



UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO
CURSO DE ENGENHARIA MECATRÔNICA E ENGENHARIA DA COMPUTAÇÃO

RELATÓRIO

ATYSON JAIME DE SOUSA MARTINS: N° 20190153956
JOSE LINDENBERG DE ANDRADE: N° 20200150293
JÚLIA COSTA CORRÊA DE OLIVEIRA: N° 20200149087
SAMUEL CAVALCANTI: N° 20200149318

Natal-RN
2021

ATYSON JAIME DE SOUSA MARTINS: N° 20190153956
JOSE LINDENBERG DE ANDRADE: N° 20200150293
JÚLIA COSTA CORRÊA DE OLIVEIRA: N° 20200149087
SAMUEL CAVALCANTI: N° 20200149318

Relatório apresentado à disciplina de Sistemas Robóticos Autônomos, correspondente à avaliação parcial da 2ª unidade do semestre 2021.2 do curso de Engenharia Mecatrônica e Engenharia da Computação da Universidade Federal do Rio Grande do Norte, sob orientação do **Prof. Pablo Javier.**

Professor: Pablo Javier Alsina.

Natal-RN
2021

RESUMO

Relatório

Lista de Figuras

1	Espaço de trabalho CoppeliaSim com Obstáculos	7
2	Obstáculos	7
3	Robô	8
4	Gráfico do espaço de trabalho com o caminho gerado pelos algoritmo de Diogo.	9
5	Gráfico do espaço configuração gerado a partir do espaço de trabalho.	9
6	Polígono exmplo para o algoritmo	10
7	Espaço de Trabalho com um grafo do tipo malha 50x50	11
8	Espaço de Configuração com um grafo do tipo malha 50x50	11
9	Cenário simples com obstáculos bem afastados	12
10	Mapa de calor do campo artificial	14
11	Caminho do campo artificial no espaço de trabalho	15
12	Caminho do campo artificial no espaço de configuração	16

Sumário

1	INTRODUÇÃO	6
2	PRIMEIRA META	7
3	SEGUNDA META	10
4	TERCEIRA META	12

1 INTRODUÇÃO

Com o intuito de desenvolver um Planejadores de Caminhos para um robô móvel, que permitam ao mesmo executar movimentos especificados em espaço povoado de obstáculos, sem colidir com os mesmos. Foi incluído obstáculos poligonais no espaço de trabalho simulado e considerado que o robô tenha um formato retangular. Esse trabalho foi dividido em três metas, a primeira meta visa implementar um algoritmo que mapeie os obstáculos do espaço de trabalho para o espaço de configuração e apresentar um caminho no espaço de configuração e espaço de trabalho. A segunda meta consiste em a partir dos obstáculos no espaço de configuração e de trabalho gerar um grafo, a qual deve-se implementar um algoritmo que navegue nesse grafo e busque encontrar os caminhos a qual um controlador deve seguir o caminho planejado. A terceira meta busca utilizar algoritmos de campos de potenciais para gerar caminhos e igualmente aos algoritmos de grafo, deve-se ser capaz visualizar esses caminhos no espaço de configuração e trabalho, além do fato de implementar um controlador que permita seguir o caminho.

2 PRIMEIRA META

Para que faça sentido ter um espaço de configuração, foram adicionados no simulador obstáculos, que podem ser vistos na figura 1; também foram adicionados pequenos objetos planares da cor roxa que representam os vértices de cada obstáculo, de modo que se possa automatizar o processo de coleta desses vértices e visualizá-los durante a simulação, o mesmo foi feito com o robô, sendo que no caso do robô, o retângulo que o representa é maior do que ele para que seja possível o robô rotacionar dentro desse retângulo. Os obstáculos podem ser vistos na figura 2 e o robô na figura 3.

