Otimização discreta na determinação de trajetórias de veículos Dubins

André César Medeiros[1], **Sebastián Urrutia**[1]

[1]Departamento de Ciência da Computação - Universidade Federal de Minas Gerais
Av. Antônio Carlos, 6627, Pampulha, Belo Horizonte - MG, CEP 31270-901, Brazil

{acesar,surrutia}@dcc.ufmg.br

Abstract. *Dubins' vehicles describe a twice differentiable curve of bounded curvature. In this work we present an algorithm for the Traveling Salesman Problem for Dubins' vehicles. In our approach, we propose using a version of the Traveling Salesman Problem that minimizes both distance and direction change angles to determine the tour specifying the order in which the points should be visited. In order to calculate the point-to-point Dubins' path we rely on a result by Dubins, and apply a shortest path algorithm on a discretized search space. Results indicate that the new algorithm obtains better solutions than the ones found in the literature in similar computation times.*

Resumo. *Veículos Dubins são veículos que descrevem curvas duas vezes diferenciáveis com restrição de curvatura. Nesse trabalho investigamos um algoritmo para o Problema do Caixeiro Viajante de veículos Dubins. Para isto, propomos o uso de uma versão do Problema do Caixeiro Viajante que minimiza distâncias e ângulos de mudança de direção para determinar a ordem em que os pontos são visitados. Para calcular as curvas Dubins ponto-a-ponto, usamos o teorema de Dubins, e aplicamos um algoritmo de caminho mínimo em um espaço de busca discretizado. Os resultados indicam que o novo algoritmo obtém soluções melhores do que as encontradas na literatura.*

1. Introdução

O planejamento de trajetórias de robôs pode se aproveitar de soluções de otimização combinatória, em especial do Problema do Caixeiro Viajante (PCV). No entanto, muitos robôs possuem restrições de movimento devido a características intrínsecas da sua mecânica, impossibilitando-os de realizarem movimentos arbitrários. Essas características, bastante específicas e variadas, dificultam a adequação do PCV para o planejamento de trajetórias. Por exemplo, trajetórias de aeronaves são bastante distintas de trajetórias de automóveis, e não se pode afirmar que a solução do PCV é o percurso que proporciona a trajetória ótima, de menor comprimento, para qualquer um dos dois veículos.

A restrição mais relevante de movimento de veículos no plano é a de curvatura: existe uma curvatura máxima para o movimento. Em outras palavras, existe um raio mínimo para a curva circular que o veículo pode realizar. Dubins [2] estudou as curvas contínuas de comprimento mínimo, com restrições de curvatura, dadas as posições e tangentes das extremidades. Ele concluiu que essas curvas são compostas por segmentos de retas e arcos circulares com o raio mínimo de curvatura. Essas curvas, desde então chamadas de

Dubins, descrevem a trajetória de menor comprimento que um veículo não-holonômico [4], como um carro sem marcha ré, realiza.

O PCV para veículos Dubins (PCVD) consiste em encontrar a trajetória Dubins fechada de menor comprimento que passa por um conjunto finito de pontos no plano. Esse problema pode ser decomposto em duas etapas: *(a)* a etapa de circuito determina o circuito, i.e., a ordem de visita aos pontos, e *(b)* a etapa de trajetória determina as trajetórias ponto-a-ponto. Visto que Dubins [2] caracterizou a curva ponto-a-ponto de menor comprimento quando as orientações inicial e final são conhecidas, a segunda etapa pode ser resolvida determinando-se a orientação do veículo em cada um dos pontos do circuito.

Savla *et al.* [6] estudou o PCVD. Eles propuseram uma heurística que soluciona a etapa de circuito primeiro, e depois a etapa de trajetória. Para a etapa de circuito, eles propuseram usar a solução exata do PCV Euclidiano (PCVE: um caso especial do PCV, onde os pontos estão dispostos no plano e as distâncias entre eles é a distância Euclidiana). Para a segunda etapa, eles propuseram o "algoritmo alternante", que atribui orientações aos pontos do circuito de tal forma que a trajetória alterna entre segmentos de reta e curvas Dubins.

