APLICAÇÃO DE ALGORITMO RRT* AO PLANEJAMENTO AUTOMÁTCO DE TRAJETÓRIAS DE NAVEGAÇÃO PARA VANTS

RESUMO

Um dos principais focos de pesquisas relacionadas a Veículos Aéreos Não Tripulados (VANTs) é o aumento de autonomia destes veículos, que consiste na diminuição do grau de dependência do VANT de operadores, transferindo parte do processo de tomada de decisão do operador para o próprio veículo. O planejamento automático de trajetórias de navegação é um exemplo de tecnologia autônoma para VANTs. O procedimento de planejamento de trajetórias pode ser dividido em três principais etapas: a modelagem computacional do ambiente de navegação do VANT; o planejamento automático de uma rota; e a suavização da rota planejada, convertendo-a em uma trajetória de navegação, isto é, em um movimento. O objetivo deste trabalho é a aplicação de três métodos computacionais nas três etapas mencionadas: a modelagem do ambiente de navegação através de grades regulares binárias; o planejamento de rotas por meio do algoritmo RRT*; e a suavização das rotas planejadas, convertendo-as em *k*-trajetórias.

PALAVRAS CHAVE. VANTs. Trajetórias de navegação. Algoritmo RRT*.

Tópicos: MH – Metaheuristicas

ABSTRACT

A major focus on research related to Unmanned Aerial Vehicles (UAVs) is increasing the autonomy of these vehicles, which consists in reducing the dependence level between the UAV and operators, transferring part of the decision making process from the operator to the vehicle itself. The automatic navigation trajectory planning is an example of autonomous technology for UAVs. The trajectory planning procedure can be divided into three main steps: the computational modeling of the UAV navigation environment; the automatic path planning; and the smoothing of the planned path, converting the path into a trajectory, i. e., a motion. The objective of this work is the application of three computational methods in the three mentioned steps: the modeling of the navigation environment through binary regular grids; the path planning, using the algorithm RRT*; and the smoothing of the planned paths, converting them into *k*-trajectories.

KEYWORDS. UAVs. Navigation trajectories. RRT* algorithm.

Paper topics MH – Metaheuristics

1. Introdução

Um dos principais focos das pesquisas relacionadas a Veículos Aéreos Não Tripulados (VANTs) é o aumento de autonomia destes veículos, que consiste na diminuição do grau de dependência do VANT de operadores, transferindo parte do processo de tomada de decisão do operador para o próprio veículo. O planejamento automático de trajetórias é essencial para a implementação da maioria das novas capacidades pretendidas com o aumento da autonomia de VANTs.

Uma trajetória de navegação para um VANT é o movimento que o veículo deve descrever entre uma posição de navegação (*waypoint*) de origem e uma posição de destino [Medeiros e Silva, 2010]. Neste trabalho, o procedimento de planejamento de uma trajetória de navegação é composto por três principais etapas: a modelagem computacional do ambiente de navegação do VANT; o planejamento automático de uma rota de navegação; e a suavização da rota planejada, convertendo-a em uma trajetória de navegação, isto é, em um movimento.

O modelo computacional de um ambiente de navegação é uma representação das regiões navegáveis e dos obstáculos à navegação do VANT nesse ambiente. O modelo computacional é usado no processo de verificação de situações de colisão das rotas e das respectivas trajetórias planejadas.

Uma rota é uma sequencia de *waypoints* que conectam o *waypoint* atual do VANT, isto é, a posição onde o VANT se encontra, e um *waypoint* de destino, ou seja, a posição que o veículo almeja alcançar ao final do movimento. Esses *waypoints* são conectados por segmentos de reta livres de situações de colisão com os obstáculos inseridos no modelo computacional do ambiente de navegação.

A última etapa é o procedimento de suavização que, como descrito anteriormente, consiste em converter a rota planejada em uma trajetória cinematicamente viável e livre de colisão. A trajetória de navegação deve permitir a navegação segura do VANT, sem violar suas restrições cinemáticas.

Há diversos estruturas e métodos que podem ser utilizados para o planejamento de trajetórias de navegação para VANTs [Medeiros, 2012].

