



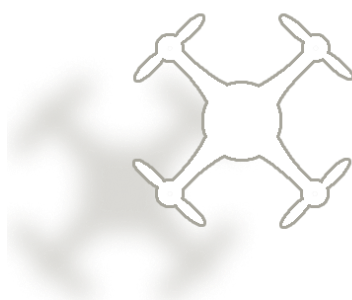
UNIVERSIDADE FEDERAL DO CEARÁ – UFC

CAMPUS SOBRAL

CURSO: ENGENHARIA DA COMPUTAÇÃO

DISCIPLINA: PARADIGMAS DE LINGUAGEM DE PROGRAMAÇÃO

## EQUIPE 2 : NAVEGAÇÃO DE DRONES (PROLOG)



ANDRÉ VERAS - 385099

FRANCISCO VILMAR RODRIGUES DA SILVA - 394029

KENEDY RODRIGUES COSTA - 478937

WILLIAM DO VALE MESQUITA - 390185

SOBRAL – 02 de DEZEMBRO de 2019

## SUMARIO

INTRODUÇÃO .....	02
OBJETIVOS .....	02
DESENVOLVIMENTO .....	03
RESULTADOS OBTIDOS .....	04
CONCLUSÕES .....	04
REFERENCIAS .....	05

## INTRODUÇÃO

- O que são drones?

Drone é um veículo aéreo não tripulado e controlado remotamente que pode realizar inúmeras tarefas. Utilizados tanto em guerras quanto para entregar pizza, estes equipamentos estão cada vez mais presentes em lugares do mundo.

- O que é o Prolog?

Prolog é uma linguagem de programação que se enquadra no paradigma de programação em lógica matemática. É uma linguagem de uso geral que é especialmente associada com a inteligência artificial e linguística computacional. Consiste numa linguagem puramente lógica, que pode ser chamada de Prolog puro, e numa linguagem correta, a qual acrescenta Prolog puro com componentes lógicos.

## OBJETIVOS

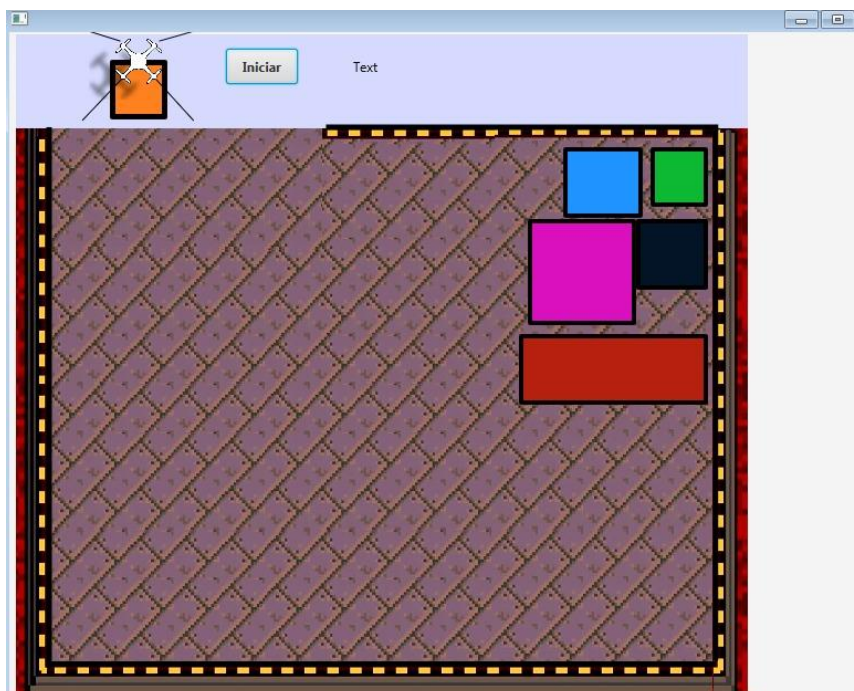
O objetivo do drone é chegar a um determinado destino, então ele deve seguir até esse ponto, mas à medida que encontrar obstáculos deverá desviar e continuar para o ponto.

## DESENVOLVIMENTO

Para fazermos a interface visual e a comunicação com o drone, utilizamos prolog para implementar a lógica dos sensores e desenvolvemos a interface para simularmos o drone, seus obstáculos e destino em JavaFx.

Por ser uma tela interativa, primeiro definimos o destino do drone movimentando o bloco com a cor correspondente em laranja. Temos a opção de movimentarmos os demais blocos para formarmos obstáculos na qual o drone deverá detectar e contornar após o processamento lógico pelo Prolog.