Le Ny *et al.* [5] observaram que a ordem de resolução das etapas pode ser trocada. Eles propuseram resolver primeiro a etapa de trajetória atribuindo uma orientação aleatória a cada ponto. Na segunda etapa, os autores calculam as trajetórias ponto-a-ponto com curvas Dubins para todo par de pontos e constroem um digrafo completo e assimétrico. A solução é então obtida resolvendo-se o PCV Assimétrico naquele digrafo de forma aproximada.

Neste trabalho, introduzimos uma nova heurística para o PCVD que resolve primeiro a etapa de circuito e depois a etapa de trajetória. Para a etapa de circuito, ao invés de usar a solução do PCVE, usamos a solução de uma versão diferente do PCV, que minimiza não apenas a distância total percorrida, mas também a soma dos ângulos de mudança de direção nos pontos a serem visitados. Esperamos que a minimização desses ângulos na primeira etapa promova a minimização do comprimento das trajetórias que serão encontradas na segunda etapa. Além disso, para a etapa de trajetória, usamos um algoritmo de caminho mínimo como uma forma eficiente de encontrar as escolhas ótimas de orientações em um espaço de busca discretizado. Combinadas, as soluções para ambas as etapas nos dão uma heurística para o PCVD.

Soluções para o PCVD podem ser aplicadas para determinar trajetórias de veículos aéreos não tripulados (VANT). Por exemplo, um VANT voando sobre um território com objetivos de vigilância precisa realizar um circuito no dado conjunto de pontos. Dadas certas configurações, a sua trajetória de menor comprimento não é segue o circuito solução do PCVE. Neste caso, a solução do PCVD é mais realista.

O restante deste artigo é organizado da seguinte forma. Na próxima seção apresentamos nosso algoritmo, descrevendo a abordagem nas duas etapas. A seção 3 apresenta resultados e comparações com outros algoritmos, enquanto a seção 4 conclui o artigo com algumas observações.

2. Abordagem

Uma curva Dubins é a curva $\gamma : [0,1] \rightarrow \mathbb{R}^2$ contínua e duas vezes diferenciável de menor comprimento que parte de um ponto $\gamma(0) = p_1 \in \mathbb{R}^2$ com orientação (ângulo de inclinação da tangente) $\phi_1 \in [0, 2\pi)$ e chega no ponto $\gamma(1) = p_2$ com orientação ϕ_2 de tal forma que sua curvatura é sempre menor que ou igual a $1/\rho$, onde $\rho > 0$ é chamado o *raio mínimo de curvatura*. Portanto, essa é a curva ótima entre uma *configuração inicial* (p_1, ϕ_1) e uma *configuração final* (p_2, ϕ_2) de um *veículo Dubins*.

Dubins [2] mostrou que a curva entre p_1 e p_2 segue um dos 6 tipos possíveis: LSL, RSR, RSL, LSR, RLR, LRL, onde S representa um segmento de reta, L representa um arco de círculo para a esquerda, e R um arco para a direita. Os arcos L e R têm raio exatamente ρ. Shkel e Lumelsky [7] providenciam fórmulas para o cálculo do comprimento dessas curvas. Observe que uma das três partes da curva pode ser vazia, i.e., de comprimento zero, e que as curvas do tipo RLR e LRL são viáveis apenas se a distância entre p_1 e p_2 for menor ou igual a 6ρ.

Dado um conjunto de pontos $P = \{p_1, p_2, \ldots, p_n\} \subset \mathbb{R}^2$, o PCVD consiste em determinar um circuito $Q = (q_1, q_2, \ldots, q_n)$ (isto é, uma ordenação) dos pontos de P, e uma orientação ϕ_i para cada q_i, tal que $\sum_{i=1}^{n} d((q_i, \phi_i), (q_{i+1}, \phi_{i+1}))$ seja mínima, onde $d((q_i, \phi_i), (q_{i+1}, \phi_{i+1}))$ é o comprimento da curva Dubins entre as configurações (q_i, ϕ_i) e (q_{i+1}, ϕ_{i+1}). Consideramos que q_{n+1} (respectivamente ϕ_{i+1}) é simplesmente uma notação conveniente para q_1 (resp. ϕ_1) e q_0 (resp. ϕ_0) é simplesmente uma notação conveniente para q_n (resp. ϕ_n).