Grades regulares binárias são matrizes quadradas que podem ser utilizadas para modelar ambientes de navegação. Cada célula da grade pode representar uma região do ambiente de navegação. Por exemplo, uma célula da grade pode assumir o valor 1 indicando que a região representada é um obstáculo, e o valor 0 indicando uma região navegável.

Árvores aleatórias de rápida exploração [LaValle, 1996], tradução de Rapidly-exploring Random Trees (RRTs), são algoritmos que vêm sendo utilizados com êxito no planejamento automático de rotas. O algoritmo RRT é um processo de amostragem de posições navegáveis de um ambiente através de um grafo do tipo árvore. O funcionamento consiste em expandir a árvore de modo aleatório do nó raiz até que uma de suas ramificações alcance uma posição final. Como cada nó possui informação de seu nó antecessor, a rota é traçada desta posição até a posição de origem e depois invertida. O algoritmo RRT* (RRT estrela) é uma melhoria do algoritmo RRT visando à redução da extensão das rotas planejadas [Karaman e Frazzoli, 2011].

As *k*-trajetórias propostas em [Anderson et al., 2005] são classes de trajetórias de navegação que podem ser construídas a partir de rotas, considerando as restrições cinemáticas de um VANT.

Assim, o objetivo deste trabalho é o planejamento de trajetórias de navegação para VANTs através de grades regulares, de uma implementação computacional do algoritmo RRT* e de *k*-trajetórias.

A Seção 2 descreve a metodologia elaborada para alcançar o objetivo proposto. Na Seção 3 são apresentados alguns resultados obtidos com a aplicação do algoritmo RRT* no planejamento de *k*-trajetórias para VANTs, considerando ambientes de navegação modelados por grades regulares binárias. Na Seção 4 são apresentadas as conclusões e observações finais sobre o trabalho desenvolvido.

2. Metodologia

Esta seção descreve os métodos utilizados para: a modelagem computacional de um ambiente de navegação; o planejamento automático de rotas de navegação para VANTs; e a conversão das rotas planejadas em trajetórias seguras e cinematicamente viáveis.

2.1. Grades Regulares Binárias

Neste trabalho, cada espaço C é a representação computacional bidimencional de um ambiente de navegação, compreendido como uma superfície de altitude constante gerada por uma grade regular binária.

A representação do ambiente de navegação foi gerado por meio de grades regulares binárias, atravéz de uma matriz quadrática. Cada célula da matriz cuja células assumem o valor 1, representam regiões não navegáveis (células pretas), as regiões cinzas são envoltórias de segurança em torno das áreas não navegáveis, a fim de aumentar a segurança das rotas e trajetórias planejadas para os ambientes de navegação. A Figura 1. Apresenta o ambiente de navegação utilizado neste trabalho.

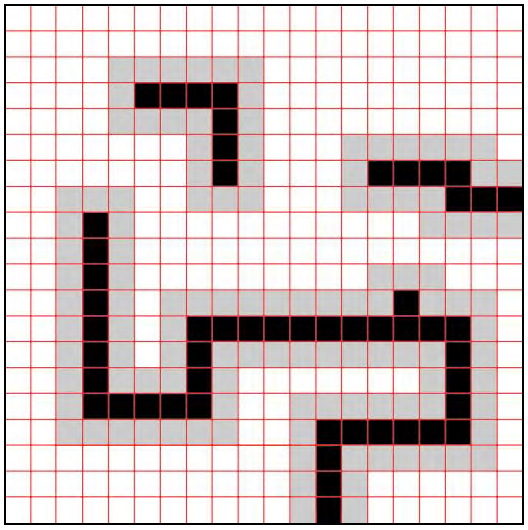


Figura 1. Representação do ambiente por meio de grade regular.

2.2. Algoritmo RRT*

Em [Karaman e Frazzoli, 2011] foi proposta a RRT*,que apresenta uma melhoria em relação à RRT. Essa melhoria consiste em buscar, a cada iteração, uma rota de menor extensão entre o novo nó q_{new} e q_{init} através da árvore existente. Com isso, há uma maior probabilidade das rotas de navegação planejadas através de uma RRT* terem extensão inferior que as rotas planejadas através de uma RRT padrão [Karaman e Frazzoli, 2011].

