

UNIVERSITÀ DEGLI STUDI DELLA TUSCIA

Corso di dottorato
Engineering for Energy and Environment

Tesi di dottorato

**Progettazione e sperimentazione di un
sistema di acquisizione dati low-cost per
il monitoraggio delle prestazioni di una
semovente per la raccolta delle nocciole**

Candidato

Ing. Gianmarco Rigon

Relatore

Prof. Ing. Danilo Monarca

Ciclo di dottorato: XXXVII

A.A. 2023/2024

Indice

Sommario	3
Abstract	4
Introduzione	6
1 Introduzione	7
1.1 Le macchine per la raccolta delle nocciole	7
1.1.1 La corilicoltura	7
1.1.2 Tipologie di macchine per la raccolta	8
1.1.3 La semovente aspiratrice Cimina 300S	9
1.2 I sistemi di acquisizione dati	18
1.2.1 Stato dell'arte sull'Agricoltura 4.0	18
1.2.2 Struttura generale dei DAQ	19
1.2.3 Sensori	21
1.2.4 Circuiti di condizionamento dei segnali	22
1.2.5 ECU	24
1.2.6 BUS di sistema e CAN-Bus	26
1.2.7 Microcontrollori Arduino	27
1.3 Obiettivi	29
Materiali e metodi	31
2 Materiali e metodi	31
2.1 Hardware	31
2.1.1 Sensori	31
2.1.2 Moduli aggiuntivi	44
2.1.3 Microcontrollori	48
2.1.4 Assemblaggio dei componenti elettronici	52

2.2	Cablaggi	56
2.2.1	Hall	56
2.2.2	Pressione, temperatura e accelerazione	58
2.2.3	CAN-BUS	67
2.2.4	GPS	68
2.3	Programmazione	69
2.3.1	Hall	76
2.3.2	Pressione, temperatura e accelerazione	79
2.3.3	CAN-BUS	86
2.3.4	GPS	89
2.4	Elaborazione dei dati	91
Risultati		93
3	Risultati	93
3.1	Test dell'impianto di ventilazione	95
3.1.1	Condizioni ambientali	95
3.1.2	Parametri motore	95
3.1.3	Prestazioni dei ventilatori	96
3.2	Test su strada: massima velocità	101
3.2.1	Posizione	102
3.2.2	Velocità	102
3.2.3	Parametri motore	104
3.2.4	Pressioni nel circuito idraulico	105
3.3	Test su strada: tragitto d'esempio	108
3.3.1	Condizioni ambientali	108
3.3.2	Velocità	109
3.3.3	Pendenza	109
3.3.4	Parametri motore	111
3.3.5	Pressioni nel circuito idraulico	112
3.3.6	Temperature dell'olio idraulico	115
3.3.7	Parametri di lavoro tipici su strada	116
3.4	Test in campo: primo giorno di raccolta	117
3.4.1	Posizione	117
3.4.2	Velocità	120
3.4.3	Parametri motore	120
3.4.4	Pendenza	121
3.4.5	Pressioni nel circuito idraulico	123

3.4.6	Prestazioni dei ventilatori	124
3.5	Test in campo: secondo giorno di raccolta	126
3.5.1	Condizioni ambientali	126
3.5.2	Velocità	127
3.5.3	Parametri motore	127
3.5.4	Pendenza	130
3.5.5	Pressioni nel circuito idraulico	130
3.5.6	Prestazioni dei ventilatori	132
3.5.7	Temperature dell'olio idraulico	133
3.6	Parametri tipici durante la raccolta	135
Conclusioni		136
4 Conclusioni		137
Lista delle figure		139
Lista delle tabelle		143
Bibliografia		146
Ringraziamenti		149

