR) e em especial para enxame de robôs (*Swarm Robotics)* tem conhecido grandes avanços. Uma das abordagens para se atingir a coordenação e cooperação entre robôs nesses sistemas têm sua inspiração no comportamento coletivo de animais sociais, como abelhas ou formigas, que apesar de poucas capacidades cognitivas são capazes de cooperar e trocar informações entre si a fim de atingir um objetivo comum ao enxame. O uso de múltiplos robôs com características simples mas capazes de realizar tarefas complexas busca emular este tipo de comportamento, tornando a execução das tarefas mais eficiente e menos custosas quando comparado a utilização de um único robô dotado de vários sensores e equipamentos (de Sá et al., 2017).

Os SMR com vários robôs simples tem sido usado em aplicações críticas, tais como: detecção de radioatividade; busca e destruição de minas terrestres; busca de sobreviventes em cenários de desastre, entre outros (Mendonça et al., 2017). Isto porque estes sistemas são dotados de escalabilidade e redundância intrínseca, sendo possível adicionar ou remover robôs do sistema, tornando-o robusto a falhas. Além destas características, um SMR é considerado mais adaptável a mudanças do ambiente, devido à maior área de cobertura de sensoreamento e capacidade de troca de informações entre os robôs (Oh et al., 2017).

Para um SMR, em que cada robô tem capacidade de processamento e sensoreamento individual limitados, a autonomia de decisão (qual tarefa realizar) e de navegação (qual trajetória seguir) torna-se um requisito crítico. Assim, dois elementos são necessários para garantir a navegação autônoma dos agentes. O primeiro diz respeito a um algoritmo de planejamento que precede a movimentação, que determina qual o próximo objetivo a ser atingido (tarefa) (Ladosz et al., 2017; Mac et al., 2016) e como chegar até ele (trajetória). O segundo diz respeito a sistemas de controle de posição e velocidade, para que o robô consiga realizar a trajetória planejada e se deslocar de forma segura (desviando-se de obstáculos) ao objetivo determinado (Abadi and Khooban, 2015; Castillo et al., 2015).

Neste contexto, o presente trabalho desenvolve uma arquitetura centralizada para a navegação autônoma de múltiplos robôs Sphero. A arquitetura proposta permite a implementação e teste de um sistema de navegação autônomo para SMR com capacidade limitada de recursos. Os al-

goritmos de planejamento de trajetórias e de controle de navegação e desvio de obstáculo de cada robô são implementados em MATLAB. A troca de informações entre o robô Sphero e o sistema de navegação autônoma proposto é baseado na plataforma *Robot Operating System* (ROS) e os controladores para navegação e desvio de obstáculos, apresentados neste trabalho, são do tipo *Fuzzy*. Dois cenários reais de busca e coleta cooperativa de alvos com posições previamente conhecidas e obstáculos com posições desconhecidas são apresentados para validar a arquitetura proposta.

2 Arquitetura Centralizada do Sistema Multirrobô proposto

O robô Sphero (Figura 1) utilizado neste trabalho é um robô móvel de acionamento diferencial, encapsulado por uma casca esférica de material plástico resistente ao impacto, com um led RGB no seu interior para diferenciação por cor e orientação, e dispõe de comunicação *Bluetooth* (Berthold and Hafner, 2015). O robô é um sistema fechado, e apesar de existir uma API para envio e recebimento de comandos, não dispõe de uma forma de reprogramar o seu hardware.

Como discutido acima, cada robô de um SMR deve possuir as seguintes características: capacidade de comunicação com os demais robôs, percepção individual do ambiente através de sensoreamento e capacidade de processamento para cálculos de planejamento e navegação (Kisdi and Tatnall, 2011). No entanto, os robôs Sphero não dispõem de nenhuma dessas características. A arquitetura proposta busca contornar a ausência desses elementos através da utilização de um sistema de navegação autônomo executando em um computador remoto.

Figura 1: Robô móvel Sphero

Considerando as restrições do robô, o sistema de navegação desenvolvido assegura que o robô é capaz de movimentar-se no ambiente em que está inserido, realizando a coleta cooperativa de alvos com posições previamente conhecidas, enquanto desvia-se de obstáculos com posições desconhecidas. Para isto, a arquitetura proposta foi dividida em quatro módulos (Percepção; Planejamento e Distribuição; Navegação e Controle; Acionamento e Execução) que comunicam-se entre si conforme ilustrado na Figura 2.