A orientação de um segmento de reta entre p_i e p_j é definida por $\alpha(p_i, p_j)$, enquanto o ângulo de mudança de direção entre os segmentos (p_i, p_j) e (p_j, p_k) é definido por $\alpha(p_i, p_j, p_k) = \arccos\left(\frac{p_j - p_i}{\|p_j - p_i\|} \cdot \frac{p_k - p_j}{\|p_k - p_j\|}\right)$.

2.1. Determinação do circuito

Conforme mencionado na seção anterior, Savla *et al.* [6] propuseram o uso da solução ótima do PCVE como o circuito do PCVD. No entanto, o comprimento da curva de Dubins depende não apenas das distâncias, mas também das orientações inicial e final. O PCVE minimiza distâncias mas não considera os as mudanças de direção. Em particular, dois circuitos com mesma distância entre os pontos mas diferentes ângulos de mudança de direção resultam em curvas Dubins de comprimentos diferentes, e o PCVE não detecta essa diferença. A figura 1 ilustra esse fato.

Diante disso, é razoável sugerir a inclusão dos ângulos de mudança de direção na função objetivo do PCV. Esperamos que esse critério proporcione circuitos cujos pontos estejam relativamente alinhados, portanto as curvas Dubins serão mais curtas mesmo quando a distância total do circuito for maior que o circuito solução do PCVE. O modelo de programação inteira 1 resolve uma versão do PCV que considera a minimização da distância percorrida e dos ângulos de mudança de direção. Esse problema, chamado Problema do Caixeiro Viajante Angular (PCVAng), foi estudado por Aggarwal *et al.* [1]. As variáveis de decisão são:

$$x_{ij} = \begin{cases} 1, & \text{se o arco } (p_i, p_j) \text{ pertence ao circuito,} \\ 0, & \text{caso contrário,} \end{cases}$$

(a) ângulo reto **(b)** ângulo agudo

Figure 1: ***Comparação entre curvas Dubins devido a ângulos de mudança de direção distintos nos caminhos. As distâncias entre os pontos do caminho nos dois exemplos são idênticas.***

$$y_{ijk} = \begin{cases} 1, & \text{se os arcos } (p_i, p_j) \text{ e } (p_j, p_k) \text{ pertencem ao circuito,} \\ 0, & \text{caso contrário,} \end{cases}$$

onde os pontos p_i, p_j, p_k são distintos.

Formulação 1 *Problema do Caixeiro Viajante Angular.*

$$\text{minimizar} \quad w_1 \sum_{i \in P} \sum_{j \in P} c_{ij}\, x_{ij} + w_2 \sum_{i \in P} \sum_{j \in P} \sum_{k \in P} \alpha_{ijk}\, y_{ijk} \tag{1}$$

$$\text{sujeito a} \quad \sum_{j \in P} x_{ij} = 1, \quad \forall\, i \in P \tag{2}$$

$$\sum_{i \in P} x_{ij} = 1, \quad \forall\, j \in P \tag{3}$$

$$y_{ijk} \geq x_{ij} + x_{jk} - 1, \quad \forall\, i,\, j,\, k \in P \tag{4}$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1, \quad \forall\, S \subset P \tag{5}$$

$$x_{ij} \in \{0, 1\}, \quad \forall\, i,\, j \in P \tag{6}$$

$$y_{ijk} \in \{0, 1\}, \quad \forall\, i,\, j,\, k \in P \tag{7}$$

onde $w_1 \geq 0$ e $w_2 \geq 0$ são pesos relativos a cada objetivo. Os custos c_{ij} são igual a $c(p_i, p_j)$, a distância euclidiana entre os pontos p_i e p_j, e $\alpha_{ijk} = \alpha(p_i, p_j, p_k)$.