Sommario

Lo sviluppo tecnologico delle semoventi per la raccolta della frutta a guscio garantisce al giorno d'oggi una riduzione della manodopera, distinte capacità di raccolta, un ottimo livello di pulizia del prodotto e migliori condizioni di sicurezza e comfort nella conduzione del mezzo. Nonostante ciò, l'adozione delle tecnologie digitali tipiche dell'Agricoltura 4.0 è ancora limitata in questo settore. L'applicazione dell'elettronica in agricoltura, al giorno d'oggi più accessibile economicamente e *open source*, permette di monitorare macchine e produzioni agricole, riducendo i costi e gli sprechi dei processi produttivi. Pertanto l'obiettivo di questo trattato è progettare e implementare un sistema di acquisizione dati *low cost* per monitorare le prestazioni di una semovente raccoglitrice di nocciole, la Cimina 300S, prodotta da FACMA. Nello specifico sono stati impiegati i microcontrollori Arduino accoppiati con sensori per misurare la velocità di avanzamento, le pressioni e le temperature nel circuito idraulico, le portate d'aria nel condotti di ventilazione, i parametri motore come potenza, coppia, regime di giri e consumi, la posizione satellitare e la pendenza affrontata. Il trattato si aprirà con un'introduzione sulle macchine per la raccolta delle nocciole, soffermandosi sulla struttura dettagliata della semovente oggetto di studio e sul funzionamento dei sistemi di acquisizione dati, installati nei veicoli agricoli. Successivamente si descriveranno i sensori e i microcontrollori impiegati nello studio, si illustreranno i cablaggi e il processo di assemblaggio dei componenti elettronici e si analizzerà la logica di programmazione delle schede digitali. Infine, si interpreteranno i risultati delle prove, studiate per testare il veicolo nelle tipiche condizioni d'impiego, che siano su strada o in campo.

Parole chiave: semovente raccoglitrice, nocciole, sistema di acquisizione dati, Arduino, Agricoltura 4.0.

Abstract

The technological advancement of self-propelled hazelnut harvesters nowadays guarantees a reduction in labour, distinctive harvesting capacities, an excellent level of product cleanliness and enhanced safety and comfort conditions for the driver. Nevertheless, the adoption of digital technologies typical of Agriculture 4.0 is still limited in this sector. The application of electronics in agriculture, which is nowadays more economically accessible and open-source, permits the monitoring of machines and agricultural production, and the reduction of costs and waste in production processes.

Therefore, the objective of this study is to engineer and implement a low cost data acquisition system to monitor the performances of a self-propelled hazelnut harvester, the Cimina 300S, manufactured by FACMA. Specifically, Arduino microcontrollers coupled with sensors were used to measure the forward speed, pressures and temperatures in the hydraulic system, air flow rates in the ventilation ducts, engine parameters such as power, torque, RPM and fuel consumption, position, and terrain slope.

This thesis will commence with an introduction to hazelnut harvesting machines, with a particular focus on the detailed structure of the self-propelled vehicle under study, and the functioning of the data acquisition systems installed in agricultural vehicles. Next, the sensors and microcontrollers used in the study will be described, the wiring and assembly process of the electronic components will be illustrated, and the programming flow chart of the digital boards will be analysed. Finally, the results of the tests, which were specifically intended to assess the vehicle's performance under typical operational conditions, whether on the road or in the field, will be analysed and interpreted.

Keywords: self-propelled harvester, hazelnut, data acquisition system, Arduino, Agriculture 4.0.

Capitolo 1

Introduzione

In questo capitolo introduttivo si esporranno i fondamenti di meccanica, ingegneria ed elettronica inerenti alle le macchine per la raccolta delle nocciole e i moderni sistemi di acquisizione dati. In primo luogo, dopo un accenno sullo sviluppo della corilicoltura mondiale e italiana, si analizzerà la macchina per la raccolta delle nocciole oggetto di studio, con un focus approfondito sulla struttura e sul funzionamento. Queste informazioni risultano necessarie per comprendere meglio i parametri operativi principali del veicolo, la scelta e il posizionamento dei sensori, e a giustificare i risultati dei campionamenti. In secondo luogo, dopo un breve excursus sullo stato dell'arte dell'Agricoltura 4.0, si approfondirà la struttura e il funzionamento dei sistemi di acquisizione dati, parte necessaria per la comprensione dei componenti elettronici impiegati e dei collegamenti elettrici.

