ESTRATÉGIA DE PERSEGUIÇÃO – PROJETO PAC-MAN

1. Integrantes do projeto e repositório do projeto

Luís Fernando Almeida – 8102 – T1

Bruno Marques da Silva – 6972 – T1

https://github.com/luissaster/pacman-clone-cpp

2. Introdução

Este documento tem como objetivo explicar e detalhar a estratégia de perseguição do Pac-Man empregada por parte dos fantasmas.

3. Estratégia de perseguição

Em nosso projeto, não foi empregado nenhum algoritmo popular de *pathfinding*, tais como o algoritmo A* ou o algoritmo Dijkstra. Como alternativa, levando em consideração as datas de entrega e o nosso conhecimento atual, optamos por uma implementação de estratégia de perseguição mais simples, a qual será descrita nesse documento.

Primeiramente, assim como no Pac-Man original, nosso projeto conta com os quatro fantasmas originais: Blinky, Pinky, Inky e Clyde. Nossa intenção é que, eventualmente, cada um deles apresente seu próprio mecanismo de perseguição do Pac-Man, buscando a máxima fidelidade ao jogo original. Para isso, a função "*chasePacman*" assume um comportamento polimórfico, na qual cada fantasma irá abordar a perseguição de uma forma diferente.

```
class Ghost : public Entity {
public:
    Ghost();
    ~Ghost();
    virtual void chasePacman(std::vector<std::vector<char>> mapa, int pacmanX, int pacmanY) = 0;
    void randomDirection(std::vector<std::vector<char>> mapa);
    void checkRandomDirection(std::vector<std::vector<char>> mapa);
    virtual void renderGhost(ALLEGRO_BITMAP* img, int sprite) = 0;
protected:
    int ghostType; // 1 - Blinky 2 - Inky 3 - Pinky 4 - Clyde
};
```

Figura 1: Classe "Ghost", a qual é herdada pelas classes "Blinky", "Pinky", "Inky" e "Clyde".

No momento atual, contamos apenas com a implementação para a parte do Blinky, ou seja, o fantasma vermelho. Assim como no jogo original, nosso Blinky também persegue ferozmente o jogador, priorizando caminhos que encurtarão a distância entre os dois. Na parte de código, implementamos essa perseguição na forma de oito checagens de condições, as quais representam todas as possibilidades de posição do Blinky em relação ao Pac-Man.