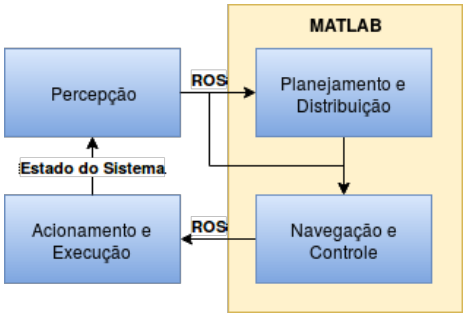


Figura 2: Visão geral da arquitetura

Esta comunicação entre os módulos ocorre através da troca de informações com publicações em tópicos próprios do *Robot Operating System* (ROS). O ROS é um *framework* contendo uma coleção de ferramentas, bibliotecas e convenções que possibilitam o desenvolvimento de aplicações robustas e complexas em robótica (Quigley et al., 2009). A comunicação através do ROS é baseada no paradigma *publisher-subscriber* em que os publicadores publicam mensagens (salvam dados) em uma fila de mensagens (tópico), e todos os interessados podem acessar a mensagem ao subscreverem-se ao canal.

A execução do sistema proposto inicia a partir das informações fornecidas pelo Módulo de Percepção aos módulos de Planejamento e Distribuição, e de Navegação e Controle. Este módulo disponibiliza a localização (coordenadas cartesianas) dos robôs, alvos (ponto de passagem para realização de tarefa) e obstáculos presentes no ambiente. Dessa forma, não é necessário utilizar técnicas de localização a partir da odometria dos robôs durante a execução, pois obtém-se o posicionamento global de cada agente, alvo e obstáculo no cenário, desde que equipados com marcadores ARTag, os quais serão descritos nos parágrafos posteriores.

O módulo de Planejamento e Distribuição responsabiliza-se por definir os objetivos do sistema, informando aos robôs quais os alvos individuais que cada um deverá coletar. Para tal, baseia-se nas informações de posicionamento fornecidas pelo Módulo de Percepção. Deve-se salientar que este algoritmo é executado somente uma única vez (offline) no início da execução do sistema. Para o planejamento são consideradas somente as posições dos alvos, enquanto a posição dos obstáculos é tida como desconhecida durante o cálculo de trajetória. Dessa forma, os obstáculos somente serão percebidos pelos robôs quando estes se encontram suficientemente próximos dos obstáculos. A percepção dos obstáculos é reali-

zada *on-line* e por meio de uma área de sensoreamento simulada no MATLAB.

Combinando as informações recebidas pelos módulos de Percepção e Planejamento e Distribuição, o módulo de Navegação e Controle é responsável por calcular o sinal de controle que será aplicado em cada robô para levá-lo em direção aos alvos a ele atribuídos e/ou para desvio de obstáculos. Para isto, este módulo calcula a distância entre o robô e seu alvo e/ou obstáculo, a partir das coordenadas recebidas pelo módulo de Percepção e das características individuais dos robôs, como por exemplo o *range* de detecção do sensor, ângulo de abertura ou ângulo de varredura, entre outros. Por sua vez, o Módulo de Acionamento e Execução é responsável pela execução das ações de controle calculadas pelo módulo de Navegação e Controle, realizando de fato a atuação dos robôs em busca dos alvos presentes no ambiente, alterando portanto o estado do sistema.

O ciclo de execução ilustrado na Figura 2 repete-se indefinidamente até que o sistema atinja o seu objetivo coletivo, ou seja, até que todos os robôs coletem todos os alvos a eles atribuídos e parem em uma posição final estabelecida a priori.

O SMR desenvolvido é composto pelos seguintes elementos: um computador central para a execução do ROS e do MATLAB; os robôs Sphero identificados por marcadores do tipo ARTag (Fiala, 2004), conforme mostrado na figura 3; uma câmera fixa conectada ao computador central, posicionada adequadamente no topo de uma área delimitada que corresponde ao ambiente de navegação e contem os alvos (pontos de passagem) que os robôs devem atingir e coletar, e os obstáculos que os robôs devem evitar. Os obstáculos e alvos são também unicamente identificados por ARTag. A seguir cada um dos módulos compondo o sistema de navegação autônomo desenvolvido é detalhado.

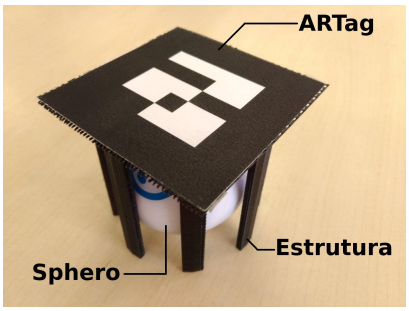


Figura 3: Estrutura de suporte a tag

O módulo de percepção é composto pela câmera e pelo pacote `ar_track_alvar` disponível no ROS que, a partir da captura da imagem digital da câmera posicionada sobre o ambiente, realiza o processamento e detecção das ARTags presentes na imagem. Em seguida, as coordenadas de cada marcador (ARTags detectadas) são continuamente publicadas em um tópico ROS.