1.1 Le macchine per la raccolta delle nocciole

1.1.1 La corilicoltura

Il settore agricolo riveste un ruolo cruciale nell'economia globale, fornendo risorse essenziali in numerose produzioni alimentari nonché di altro genere. Uno degli ambiti in maggior sviluppo dell'agricoltura riguarda la frutta a guscio, più in particolare le nocciole.

Il rapido sviluppo del settore corilicolo è legato alla crescente domanda di prodotto da parte dell'industria di trasformazione, in particolare quella dolciaria. Basti pensare che nell'ultimo decennio la produzione mondiale di nocciole è aumentata di circa il 25%, con l'Italia che figura al secondo

posto dopo la Turchia tra i principali produttori. Per rendere meglio l'idea, la superficie totale ricoperta da terreni dedicati alla raccolta di nocciole nel nostro paese ha subito un incremento del 22% nel decennio compreso tra il 2010 e il 2020, passando da una superficie coltivata di 70000 ettari a 90000 ettari. La produzione corilicola risulta inoltre essere molto concentrata, con l'80% della superficie dedicata localizzata principalmente in tre regioni, Piemonte (29%), Lazio (28%) e Campania (24%). La restante parte (19%) è suddivisa tra altre quattro regioni [9]. In Italia, sebbene largamente diffusa anche nelle altre regioni menzionate in precedenza, la coltivazione del nocciolo ha considerevole rilevanza nella provincia di Viterbo. La corilicoltura viterbese presenta, infatti, caratteristiche di notevole specializzazione e si estende su circa 20000 ettari del territorio dei Monti Cimini, tanto da costituirne uno degli elementi caratteristici del paesaggio agricolo e forestale della zona stessa, e occupa gran parte delle aree rurali intorno al Lago di Vico (Ronciglione).

In queste zone, pertanto, necessitando la coltura di elevati livelli di meccanizzazione, si è assistito a un grande sviluppo di officine meccaniche che hanno concentrato gran parte delle attività nella ricerca di soluzioni tecniche in grado di permettere ai coltivatori locali di operare con mezzi e strumenti sempre più moderni e tecnologici.

1.1.2 Tipologie di macchine per la raccolta

L'Italia è uno dei principali produttori di macchine raccoglitrici. La necessità di adattarsi ad areali di produzione diversi ha dato vita a un ampio mercato che offre soluzioni diversificate per qualsiasi azienda, sia all'interno che all'esterno del Paese.

Attualmente le macchine raccoglitrici possono essere dotate di sistema autonomo di propulsione (semoventi) o dipendere il loro funzionamento da una macchina trattrice (trainate, semiportate). Un'ulteriore distinzione può essere effettuata considerando il principio di raccolta del prodotto da terra: alcune effettuano l'aspirazione del prodotto dal suolo tramite ventilatori (aspiratrici), mentre altre sollevano meccanicamente il prodotto da terra attraverso organi rotanti (meccaniche o raccattatrici). Combinando le varie categorie si ottengono le soluzioni commerciali oggi più diffuse: aspiratrici trainate, aspiratrici semoventi, raccattatrici semiportate e raccattatrici semoventi.

1.1.3 La semovente aspiratrice Cimina 300S

Tra le aziende più importanti nel Viterbese, nonché su tutto il territorio nazionale e in larga scala anche mondiale attraverso una rete di rivenditori in differenti nazioni, figura la FACMA S.r.l.. La società, con sede presso Vitorchiano, si occupa di progettazione e costruzione di macchine agricole per la lavorazione del terreno e per la raccolta meccanizzata della frutta a guscio, ma anche di impianti di essiccazione, pulizia e stoccaggio del raccolto [2].

