TO do:

-Obiettivo del progetto

- Vincoli labirinto

- Software e Linguaggi utilizzati

1. Python
2. Pycharm
3. Spyder
4. GitHub
5. CoppeliaSim

-Come funziona l’algoritmo

1. Learning from past experiences (SLIDE lect3 10) -> albero
2. Modeling the environment -> albero
3. Albero
   1. Requisiti per creazione dei nodi
4. Stratificazione dei controllori. Fare uno schema sense, think act (?). Vedere slide lect2 del capitolo 2
5. Uso di stati
6. Rotazione

-Cose sparse:

1. Log
2. Redis
3. File di configurazione

-Class diagram

-Coppelia

-Parte fisica

-Problemi che abbiamo avuto:

1. Thread di Coppelia
2. Sensore per la rotazione
3. Balance

- Miglioramenti futuri e implementazioni future:

Intelligenza:  
intelligence = low/mid/high  
low: prende direttamente la lista di priorità nel file di conf  
mid: in modo casuale (distribuzione uniforme?) sceglie l'ordine della lista di priorità sia all'inizio sia durante l'esplorazione  
high: Metodo che in base allo storico dei risultati prodotti nell'esplorazione di diversi labirinti  
 sceglie la lista di priorità migliore che permette di esplorare il labirinto:  
 i) con minor tempo possibile  
 ii) con un numero di nodi più piccolo possibile  
 iii) con un numero di nodi dead end più piccolo possibile

Obiettivo del progetto

L’obiettivo del progetto consiste nel far esplorare al robot a quattro ruote FreeNove 4WD un labirinto la cui struttura non è nota al robot, in modo autonomo e intelligente. L’algoritmo che abbiamo ideato è una versione alternativa dell’algoritmo di ricerca in profondità DFS. Il robot inizia la sua esplorazione da un punto iniziale, chiamato radice o stato iniziale e, in base ai valori dei sensori di cui è dotato, l’algoritmo elabora i dati e decide la direzione e verso in cui si deve muovere il robot. Se il robot trova un vicolo cieco dal quale non è più possibile proseguire in avanti, l’algoritmo deciderà di far ripercorrere al robot il percorso ~~fatto fino ad ora~~ intrapreso fino a quando non trova un nuovo percorso da esplorare. L’esplorazione si conclude quando il robot trova il punto finale del labirinto. Nella presente documentazione, Agente e Robot assumono lo stesso significato.

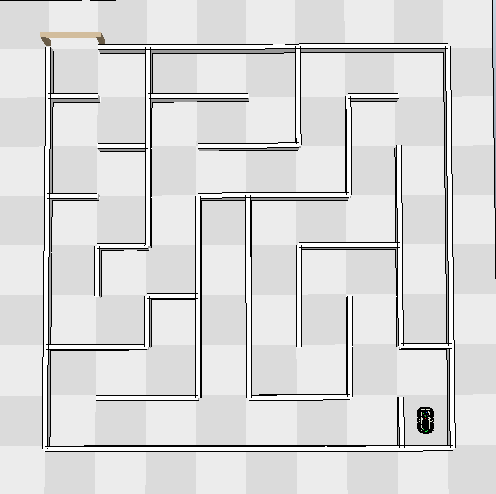
Vincoli del labirinto

Il labirinto deve avere una struttura che deve rispettare i/le seguenti vincoli/condizioni:

1. Struttura statica e non può mutare nel tempo
2. Un solo punto iniziale (radice/root)
3. Un solo punto finale/uscita dal labirinto (final)
4. Può essere rappresentato attraverso un albero semplice:
   1. Non ci sono cicli
   2. Non è possibile passare per uno stesso punto se si esplora in profondità
5. Presenza di vicoli ciechi
6. Percorso formato da muri che formano angoli di 90° e 180°
7. Esiste un percorso più breve che permette all’agente di andare dallo stato iniziale allo stato finale
8. Nella robotica virtuale:
   1. Distanza tra due muri deve essere almeno di 0.45 m.
   2. Spessore deve essere almeno di 0.10 m
9. Fisico:
   1. Distanza tra due muri deve essere almeno di ...
   2. Spessore deve essere almeno di ….

Inoltre:

1. Labirinto sconosciuto all’agente e quindi non può sapere a priori il percorso più breve dall’ingresso fino all’uscita (detto anche punto finale)
2. Uscita sconosciuta all’agente
3. L’agente tiene traccia delle decisioni che ha eseguito e lo fa senza la nozione di tempo
4. Al fine di prendere una decisione informata riguardo a quale azione eseguire l’agente deve essere in grado di osservare ciò che è attorno a lui



Algoritmo

L’algoritmo che abbiamo pensato si basa sulla ricerca in profondità, ma è molto differente alla struttura di un algoritmo DFS. L’algoritmo prende decisioni in base [trovare sinonimo] ai valori dei sensori di prossimità, allo stato (State), alla posizione logica della macchina (Position) e alla modalità di esplorazione (Mode).