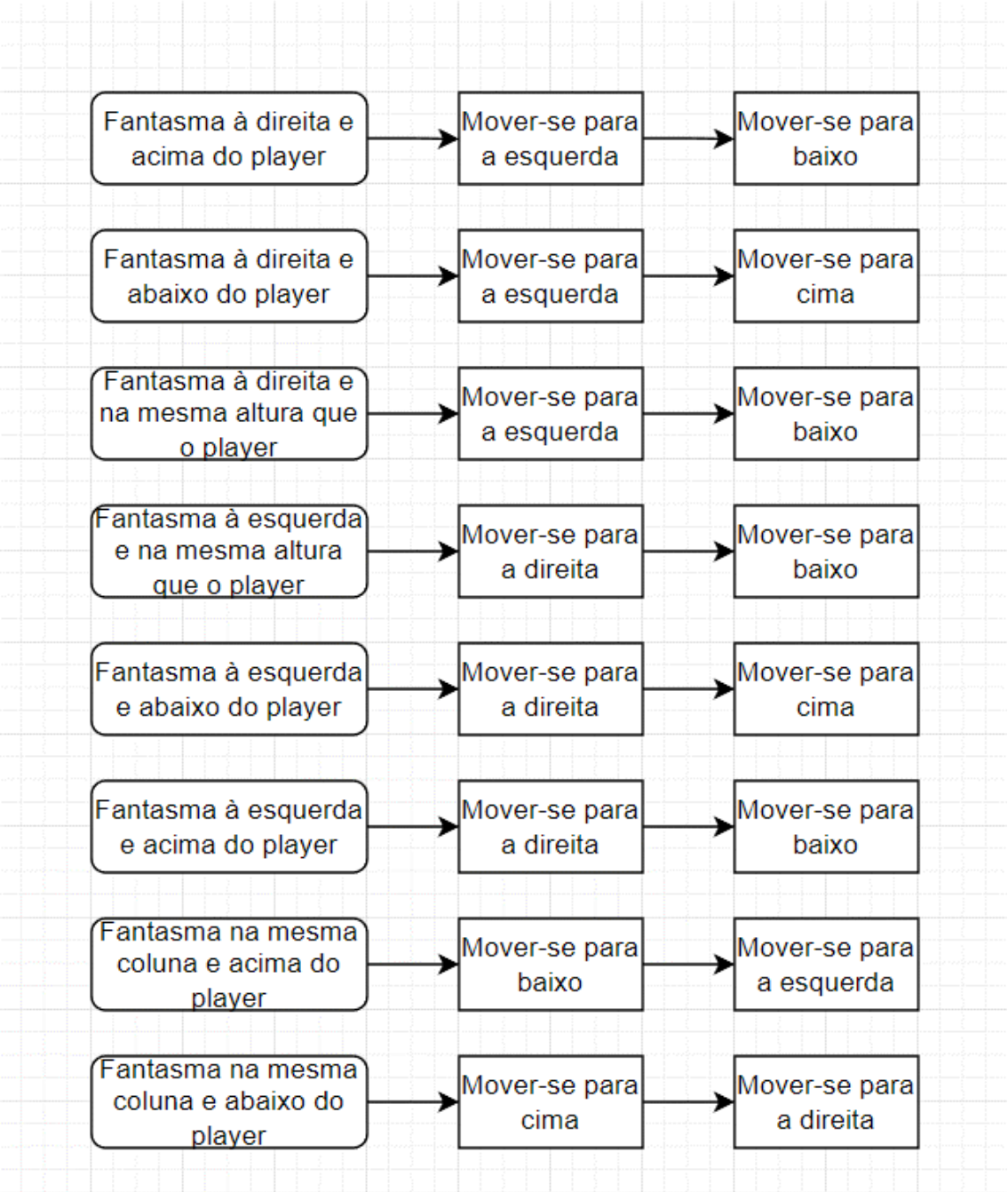

Figura 2: Diagrama representando todas as possibilidades de posições.

Pode-se perceber que não há nenhuma checagem pra quando o fantasma está na mesma coluna e na mesma linha que o player, porque isso implicaria que o fantasma colidiu com o player, o que resulta no fim do jogo. Tal checagem é feita em uma função separada, na classe do player.

Na implementação do código, essa mentalidade representada no diagrama resultou numa série de *IFs* e *ELSEs*, os quais são constantemente checados para decidir o próximo movimento do fantasma vermelho.

```
else if (blinkyX > pacX && blinkyY < pacY) { // Blinky is to the right of Pacman and above Pacman
    // PRIORITY: MOVE LEFT -> MOVE DOWN
    if (currentMove != RIGHT && checkEntityCollisionLeft(map)) {
        this->setNextMove(ALLEGRO_KEY_LEFT, map);
        return;
    }
    else if (currentMove != UP && checkEntityCollisionDown(map)) {
        this->setNextMove(ALLEGRO_KEY_DOWN, map);
        return;
    }
    else if (currentMove != LEFT && checkEntityCollisionRight(map)) {
        this->setNextMove(ALLEGRO_KEY_RIGHT, map);
        return;
    }
    else if (currentMove != DOWN && checkEntityCollisionUp(map)) {
        this->setNextMove(ALLEGRO_KEY_UP, map);
        return;
    }
}
```

Figura 3: Trecho de código que implementa a primeira condição do diagrama.

Sendo assim, essa função polimórfica é chamada constantemente dentro do *loop* principal do jogo, para que o fantasma sempre atualize sua posição atual em relação ao Pac-Man e priorize o melhor movimento possível.

Essa estratégia de perseguição, apesar de não ser a mais eficiente ou a mais acurada, gerou um resultado agradável no qual o fantasma aparenta estar sempre indo em direção ao player, em alguns momentos até tentando "cortar" o jogador, tomando rotas que o coloca diretamente na frente do jogador.

4. Referências

GameInternals - Understanding Pac-Man Ghost Behavior. Disponível em: <https://gameinternals.com/understanding-pac-man-ghost-behavior>.

Pac-Man Ghost AI Explained. **YouTube**, 13 jul. 2019. Disponível em: <https://www.youtube.com/watch?v=ataGotQ7ir8>