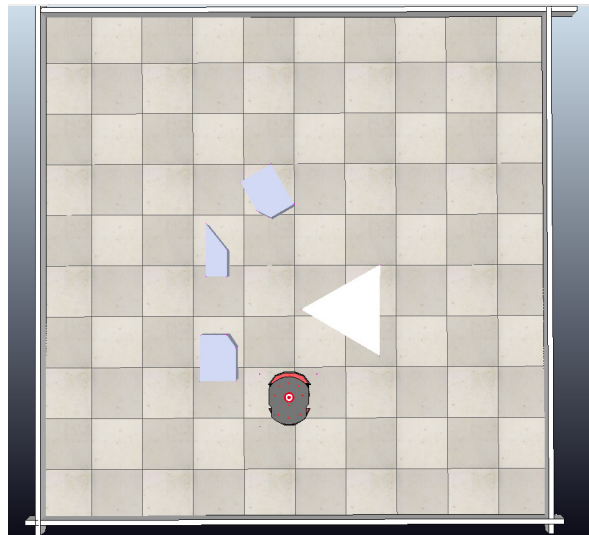


Figura 1: Espaço de trabalho CoppeliaSim com Obstáculos

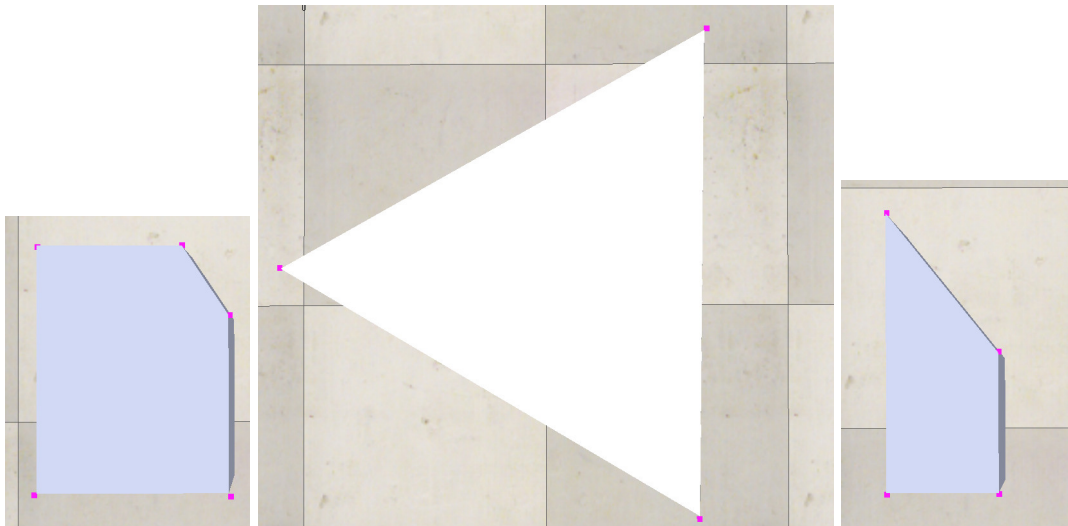


Figura 2: Obstáculos

O motivo pelo qual o robô tem um retângulo que lhe representa maior que o seu verdadeiro tamanho é para facilitar na geração do espaço de configuração, uma vez que tendo o espaço extra o robô pode girar e, dessa forma, o espaço de configuração pode ser obtido apenas transladando o retângulo. Tendo todos os vértices necessários, foi gerado o espaço de configuração seguindo o seguinte algoritmo:

- Escolhendo o sistema de coordenadas tendo como vista o centro geométrico do polígono do robô, caso esse polígono seja convexo, a média dos vértices é o centro geométrico.
- Refletindo os vértices do Robô, ou seja, dada o conjunto de vértices do robô A é calculado $-A = \{-a | a \in A\}$
- Calculado a soma de Minkowski entre o conjunto $-A$ e um conjunto de vértices de um o obstáculo P_i
- Calculado o convex hull do resultado da soma de Minkowski, onde o resultado do convex hull é o conjunto de vértices de um obstáculo no espaço de configuração.

Esse algoritmo foi retirado de Paulina Varshavskaya, a qual foi passado o texto original para uma versão escrita em markdown [5].

