DESENVOLVIMENTO DE UMA ARQUITETURA DE NAVEGAÇÃO DELIBERATIVA PARA ROBÔS MÓVEIS UTILIZANDO A TEORIA DE CONTROLE SUPERVISÓRIO

Lucas Molina*, João C. Basilio*, Eduardo O. Freire†, Marcos V. Moreira*

*Programa de Engenharia Elétrica da Universidade Federal do Rio de Janeiro - PEE/COPPE/UFRJ, Rio de Janeiro-RJ, Brasil.

†Departamento de Engenharia Elétrica da Universidade Federal de Sergipe - DEL/UFS, São Cristóvão-SE, Brasil.

Emails: lmolina@ufs.br, basilio@dee.ufrj.br, eofreire@ufs.br, moreira@dee.ufrj.br

Abstract— This paper presents a development and implementation of a navigation architecture predominantly deliberative using behavior coordination by using automata and a planning criterion based on a measure of language (μ). The trajectory planning system models all robot possible moves and is basically formed of a planner automaton that models the navigation environment and the robot actions in this environment. By taking into account all possible robot trajectories from an initial to a final point as the sub-language of the language generated by the planner automaton, it is possible to use parameter μ as a measure of performance for these trajectories so as to choose the best trajectory according to the adopted criterion. Simulation exercises carried out using MobileSim software, made available by P3-DX robot manufacturer, demonstrate the efficiency of the proposed navigation architecture.

Keywords— Mobile robot navigation, supervisory control, path planning.

Resumo— Nesse artigo é apresentado o desenvolvimento e a implementação de uma arquitetura de navegação predominantemente deliberativa, utilizando coordenação de comportamentos modelados utilizando-se autômatos e um critério de planejamento baseado em uma medida de linguagem (μ). O sistema de planejamento de trajetória modela todos os possíveis movimentos do robô, sendo formado, basicamente, por um autômato de planejamento, que modela o ambiente de navegação e as ações do robô nesse ambiente. Considerando as trajetórias que levam o robô de um ponto inicial a um ponto de destino como sendo sublinguagens da linguagem gerada pelo autômato de planejamento, é possível utilizar o parâmetro μ como medida de desempenho dessas trajetórias para escolher a melhor trajetória segundo o critério adotado. Exercícios de simulação realizados utilizando-se o *software* MobileSim, disponibilizado pela fabricante do robô P3-DX, comprovam a eficácia da arquitetura de navegação proposta nesse trabalho.

Palavras-chave— Navegação de robôs móveis, controle supervisório, planejamento de trajetória.

1 Introdução

Robôs móveis são utilizados, atualmente, na execução de diversas tarefas em que é necessário algum tipo de locomoção como por exemplo em missões de resgate e exploração. A complexidade das tarefas as quais um robô é submetido tem aumentado muito e com ela, cresceu a demanda por sistemas de navegação mais complexos, onde a preocupação não é simplesmente concluir uma tarefa e sim concluí-la atendendo diversas especificações (comportamentos). Para atender a esses objetivos ao mesmo tempo, surgiram as teorias de composição de comportamentos, nas quais sistemas de controle independentes são desenvolvidos para alcançar diferentes objetivos (comportamentos).

A elaboração de um sistema de composição de comportamentos cooperativos nem sempre é uma tarefa fácil. Assim sendo, diversos pesquisadores buscaram em áreas afins, técnicas e formalismos que possibilitassem a realização da composição de comportamentos de forma mais simples e bem fundamentada. Nesse contexto, a modelagem de comportamentos através da teoria de sistemas a eventos discretos (SED) (Cassandras e Lafortune, 2006), mais especificamente autômatos, surge como uma ferramenta que possibilita modelar a interação entre os diversos comportamentos de um sistema de forma simples, permitindo ainda a inclusão de novos comportamentos no sistema, sem que para isso seja necessária qualquer alteração nos demais comportamentos já modelados. Em Kosecká e Bajcsy (1993) foi utilizada formalmente a teoria de autômatos para demonstrar a viabilidade da utilização dessa teoria na modelagem, síntese e coordenação de diferentes comportamentos. Após a comprovação da viabilidade da aplicação de SED para tarefas de navegação, foi apresentado em Kosecká e Bogoni (1994) um estudo mais detalhado a respeito da controlabilidade e observabilidade de comportamentos de navegação baseados em autômatos. Em Kosecká (1996), a teoria de controle supervisório (Ramadge e Wonham, 1987) foi considerada na construção e composição de diferentes comportamentos de navegação.

Com a evolução da teoria de controle supervisório surgiram diferentes métodos de desenvolvimento de um controlador e, com eles, cresceu a busca pela otimalidade dos sistemas controlados. Em Ray e Phoha (2003) e Ray et al. (2005) foi formalizada uma medida de linguagem quantitativa (μ), capaz de avaliar o desempenho de sistemas controlados. Essa medida foi utilizada em Wang

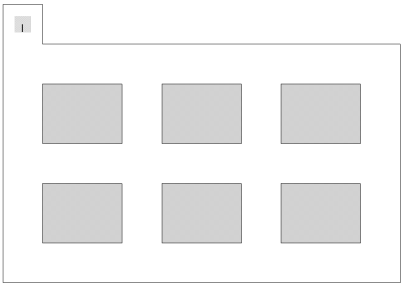

Figura 1: Ambiente de navegação considerado.

