Il progetto consiste in un robot in grado di muoversi autonomamente, evitando di scontrarsi con pareti ed ostacoli, e creando al contempo una mappa bidimensionale dell’ambiente circostante.

Il robot è dotato di due ruote collegate a due servomotori, che permettono di tener traccia degli spostamenti e quindi della posizione del robot con estrema precisione. Un sensore ad ultrasuoni (distanza massima 100 cm) è montato in cima al robot, e grazie a un terzo servomotore può ruotare di 360°. Il movimento del sensore è regolato da un arduino, mentre lettura, interpretazione ed elaborazione dei dati è affidata ad una Raspberry P3; la parte strutturale è stata stampata in 3D.

Vengono tenuti in memoria in un database MySql lite 3 le coordinate del robot, il suo orientamento, e la mappa dell’ambiente, che consiste di una griglia di caselle di 1×1 cm, le quali hanno tre stati possibili: libera, contenente un ostacolo, e un terzo stato utile al movimento intelligente del robot.

Il sensore viene attivato solo con il robot fermo; dopo una rilevazione, il sensore ruota di 5° e se ne effettua un’altra, fino a svolgere un giro completo. Se una data misurazione rileva un ostacolo a una certa distanza, partendo dalle coordinate del robot, dal suo orientamento e dall'angolo del sensore è facile risalire alla casella della griglia contenente l'ostacolo: tale casella passa quindi allo stato “contenente un ostacolo”.

Abbiamo ideato e sviluppato da zero l’algoritmo di movimento, che permette al robot di creare una mappa con grande efficienza, e senza scontrarsi con alcun ostacolo:

quando il robot si trova in una casella vuota, farà 72 rilevazioni attorno a sé per determinare la posizione di eventuali ostacoli, e assocerà alle caselle in cui si trova un terzo stato grazie al quale eviterà di ripetere rilevazioni dalla stessa posizione. Dopodiché, se le caselle di fronte sono libere (né contengono ostacoli, né alcuna misurazione è stata fatta quando il robot si trovava su di esse), il robot procederà in avanti di X\* m; altrimenti, le altre caselle circostanti saranno analizzate, e in seguito il robot ruoterà, muovendosi in una direzione ortogonale all’orientamento attuale, o tornerà indietro. Le direzioni con caselle libere hanno la precedenza, avanzare frontalmente è preferito a ruotare di 90° o 180°, le direzioni contenenti ostacoli sono sempre evitate.

In sintesi, il robot si muove lungo due direzioni ortogonali, una volta giunto in posizioni nuove farà una rilevazioni per cercare ostacoli, eviterà le collisioni e tenderà a visitare sempre zone nuove, fino al completamento della mappatura, senza ripetere rilevazioni già fatte.

Alcune grandezze in gioco possono essere modificate (grandezza delle caselle della griglia, angolo di movimento del sensore, raggio massimo del sensore, distanza percorsa dal robot tra una misurazione e la successiva). Modificandole, si può sacrificare del tempo (rendendo il processo di mappatura più lento) in favore di una maggiore precisione, e viceversa.

In questo progetto sono stati alcuni framework OpenSurce come Django per gestione del sito, db e server, Chart.js per la creazione del grafico.

L’ Arduino si occupa della gestione dei sensori mediante codice c++

Il Raspberry si occupa di lettura e elaborazione dati mediate codice python3, e si occupa della gestione del server mediante un servizio in bach che si avvia automaticamente all’accensione

Tutto il codice è disponibile su questa repositori GitHub: https://github.com/Ro0t-set/Roobot-2d

Qui in seguito illustrata una minima parte del codice impiegata per il progetto.

1)Inizia mappatura: un’Arduino ordina di fare fare una rotazione di 5° ad un servo sino ad arrivare a 180° prendendo 72 rilevamenti inviati a una porta seriale del Raspberry.

|  |
| --- |
| Avvio e rotazione servo radar + lettura dati  rotazione%20servo.png |

|  |
| --- |
| Avvio lettura seriale  lettura%20del%20serial.png |

|  |
| --- |
| Elaborazione dati  elaborazione%20dati.png  il codice viene messo in loop e le variabili rinnovate  dopo ogni movimento |

|  |
| --- |
| Reset & Reload dati seriale  rinnovo%20dati%20seriali.png |

|  |
| --- |
| viene creato il grafico dei dati elaborati  geneirazione%20grafico.png  Uso un codice py per generare codice js compatibile  con il Framework Charh.js |