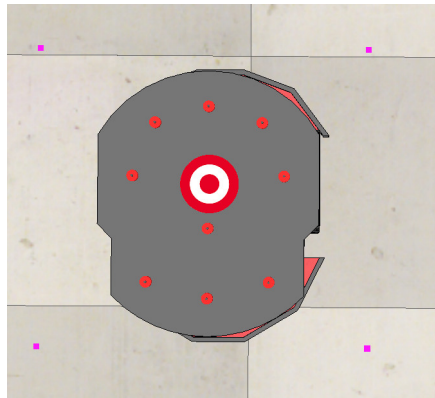


Figura 3: Robô

Utilizando o script que extrai os vértices da simulação, foi criado o gráfico do espaço de configuração, e com o caminho obtido utilizando polinômios interpoladores de terceiro grau foi gerada a figura 4. Utilizando o algoritmo e o robô em formato retangular foi obtido o espaço de configuração da figura 5. Podemos observar nessa figura um caminho gerado pelos polinômios interpoladores de Diogo que é representado linha azul. O espaço de configuração demonstra que existe um risco do robô colidir com os obstáculos.

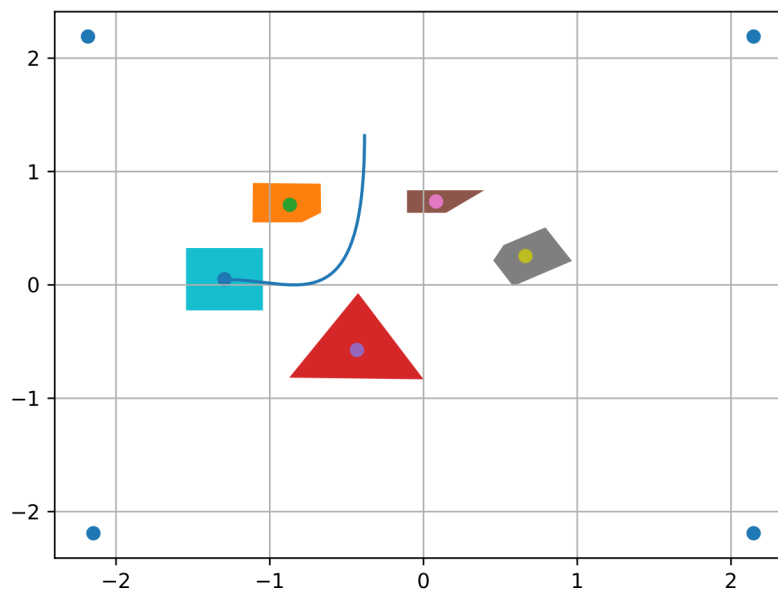


Figura 4: Gráfico do espaço de trabalho com o caminho gerado pelo algoritmo de Diogo.

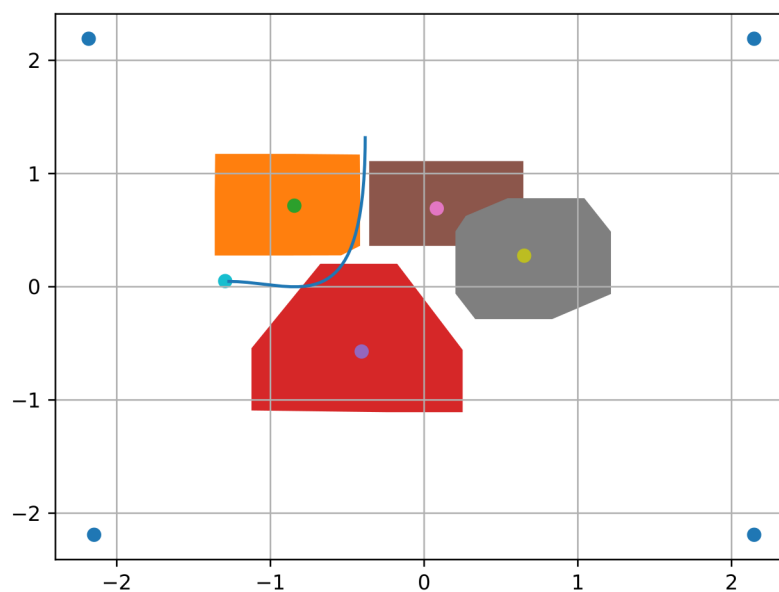


Figura 5: Gráfico do espaço configuração gerado a partir do espaço de trabalho.

