

Università degli studi di Modena e Reggio Emilia
Dipartimento di ingegneria "Enzo Ferrari"

2024

Implementazione di un sistema di
controllo remoto per veicoli
semi-autonomi connessi

Relatore: Paolo Burgio
Candidato: Alessandro Appio

Anno accademico: 2023/2024

Contents

1	Introduzione	3
1.1	Guida remota e guida autonoma	3
1.2	Scopo della tesi	3
2	Piattaforma di sviluppo	3
2.1	Rover AgileX	3
2.2	GPGPU e sensore lidar	3
3	Funzionamento	4
3.1	Informazioni scambiate tra veicolo e server	4
3.2	Funzionamento lato veicolo	4
3.3	Funzionamento lato server	4
4	ROS	5
4.1	Nodi	5
4.2	Comunicazione tra nodi	5
5	MQTT	5
5.1	Formattazione messaggi	5
5.2	Topic	5
6	Sviluppi futuri e conclusione	5
6.1	Problemi	5
6.2	Soluzioni	5
6.3	Applicazioni pratiche	5

1 Introduzione

1.1 Guida remota e guida autonoma

Con guida autonoma si intende un insieme di tecnologie, metodi e tecniche che permettono ad un veicolo di muoversi senza l'intervento umano. Un veicolo autonomo è infatti capace di esaminare l'ambiente, calcolare un percorso da seguire in base a quanto esaminato e seguire tale percorso. Generalmente questi 3 step vengono chiamati: perception, planning e control. Quando parliamo di guida remota invece intendiamo un tipo di guida in cui, chi effettivamente prende decisioni su come il veicolo deve muoversi e su che percorso esso debba seguire, è un'essere umano, ma esso fa utilizzo di tecnologie come sensori, attuatori e la rete in generale per la guida.

1.2 Scopo della tesi

Lo scopo di questa tesi è di sviluppare un veicolo ibrido che sia capace di avvalersi della guida autonoma in condizioni normali, ma che sia anche capace di essere guidato da un essere umano a distanza al momento del bisogno.

2 Piattaforma di sviluppo

In questa sezione si descrive come è stata assemblata la piattaforma per lo sviluppo e il testing.

2.1 Rover AgileX

Il veicolo scelto per lo svolgimento della tesi è un rover di marca AgileX e modello Hunter, è un veicolo terrestre comandabile tramite interfaccia CAN. Il veicolo è capace di fornire, oltre ad un'interfaccia per il controllo del suddetto, una serie di dati sullo stato del veicolo, come l'odometria (ovvero lo spostamento del veicolo calcolato a partire dal movimento delle ruote).

2.2 GPGPU e sensore lidar

Altra parte importante per lo svolgimento della tesi è stato trovare un calcolatore embedded per eseguire tutta la logica di controllo ed il planning ed un sensore che dia informazioni utili per la perception. Per quanto riguarda il calcolatore è stato scelto di utilizzare una GPGPU (General Purpose Graphic Processing Unit) con il sistema operativo Ubuntu 20.04, mentre per il sensore è stato scelto di utilizzare un sensore Lidar, ovvero un sensore laser capace di calcolare la distanza di vari punti nell'ambiente in base al tempo che un raggio laser impiega per tornare alla sorgente.

3 Funzionamento

Nella seguente sezione si descrive il funzionamento generale dello stack di guida remota e come un server si può interfacciare con il veicolo per svolgere tali operazioni

3.1 Informazioni scambiate tra veicolo e server

Per adempiere allo scopo di guida remota sono necessarie due figure distinte: Il veicolo ed un server. Il veicolo è stato descritto nella sezione precedente, il server invece è quella figura del sistema con cui l'umano si interfaccia e che fornisce sia i dati prelevati dal veicolo sia un metodo per il controllo del veicolo. Per garantire il funzionamento della struttura è quindi necessario lo scambio di informazioni tra le due parti. Queste informazioni sono: - I messaggi di controllo per i motori - La pointcloud fornita dal lidar - l'odometria calcolata dal veicolo Le informazioni tra il server ed il veicolo sono scambiate tramite protocollo di rete MQTT, un protocollo studiato per il mondo dell'IOT e per tali applicazioni. Sia a bordo del veicolo che sul server è istanziata un versione di ROS (Robotic Operative System) ovvero un software utilizzato per creare diversi nodi (o processi paralleli) e per permettere lo scambio di informazioni tra essi (Tramite topic e messaggi).

3.2 Funzionamento lato veicolo

A bordo del mezzo sono istanziati due nodi ROS: uno incaricato di gestire l'invio dei dati del sensore e dell'odometria, ed uno incaricato di ricevere i messaggi di controllo. Per comodità li chiameremo rispettivamente *telemetry node* e *control node*. Come descritto prima il *telemetry node* si incarica di ricevere messaggi da ROS contenenti la pointcloud del lidar e l'odometria del mezzo, per poi formattare tali messaggi come stringhe JSON ed inviare questi messaggi tramite protocollo MQTT al server in due topic dedicati. Per quanto riguarda il *control node* invece, questo si incarica di ricevere messaggi dal server che riguardano il controllo del mezzo, una volta ricevuti i messaggi come stringa JSON questi vengono convertiti in messaggi ROS e vengono inviati ai driver del rovere che permettono poi il controllo tramite CAN del veicolo.

3.3 Funzionamento lato server

Il server, a differenza del veicolo, esegue un solo nodo ros. Lo scopo di tale nodo non è tanto quello di prendere decisioni sul movimento del veicolo ma è quello di ricevere i dati dal veicolo, farli processare da un terzo ente, ricevere le decisioni di questo terzo ente espresse come azioni che il veicolo deve eseguire ed inviarle al veicolo. Nello specifico tale nodo riceve tramite protocollo MQTT i dati dal veicolo, per poi formattarli come messaggi ros e pubblicarli sui relativi topic ros. Una volta che questi dati vengono ripubblicati questi possono essere elaborati o da uno stack di guida autonoma o da un effettivo pilota che a distanza vedrà

i dati rilevati dal veicolo e prenderà decisioni sul controllo. Una volta prese queste decisioni dal terzo ente, queste verranno pubblicate sul topic ros addetto al trasporto di tali informazioni e a cui il nodo ros del server è iscritto. Una volta che il nodo avrà letto il dato, lo formatterà in una stringa JSON e la invierà tramite MQTT al veicolo che eseguirà tali comandi

4 ROS

4.1 Nodi

4.2 Comunicazione tra nodi

5 MQTT

5.1 Formattazione messaggi

5.2 Topic

6 Sviluppi futuri e conclusioni

6.1 Problemi

6.2 Soluzioni

6.3 Applicazioni pratiche