A partir deste ponto, uma *procedure* em MATLAB lê os tópicos do ROS e armazena três listas que relacionam os identificadores únicos dos marcadores ARTag com seus papéis no sistema (alvo, robô ou obstáculo), sendo que um marcador pode assumir somente um dentre estes três papeis. Essas listas são disponibilizadas aos outros módulos.

O Módulo de Planejamento e Distribuição leva em consideração as informações recebidas do Módulo de Percepção e seus conhecimentos a priori sobre o ambiente e objetivos do SMR. A partir das coordenadas dos robôs e dos alvos retiradas das listas de ARTags, o módulo executa o algoritmo de distribuição de alvos e cálculo de trajetória de forma conjunta para todos os robôs. O planejamento de trajetória é modelado a partir da compreensão do problema de distribuição de alvos como um caso particular do problema de Problema do Caixeiro Viajante Múltiplo (MTSP), em que existem múltiplos caixeiros (robôs) que devem visitar várias cidades (alvos) no menor percurso possível (trajetórias planejadas). O MTSP assim formulado é solucionado por um algoritmo genético que tem como saída uma lista de rotas passando por subconjunto de alvos, que será atribuída a cada robô, de modo que o percurso total realizado pelo conjunto de robôs seja o menor possível. A implementação utilizada neste trabalho é a mesma encontrada em (Almeida et al., 2017) e não será detalhada por falta de espaço.

Por fim, este módulo responsabiliza-se ainda por manter um registro dos alvos que já foram coletados, informando-os aos outros módulos que esses alvos agora são considerados obstáculos que devem ser evitados. Uma vez obtida a lista dos alvos para cada robô, o primeiro alvo da lista de cada robô que deverá ser coletado é informado ao Módulo de Navegação e Controle.

O Módulo de Navegação e Controle executa de forma paralela, para cada robô, um controlador *fuzzy* de posição e velocidade. Apesar do controlador ser o mesmo para cada robô, sua operação depende das coordenadas do alvo que deve ser coletado por cada um deles. Dada uma lista ordenada de alvos a coletar (coordenadas) e a coordenada atual do robô, esse módulo gera as ações de controle necessárias para deslocar o robô até a coordenada do próximo alvo não coletado. Durante o deslocamento, caso o robô aproxime-se de um obstáculo, este é detectado através de um sensor totalmente simulado, que é considerado em cada robô como um sensor de proximidade de formato circular, que envolve todo o robô. Após efetuar o desvio, o robô retorna à trajetória previamente planejada. Uma vez que um alvo foi coletado, este também passa a ser considerado um obstáculo que deve ser evitado por todos os robôs. Detalhes sobre o sensor simulado serão descritos na Seção 3.

O Módulo de Acionamento e Execução é res-

ponsável pela comunicação com o robô físico, propriamente dito. As ações de controle determinadas pelo Módulo de Navegação são comunicadas aos robôs via *Bluetooth* através do ROS, intermediados pelo pacote ROS `sphero_swarm`. Os robôs deverão executar o acionamento de seus atuadores, deslocando-se no ambiente, de acordo com as ações de controle calculadas. Esse pacote ROS conecta-se com robôs Sphero previamente registrados através de seu *Bluetooth Address*, criando diversos tópicos ROS individuais para cada robô, dentre os quais o tópico `/cmd_vel`. A partir das mensagens publicadas neste tópico, os robôs acionam seus atuadores para deslocamento no ambiente de acordo com a trajetória planejada.

3 Sistema Autônomo de Navegação

O sistema de navegação autônomo desenvolvido pressupõe um controlador para cada robô do SMR. Isto é implementado através da execução em paralelo da *procedure* de controle tantas vezes quantas sejam o número de robôs do SMR. Cada chamada executa considerando os dados de seu robô, os quais são recebidos do módulo de Planejamento e Distribuição. Com isto, assegura-se que cada robô possui o seu controlador exclusivo para navegação e desvio de obstáculo. Cada controlador é composto por três elementos: um controlador *fuzzy* de busca de alvo, um controlador (*procedure*) de desvio de obstáculo, e um interpolador *fuzzy*, responsável por ponderar as ações de controle de busca e de desvio, e calcular a ação de controle final a ser efetivamente aplicada ao robô. A Figura 4 ilustra a arquitetura utilizada para implementar o controle de navegação e desvio de obstáculos, onde:

- e_x e e_y são respectivamente as distâncias (erros) entre as coordenadas x e y da posição atual do robô e de seu alvo assinalado;
- I_s é a leitura do sensor do robô;
- x_o e y_o são respectivamente as coordenadas x e y do obstáculo no *range* do sensor do robô;
- τ é o fator de ponderação;
- $d'_{x,y} \in R^2$ é a ação de controle de desvio;
- $b'_{x,y} \in R^2$ é a ação de controle de busca de alvo;
- $ac'_{x,y} \in R^2$ é o resultado da ponderação entre as ações de controle de busca e desvio.