3 SEGUNDA META

Nessa meta é mapeado o espaço de configuração para um grafo e utilizado A^* a fim de gerar um caminho entre um ponto a outro nesse grafo. Para gerar o grafo foi criada uma malha igualmente espaçada 50x50, totalizando um grafo com 2500 nós, no entanto parte desses nós estão dentro dos polígonos do espaço de configuração que representam os obstáculos, ou seja, esses pontos não geram um caminho válido. Para resolver esse problema, foi utilizado um algoritmo que verifica se um ponto está ou não contido em um polígono, que consiste em três passos simples:

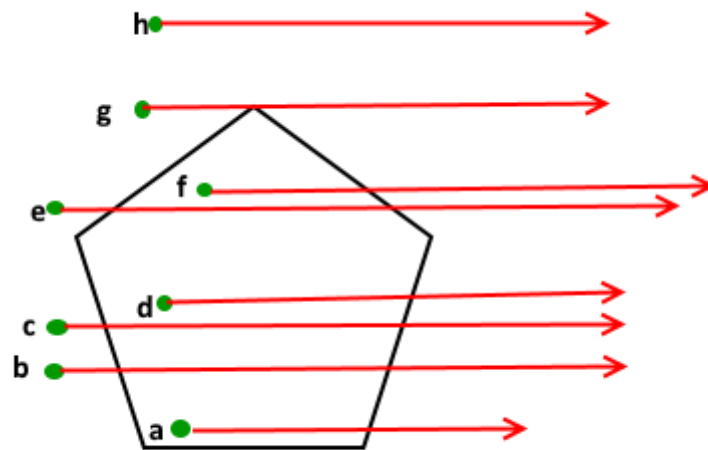


Figura 6: Polígono exemplo para o algoritmo

- desenhe uma linha horizontal a partir do lado direito do ponto até infinito
- contar o número de vezes que essa linha intersecta com os lados do polígono
- Se o ponto estiver dentro do polígono, o número de interseções é ímpar ou o ponto se encontra em cima do lado do polígono, caso contrário o ponto está fora do polígono.

Esse algoritmo foi retirado do **geeksforgeeks**. Utilizando a malha e o algoritmo para calcular se um ponto está ou não dentro do polígono e o algoritmo A^* para determinar o melhor caminho, foram geradas duas figuras: a figura 7, que demonstra o funcionamento do o algoritmo de colisão do polígono com o ponto e também mostra, através do seguimento laranja, o caminho sugerido pela A^* . Já a figura 8 mostra as mesmas informações só que no espaço de configuração. Também foi feito um vídeo mostrando o controlador Frederico no seu trabalho de seguir o caminho gerado pela A^* no espaço de configuração [2]