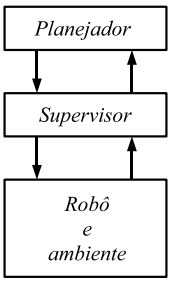

Figura 2: Diagrama de blocos da arquitetura de navegação proposta.

et al. (2004) para definir uma maneira de selecionar os diferentes comportamentos de um sistema de navegação. Em Wang et al. (2005), a medida de linguagem μ foi utilizada juntamente com a teoria de sistemas supervisores para o desenvolvimento de um sistema de navegação ótimo para robôs móveis, segundo o critério μ.

Esse artigo apresenta o desenvolvimento e a implementação de uma arquitetura de navegação predominantemente deliberativa, utilizando coordenação de comportamentos modelados por autômatos e um critério de planejamento baseado em uma medida de linguagem (μ). No sistema aqui proposto, a medida de linguagem μ é modificada para funcionar como um critério de planejamento de trajetória, possibilitando a escolha de uma trajetória ótima segundo esse critério.

Este trabalho está estruturado da seguinte forma. Na seção 2 é apresentado o problema a ser abordado. Na seção 3, os autômatos que modelam o robô e os comportamentos admissíveis no ambiente de navegação são construídos e utilizados para compor um único modelo que representa todo o sistema. Na seção 4 é apresentada a construção do autômato de planejamento e a sua utilização para a determinação da melhor trajetória. Na seção 5 são apresentados os resultados obtidos, em simulação. Finalmente, na seção 6 são apresentadas as conclusões.

2 Ambiente e arquitetura de navegação proposta

Para definir os movimentos necessários ao robô na tarefa de navegação, é preciso definir inicialmente o ambiente onde o robô irá navegar. O ambiente considerado nesse trabalho está representado na figura 1, no qual a tarefa do sistema de navegação é levar o robô da posição atual a uma posição de destino especificada, para a retirada de um item do depósito.

O arquitetura de navegação proposta (figura 2) é formado por dois grandes blocos: o sistema supervisor e o sistema de planejamento. O *planejador* é o responsável por escolher a melhor trajetória que leva o robô de um estado (posição inicial) a outro (posição de destino), enquanto o *supervisor* é o responsável por garantir a execução da sequência planejada, utilizando para isso uma composição de comportamentos.

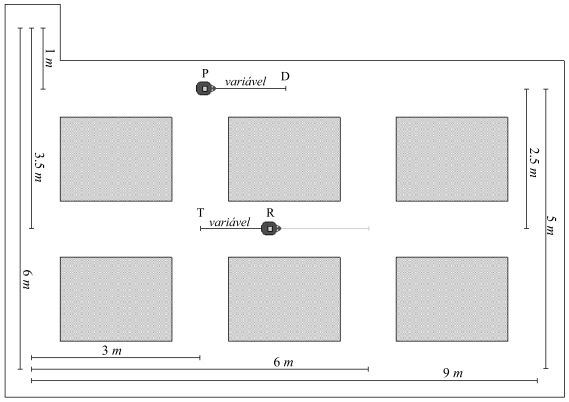

Figura 3: Distâncias de translação necessárias ao robô para navegar no ambiente considerado.

3 O supervisor

Para navegar no ambiente ilustrado na figura 1 um robô precisa apenas de movimentos desacoplados de translação e rotação, sendo que as rotações devem ocorrer apenas em alguns pontos de interesse (intersecção dos corredores). Esses pontos são chamados de *pontos de transição*. Analisando o mapa da figura 3, é possível observar 12 diferentes valores de distâncias de translação necessárias ao robô para navegar neste ambiente, dos quais 8 são fixos (que indicam as distâncias entre os pontos de transição) e os demais variáveis, que indicam ou a distância entre a posição atual do robô (R) e o último ponto de transição válido (T), ou a distância necessária para ir do ponto de transição atual (P) ao destino (D).

As distâncias ilustradas na figura 3 estão associadas a eventos de acordo com a tabela 1. No que diz respeito à rotação, devido à simplicidade do mapa escolhido, é necessário apenas três valores de rotação para o robô, quais sejam: (i) 90 graus em sentido anti-horário; (ii) 90 graus em sentido horário e; (iii) 180 graus. Essas rotações estão associadas a eventos de acordo com a tabela

Tabela 1: Eventos que indicam os comandos de ação do robô.