Através do Java, foi criado o cenário em que o drone iria percorrer e com a auxílio do Prolog para tomadas de decisão. Algumas funções no Java são de essencial importância para o drone, entre elas pode-se destacar a função colisão, movimento e trajeto.



interface do programa – Fonte: Autor

Para a animação foi usada a Classe `TranslateTransition` do `javaFX`, onde é possível setar um ponto de destino usando os métodos `SetToX` e `SetToY`, assim como obter as coordenadas desejadas pelo `getTranslateX` e `getTranslateY`



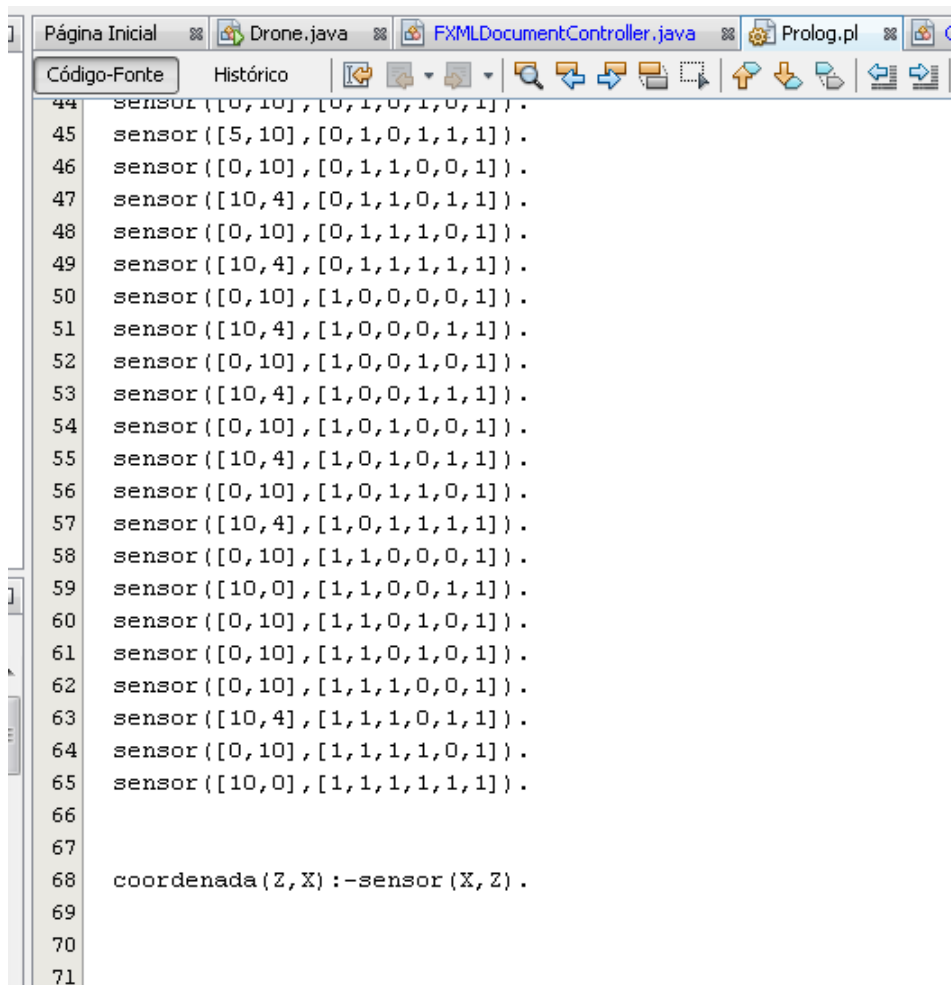
Aproveitando os recursos do JavaFX foi criado um método que deverá testar se um ou mais sensores estão colidindo com os obstáculos, o método **colisao()** retorna true caso haja a colisão dos sensores e implementa um conjunto de regras para controlar a orientação do eixo X e Y.

```
private boolean colisao() {  
  
    int controle = 0;  
  
    if (Shape.intersect(s1, barreira2).getBoundsInLocal().isEmpty() == false  
        || Shape.intersect(s1, barreira3).getBoundsInLocal().isEmpty() == false  
        || Shape.intersect(s1, barreira4).getBoundsInLocal().isEmpty() == false  
        || Shape.intersect(s1, barreira).getBoundsInLocal().isEmpty() == false  
        || Shape.intersect(s1, barreira5).getBoundsInLocal().isEmpty() == false  
        || Shape.intersect(s1, barreira6).getBoundsInLocal().isEmpty() == false  
        || Shape.intersect(s1, barreira7).getBoundsInLocal().isEmpty() == false  
        || Shape.intersect(s1, barreira8).getBoundsInLocal().isEmpty() == false  
        || Shape.intersect(s1, barreira9).getBoundsInLocal().isEmpty() == false) {  
  
        senso[0] = 1;  
        controle++;  
  
        s1.setStroke(Color.RED);  
        s1.setVisible(true);  
    }  
}
```

Método de Colisão

Por fim para tomar as decisões baseadas nos sensores ocupados foi usada uma ligação com prolog, de modo a facilitar os testes, a conexão foi feita via classe java usando a biblioteca nativa do prolog *jpl*.

O *jpl* basicamente recebe uma lista em binário de 6 posições, uma para cada sensor, onde o 1 representa a colisão do sensor e retorna uma lista de duas posições indicando a coordenada na qual o drone deve seguir para desviar do obstáculo.



