**RELAZIONE**

Indice:

1. Descrizione formale del dominio e dei vincoli
2. Acquisizione e classificazione degli input, stato iniziale e goal
3. Esecuzione e prestazioni

**Descrizione formale del dominio e dei vincoli**

Per risolvere questo problema di SmartVacuum, proponiamo l'utilizzo di un algoritmo di ricerca informata A\*, che permette di trovare la sequenza di azioni ottimale per il nostro robot aspirapolvere.

Definiamo la nostra ricerca come uno spazio degli stati, in cui ogni stato è definito dalla disposizione delle stanze e la posizione del robot.

Per la rappresentazione delle stanze utilizzeremo una matrice in cui ogni cella rappresenta una stanza. Ogni cella conterrà una lettera, che rappresenta la condizione della stanza; in particolare:

|  |  |  |
| --- | --- | --- |
| Lettera | Numeri per l’assegnazione | Descrizione |
| S | 0 | Stanza iniziale |
| C | 1 | Stanza pulita |
| D | 2 | Stanza sporca |
| V | 3 | Stanza molto sporca |
| X | 4 | Stanza non accessibile |
| F | 5 | Stanza finale |

Per quanto riguarda le azioni che il robot può compiere definiamo:

|  |  |  |
| --- | --- | --- |
| Azione | Spiegazione | Peso |
| Up | Movimento verso l’alto | 1 |
| Down | Movimento verso il basso | 1 |
| Right | Movimento verso destra | 1 |
| Left | Movimento verso sinistra | 1 |
| Clear | Pulisce la stanza sporca | 1 |
| Sclear | Pulisce la stanza molto sporca | 2 |

Sulla base delle azioni e della posizione dello smart vacuum, definiamo dei vincoli:

* Il robot ha la possibilità di muoversi solo tra stanze adiacenti e compiere un solo passo alla volta
* Quando ci si trova in una stanza sporca (D) o molto sporca (V) il robot non può spostarsi fino al completamento di pulizia della stanza.
* Se si incontra una stanza inaccessibile (X), il robot non potrà muoversi in quella stanza, quindi non avrà la possibilità di compiere il movimento che gli permette di entrare (se ad esempio la stanza inaccessibile si trova a destra il robot potrà muoversi solo in alto, in basso o a sinistra).

Lo stato del sistema può essere rappresentato come una tupla (s, r) dove s rappresenta lo stato delle stanze e r è la posizione attuale del robot.

**Acquisizione e classificazione degli input, stato iniziale e goal**

In questo problema gli input sono rappresentati da una matrice con al suo interno delle lettere.

L’acquisizione della matrice può avvenire in due modi:

* Scattando una foto della matrice
* Caricando l’immagine

In entrambi i casi l’immagine deve avere la griglia disegnata a computer mentre le lettere possono essere scritte sia manualmente che a computer.

Per la classificazione degli input utilizziamo eMNIST, un dataset contenente immagini di lettere minuscole e maiuscole scritte a mano, che fornisce un ampio spettro di dati per l'addestramento e la verifica di modelli di apprendimento automatico.

In particolare, nel progetto da noi realizzato per l’allenamento del modello da utilizzare per il riconoscimento delle lettere abbiamo utilizzato il 70% del dataset eMNIST per la fase di training e il restante 30% per la fase di test.

Il modello è stato allenato per un totale di 10 epoche, di seguito sono riportati i grafici della funzione Loss e l’accuratezza del modello.

Immagine che contiene testo, diagramma, linea, schermata

Descrizione generata automaticamente

La scelta di allenare il modello in 10 epoche è conseguente all’andamento del grafico; se avessimo allenato ulteriormente il modello si sarebbe infatti verificato overfitting; i valori di training avrebbero

superato quelli di test.

Immagine che contiene testo, diagramma, schermata, Diagramma

Descrizione generata automaticamente

In questo grafico possiamo vedere la misura di errore tra le previsioni effettuate dal modello e i valori reali dei dati.

Stato goal

L'obiettivo della ricerca è trovare una sequenza di azioni che porti il robot a pulire tutte le stanze e raggiungere la posizione finale.

Per la risoluzione del problema partiamo dall’identificazione dello stato iniziale; lo stato sarà ora costituito da tutti gli elementi della matrice, e la testina che corrisponde alla posizione di “S” nella matrice (problem.initial).

Il nostro stato goal è univoco e corrisponde ad una lista di interi e la posizione della testina nella stanza finale “F”, con la caratteristica che tutte le stanze sporche e molto sporche siano state pulite prima.

La funzione di valutazione euristica prenderà in considerazione la posizione iniziale e finale che il robot deve avere, valutando il precorso migliore al fine di raggiungere lo stato goal.

**Esecuzione e prestazioni**

Al fine di trovare la soluzione migliore per il problema abbiamo fatto diversi test utilizzando sia la ricerca informata che non informata.

* **Ricerca non informata**, breadth search graph, un algoritmo di ricerca in ampiezza basato sui grafi.
* **Ricerca informata**, utilizza l’algoritmo A\* con i diversi tipi di euristiche:

1. Path\_cost
2. Manhattan
3. Manhattan + path\_cost

Prestazioni:

Non possiamo parlare di prestazioni migliori in assoluto; dobbiamo relazionarle al path\_cost o al minor tempo impiegato.

* Se per prestazioni migliori vogliamo considerare la ricerca informata che ha il costo minore, allora le euristiche migliori sono quelle che utilizzano il path\_cost.
* Se per prestazioni migliori vogliamo considerare un minor tempo impiegato nella risoluzione del problema, allora l’euristica migliore è quella che utilizza Manhattan.

Di seguito sono riportati alcuni esempi di misurazioni (contenenti input, costi e tempo), mettendo a confronto ricerca informata e non informata.

|  |  |  |
| --- | --- | --- |
| breadth search graph | costo | tempo |
| 3x3 | 18 | 0.04621577262878418 |
| 5x5 | 40 | 0.24764537811279297 |
| 6x6 | 74 | 10 min 31s |

|  |  |  |
| --- | --- | --- |
| A\* con path-cost | costo | tempo |
| 3x3 | 18 | 0.03491806983947754 |
| 5x5 | 40 | 0.8689327239990234 |
| 6x6 | 74 | 21min 47s |
| 7x7 | n.a. | >30 min |

|  |  |  |
| --- | --- | --- |
| A\* con Manhattan | costo | tempo |
| 3x3 | 18 | 0.039154767990112305 |
| 5x5 | 42 | 0.054010868072509766 |
| 6x6 | 90 | 0.6314210891723633 |
| 7x7 | 163 | 2.3893775939941406 |

|  |  |  |
| --- | --- | --- |
| A\* con Manhattan + path-cost | costo | tempo |
| 3x3 | 18 | 0.0907137393951416 |
| 5x5 | 40 | 2.00551438331604 |
| 6x6 | 74 | 21 min 40s |
| 7x7 | n.a. | >30 min |

Degno di nota è il fatto che, per matrici piccole, il tempo impiegato utilizzando la ricerca non informata e la ricerca informata non presenta grande differenza; questo poiché per rendere più efficiente la risoluzione del problema abbiamo deciso di limitare in parte le azioni possibile del robot; se infatti questo si trova in una stanza sporca o molto sporca lo smart vacuum può solo pulire, il che diminuisce di molto il branching-factor.