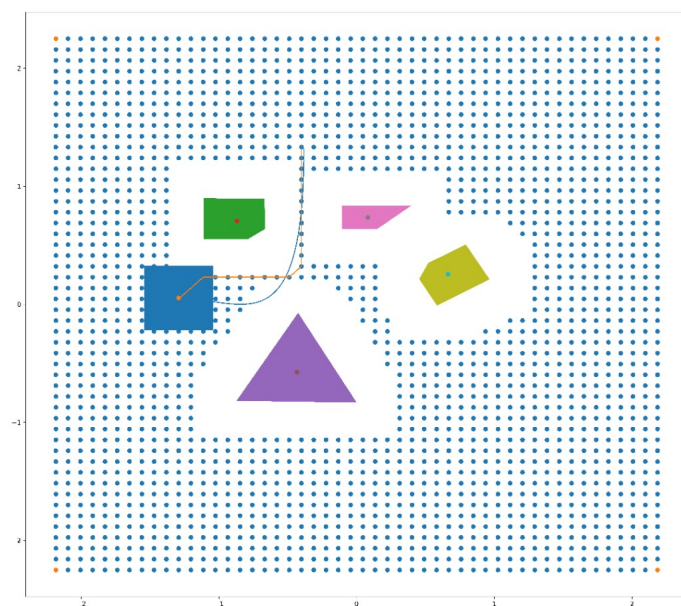


Figura 7: Espaço de Trabalho com um grafo do tipo malha 50x50

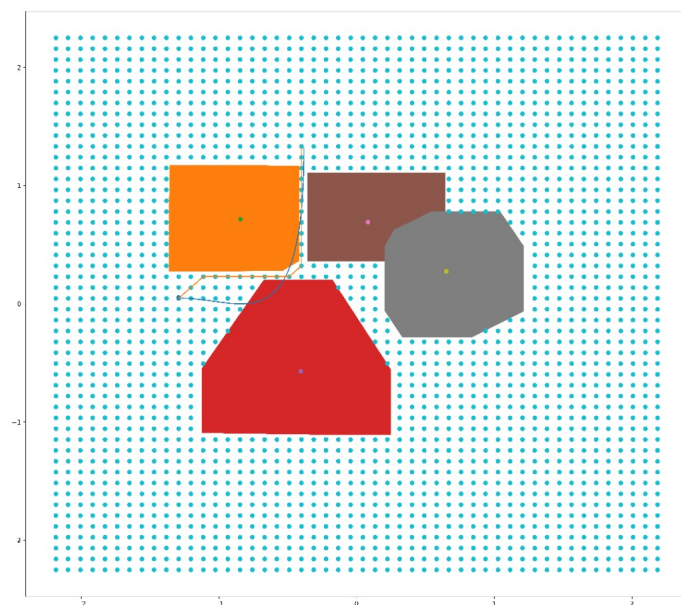


Figura 8: Espaço de Configuração com um grafo do tipo malha 50x50

4 TERCEIRA META

Nessa meta foi utilizado a ideia de campos potenciais artificiais para criação de caminhos. Onde a orientação final é uma partícula atrativa, que emite um campo artificial P_a dado por:

$$P_a = \frac{1}{2} K_p \|s_i - s_g\| \quad (1)$$

onde K_p é uma constante que nos nossos testes foi assumido 0.5 e $\|s_i - s_g\|$ é a distância euclidiana entre as posição s_i e a posição objetivo (goal) s_g , ou seja, quando maior a distância entre o objetivo e a posição, maior é o valor campo. Já os obstáculos emitem um campo artificial repulsivo P_o dado por:

$$\begin{aligned} & \frac{1}{2} \eta \left(\frac{1}{\|s_i - s_o\|} - \frac{1}{RR} \right)^2 & \text{se } \frac{\|s_i - s_o\|}{2} \leq RR \\ & 0 & \text{caso contrário} \end{aligned} \quad (2)$$

onde, nos nossos testes $\eta = 110$ e RR é um suposto raio do robô que no caso é a distância entre o centro do quadrilátero com o vértice mais distante do centro. No caso foi adotado que P_i é todo vértice de um obstáculo, por exemplo, o obstáculo cujo o formato é de um triângulo possui três fontes de campo repulsivo. Nessa etapa foi utilizado 2 algoritmos no campo potencial, o primeiro foi a decida do gradiente, uma vez que o valor do campo tende a decrescer com a distância entre a posição atual do robô e a posição desejada, no caso de cenários mais simples como no caso da figura 9, funciona perfeitamente.

