Rilevamento di ostacoli in movimento utilizzando un telemetro laser

Andrea TRESOLDI Emanuele RIZZI Vehat SAKTI

Università di Bergamo, Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione

May 24, 2022

Abstract

Il robot è in grado di seguire un percorso generato a partire da una mappa di occupazione e lungo la traiettoria potrebbe incontrare degli ostacoli in movimento non rappresentati, per esempio delle persone che si muovono lungo un corridoio in senso opposto. L'obiettivo del progetto consiste nello sviluppare una libreria software C++ che implementa la funzionalità di rilevamento degli ostacoli in movimento. A questo proposito viene utilizzato un telemetro laser che permette al robot di acquisire informazioni sull'ambiente in esso cui si muove.

1 Rilevamento di ostacoli in movimento

Durante i propri spostamenti il robot esegue continuamente delle scansioni utilizzando un telemetro laser per acquisire informazioni sull'ambiente. Queste misure di distanza fornite dal laser vengono utilizzate dal robot per costruire una mappa di occupazione in modo tale da avere una rappresentazione interna dell'ambiente circostante. Dal confronto di due mappe di occupazione successive il robot è così in grado di rilevare gli ostacoli in movimento.

1.1 Costruzione e confronto delle mappe di occupazione

Ad ogni scansione effettuata, il robot costruisce una nuova mappa di occupazione (mappa corrente) centrata nella posizione corrente del laser in cui è stata effettuata la rilevazione. All'interno di questa mappa vengono etichettate come occupate tutte le celle in cui è presente un'ostacolo. Confrontando due mappe di occupazione successive (corrente e precedente), cella per cella, il robot è in grado di individuare tutte le celle relative agli ostacoli in movimento. In dettaglio, la logica che guida il confronto tra due mappe successive è la seguente:

- se nella mappa di occupazione corrente sono presenti delle celle etichettate come occupate che nella mappa precedente risultano invece essere libere, allora queste celle sono riferite ad uno o più ostacoli in movimento nell'ambiente.

Tuttavia, affinchè il robot possa eseguire un confronto tra due mappe di occupazioni successive in modo corretto ed efficace per raggiungere il suo obiettivo, è fondamentale che le misure del laser prodotte dalla scansione precedente, siano prima attualizzate rispetto alla posizione corrente del laser, ovvero la posizione in cui è stata effettuata la nuova scansione. Dopo aver adattato la mappa precedente a quella corrente, centrandola nella posizione corrente del laser e

utilizzando le misure attualizzate, è così possibile eseguire un confronto esatto, cella per cella, tra le due mappe per rilevare gli ostacoli in movimento, essendo queste centrate nello stesso punto (nella posizione corrente del laser). Per poter attualizzare le misure precedenti del laser alla sua posizione corrente, il robot utilizza la stima odometrica dei suoi spostamenti che viene fornita dagli encoder montati sulle ruote.

1.2 Clustering e filtraggio del rumore

Una volta identificate tutte le celle in movimento, il robot utilizza un algoritmo di clustering per poter associare ognuna di queste ad un ostacolo. L'algoritmo [1] che è stato integrato all'interno della libreria prende il nome di DBSCAN (Density-Based Spatial Clustering of Applications with Noise [2]): questo algoritmo basa il raggruppamento sulla densità dei punti all'interno di uno spazio N-dimensionale connetendo regioni di punti con densità sufficientemente alta. Il clustering permette al robot non sono di costruire cluster di punti associati agli ostacoli in movimento, ma permette anche di rimuovere i punti associati al rumore presenti a causa dalla discretizzazione che viene effettuata per poter costruire le mappe di occupazione a partire dalle misure sensoriali fornite dal laser.