No método RRT*, cada novo nó q_{new} tem como antecessor um nó q_{min} da árvore. Este nó q_{min} corresponde a um nó dentro de uma região $V(q_{new},r(n_v))$ em torno do nó q_{new}, tal que a extensão da rota do nó raiz q_{init} até q_{new}, passando por q_{min}, é a menor dentre todos os nós pertencentes a $V(q_{new},r(n_v))$. Cada nó v_j de $V(q_{new},r(n_v))$, que possui uma rota até o nó q_{init} com extensão superior à extensão da rota entre q_{init} e v_j, passando por q_{new}, tem o seu nó antecessor substituído por q_{new}. Quando ocorre essa substituição, todos os nós sucessores de v_j têm as extensões de suas rotas até q_{init} atualizadas. Com esse procedimento de atualização, a RRT* procura, a cada iteração, criar ramificações com a menor extensão possível em relação ao nó raiz da árvore. O algoritmo correspondente ao método RRT* é apresentado na Tabela 1.

Tabela 1. Algoritmo RRT* para o planejamento de rotas.

Etapas	Descrição
1	inserir a raiz q_{init} na árvore G
2	$n_v \leftarrow 1$
3	$s \leftarrow 0$
4	enquanto s = 0 faça
5	$q_{rand} \leftarrow$ RAND_CONFIG(C)
6	$q_{near} \leftarrow$ NEAREST_VERTEX(q_{rand},G)
7	$q_{new} \leftarrow$ NEW_CONF(q_{near}, Δq)
8	se $\overline{q_{near}q_{new}}$ não intercepta qualquer obstáculo faça
9	$r(n_v) \leftarrow \frac{\beta \log(n_v)}{n_v}$
10	para cada nó q_i da árvore G faça
11	se distância(q_i,q_{new}) $\leq$ r(n_v) faça
12	inserir o nó q_i na lista V(q_{new}, r(n_v)) de nós vizinhos de q_{new}
13	inserir o nó q_{new} na árvore G
14	$n_v \leftarrow n_v + 1$
15	$q_{min} \leftarrow q_{near}$
16	$c_{min} \leftarrow$ extensão_rota(q_{init},q_{near}) + distância(q_{near},q_{new})
17	para cada nó v_j da lista V(q_{new}, r(n_v)) faça
18	se ((extensão_rota(q_{init},v_j) + distância(v_j,q_{new})) < c_{min}) e ($\overline{v_j q_{new}}$ não intercepta qualquer obstáculo) faça
19	$c_{min} \leftarrow$ (extensão_rota(q_{init},v_j) + distância(v_j,q_{new}))
20	$q_{min} \leftarrow v_j$
21	inserir a aresta que une q_{min} a q_{new} na árvore G
22	antecessor(q_{new}) $\leftarrow q_{min}$
23	para cada nó v_j da lista V(q_{new}, r(n_v)) faça
24	se ((extensão_rota(q_{init},q_{new}) + distância(q_{new},v_j)) < (extensão_rota(q_{init},v_j)) e ($\overline{q_{new}v_j}$ não intercepta qualquer obstáculo) faça
25	extensão_rota(q_{init},v_j) $\leftarrow$ (extensão_rota(q_{init},q_{new}) + distância(q_{new},v_j))
26	antecessor(v_j) $\leftarrow q_{new}$
27	atualizar as extensões das rotas entre q_{init} e todos os sucessores de v_j
28	se ($d(q_{new}, q_{dest}) \leq l_d$ e ($\overline{q_{new}q_{dest}}$ não intercepta qualquer obstáculo)) faça
29	inserir o nó q_{dest} na árvore G
30	inserir a aresta que une q_{new} a q_{dest} na árvore G
31	antecessor(q_{dest}) $\leftarrow q_{new}$
32	$s \leftarrow 1$
33	$q \leftarrow q_{dest}$
34	enquanto q $\neq q_{init}$ faça
35	armazenar q na pilha R, que representa a rota planejada
36	q $\leftarrow$ antecessor(q)
37	se q = q_{init} faça
38	armazenar q na pilha R