Esta formulação é uma generalização do PCVE. Ignorando as variáveis y_{ijk} e usando os valores $w_1 = 1$ e $w_2 = 0$, o modelo do PCVAng se transforma na formulação de Dantzig-Fulkerson-Johnson [3] para o PCVE.

Tanto w_1 quanto w_2 são relevantes para o comprimento das curvas Dubins. Neste trabalho propomos usar $w_1 = 1$ e $w_2 = \rho$, pois a solução para o PCVAng é o circuito de tempo mínimo para um robô de tração diferencial [4] de duas rodas separadas por uma distância de 2ρ. Esse fato pode ser provado, mas omitimos os detalhes neste artigo. É interessante notar que tanto a curva descrita por uma roda do robô de tração diferencial quando a curva Dubins consistem em segmentos de reta e arcos circulares de raio ρ.

O modelo anterior do PCVAng pode ser simplificado, utilizando apenas as variáveis y_{ijk}. O modelo simplificado foi o que utilizamos na implementação.

2.2. Determinando trajetórias ponto-a-ponto

Dado um circuito $Q = (q_1, \ldots, q_n)$ para o PCVD, a etapa restante consiste em computar a orientação em cada ponto de tal forma que a trajetória Dubins pode ser determinada. O "Algoritmo Alternante" de Savla *et al.* [6] atribui orientações $\Phi = (\phi_1, \ldots, \phi_n)$ aos pontos $q_1, \ldots, q_n$ tal que a trajetória Dubins alterna, a cada dois pontos do circuito, entre segmentos de reta e curvas Dubins.

Definimos o custo reduzido de uma curva Dubins entre as configurações (p_1, ϕ_1) e (p_2, ϕ_2) por $\text{cr}(p_1, \phi_1, p_2, \phi_2) = d(p_1, \phi_1, p_2, \phi_2) - c(p_1, p_2)$ e o comprimento Euclidiano do circuito Q como $c(Q)$. O custo da trajetória Dubins visitando os pontos na ordem dada por Q com orientações Φ é $\sum_{i=1}^{n} d(q_i, \phi_i, q_{i+1}, \phi_{i+1})$, sendo igual a $c(Q) + \sum_{i=1}^{n} \text{cr}(q_i, \phi_i, q_{i+1}, \phi_{i+1})$. Como Q é fixo, $c(Q)$ é constante, portanto, para minimizar o comprimento da trajetória Dubins, precisamos minimizar a soma dos custos reduzidos.

Propomos fazer a busca pelo melhor conjunto de orientações em um espaço de busca discretizado. Considere que as orientações nos pontos podem assumir apenas os valores no conjunto finito $\Lambda = \{(\frac{k}{K})2\pi \ : \ k = 0, \ldots, K-1\}$ consistindo em K ângulos distintos e igualmente separados. Λ discretiza o intervalo $[0, 2\pi)$ originalmente permitido para as orientações. Definimos o ponto inicial da trajetória como $q_s, \ s = \arg\max_{i=1,\ldots,n}\{\min\{c(q_{i-1}, q_i), \, c(q_i, q_{i+1})\}\}$, i.e., o ponto com a maior aresta adjacente mínima no circuito Q. Fixamos a orientação desse ponto inicial como a média dos ângulos $\alpha(q_{s-1}, q_s)$ e $\alpha(q_s, q_{s+1})$. Dada essa fixação, procuramos as orientações ótimas $\Phi^* \in \Lambda^n$.

Construimos um digrafo da seguinte forma:

- Para o ponto inicial q_s criamos dois nós: o nó inicial e o nó final.
- Para cada ponto $q_i \neq q_s$, criamos K nós (q_i, ϕ_λ), um para cada orientação $\phi_\lambda \in \Lambda$.
- Para cada par de nós $v_1 = (q_i, \phi_\lambda)$ e $v_2 = (q_{i+1}, \phi_{\lambda'})$ que representam configurações de pontos consecutivos no circuito, criamos um arco de v_1 para v_2 com custo $\text{cr}(v_1, v_2)$. O nó inicial tem apenas arcos de saída, e o nó final tem apenas arcos de entrada.