1.3 Calcolo della circonferenza minima

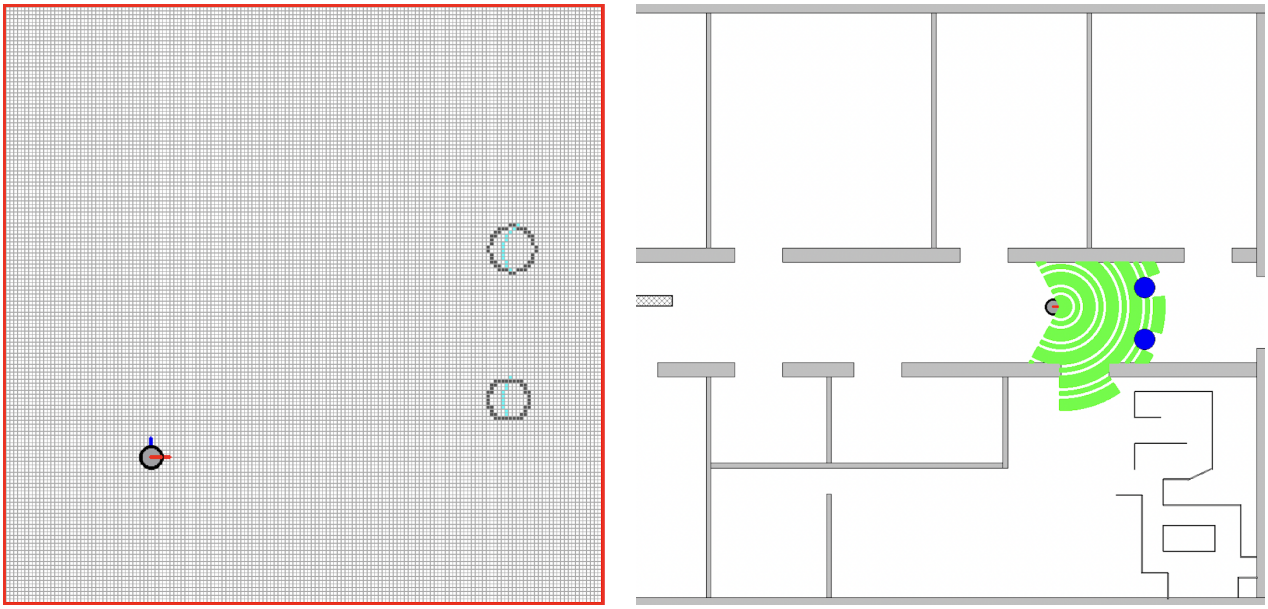
Terminata la fase di clustering, il robot calcola infine raggio e centro della circonferenza minima associata ad ogni cluster che contenga tutti i suoi punti. Questo problema, noto come problema del cerchio minimo [3], consiste nel trovare la circonferenza più piccola che contenga un insieme di punti nel piano. L'algoritmo [4] viene applicato a tutti i cluster rilevati dal robot in modo tale da avere una stima sia della posizione del centro (il centro della circonferenza) che delle dimensioni (il raggio) di ogni ostacolo in movimento all'interno dell'ambiente. In figura 1 e in figura 2 sono mostrate due differenti simulazioni che mostrato il funzionamento della libreria implementata.

[1] il codice sorgente e alcune informazioni sull'algoritmo sono presenti al seguente link: `https://github.com/james-yoo/DBSCAN`.

[2] `https://it.wikipedia.org/wiki/Dbscan`.

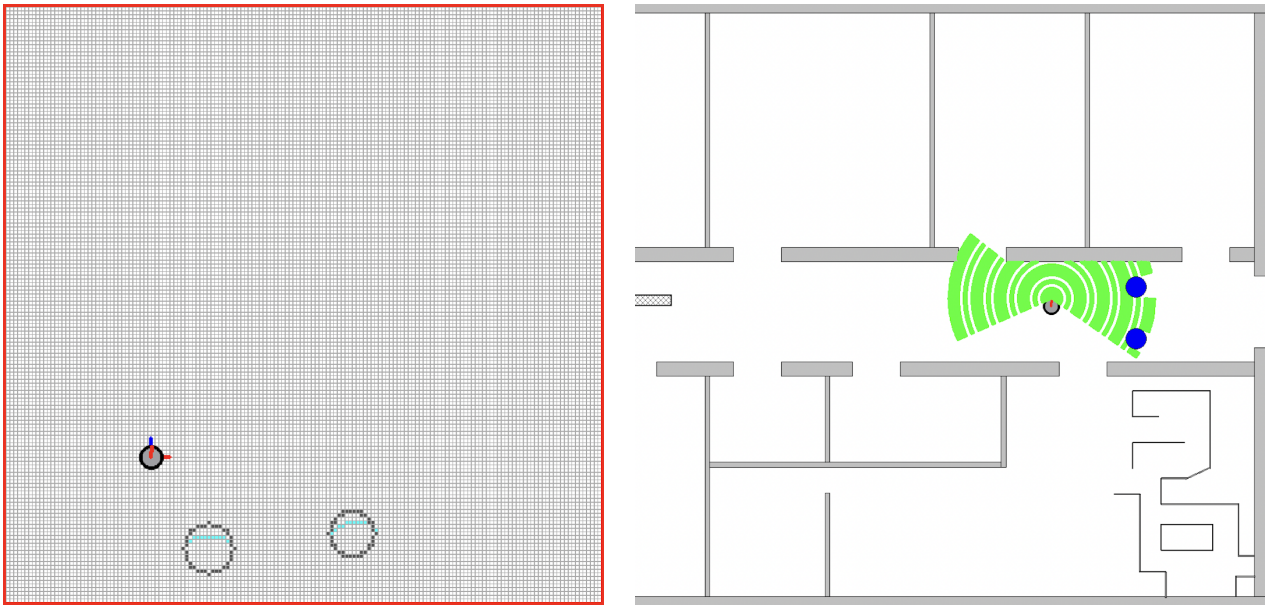
[3] `https://it.frwiki.wiki/wiki/Problème_du_cercle_minimum`.

[4] il codice sorgente e alcune informazioni sull'algoritmo sono presenti al seguente link: `https://www.nayuki.io/page/smallest-enclosing-circle`.



(a) (b)

Figure 1: (a) in azzurro sono rappresentati i punti delle frontiere degli ostacoli in movimento mentre in nero le circonferenze minima associate ad ogni cluster. (b) il robot si muove all'interno dell'ambiente e due ostacoli si muovono verso di lui.



(a) (b)

Figure 2: (a) in azzurro sono rappresentati i punti delle frontiere degli ostacoli in movimento mentre in nero le circonferenze minima associate ad ogni cluster. (b) il robot si muove all'interno dell'ambiente e due ostacoli si muovono verso di lui.