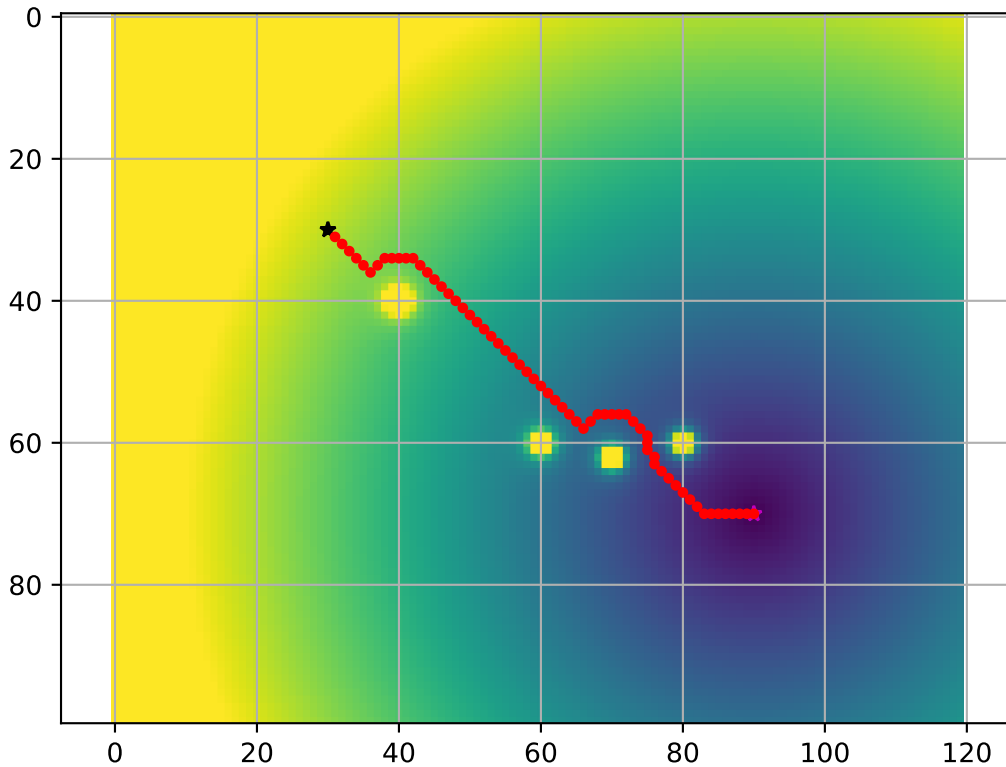


Figura 9: Cenário simples com obstáculos bem afastados

Já no nosso cenário, onde temos três obstáculos muito próximos um dos outros. Observamos que a decida do gradiente não funciona, sua tentativa pode ser observada pela sequência de pontos vermelhos na figura 10 que em um dado momento se recusa a se afastar do ponto cinza, para futuramente encosta-la. Então aproveitando a implementação do algoritmo A^* foi utilizado o campo potencial artificial como heurística e também não é considerado nenhum valor de distância entre os pontos da malha dessa forma o custo entre dois nós do grafo é dada pela seguinte equação:

$$C(o, t) = |P(t) - P(o)| \quad (3)$$

onde $||$ representa o módulo e o é o nó de origem e t o nó de destino então no caso o custo entre dois nós é a diferença em modulo dos valores dos campos artificiais do nó alvo(target) e nó origem, dessa forma a A^* vai buscar minimiza além nos menores valores do campo, como também as menores transições, lembrando que devido a equação de repulsão no momento que a distância da posição atual s_i é duas vezes menor que menor o raio do robô, então ocorre uma transição elevada, que a A^* por si só evita. Também foi feita uma segunda modificação, que durante o algoritmo a partir de um nó busca o seus vizinhos, é feito uma filtragem desses vizinho que só serão ser possíveis de serem visitados se o valor de seu campo for menor que T_p , onde nos nossos testes $T_p = 50$. Dessa forma o algoritmo busca o menor potencial, com a transição menos brusca e em nós com valores de campo menores que 50. Podemos ver o caminho gerado pelo algoritmo A^* pela sequência de pontos da cor ciano na figura 10. Cores claras no mapa de calor significa um alto valor de campo, cores mais escuras um baixo valor de campo.