Evento	Descrição
v_1	efetuar translação de 1 m (v)
v_2	efetuar translação de 2,5 m (v)
v_3	efetuar translação de 5 m (v)
v_4	efetuar translação de 3,5 m (v)
v_5	efetuar translação de 6 m (v)
v_f	efetuar translação para o destino (v)
v_r	efetuar translação de retorno (v)
h_1	efetuar translação de 3 m (h)
h_2	efetuar translação de 6 m (h)
h_3	efetuar translação de 9 m (h)
h_f	efetuar translação para o destino (h)
h_r	efetuar translação de retorno (h)
r_1	efetuar rotação de 90 graus
r_2	efetuar rotação de -90 graus
r_3	efetuar rotação de 180 graus

1. Todos esses eventos são controláveis, uma vez que, em sua essência, são comandos de ação dados ao robô para que este os execute. Para fins de simplificação de notação, serão utilizados os conjuntos de eventos

$$\begin{aligned} E_t &= \{v_1, v_2, v_3, v_4, v_5, v_f, v_r, h_1, h_2, h_3, h_f, h_r\}, \\ E_r &= \{r_1, r_2, r_3\}, \\ E &= E_r \dot{\cup} E_t, \end{aligned}$$

em que E_t é o conjunto dos eventos de translação, E_r é o conjunto dos eventos de rotação e E é o conjunto de todos os eventos de movimento.

Para a obtenção do autômato de navegação, é necessário construir o modelo a eventos discretos G do robô e seus movimentos admissíveis no ambiente. O autômato de navegação G será obtido a partir da composição de três comportamentos mais simples, G_1, G_2 e G_3, quais sejam: (i) translação e rotação desacoplados (G_1); (ii) replanejar ao detectar um obstáculo (G_2) e; (iii) interface operação-planejamento-execução (G_3).

3.1 Comportamento G_1

Para navegar no ambiente de navegação proposto, o robô precisa apenas de movimentos de translação e rotação desacoplados, ou seja, não há necessidade de movimentos em arco, por exemplo. Além disso, é necessário garantir que uma nova ordem de movimento só poderá ser dada ao robô caso o robô não esteja executando algum outro movimento. O objetivo do autômato G_1 é modelar esse comportamento. Para isso, é necessário utilizar dois eventos, quais sejam: (i) w_0, que indica que a velocidade de rotação do robô é zero e; (ii) v_0, que indica que a velocidade de translação do robô é zero. Esses eventos são não controláveis, pois sua ocorrência não pode ser desabilitada por um supervisor. Na tabela 2 são apresentados esses eventos, juntamente com o evento tr (também não controlável) que indica que o robô terminou a navegação, ou seja, alcançou o ponto de destino especificado. Para fins de simplificação de notação, será utilizado o conjunto de eventos

$$E_f = \{v_0, w_0, tr\}$$

para denotar o conjunto de todos os eventos que indicam o término de um movimento, seja de rotação, translação ou da tarefa como um todo. O diagrama de transição de estados do autômato G_1, que modela esse comportamento, é apresentado na figura 4.

Tabela 2: Eventos que indicam o término de uma tarefa de translação, rotação ou da navegação.

Evento	Descrição
v_0	indica velocidade linear nula
w_0	indica velocidade angular nula
tr	tarefa realizada com sucesso

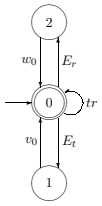

Figura 4: Diagrama de transição de estados de G_1.

3.2 Comportamento G_2

Mesmo em um sistema de navegação prioritariamente deliberativo, é necessário incluir na arquitetura de navegação a capacidade de evitar colisões, fazendo com que o robô, ao observar um obstáculo em seu caminho, possa interromper a sua ação e replanejar a execução da tarefa. O objetivo do autômato G_2 é modelar esse comportamento. Para fazer isso é necessário fazer com que, sempre que o robô detectar a presença de um obstáculo em seu caminho, ele execute uma sequência de eventos capaz de levá-lo ao último ponto de transição válido, onde a execução da tarefa será replanejada. Para realizar a modelagem desse comportamento, são necessários os eventos definidos na tabela 3. Para fins de simplificação de notação, será utilizado o conjunto de eventos

$$E_o = \{od, pr, rp, rpe\}.$$

O diagrama de transição de estados do autômato G_2 que modela a detecção de obstáculos e o planejamento está representado na figura 5.

Tabela 3: Eventos relacionados à detecção de obstáculos e replanejamento.

Evento	Descrição
od	obstáculo detectado pelo robô
pr	ordem para parar o robô
rp	ordem para replanejar a tarefa
rpe	replanejamento executado

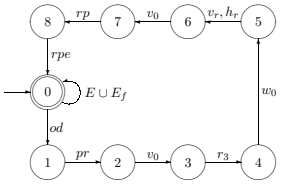

Figura 5: Diagrama de transição de estados de G_2.

3.3 Comportamento G_3

O comportamento G_3 tem como objetivo modelar a sequência de etapas necessárias ao robô, antes que este inicie a navegação propriamente dita e estabelecer a relação entre o operador (pessoa responsável por informar o ponto de destino ao robô e pela ativação/desativação da navegação) e o sistema. O diagrama de transição de estados do autômato G_3, que modela esse comportamento, é apresentado na figura 6 e os eventos associados a esse modelo são apresentados na tabela 4.