No algoritmo de planejamento de rota utilizando o método RRT*: *G* é o grafo que representa a árvore RRT; q_{init} é uma posição que corresponde à raiz da árvore; q_{rand} é uma posição gerada aleatoriamente no espaço *C*; q_{near} é o nó da árvore mais proximo da posição q_{rand}; q_{new} é um novo nó da árvore gerado no segmento de reta que une q_{near} a q_{rand}, e cuja a distância em relação a q_{near} é igual a Δq; *RAND_CONF* é uma função que gera aleatóriamente uma posição contida no espaço *C*, tal que o segmento de reta $(\overline{q_{near}q_{new}})$ não deve interceptar qualquer obstáculo contido em *C*; *NEAREST_VERTEX* é uma função que retorna o nó q_{near} mais próximo de q_{rand}; e *NEW_CONFIG* é uma função que gera q_{new} seguindo as restrições descritas anteriormente.

O parâmetro β é uma constante utilizada para definir o raio $r(n_v)$ da vizinhança do nó q_{new}. Através de uma análise da equação da linha 9, pode-se verificar que o valor de *r* diminui à medida que o número de nós/vértices (n_v) da ávore aumenta. Esse fato pode ser observado no gráfico da Figura 2. A diminuição é mais acentuada no inicío do processo e vai se tornando menos acentuada com o aumento de n_v. O intuito dessa formulação é permitir uma maior atualização de vizinho de q_{new} no início do processo de construção da árvore. A vizinhança de q_{new} vai sendo reduzida à medida em que a árvore é expandida, isto é, à medida em que aumenta a expectativa de que o próximo nó q_{new} seja criado a uma distância inferior ou igual a l_d, que é a

condição de parada do algoritmo. Assim, o controle da variação do tamanho da vizinha de q_{new} é feito através do parâmetro β.

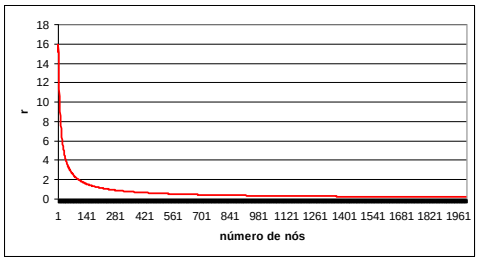


Figura 2. Variação de $r(n_v)$ para β=100.

Algumas etapas do algoritmo RRT* são exemplificadas na Figura 3. A etapa da Figura 3a corresponde à parcela do algoritmo delimitada pelas linhas 9 e 14. A etapa da Figura 3b corresponde à parcela do algoritmo delimitada pelas linhas 15 e 22. A etapa da Figura 3c corresponde à parcela do algoritmo delimitada pelas linhas 23 e 27. A etapa da Figura 3d é o resultado final das etapas anteriorres.

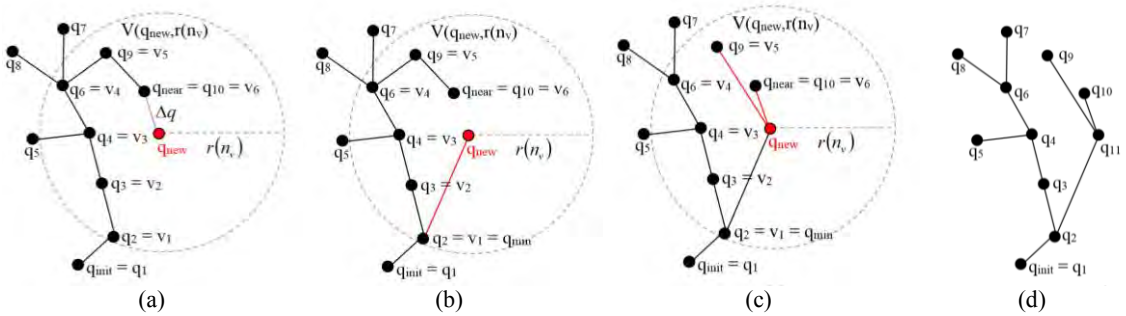


Figura 3. Etapas do algoritmo RRT*: (a) criação da vizinhança de q_{new} *(V(qnew,r(nv)));* (b) determinação de q_{min}; (c) substituição do nó antecessor de v_j por q_{new}, quando a rota entre q_{init} e v_j passando pelo seu antecessor por maior que a rota entre q_{init} e v_j passando por q_{new}; e (d) árvore atualizada após a inserção de q_{new}.