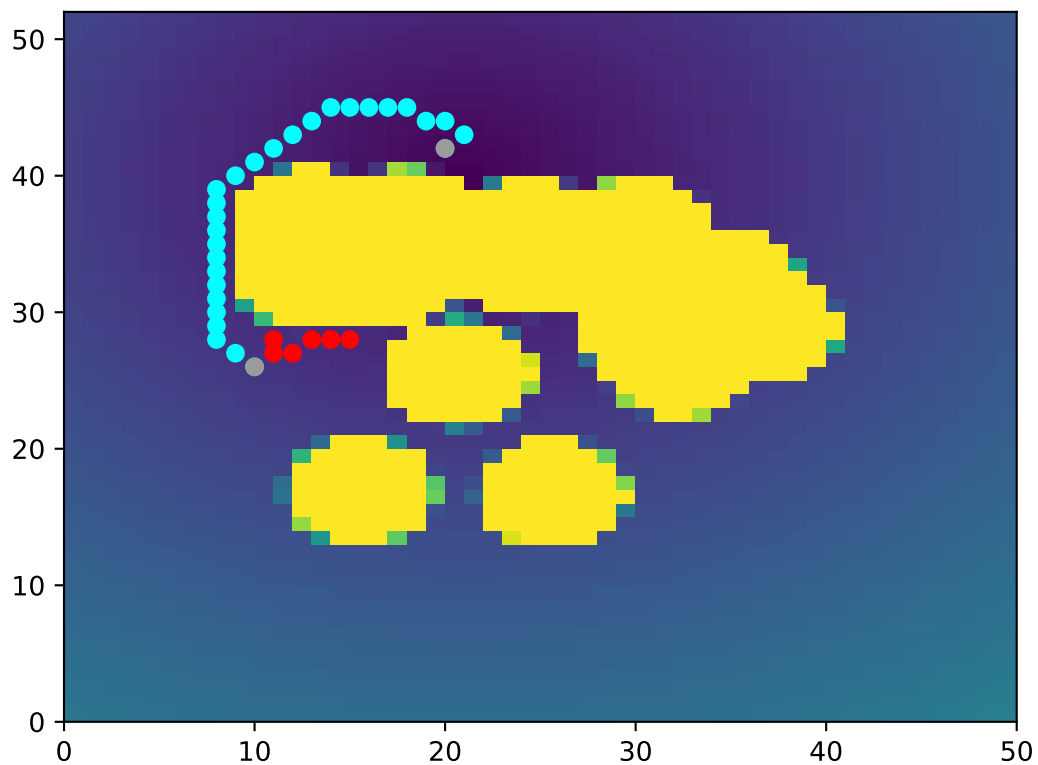


Figura 10: Mapa de calor do campo artificial

No espaço de configuração o caminho da A^* é a sequência de pontos amarelos e o caminho de pontos cianos é o algoritmo da decida do gradiente que podem ser visto na figura 11, essa mesma sequência de cores foi adotada no gráfico da figura 12 que mostram o caminho no espaço de configuração. Também foi feito um vídeo mostrando o controlador Frederico buscando seguir o caminho gerado pela A^* no campo artificial potencial: [3]