De acordo com a Figura 4, o controle de busca de alvo é implementado através de um controlador *fuzzy* com duas entradas e duas saídas, definidas no intervalo $[-1, 1]$. As funções de pertinência utilizadas para as entradas e saídas do controlador de busca são idênticas. Foram definidos 7

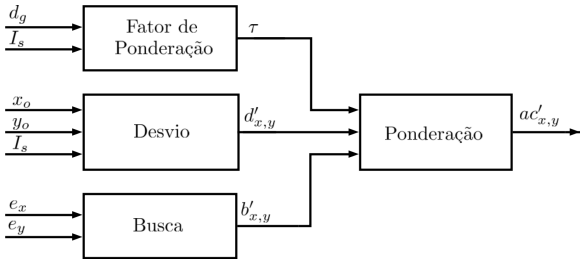


Figura 4: Arquitetura de controle da navegação

conjuntos *fuzzy*, representados pelas variáveis linguísticas Negativo Grande (NG), Negativo Médio (NM), Negativo Pequeno (NP), Zero (ZE), Positivo Pequeno (PP), Positivo Médio (PM) e Positivo Grande (PG). A Figura 5 ilustra os conjuntos *fuzzy* descrevendo estas variáveis linguísticas. A Tabela 1 contém a base de regras utilizada no controle de busca.

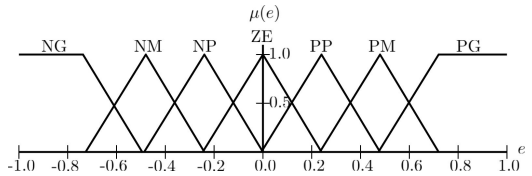


Figura 5: Funções de pertinência do controlador de busca de alvos

Por sua vez, o cálculo da ação de controle de desvio leva em consideração a intensidade de leitura do sensor de proximidade e o posicionamento relativo do robô, do obstáculo e do alvo atual que o robô busca atingir, conforme verifica-se na Figura 4. A saída do controle de desvio é um vetor com componentes no intervalo $[-1, 1]$, calculadas de acordo com a Figura 6, onde:

- x'_r e y'_r representam o plano de referência do robô;
- x'_o e y'_o representam as coordenadas do obstáculo no plano de referência do robô;
- x'_g e y'_g representam as coordenadas dos alvos no plano de referência do robô;
- ρ_o e ρ_g representam a distância entre o robô e alvo e entre o robô e obstáculo;
- α_1 e α_2 representam o ângulo formado entre o eixo x'_r e a posição do alvo e do obstáculo, respectivamente;
- $d*_{x,y}$ e $d'_{x,y}$ representam as ações de desvio inicial e rotacionada, respectivamente.
- ϕ é o ângulo de rotação do vetor de desvio inicial

Apenas o obstáculo mais próximo do robô é considerado. Quando um obstáculo é detectado

Tabela 1: Base de regras *fuzzy* do controle de busca

e_x/e_y	NG	NM	NP	ZE	PP	PM	PG
NG	PG/PG	PM/PG	PP/PG	ZE/PG	NP/PG	NM/PG	NG/PG
NM	PG/PM	PM/PM	PP/PM	ZE/PM	NP/PM	NM/PM	NG/PM
NP	PG/PP	PM/PP	PP/PP	ZE/PP	NP/PP	NM/PP	NG/PP
ZE	PG/ZE	PM/ZE	PP/ZE	ZE/ZE	NP/ZE	NM/ZE	NG/ZE
PP	PG/NP	PM/NP	PP/NP	ZE/NP	NP/NP	NM/NP	NG/NP
PM	PG/NM	PM/NM	PP/NM	ZE/NM	NP/NM	NM/NM	NG/NM
PG	PG/NG	PM/NG	PP/NG	ZE/NG	NP/NG	NM/NG	NG/NG