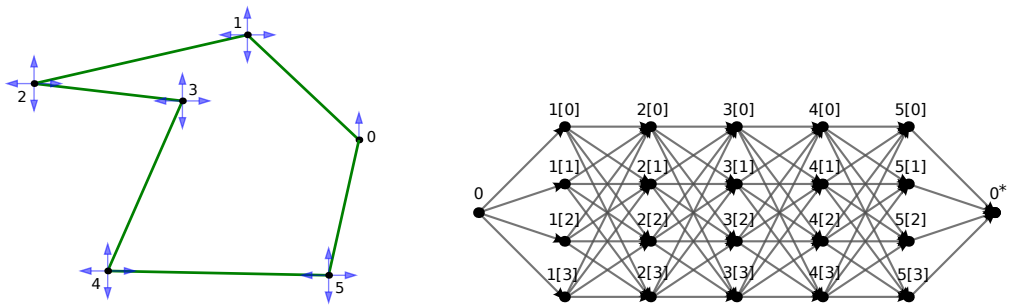
Então, aplicamos o algoritmo de Dijkstra entre o nó inicial e o nó final. O resultado dessa execução é a trajetória Dubins de menor comprimento que pode ser obtida com o dado circuito Q, a discretização Λ das orientações, e a fixação da orientação do ponto inicial. Note que o digrafo descrito acima não precisa ser construído antes da execução do algoritmo, mas criamos nós e arcos à medida que for necessário. A figura 2 ilustra como o digrafo é criado.

A escolha do ponto inicial é baseada na suposição de que pontos consecutivos no circuito que estão distantes um do outro resultam em comprimentos de curvas Dubins menos sensíveis às orientações em relação a pontos consecutivos próximos.

3. Resultados

Quatro algoritmos, que resolvem a etapa de circuito e depois a etapa de trajetória, foram implementados na linguagem C++ e executados em um computador Core 2 Quad 2.5GHz e sistema operacional Linux:

1. EAA: circuito ótimo do PCVE e orientações dadas pelo Algoritmo Alternante. Este é o método encontrado em Savla *et al.* [6].



(a) Discretização Λ nos pontos

(b) Digrafo construído

Figure 2: ***Processo de construção do digrafo. Dada a discretização*** Λ***, neste caso com 4 ângulos, obtemos o digrafo com 4 nós para cada ponto exceto o ponto inicial.***

2. ED: circuito ótimo do PCVE e orientações dadas pelo algoritmo de Dijkstra.
3. AngAA: circuito ótimo do PCVAng e orientações dadas pelo Algoritmo Alternante.
4. AngD: circuito ótimo do PCVAng e orientações dadas pelo algoritmo de Dijkstra. Essa é a nossa proposta de heurística.

O desempenho dos algoritmos foi comparado considerando o comprimento da trajetória Dubins para o PCVD em instâncias de teste de 20 ou 21 pontos aleatórios selecionados conforme uma distribuição uniforme no quadrado $(0, 500) \times (0, 500)$. Para todas as instâncias, assumimos que $\rho = 50$ (o raio mínimo de curvatura), e $K = 40$ como o nível de discretização na etapa de cálcula das orientações.

Para encontrar soluções exatas do PCVAng e PCVE, utilizamos o solver ILOG CPLEX® 10.2 na formulação simplificada da Formulação 1 e na formulação de Dantzig-Fulkerson-Johnson [3]. Note que as restrições (5) na Formulação 1, e similares na formulação do PCVE, são dadas ao solver apenas quando necessário para evitar circuitos pré-hamiltonianos.