3.4 Autômato de navegação G

O autômato de navegação G é obtido através da composição paralela dos três comportamentos (G_1, G_2 e G_3) que modelam a navegação do robô, ou seja:

$$G = G_1 \| G_2 \| G_3. \quad (1)$$

O diagrama de transição de estados do autômato G está representado na figura 7. Note que os estados foram renomeados com o intuito de simplificar a notação. Analisando esse autômato é possível observar que existem sequências de eventos indesejadas no sistema, como aquelas que terminam na detecção de um obstáculo enquanto o robô está parado (evento *od* do estado 3 para o estado 24), por exemplo. Uma das maneiras de se corrigir esse problema é utilizando a teoria de controle supervisório que, a partir de especificações associadas ao modelo do robô, produzirá um sistema controlado, cuja linguagem representa exatamente o comportamento desejado para o sistema de navegação, eliminando, assim, as sequências indesejadas.

Tabela 4: Eventos relacionados à interface operação-planejamento-execução.

Evento	Descrição
p	ordem para planejar a tarefa
pe	planejamento executado
an	ordem para ativar a navegação
dn	ordem para desativar a navegação

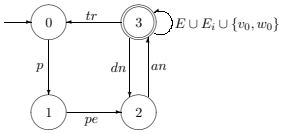

Figura 6: Diagrama de transição de estados de G_3.

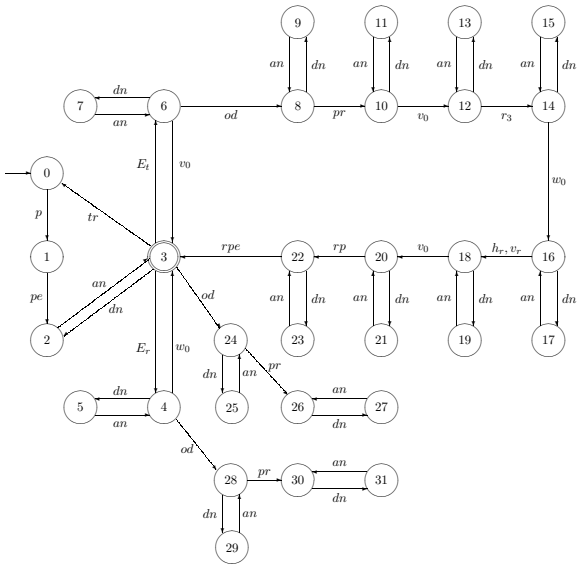

Figura 7: Diagrama de transição de estados de G.

3.5 Especificação de controle E_1

No modelo do sistema de navegação G, a capacidade de detectar obstáculos está ativa independente da ação que o robô esteja executando. Isso é desnecessário, uma vez que o robô, parado ou executando uma rotação, não precisa detectar obstáculo (para o caso de obstáculos estáticos, como os considerados nesse trabalho). Esse fato não representa problema algum para o comportamento do sistema de navegação G, mas aumenta a complexidade desse modelo significativamente. Para fazer com que a capacidade de detectar obstáculos do robô esteja ativa somente quando este estiver realizando uma translação foi criada a especificação de controle E_1, cujo diagrama de transição de estados está representado na figura 8.

3.6 Autômato de navegação controlado S/G

Uma vez definido o autômato de navegação G e a especificação de controle E_1, é preciso construir o autômato S/G que representa a especificação de controle e o comportamento do robô em um único

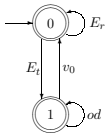

Figura 8: Diagrama de transição de estados de E_1.

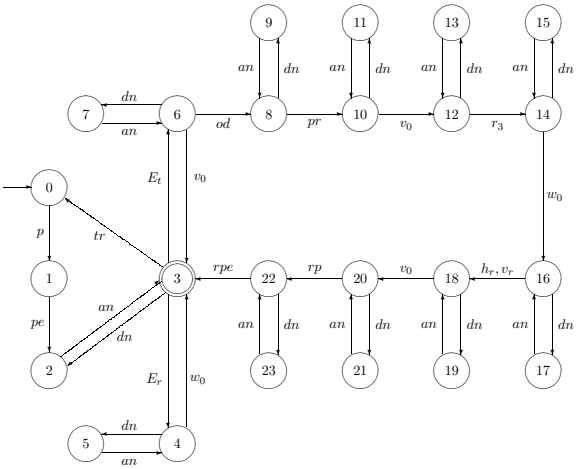

Figura 9: Diagrama de transição de estados de S/G.

modelo. Para tanto, torna-se necessário obter o seguinte resultado.