The image shows a screenshot of a Prolog IDE window. The title bar contains several tabs: 'Página Inicial', 'Drone.java', 'FXMLDocumentController.java', 'Prolog.pl', and a partially visible 'C'. Below the title bar is a toolbar with icons for file operations and editing. The main editor area displays Prolog code. Lines 44 through 65 contain a series of 'sensor' facts, each with two arguments: a coordinate pair [X, Y] and a 7-bit binary vector. The coordinates alternate between [0, 10] and [10, 4]. The binary vectors vary across the facts. Line 68 defines a predicate 'coordenada(Z, X) :- sensor(X, Z)'. Lines 66, 69, 70, and 71 are empty.

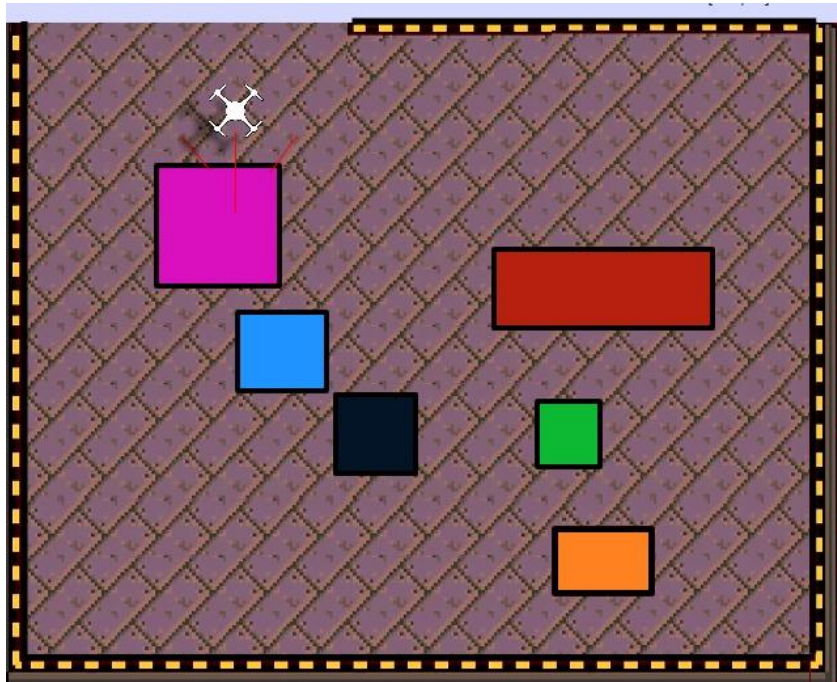
```
44 sensor([0,10],[0,1,0,1,0,1]).
45 sensor([5,10],[0,1,0,1,1,1]).
46 sensor([0,10],[0,1,1,0,0,1]).
47 sensor([10,4],[0,1,1,0,1,1]).
48 sensor([0,10],[0,1,1,1,0,1]).
49 sensor([10,4],[0,1,1,1,1,1]).
50 sensor([0,10],[1,0,0,0,0,1]).
51 sensor([10,4],[1,0,0,0,1,1]).
52 sensor([0,10],[1,0,0,1,0,1]).
53 sensor([10,4],[1,0,0,1,1,1]).
54 sensor([0,10],[1,0,1,0,0,1]).
55 sensor([10,4],[1,0,1,0,1,1]).
56 sensor([0,10],[1,0,1,1,0,1]).
57 sensor([10,4],[1,0,1,1,1,1]).
58 sensor([0,10],[1,1,0,0,0,1]).
59 sensor([10,0],[1,1,0,0,1,1]).
60 sensor([0,10],[1,1,0,1,0,1]).
61 sensor([0,10],[1,1,0,1,0,1]).
62 sensor([0,10],[1,1,1,0,0,1]).
63 sensor([10,4],[1,1,1,0,1,1]).
64 sensor([0,10],[1,1,1,1,0,1]).
65 sensor([10,0],[1,1,1,1,1,1]).
66
67
68 coordenada(Z,X):-sensor(X,Z).
69
70
71
```

Predicados do prolog

Com as ferramentas em mãos vamos aos resultados.

## RESULTADOS OBTIDOS

O drone consegue visualizar o trajeto que deverá percorrer até alcançar o destino, traçando uma linha ideal, ou seja, uma reta. No entanto, por ter obstáculos no trajeto o drone recebe novas coordenadas através do Prolog e consegue desviar de forma eficiente em grande parte dos testes realizados.



Detecção de colisão – Fonte: Autor

## CONCLUSÕES

Em todo o conjunto, ficamos satisfeito com o resultado obtido no projeto, além de conhecer uma linguagem nova e que poderá nos ajudar em projetos futuros. Ocorreram imprevistos, mas conseguimos contornar grande parte através de atualizações no código e realizações de testes em que podemos visualizar uma vasta possibilidade de movimentos que o drone poderá a vir a fazer.



## REFERÊNCIAS

<https://canaltech.com.br/produtos/o-que-e-drone/>

<https://pt.wikipedia.org/wiki/Prolog>

<https://www.youtube.com/watch?v=PwT72Yvrc5c>