Memoria Práctica 2

Daniel Alconchel Vázquez

Nivel 0

Indicaciones del tutorial.

Nivel 1

Reutilizo el código del tutorial, pero cambio la estructura interna de los **Abiertos**, que pasa de ser `stack` a `queue`. Además, para optimizar un poco el algoritmo, compruebo que el nodo destino esté en abiertos (ya que sabemos que el primero que entre en abiertos será nuestra solución)

Nivel 2

Vuelvo a cambiar la estructura de los **Abiertos**, esta vez por un `priority_queue`. Además, modifico las estructuras de los nodos para tener más información, como la posesión de zapatillas o bikini o el coste acumulado de ir hasta ese nodo, desde el origen.

```
struct nodoBase {
    list<Action> secuencia;
};

struct nodo : public nodoBase
{
    estado st;

};

struct superNodo : public nodoBase {

    estadoCompleto st;
    int costeReal, costeTotal;

    // @pre bikini y zapatillas no ambas true - inconsistente
    explicit superNodo(const estado & _st, bool bikini, bool zapatillas) :
st(_st) {
        if(bikini && zapatillas) std::cerr << "incoherncia construccion
superNodo" << std::endl;
        if(bikini) st.acquireBikini();
        else if(zapatillas) st.acquireZapatillas();
    }

    int dameCosteHeuristico(const estado & destino) const {
        //return 0;
        return std::max(std::abs(destino.fila - st.fila),
std::abs(destino.columna - st.columna));
    }

    // @pre costeReal has been set
    void resolverCosteTotal(const estado & destino) {
        costeTotal = costeReal + dameCosteHeuristico(destino);
    }

};
```

Además, implemento una función para calcular los costes en cada caso y la heurística, que se basa en usar la norma infinito.

Nivel 3

No implementado

Nivel 4

No implementado