Teorema 1 *A linguagem gerada por E_1 ($\mathcal{L}(E_1)$ é controlável em relação a à linguagem $\mathcal{L}(G)$ e ao conjunto dos eventos não controláveis de G, $E_{uc} = \{v_0, w_0, tr, rpe, pe\}$.*

Demonstração. A linguagem $\mathcal{L}(E_1)$ será controlável em relação a $\mathcal{L}(G)$ e $E_{uc} = \{v_0, w_0, tr, rpe, pe\}$, se, e somente se, a seguinte relação for ser verdadeira:

$$\overline{\mathcal{L}(E_1)}E_{uc} \cap \mathcal{L}(G) \subseteq \overline{\mathcal{L}(E_1)}, \qquad (2)$$

em que $\mathcal{L}(E_1) = \overline{\mathcal{L}(E_1)}$, isto é, toda continuação não controlável da linguagem $\mathcal{L}(E_1)$ presente na linguagem $\mathcal{L}(G)$ deve também pertencer a $\mathcal{L}(E_1)$. O único evento não controlável de E_1 é o evento v_0. Uma vez que toda transição associada a esse evento em $\mathcal{L}(G)$ também aparece em $\mathcal{L}(E_1)$, a relação apresentada na equação (2) é verdadeira, logo, a linguagem $\mathcal{L}(E_1)$ é controlável. □

De acordo com o teorema 1 pode-se concluir, então, que o supervisor S/G é realizável, podendo o seu modelo ser obtido através da composição paralela entre G e E_1. O autômato S/G que modela o sistema de navegação controlado é apresentado na figura 9.

Com o modelo do autômato S/G, o robô é capaz de executar a sequência de eventos definida pelo planejador (figura 2) de forma segura, realizando movimentos de translação, movimentos de rotação e solicitando um replanejamento da trajetória sempre que encontrar um obstáculo durante uma translação. Além disso, são permitidas intervenções do operador, habilitando ou desabilitando a navegação, sempre que julgar necessário.

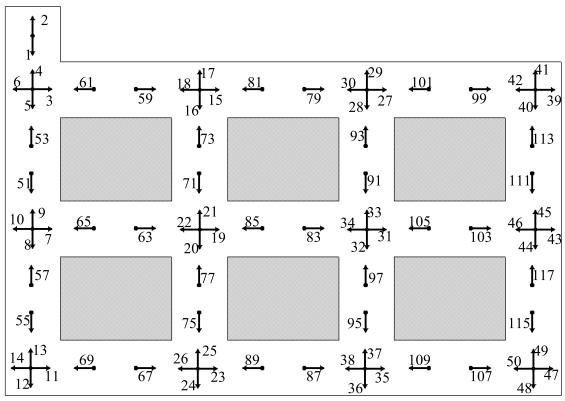

Figura 10: Representação dos estado de P no mapa.

4 O planejador

O planejamento de trajetória proposto leva em consideração os eventos de movimento admissíveis do robô E e o mapa do ambiente de navegação (figura 1) para possibilitar a construção de um autômato que modele todas as possíveis trajetórias que podem ser executadas pelo robô nesse ambiente. Esse autômato é denominado *autômato de planejamento*. Um vez construído o autômato de planejamento, a estratégia utilizada para escolher a melhor trajetória (sequência de eventos) consiste em separar as trajetórias que levam o robô do ponto atual ao ponto de destino e, em seguida, escolher a melhor trajetória, segundo o *critério de planejamento* μ.

4.1 Autômato de planejamento P

O autômato P é um autômato que modela todas as possíveis trajetórias que podem ser executadas pelo robô no ambiente proposto. Para construir esse autômato, é preciso discretizar as posições do robô no ambiente e associar cada posição a um estado do autômato P. Existem dois tipos de estados no autômato P: os *estados de transição* e os *estados finais*. Os *estados de transição* são os estados associados às posições de interseção dos corredores e os *estados finais* são os estados associados aos corredores. Todos esses estados estão representados na figura 10.

Definidos os estados do autômato P, as transições entre estes estados é determinada pelos eventos em E de forma direta, utilizando os valores de distância apresentados na figura 3. Por exemplo, para o robô sair do estado 1 e ir para o estado 5 é preciso uma translação de 1 metro na vertical

(evento v_1), para sair do estado 5 e ir para o estado 3 é preciso uma rotação de -90 graus (evento r_2) e assim por diante. Associando todos os possíveis movimentos do robô, em cada um dos estados do autômato P, aos eventos de movimento E, obtém-se o diagrama de transição de estados de P, que possui 118 estados, 15 eventos (eventos em E) e 219 transições.

4.2 Possíveis trajetórias

Utilizando o autômato de planejamento descrito na seção 4.1, é possível definir uma trajetória como uma sequência de eventos que leva o robô de um estado para outro. No entanto existem infinitas trajetórias que levam o robô de um estado inicial x_i a um estado de destino x_r. Isso ocorre porque a estrutura do ambiente permite ao robô revisitar o mesmo ponto de transição (interseção de corredores) infinitas vezes antes de chegar ao seu destino, criando assim laços de repetição infinitos. Para contornar esse problema, foi utilizado um algoritmo de planejamento para realizar uma busca pelas trajetórias candidatas, criando uma árvore com todas as sequências desde o estado atual do robô até o estado de destino especificado. Essa busca possui quatro critérios de parada que indicam o fim de um dos ramos da árvore de busca, quais sejam:
1) CP1: A trajetória chegou a um estado pertencente a um ponto de transição já visitado;
2) CP2: A trajetória chegou a um *estado final* diferente daquele associado ao ponto de destino do robô;
3) CP3: A trajetória contém duas rotações seguidas;
4) CP4: A trajetória chegou ao *estado final* associado ao ponto de destino do robô.
Os ramos da árvore que forem interrompidos com os critérios de parada CP1, CP2 e CP3 serão excluídos da busca, enquanto que os ramos que terminarem com o critério CP4 serão as possíveis trajetórias selecionadas. O algoritmo de busca baseado na construção de uma árvore, é apresentado a seguir.