Visto que $c(p_i, p_j) \leq d(p_i, \phi_i, p_j, \phi_j)\ \forall i, j$, fica claro que o comprimento do circuito solução do PCVE é um limite inferior para o comprimento da solução ótima do PVCD. A Tabela 1 mostra os resultados computacionais para cada algoritmo aplicado nas 20 instâncias (as 10 primeiras têm 20 pontos, e as 10 últimas têm 21 pontos). Para cada algoritmo, a Tabela 1 apresenta o custo da solução, i.e., o comprimento da trajetória Dubins, e a razão entre esse custo e o limite inferior.

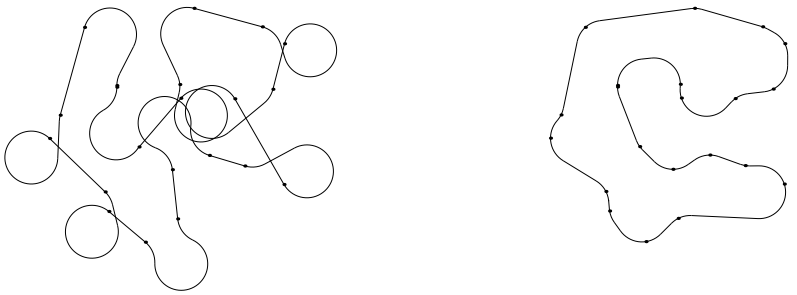
A Figura 3 ilustra as soluções obtidas pelos algoritmos EAA e AngD aplicados na instância 11.

Dentre os quatro algoritmos, ED alcançou o menor comprimento da trajetória em 25% das instância, e o AngD alcançou o menor comprimento em 85% das instâncias. Nem EAA nem AngAA alcançaram o menor comprimento em qualquer instância.

Se compararmos o EAA com AngAA e o ED com AngD, veremos que a solução do PCVAng se mostra melhor do que a solução do PCVE para a etapa de circuito dos algoritmos. Para a etapa de trajetória, se compararmos EAA com ED e AngAA com AngD, veremos que o uso do algoritmo de Dijkstra certamente é um método melhor do que o Algoritmo Alternante. A média do excesso sobre o limite inferior é 122% para o

	Comprimento trajetória Dubins				Razão com limite inferior			
Instância	**EAA**	**ED**	**AngAA**	**AngD**	**EAA**	**ED**	**AngAA**	**AngD**
1	4305,85	2688,15	4357,87	**2462,35**	2,17	1,36	2,20	1,24
2	4231,56	2698,42	3682,61	**2628,88**	2,15	1,37	1,87	1,34
3	3949,40	2628,16	3949,82	**2624,98**	2,54	1,69	2,54	1,69
4	4469,56	**2647,04**	3657,88	2770,28	2,24	1,33	1,84	1,39
5	4135,17	3173,63	3571,63	**2876,89**	1,99	1,52	1,72	1,38
6	4027,10	**2570,08**	3712,75	2759,25	2,27	1,45	2,09	1,56
7	4413,45	**3195,50**	4118,16	3197,21	2,23	1,62	2,09	1,62
8	4047,18	**2805,82**	4000,08	**2805,82**	1,88	1,30	1,86	1,30
9	3862,22	2902,28	4252,23	**2701,50**	2,39	1,80	2,63	1,67
10	4033,08	3030,89	3956,10	**2965,58**	2,12	1,59	2,08	1,56
11	4553,01	2256,05	4494,25	**2180,05**	2,31	1,15	2,28	1,11
12	4325,40	3117,82	3821,44	**3117,75**	2,18	1,57	1,92	1,57
13	4547,60	3240,19	3801,51	**3143,34**	2,16	1,54	1,80	1,49
14	4534,95	3064,07	4041,18	**2856,48**	2,12	1,44	1,89	1,34
15	5004,55	2899,85	3825,48	**2510,49**	2,34	1,36	1,79	1,17
16	3565,93	2594,33	3230,83	**2395,70**	2,05	1,49	1,85	1,37
17	4394,56	3007,35	3774,31	**2681,42**	2,43	1,66	2,08	1,48
18	4503,15	3164,34	4383,80	**3034,21**	2,31	1,63	2,25	1,56
19	4563,84	**2773,08**	4563,84	**2773,08**	2,50	1,52	2,50	1,52
20	4460,31	3307,51	3718,95	**2901,43**	2,10	1,56	1,75	1,37
				Média	2,22	1,50	2,05	1,44