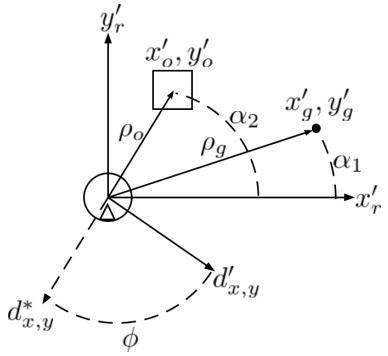


Figura 6: Representação da ação de desvio de obstáculo

no *range* do sensor do robô, um vetor $d^*_{x,y}$ na direção oposta àquela formada pelo centro do robô e o centro do obstáculo, é calculado de acordo com a equação 1.

$$d^*_{x,y} = \begin{bmatrix} -x'_o \\ -y'_o \end{bmatrix} \tag{1}$$

O vetor $d^*_{x,y}$ é rotacionado para a esquerda ou para direita dependendo do posicionamento do alvo do robô. Se $\alpha_1 > \alpha_2$, então o objetivo encontra-se à esquerda do obstáculo. Se $\alpha_1 < \alpha_2$, então o objetivo encontra-se à direita do obstáculo. Se $\alpha_1 = \alpha_2$, por opção, tenta-se efetuar o desvio pela direita. O ângulo de rotação de $d^*_{x,y}$ é denotado por ϕ e é calculado pela Equação 2.

$$\phi = \begin{cases} 90 - 45 \cdot I_s(r_o, \rho_o) & \text{se } \alpha_1 \geq \alpha_2 \\ -90 + 45 \cdot I_s(r_o, \rho_o) & \text{se } \alpha_1 < \alpha_2 \end{cases} \tag{2}$$

Após o cálculo do ângulo, $d^*_{x,y}$ é rotacionado aplicando-se a matriz de rotação R conforme Equação 3.

$$d'_{x,y} = R \cdot d^*_{x,y} \tag{3}$$

$$R = \begin{bmatrix} \cos(\phi) & -sin(\phi) \\ sin(\phi) & cos(\phi) \end{bmatrix} \tag{4}$$

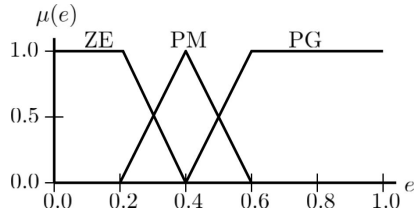
Conforme a Figura 4, o cálculo do fator de ponderação é realizado através de um interpolador *fuzzy* cujas entradas são a leitura da intensidade do sensor de proximidade e a distância do agente em relação a seu alvo. O resultado é um fator de ponderação escalar $\tau \in [0, 1]$.

A base de regras utilizada no cálculo do fator de ponderação está ilustrada na Tabela 2, onde e_o é a distância euclideana (erro) entre o robô e o alvo, conforme a Equação 5. As funções de pertinência para as entradas e a saída estão ilustradas na Figura 7.

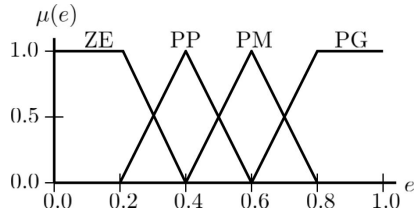
Tabela 2: Base de regras do fator de ponderação

e_o/I_s	ZE	PM	PG
ZE	PG	PG	PG
PM	PG	PM	PM
PG	ZE	PP	PP

$$e_o = \sqrt{e_x^2 + e_y^2} \tag{5}$$



(a) Variáveis de entrada



(b) Variável de saída

Figura 7: Funções de pertinência do interpolador fuzzy

A ação de controle resultante é a combinação das ações de controle de busca e desvio, ponderadas pelo fator τ, conforme a Equação 6.

$$ac'_{x,y} = \begin{bmatrix} \tau b'_x + (1-\tau)d'_x \\ \tau b'_y + (1-\tau)d'_y \end{bmatrix} \tag{6}$$

4 Resultados

Para testar e validar a arquitetura do SMR proposto, dois experimentos são realizados. No primeiro há apenas um robô, um obstáculo e um alvo e servirá para validar as funcionalidades do sistema de navegação autônomo da Seção 3. O segundo cenário contém 4 robôs e 6 alvos e será destinado à validação da arquitetura completa.

4.1 Cenário 01 - Um robô, um obstáculo e um alvo

Neste cenário, um único robô, um obstáculo e um alvo são alinhados horizontalmente, com o obstáculo posicionado entre o robô e o alvo. Espera-se que o robô desvie do obstáculo no centro e colete o alvo à direita. A Figura 8 traz os principais momentos da execução, mostrando que de fato o robô realiza a trajetória esperada. O experimento tem início na Figura 8a. Após alguns instantes, o robô detecta o obstáculo a sua frente, e toma a ação de desvio demonstrada nas figuras 8b e 8c. Finalmente, após terminar a ação de desvio, o robô continua em busca do alvo até coletá-lo, conforme a Figura 8d.