Algoritmo 1

Passo 1 *Defina como a raiz da árvore o estado atual do robô no autômato P.*

Passo 2 *Crie os descendentes da raiz da seguinte forma:*
Suponha que $|\Gamma(x_i) \setminus \{v_3, v_4, v_5, h_2, h_3\}| = n_i$. Crie, então, n_i descendentes de x_i e rotule-os como $x' = f(x_i, \sigma)$, $\sigma \in \Gamma(x_i) \setminus \{v_3, v_4, v_5, h_2, h_3\}$. Atribua aos nós criados o status de ativo e à raiz o status de inativo.

Passo 3 *Para cada um dos nós ativos x_a da árvore faça:*

- **(i)** *se pelo menos uma das condições CP1, CP2 ou CP3 for verdadeira, elimine o nó x_a e atribua a ele o status de inativo.*
- **(ii)** *se a condição CP4 for verdadeira, então, o nó x_a é uma folha. Atribua a ele o status de inativo.*
- **(iii)** *se nenhuma das condições CP1, CP2, CP3 ou CP4 for verdadeira, então, crie os descendentes do nó ativo x_a da seguinte forma: suponha que $|\Gamma(x_a) \setminus \{v_3, v_4, v_5, h_2, h_3\}| = n_a$. Crie, então, n_a descendentes de x_a e rotule-os como $x' = f(x_a, \sigma)$, $\sigma \in \Gamma(x_a) \setminus \{v_3, v_4, v_5, h_2, h_3\}$. Atribua aos nós criados o status de ativo e ao nó x_a o status de inativo.*

Passo 4 *Se existir algum nó ativo na árvore volte para o Passo 3. Caso contrário, vá para o Passo 5.*

Passo 5 *As possíveis trajetórias são as menores sequências de eventos que partem da raiz até as folhas. O número de possíveis trajetórias é igual ao número de folhas.*

O resultado do algoritmo 1 é um conjunto não ordenado de sequências de eventos que levam o robô do estado atual ao estado de destino.

4.3 Critério de planejamento utilizando μ

Definidas as possíveis trajetórias para alcançar um ponto de destino, é preciso, agora, ordenar essas trajetórias, sob algum critério, possibilitando a escolha da melhor trajetória e finalizando, assim, a etapa de planejamento. Seja X_P o espaço de estados do autômato P, $\mathcal{L}(P_i)$ a linguagem gerada pelo autômato P a partir do estado i, ou seja,

$$\mathcal{L}(P_i) = \{s \in E^* : f(i, s) \in X_P\}, \qquad (3)$$

e

$$\mathcal{L}_j = \overline{\{s_j\}},\ j = 1, 2, ..., t\ , \qquad (4)$$

em que s_j é a *j-ésima* possível trajetória que leva o robô do estado atual i ao seu destino, $\mathcal{L}_j$ é a linguagem definida pelo fecho do prefixo da trajetória s_j e t é o número de possíveis trajetórias encontradas pelo algoritmo 1. Assim sendo, as linguagens $\mathcal{L}_j$ são sublinguagens de $\mathcal{L}(P_i)$, isto é:

$$\mathcal{L}_j \subset \mathcal{L}(P_i),\ j = 1, 2, ..., t. \qquad (5)$$

Analisando sob essa perspectiva, é possível transformar um conjunto não ordenado ou parcialmente ordenado de sublinguagens de $\mathcal{L}(P_i)$, em um conjunto de sublinguagens totalmente ordenado, utilizando o parâmetro de avaliação de desempenho de controladores, ou sublinguagens, μ. Para tanto, é preciso definir dois parâmetros do autômato P: o vetor característico, $\underline{\mathcal{X}}$, e a matriz de transição de estados $\mathbf{\Pi}$, a partir dos quais

é possível calcular o vetor $\boldsymbol{\mu}$ através da seguinte equação:

$$\boldsymbol{\mu} = (\mathbf{I} - \mathbf{\Pi})^{-1} \underline{\mathcal{X}}. \quad (6)$$

4.3.1 Vetor característico ($\underline{\mathcal{X}}$):

Considerando a lei de formação do vetor característico apresentada em Ray et al. (2005), deve-se associar um valor $\mathcal{X}_i$, diferente de zero, a cada estado i do autômato P, de acordo com o impacto desse estado no sistema modelado. No caso do autômato de planejamento desenvolvido, a construção do vetor característico $\underline{\mathcal{X}}$ segue a seguinte lei de formação:

$$\mathcal{X}_i = \begin{cases} 0, & \text{se } i = x_r; \\ 1, & \text{se } i = x_f; \\ \tau < 0, & \text{do contrário.} \end{cases} \quad (7)$$

em que x_r é o estado do robô no autômato P no momento da solicitação do planejamento, x_f é o estado do autômato P que corresponde ao destino especificado para o robô e τ é um valor negativo, para indicar os estados marcados indesejáveis. Para o sistema de planejamento proposto adotou-se $\tau = -1 \times 10^{-4}$. Esse valor foi alterado em diversos experimentos, para valores menores que zero e maiores que -1×10^{-2}, sem, no entanto, apresentar alterações na trajetória escolhida pelo planejamento.