2.3. *k*-Trajetórias

Esse trabalho aborda o planejamento automático de trajetória ou movimento para VANTs considerando constantes a altitude e o módulo da velocidade de navegação. Fundamentando-se nestas considerações, o movimento bidimensional de um VANT pode ser modelado matematicamente por:

$$\begin{cases} \dot{x}_v = v_v \cos\theta_v \\ \dot{y}_v = v_v sen\theta_v \\ \dot{\theta}_v = \omega_v \in \left\{ -\dfrac{v_v}{r_c}, 0, \dfrac{v_v}{r_c} \right\} \end{cases} \quad (1.1)$$

Em que: (x_v, y_v) é a posição do VANT; v_v é a velocidade do VANT; θ_v é o ângulo formado pelo eixo x e o eixo longitudinal do VANT; ω_v é a velocidade angular do VANT; e r_c é o raio de curva da trajetória do VANT.

Na abordagem adotada nesse trabalho, uma rota é planejada por meio do método RRT* e depois é convertida em uma trajetória da classe *k*-trajetórias proposta em [ANDERSON et al., 2005].

Uma *k*-trajetória é uma sequencia de primitivas de movimento definidas pela Equação 1.1, cujo comprimento e forma podem ser ajustados por meio de um parâmetro *k*. Dada uma rota composta pelos *waypoints* w_{i-1}, w_i e w_{i+1}, uma *k*-trajetória é a única trajetória de menor extensão

que permite a navegação do *waypoint* w_{i-1} para o *waypoint* w_{i+1} passando diretamente pelo ponto $p(k)$, calculada em função de w_i. O ponto $p(k)$ é definido por [Medeiros, 2012]

$$p(k) = w_i + kr_c\left(\frac{1}{sen\left(\frac{\beta}{2}\right)} - 1\right)\vec{q} \tag{1.2}$$

Em que: c_i é o centro da circunferência que define $p(k)$; w_{ie1} e w_{ie2} são coordenadas de entrada dos arcos definidos pelas circunferências de centro c_{i-1} e c_i, respectivamente; w_{is1} e w_{is2} são coordenadas de saída dos arcos definidos pelas circunferências de centro c_i e c_{i+1}, respectivamente; $\vec{q}$ é um vetor unitário; r_c é o raio de curva do VANT.