Table 1: ***Comprimento da trajetória Dubins das soluções dos quatro algoritmos em 20 instâncias aleatórias. Em cada instância, o valor em negrito é a melhor solução encontrada.***



(a) EAA **(b)** AngD

Figure 3: ***Comparação entre as trajetórias Dubins obtidas por dois algoritmos na instância 11.***

EAA, 50% para o ED, 105% para o AngAA e 44% para o AngD. Todos os algoritmos executaram em menos de 30 segundos para todas as instâncias, sendo a etapa de circuito a mais demorada.

4. Conclusão

Neste trabalho, desenvolvemos uma nova heurística para o Problema do Caixeiro Viajante Dubins. A nova heurística usa o mesmo esquema de decomposição de Savla *et al.* [6], mas resolve ambas as etapas de forma diferente. Para a etapa de circuito, nós propusemos resolver o PCVAng no lugar do PCVE. Para a etapa de trajetória, usamos um algoritmo de caminho mínimo em um espaço de busca discretizado das orientações no lugar no Algoritmo Alternante.

Minimizar os ângulos de mudança de direção no circuito proporciona trajetórias Dubins mais curtas. Além disso, mostramos que o uso de um algoritmo de caminho

mínimo para computar as orientações em cada ponto resultam em trajetórias melhores do que as obtidas pelo método alternante. A nova heurística obteve soluções muito melhores do que aquelas obtidas pelo melhor método da literatura em todas as instâncias de teste com tempos de execução semelhantes.

Trabalhos para o futuro incluem um estudo cauteloso dos valores adequados para os pesos de cada objetivo no PCVAng, o uso de heurísticas para resolver o PCVAng permitindo a execução do algoritmo em instâncias maiores (veja figura 4), e o uso de técnicas de otimização não-linear para melhorar as orientações encontradas na etapa de trajetória do algoritmo.

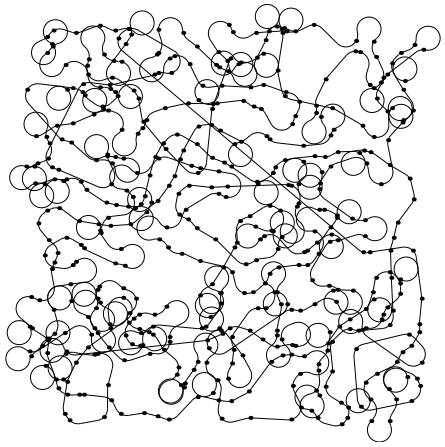


Figure 4: ***Solução de um método heurístico, em estudo, para o PCVD com 500 pontos.***

References

[1] Aggarwal, A., D. Coppersmith, S. Khanna, R. Motwani, and B. Schieber, *The angular-metric traveling salesman problem*, Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (1997), 221–229.

[2] Dubins, L.E., *On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents* American Journal of Mathematics **79** (1957), 497–516.

[3] Dantzig, G.B., D.R. Fulkerson, and S.M. Johnson, *Solution of large scale traveling salesman problem*, Oper.Res. **2** (1954), 393–410.

[4] LaValle, S.M., "Planning Algorithms", Cambridge University Press (2006) (também disponível em http://msl.cs.uiuc.edu/planning/).

[5] Le Ny, J., E. Frazzoli, and E. Feron, *The curvature-constrained traveling salesman problem for high point densities*, IEEE Conf. on Decision and Control (2007), 5985–5990.

[6] Savla, K., E. Frazzoli, and F. Bullo, *On the point-to-point and traveling salesperson problems for Dubins' vehicle*, American Control Conference, Portland, OR (2005), 786–791.

[7] Shkel, A.M., and V. J. Lumelsky, *Classification of the Dubins set*, Robotics and Autonomous Systems (2001), vol. 34, 179–202.