4.3.2 Matriz de transição de estados ($\mathbf{\Pi}$):

Para definir os elementos da matriz $\mathbf{\Pi}$, segundo os resultados apresentados em Wang e Ray (2004) e Ray et al. (2005), é preciso conhecer a probabilidade $\tilde{\pi}$ de ocorrência dos evento em P. No caso deste trabalho, como o autômato P não representa uma planta física, a probabilidade de ocorrência de um evento depende da posição inicial do robô e do ponto de destino do mesmo, que não podem ser determinados *a priori*. Para contornar esse problema foi considerado que a probabilidade do robô, estando em um estado qualquer x, executar um evento qualquer e_1 é a mesma de executar um outro evento e_2, desde que $\{e_1, e_2\} \subset \Gamma(x)$, e que a probabilidade do robô executar um evento e_3 é zero se $e_3 \notin \Gamma(x)$. Dessa forma, foi definida a seguinte lei de formação para os elementos $\tilde{\pi}$:

$$\tilde{\pi}_{it} = \begin{cases} \varsigma, & \text{se } e_t \in \Gamma(x_i); \\ 0, & \text{se } e_t \notin \Gamma(x_i), \end{cases} \quad (8)$$

para qualquer $e_t \in E$ e qualquer $x_i \in X_P$. O valor de ς utilizado nesse trabalho foi 0,2 para respeitar a formalização da medida de linguagem μ (Wang e Ray, 2004; Ray et al., 2005), que diz que

$$\sum_j \tilde{\pi}_{it} < 1, \ \forall x_i \in X_p \text{ e } \forall e_t \in E. \quad (9)$$

Dessa forma, os elementos π_{ij} da matriz $\mathbf{\Pi}$ podem ser definidos em função de $\tilde{\pi}_{it}$ como sendo:

$$\pi_{ij} = \ \sum_{e_t \in E} \tilde{\pi}_{it}, \quad \text{se } f(x_i, e_t) = x_j. \quad (10)$$

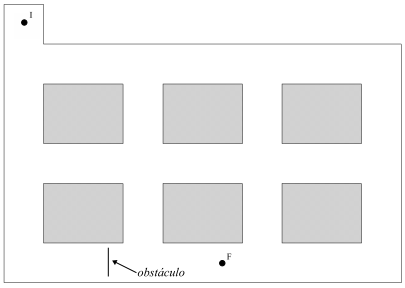

Figura 11: Ponto inicial I e ponto de destino F para os experimentos 1 e 2 e o obstáculo do experimento 2.

Tabela 5: Resultado do planejamento para alcançar o destino F, partindo do ponto I.

Sequência	Eventos	μ_1
s_1	$v_5 r_1 h_1 h_f$	1,0000
s_2	$v_1 r_1 h_1 r_2 v_3 r_1 h_f$	0,2500
s_3	$v_4 r_1 h_1 r_2 v_2 r_1 h_f$	0,2500
s_4	$v_1 r_1 h_2 r_2 v_3 r_2 h_f$	0,1249
s_5	$v_4 r_1 h_2 r_2 v_2 r_2 h_f$	0,1249

5 Resultados obtidos em simulação

Para ilustrar o funcionamento do sistema de navegação proposto foi utilizado um exemplo de seu funcionamento, em que o robô parte do ponto I com destino ao ponto F com um obstáculo na posição indicada na figura 11.

Nesse experimento, a posição $x = 5,5$ e $y = 0,5$ (ponto F) foi definida como destino para o robô e em seguida o planejamento foi ativado. As cinco melhores trajetórias encontradas pelo planejamento são apresentadas na tabela 5, em que a sequência s_1 representa a trajetória escolhida pelo planejamento.

O robô inicializa a execução da trajetória planejada normalmente até o momento em que um obstáculo é detectado (figura 12), quando ocorre o evento *od*. Nesse momento, em virtude do comportamento modelado pelo autômato G_2, o robô recebe o comando para interromper a execução da tarefa (evento *pr*) e, após parar completamente (evento v_0), ele retorna ao último ponto de transição válido (sequência $r_3 w_0 h_r v_0$). Após a ocorrência desses eventos o sistema de planejamento é acionado novamente para replanejar a trajetória, definindo, desta vez, a sequência de eventos que levará o robô da posição atual até o ponto de destino especificado F. Para evitar que, no replanejamento, o robô seja mandado novamente na direção do obstáculo, o evento r_3 (rotação de 180 graus) é desabilitado, garantindo que, no caminho replanejado, o robô não passará pelo corredor onde se encontra o obstáculo. Na tabela 6 são apresentadas as cinco melhores trajetórias escolhidas pelo sistema de planejamento, ordenadas se-

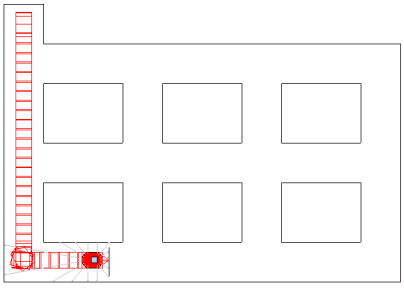

Figura 12: Trajetória executada pelo robô no experimento 2, até a detecção de um obstáculo.