As ações de controle calculadas no primeiro experimento são mostradas nos gráficos da Figura 9 em termos das iterações de software para o envios dos comandos de acionamento dos robôs. Inicialmente ($0 \leq k \leq 54$), o robô realiza unicamente a ação de busca, pois não ocorre detecção (Figura 9b) de obstáculos na vizinhança, conforme verificam-se nas figuras 9a, 9c e 9e, onde a ação de controle resultante (9e), representada pela soma ponderada de 9a e 9c, é exatamente igual à ação de busca (9a) no intervalo de tempo descrito. Aproximadamente em $k = 55$, registra-se uma leitura no sensor de aproximação, com pico ocorrendo no instante de tempo $k = 60$, conforme a Figura 9b. Neste instante, o fator de ponderação τ é alterado (Figura 9d), e a ação de desvio passa a influenciar a ação resultante (Figura 9e).

Os resultados deste experimento validam a arquitetura em módulos, a troca de informação e o sistema de navegação autônoma.

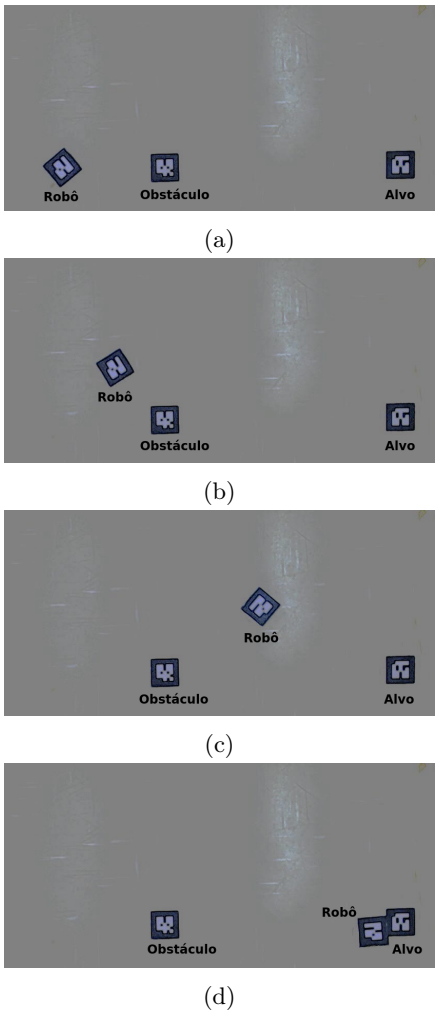
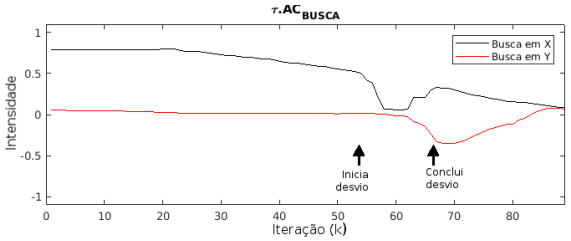


Figura 8: Cenário 1: Robô desvia de um obstáculo para coleta do alvo

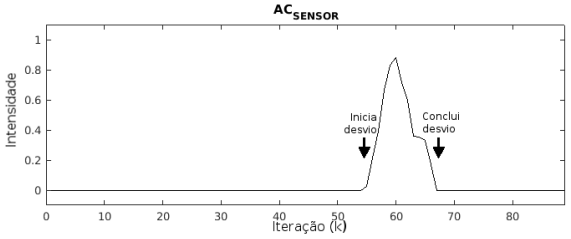
4.2 Cenário 02 - Quatro robôs e seis alvos

No segundo experimento, um cenário de dimensões 2,5x1,5 metros, com quatro robôs e seis alvos dispostos aleatoriamente no ambiente é considerado. Os robôs são inicialmente posicionados no canto esquerdo do cenário e espera-se que ao final da execução cada robô tenha conquistado pelo menos um alvo, atingindo uma posição final prédefinida para cada robô, no canto direito do cenário.