Lo stato *State* del robot può assumere diversi valori come:

1. STARTING = -1
2. STOPPED = 0
3. RUNNING = 1
4. ROTATING = 2
5. SENSING = 3

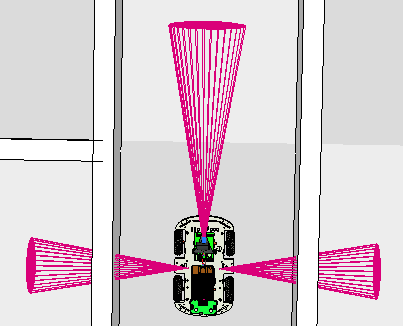
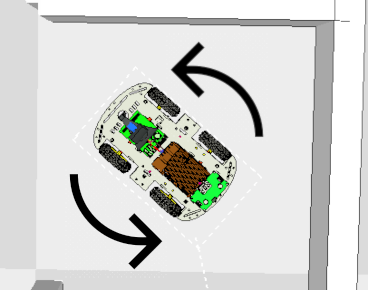
STARTING: Primo stato che assume il robot, nella fase di inizializzazione. Non può tornare più in questo stato durante l’esplorazione.

STOPPED: Quando il robot ha i motori spenti ed è fermo.

RUNNING: Robot in corsa durante il quale analizza continuamente, tramite i sensori, lo spazio circostante.

ROTATING: Stato in cui il Robot sta ruotando su sé stesso (asse z)

SENSING: Stato in cui il Robot è fermo e analizza, tramite i sensori, lo spazio circostante.



La posizione *Position* della macchina non è la posizione in coordinate, ma assume un altro significato. Può avere valori come:

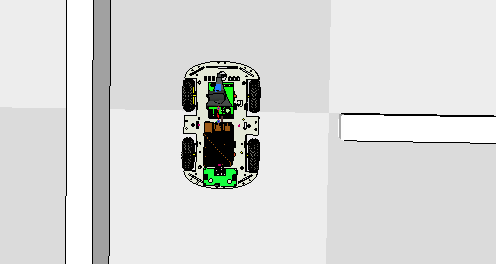
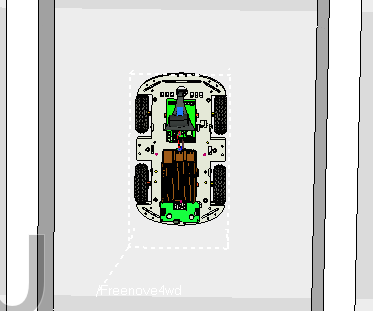
1. UNKNWON = 0
2. INITIAL = 1
3. CORRIDOR = 1
4. JUNCTION = 2

UNKNWON: Posizione sconosciuta al robot. Il robot deve analizzare lo spazio circostante tramite i sensori in modo da potersi posizionare nella posizione corretta. Dato che i sensori del robot sono Front, Left e Right è necessario ruotare il robot per capire se dietro di esso c’è un muro o meno. Una volta fatte le dovute rotazioni, la parte posteriore si affaccia al muro e il valore di Position diventa INITIAL.

INITIAL: Posizione iniziale, fase di inizializzazione. Il robot deve essere posizionato rispettando i seguenti vincoli:

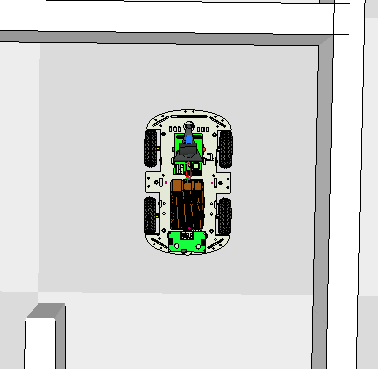
* La parte posteriore del robot deve essere contro il muro
* Il robot non può essere posizionato in un una giunzione/intersezione con 4 direzioni/strade disponibili
* Il robot non può essere posizionato in mezzo al corridoio dove è possibile andare in due sensi (in avanti e dietro)

CORRIDOR: Il Robot si trova nel corridoio. Definizione: *spazio ristretto dove la macchina può andare avanti o indietro ma non può svoltare né a sinistra né a destra*.



Img2: il bordo del muro a destra viene considerato come muro di un corridoio.

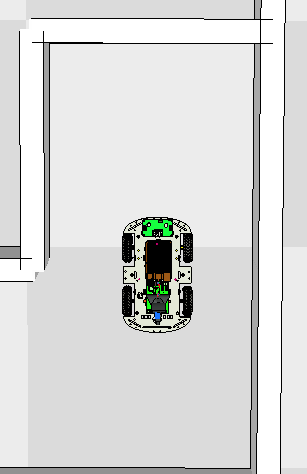
JUNCTION: Il Robot si trova in una giunzione. Definizione: *spazio dove la macchina può andare anche a destra e a sinistra oltre alla possibilità di andare avanti e indietro. La condizione necessaria e sufficiente è che può andare a sinistra o a destra.*



La modalità di esplorazione:

1. EXPLORING = 0
2. ESCAPING = 1

EXPLORING: Modalità di default. La macchina si trova in questa modalità quando deve esplorare spazi non conosciuti del labirinto. Durante questa fase, l’algoritmo crea in modo progressivo dei nodi che appenderà all’albero in modo sistematico. Tale albero mappa il labirinto in base a determinate condizioni che saranno spiegate dettagliatamente più avanti.

ESCAPING: Modalità che viene attivata quando la macchina trova un vicolo cieco. L’algoritmo deciderà di far ripercorrere al robot il percorso fatto fino ad ora fino a quando non trova un nuovo percorso da esplorare e, in tal caso, verrà riattivata la modalità EXPLORING.

Il cuore dell’algoritmo è all’interno del Controller. […]