Tabela 6: Resultado do replanejamento para alcançar o destino F.

Sequência	Eventos	μ_{14}
s_1	$r_2v_2r_2h_1r_2v_2r_1h_f$	1,0000
s_2	$r_2v_2r_2h_2r_2v_2r_2h_f$	0,4999
s_3	$r_2v_3r_2h_1r_2v_3r_1h_f$	0,2498
s_4	$r_2v_2r_2h_3r_2v_2r_2h_1h_f$	0,1248
s_5	$r_2v_3r_2h_2r_2v_3r_2h_f$	0,1248

gundo o critério de planejamento μ_{14} (o índice 14 é determinado pelo estado atual do robô no autômato de planejamento P, estado 14). A trajetória escolhida para ser executada pelo robô é aquela definida pela sequência s_1 da tabela 6, cujo resultado obtido é ilustrada na figura 13. Como na nova trajetória não existem obstáculos no caminho do robô, a execução da tarefa ocorre normalmente, com o robô alcançando o destino especificado.

6 Conclusão

Nesse trabalho foi apresentado o desenvolvimento e a implementação de uma arquitetura de navegação predominantemente deliberativa utilizando coordenação de comportamentos modelados por autômatos e um critério de planejamento baseado na medida de linguagem μ.

O sistema de planejamento de trajetória, utilizou um modelo em sistemas a eventos discretos (autômato de planejamento P) do ambiente de navegação e das ações do robô nesse ambiente

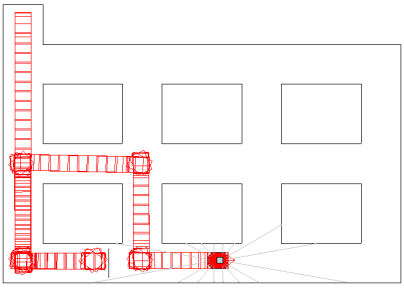

Figura 13: Trajetória completa executada pelo robô no experimento 2.

para representar todos os possíveis movimentos do robô. A linguagem $\mathcal{L}(P_i)$ do autômato P foi utilizada na obtenção do vetor característico ($\underline{\mathcal{X}}$) e da matriz de transição de estados ($\mathbf{\Pi}$), necessários para a construção da medida de linguagem μ. Considerando as trajetórias que levam o robô de um ponto inicial I a um ponto de destino F como sendo sublinguagens de $\mathcal{L}(P_i)$, foi possível utilizar o parâmetro μ como medida de desempenho dessas trajetórias e, dessa forma, escolher a melhor trajetória segundo o critério adotado. O sistema supervisor final funcionou conforme esperado, garantindo a correta execução da sequência de eventos planejada.

Agradecimentos

Os autores gostariam de agradecer à CAPES e ao CNPq pelo apoio financeiro que tornou possível a realização desse trabalho.

Referências

Cassandras, C. G. e Lafortune, S. (2006). *Introduction to Discrete Event Systems*, Springer-Verlag New York, Inc.

Kosecká, J. (1996). Supervisory control theory for autonomous mobile agents.

Kosecká, J. e Bajcsy, R. (1993). Discrete event systems for autonomous mobile agents, *Robotics and Autonomous Systems* **12**: 187–198.

Kosecká, J. e Bogoni, L. (1994). Application of discrete events systems for modeling and controlling robotic agents, *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 2257–2262.

Ramadge, P. J. G. e Wonham, W. M. (1987). Supervisory control of a class of discrete event process, *SIAM Journal of Control Optim.* **25**(1): 206 – 230.

Ray, A. e Phoha, S. (2003). Signed real measure of regular languages for discrete-event automata, *International Journal of Control* **76**: 1800 – 1808.

Ray, A., Phoha, V. V. e Phoha, S. (2005). *Quantitative Measure for Discrete Event Supervisory Control*, 1 edn, Springer Science + Business Media.

Wang, X., Fu, J., Lee, P. e Ray, A. (2004). Robot behavioral selection using discrete event language measure, *Proceedings of the 2004 American Control Conference*.

Wang, X. e Ray, A. (2004). A language measure for performance evaluation of discrete-event supervisory control systems, *Applied Mathematical Modeling* **28**: 817 – 833.

Wang, X., Ray, A., Lee, P. e Fu, J. (2005). Optimal control of robot behavior using language measure, *Quantitative Measure for Discrete Event Supervisory Control* pp. 157 – 182.