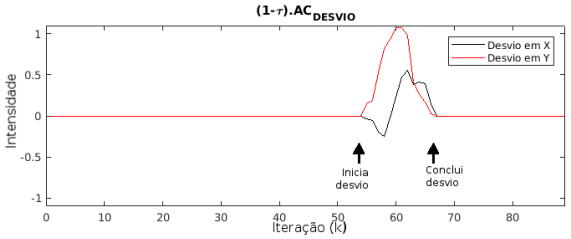
As trajetórias planejadas pelo módulo de Planejamento e Distribuição encontram-se na Figura 10, onde os círculos pretos representam os



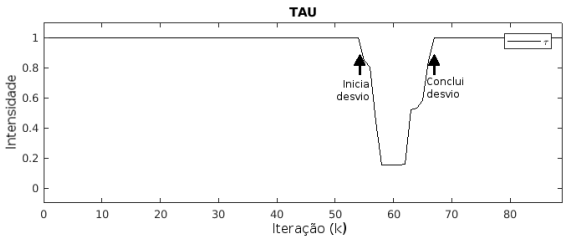
(a) Ação de busca



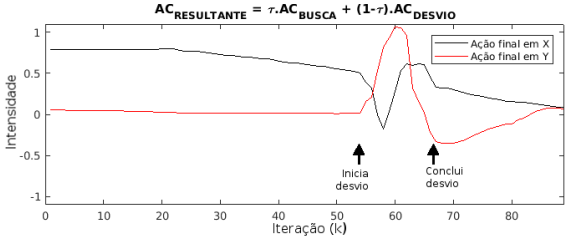
(b) Intensidade do sensor



(c) Ação de desvio



(d) Variação de τ



(e) Ação resultante

Figura 9: Variáveis do sistema de navegação autônoma

robôs, os círculos vermelhos os alvos e os quadrados pretos os pontos finais de cada robô. A execução do experimento está ilustrado na Figura 11. Para facilitar o entendimento, as trajetórias de apenas dois robôs ($R1$ e $R4$) são traçadas nas imagens utilizando as mesmas cores das trajetórias da Figura 10 para facilitar a identificação. O robô $R1$ (trajetória vermelha) deve conquistar somente um alvo ($A4$), desviando-se dos demais elementos no ambiente. Após conquistar o alvo, desloca-se para sua posição final. Já o robô $R4$ (trajetória verde)

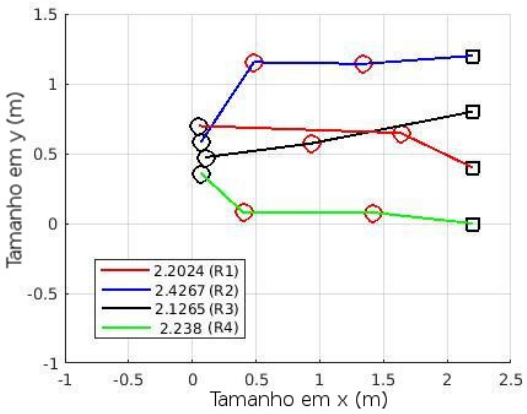


Figura 10: Planejamento de trajetória

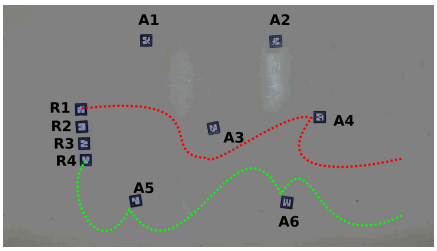
deve coletar dois alvos antes de deslocar-se para sua posição final. O momento inicial do experimento com os robôs alinhados à esquerda é mostrado na Figura 11a. Os robôs $R1$ e $R3$ se aproximam (Figura 11b) e iniciam manobra de desvio resultando nas posições mostradas na Figura 11c. Como esta manobra de desvio não é considerada na etapa de planejamento, as trajetórias desses robôs se cruzam antes do momento indicado pelo planejamento (veja na Figura 10 que as trajetórias vermelha e preta se cruzam após o robô $R1$ passar no alvo $A4$). Na Figura 11d o robô $R1$ alcança o alvo $A4$ e se desloca para sua posição final. Por fim, a execução termina com todos os robôs tendo coletado seus alvos e atingido suas posições finais, conforme a Figura 11e.

Estes experimentos apesar de simples validam a arquitetura proposta e o sistema multirobô desenvolvido.

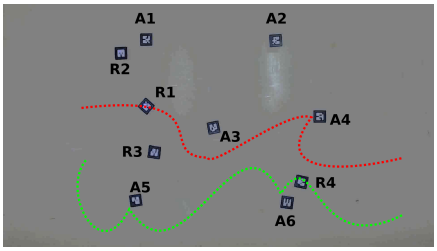
5 Conclusões

Este artigo apresentou uma arquitetura para sistemas multirrobôs com restrições de sensoriamento e comunicação. Foi também implementado um módulo de planejamento de trajetória baseado em algoritmos genéticos e um sistema de navegação autônoma baseado em controladores *fuzzy*. O SMR foi validado em experimento reais com robôs Sphero, apresentando bom desempenho para experimentos com até 5 robôs. A medida que se aumenta o número de robôs o sistema se torna lento devido à natureza *single-threaded* do MATLAB, plataforma em que são desenvolvidos os controladores.