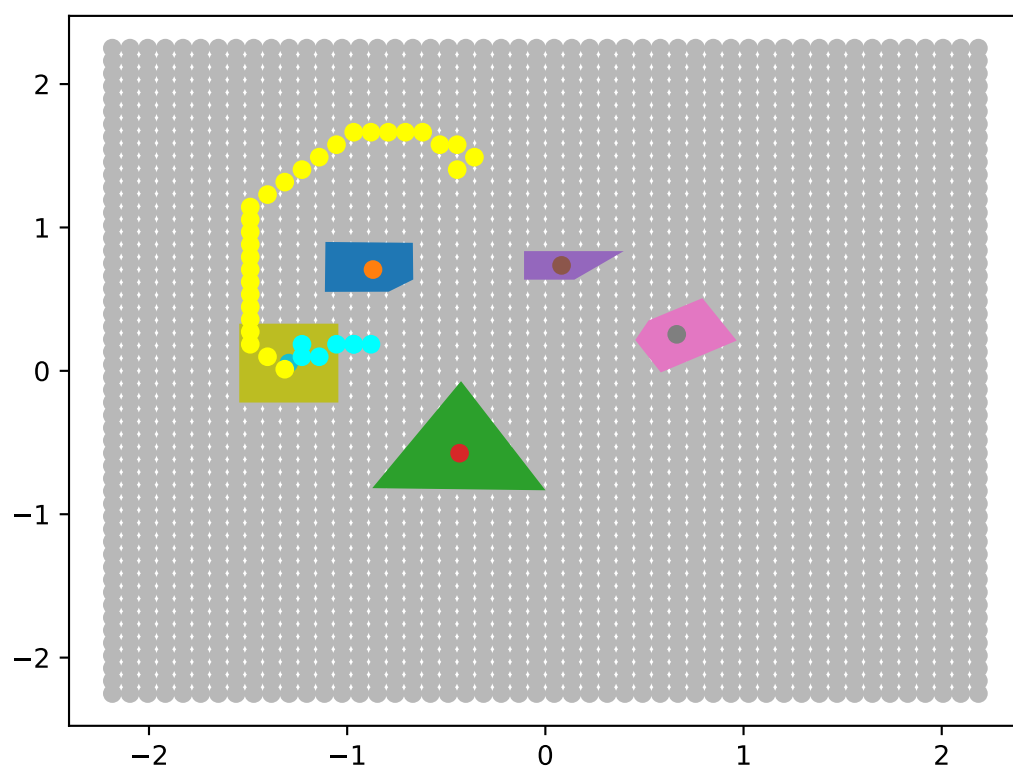


Figura 11: Caminho do campo artificial no espaço de trabalho

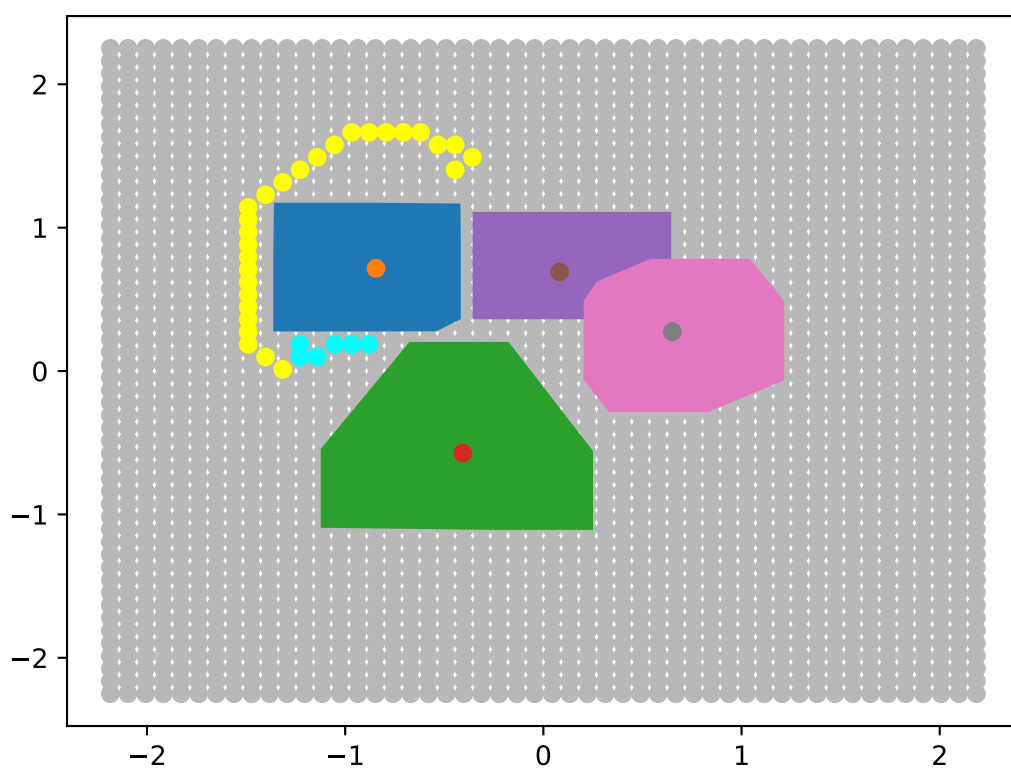


Figura 12: Caminho do campo artificial no espaço de configuração

Referências

- [1] Simulador CoppeliaSim. Coppelia Robotics, 2021. Disponível em:
<https://www.coppeliarobotics.com/>. Acesso em 22 de nov. de 2021.
- [2] Controlador Federico Seguindo Caminho gerado pela A^* no Espaço de Trabalho:
<https://youtu.be/MZ1BS-d60-Y>. Acesso em 10 de jan. de 2022.
- [3] Controlador Federico Seguindo Caminho gerado pela A^* no Campo Potencial Artificial:
<https://youtu.be/jtnPh2pygI4>. Acesso em 10 de jan. de 2022.
- [4] Repositório contendo todo código do projeto:
https://github.com/samuel-cavalcanti/controle_cinematico_sistemas_autonomos_ufrn. Acesso em 10 de jan. de 2022.
- [5] Obstacles in Configuration space by Paulina Varshavskaya:
https://github.com/samuel-cavalcanti/controle_cinematico_sistemas_autonomos_ufrn/blob/main/docs/notes_on_configuration_space.md