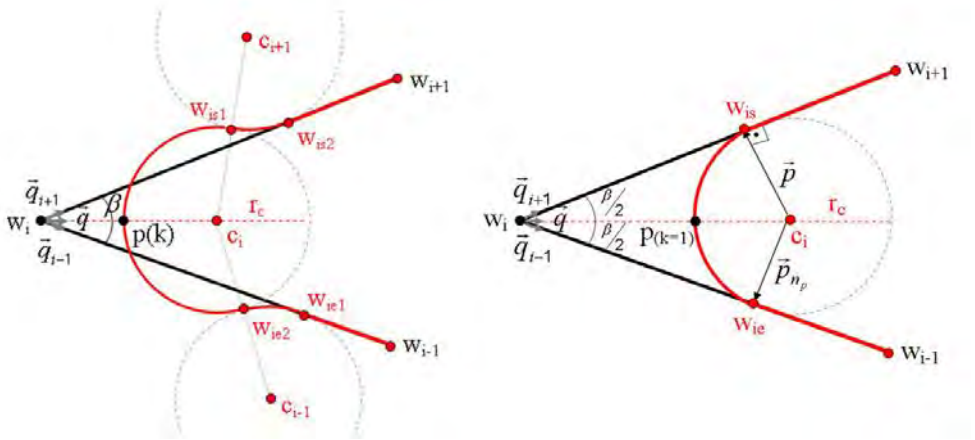


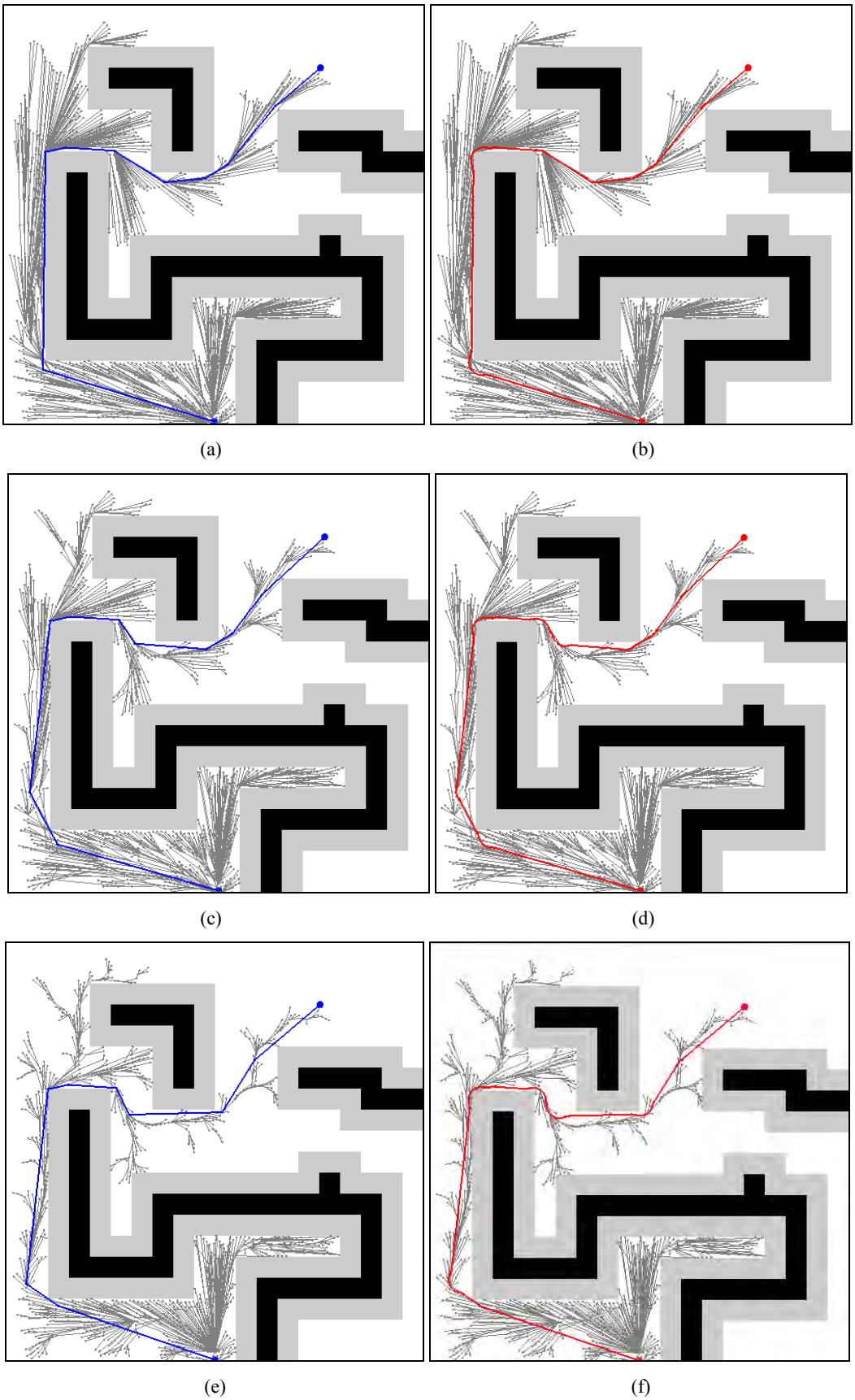
Figura 4. (a) exemplo de uma *k*-trajetória (b) exemplo de uma 1-trajetória.
Fonte: [Medeiros, 2012]