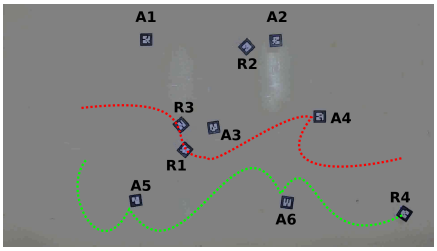
Com vistas a aplicações em robótica de enxame pretende-se remover o MATLAB da aplicação, desenvolvendo todos os algoritmos em ROS. Também espera-se realizar experimento com a odometria dos próprios robôs a fim de substituir a câmera que restringe o tamanho do ambiente explorado.



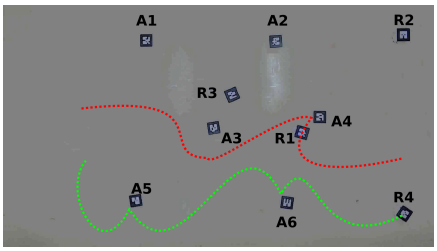
(a) Início da execução



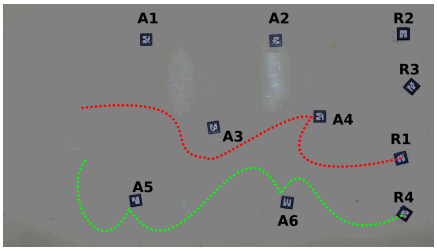
(b) $R1$ e $R3$ se aproximam



(c) $R1$ desvia-se de $R3$



(d) O robô $R1$ coleta o alvo $A4$



(e) Final de execução

Figura 11: Cenário 2 com 4 robôs e 6 alvos

Agradecimentos

Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq): processos 305816/2014-4 e 309119/2015-4.

Referências

Abadi, D. N. M. and Khooban, M. H. (2015). Design of optimal mamdani-type fuzzy controller for nonholonomic wheeled mobile robots, *Journal of King Saud University-Engineering Sciences* **27**(1): 92–100.

Almeida, J. P. L. S., Arruda, L. V. R. and Neves-Jr, F. (2017). Planejamento de rota por meio de algoritmo genético para um enxame de robôs, *XIII Simpósio Brasileiro de Automação Inteligente*.

Berthold, O. and Hafner, V. V. (2015). Closed-loop acquisition of behaviour on the sphero robot, *Proceedings of the European Conference on Artificial Life*.

Castillo, O., Neyoy, H., Soria, J., Melin, P. and Valdez, F. (2015). A new approach for dynamic fuzzy logic parameter tuning in ant colony optimization and its application in fuzzy control of a mobile robot, *Applied soft computing* **28**: 150–159.

de Sá, A. O., Nedjah, N. and de Macedo Mourelle, L. (2017). Distributed and resilient localization algorithm for swarm robotic systems, *Applied Soft Computing* **57**: 738 – 750.

Fiala, M. (2004). Artag revision 1, a fiducial marker system using digital techniques.

Kisdi, A. and Tatnall, A. R. (2011). Future robotic exploration using honeybee search strategy: Example search for caves on mars, *Acta Astronautica* **68**(11–12): 1790 – 1799.

Ladosz, P., Oh, H. and Chen, W.-H. (2017). Trajectory planning for communication relay unmanned aerial vehicles in urban dynamic environments, *Journal of Intelligent & Robotic Systems* .

Mac, T. T., Copot, C., Tran, D. T. and Keyser, R. D. (2016). Heuristic approaches in robot path planning: A survey, *Robotics and Autonomous Systems* **86**: 13 – 28.

Mendonça, M., Chrun, I. R., Neves-Jr, F. and Arruda, L. (2017). A cooperative architecture for swarm robotic based on dynamic fuzzy cognitive maps, *Engineering Applications of Artificial Intelligence* **59**: 122 – 132.

Oh, H., Shirazi, A. R., Sun, C. and Jin, Y. (2017). Bio-inspired self-organising multi-robot pattern formation: A review, *Robotics and Autonomous Systems* **91**(Supplement C): 83 – 100.

Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., Wheeler, R. and Ng, A. Y. (2009). Ros: an open-source robot operating system, *ICRA Workshop on Open Source Software*.