Desta forma, analisando a Figura 4b, uma 1-trajetória é composta por: uma sequencia de segmentos de reta $\overline{w_{i-1}w_{ie}}$; um arco de w_{ie} a wi_s; e um segmento de reta $\overline{w_{is}w_{i+1}}$, para i variando de 1 a n_t-1, onde n_t é o número de elementos de trajetória planejada. Na Figura 4, $\vec{q}$ é um vetor unitário definido pelos vetores unitários $\vec{q}_{i-1}$ e $\vec{q}_{i+1}$. Os *waypoints* w_1 e w_n são a origem q_{init} e o destino q_{init}, respectivamente. Os *waypoints* w_{ie} a wi_s são respectivamente, o *waypoint* de entrada e o *waypoint* de saída da curva definida pelo raio de curva r_c do VANT.

3. Resultados e Discussão

Um conjunto de experimentos de planejamento de trajetórias foi realizado com a aplicação do método RRT* implementado neste trabalho considerando o ambiente de navegação apresentado na Seção 2.

Exemplos de *k*-trajetórias planejadas por meio do método RRT* são apresentados na Figura 5, manteve-se a mesma sequencia de números aleatórios nos experimentos e variou-se o parâmetro β afim de mostrar seu efeito na construção da árvore RRT*. Nesta figura, os obstáculos do ambiente de navegação são representados pelos polígonos na cor preta e a camada em torno dos obstáculos é uma envoltória de segurança. A RRT* é apresentada na cor cinza. As rotas são apresentadas em azul. As trajetórias são apresentadas em vermelho.



(a) (b)

(c) (d)

(e) (f)

Figura 5. Rotas planejadas com RRT*, considerando: (a) $\beta = 20$ (c) $\beta = 10$ (d) $\beta = 5$. Rotas convertidas em trajetórias em (b), (d) e (f).

O teste das Figura 4a e Figura 4b foi executado em 3.16 segundos, o teste das Figura 4c e Figura 4d foi executado em 2.52 segundos e o teste das Figuras 4e e Figura 4f foi executado em 2.41 segundos. Os testes foram efetuados em um computador com processador de 2.3 GHz e com 8GB de memória RAM.

4. Conclusões

Através dos resultados apresentados na Seção 3, verifica-se que é possível planejar trajetórias de navegação seguras e cinematicamente viáveis para VANTs, através da aplicação do algoritmo RRT* com ambientes de navegação definidos por grades regulares. Embora uma RRT* não assegure o planejamento de rotas com a menor extensão possível, ela pode ser utilizada de modo satisfatório para o replanejamento de trajetórias, quando há remodelagem do ambiente de navegação. Assim, pode-se concluir que o objetivo deste trabalho foi alcançado.

Como trabalhos futuros, serão utilizados outros ambientes de navegação definidos por diferentes ambientes de navegação. Também será comparado as trajetórias planejadas por meio do método RRT e RRT* visando o planejamento de rotas menos extensas.

5. Referências Bibliográficas

Anderson, E. P.; Beard, R. W.; Mclain, T. W. (2005). Real-time dynamic trajectory smoothing for unmanned air vehicles. IEEE Transactions on Control Systems Technology, vol. 13, no. 3, p. 471-477.

LaValle, S. M. (1996). Rapidly-exploring random trees: A new tool for path planning. Computer Science Dept., Iowa State University, October.

Medeiros, F. L. L. (2012). Planejamento de trajetórias para veículos aéreos não tripulados usando modelagem computacional de ambientes de navegação através de grafos de visibilidade e modelos digitais de elevação. Tese de doutorado do curso de Computação Aplicada do Instituto Nacional de Pesquisas Espaciais (INPE), 238 p.

Medeiros, F. L. L.; Silva, J. D. S. (2010). A Dijkstra Algorithm for Fixed-Wing UAV Motion Planning Based on Terrain Elevation. Advances in Artificial Intelligence - SBIA 2010: 20th Brazilian Symposium on Artificial Intelligence, Series: Lecture Notes in Computer Science, v. 6404, p. 213-222.

TSOURDOS, A.; WHITE, B. A.; SHANMUGAVEL, M. (2011). Cooperative path planning of unmanned aerial vehicles. Wiley, 214 p.