In questo trattato ci limiteremo a descrivere nel dettaglio il caso studio di monitoraggio: la semovente aspiratrice FACMA Cimina 300S (Fig. 1.1). Il principio dell'aspirazione del prodotto da terra permette alle macchine aspiratrici di lavorare anche su terreni disomogenei e/o caratterizzati da pendenze considerevoli [7]. La capacità di raccolta in condizioni lavorative ottimali si attesta sulle 2 h/ha [4] e la macchina contiene al suo interno dispositivi per la cernita che effettuano una pulizia accurata a più stadi del prodotto raccolto. Le potenze erogate dai motori installati sulle macchine aspiratrici arrivano fino a 55 kW e la propulsione è di tipo completamente idraulico. La macchina possiede 3 ruote motrici e 2 gamme di velocità, una adatta alla raccolta e una alla marcia su strada. I costi di esercizio e le prestazioni di raccolta rendono conveniente l'impiego di tali macchine per aziende con superfici coltivate superiori ai 15 ha [16].

Recenti miglioramenti tecnologici

Da alcuni anni è possibile installare anteriormente sulla macchina il raccogliatore meccanico RM170, che comporta, a scapito di un maggior ingombro complessivo del mezzo e, di conseguenza, di una peggior manovrabilità, diversi vantaggi, tra cui:

- un incremento della capacità oraria di raccolta, infondendo alla macchina la natura di una raccattatrice e mantenendo al tempo stesso i vantaggi di una aspiratrice;
- una riduzione delle polveri aerodisperse [19];
- una maggiore pulizia del prodotto, grazie all'aggiunta di ulteriori stadi di cernita.

Di recente la macchina è stata munita di cabina di guida, un *optional* che:



Figura 1.1: *Raccoglitrice semovente FACMA Cimina 300S, munita di raccoglitore meccanico RM170 e cabina di guida.*

- possiede la primaria funzione del ROPS di tutela e salvaguardia del conducente nel caso di ribaltamento del mezzo [12] [21];
- assicura l'isolamento dagli agenti esterni e dalle polveri, la cui concentrazione può giungere a valori tra gli 8 e $21\text{mg}/\text{m}^3$, in funzione delle tipologie di macchine impiegate e dello stato vegetativo del terreno [17];
- permette un'attenuazione sostanziale del rumore percepito dall'operatore; infatti i livelli di pressione sonora rilevati in prossimità dell'orecchio degli addetti alla raccolta possono superare

i $90\text{ dB}(A)$ [17];

- assicura un ambiente chiuso idoneo per l'installazione degli apparati di climatizzazione, fondamentali per migliorare le condizioni di comfort termico dell'operatore [7];
- fornisce un ambiente isolato dagli agenti esterni come polvere e pioggia, idoneo all'installazione di dispositivi elettrici ausiliari come visualizzatori, comandi elettrici.

Modello 3D del veicolo

La prototipazione virtuale rappresenta ormai uno strumento fondamentale per la progettazione di macchine.

Il CAD (*Computer Aided Design*) indica il settore dell'informatica volto all'utilizzo di tecnologie software e specificamente della computer grafica, per supportare l'attività di disegno tecnico [23]; i vantaggi associati all'utilizzo del CAD sono [21]:

- visualizzazione accurata del prodotto finale;
- riduzione degli errori di progettazione;
- documentazione completa dei progetti a livello di geometrie quote, distinte base e altro;
- agevole riutilizzo dei dati di progettazione;
- riduzione dei tempi di progettazione;
- riduzione dei costi di realizzazione dei prototipi fisici;
- possibilità di eseguire, già in fase di studio o di progetto, le ottimizzazioni che altrimenti dovrebbero essere eseguite sul prototipo fisico.

Avvalendosi di SOLIDWORKS, software di modellazione CAD, è stato realizzato il modello 3D del veicolo il cui assieme comprende più di 6000 parti. Ai fini di una più semplice comprensione della struttura e del funzionamento della macchina si riportano, in Fig. 1.2, le viste isometriche del modello 3D, e, in Fig. 1.3, la vista in sezione parziale con illustrazioni sui principali componenti.

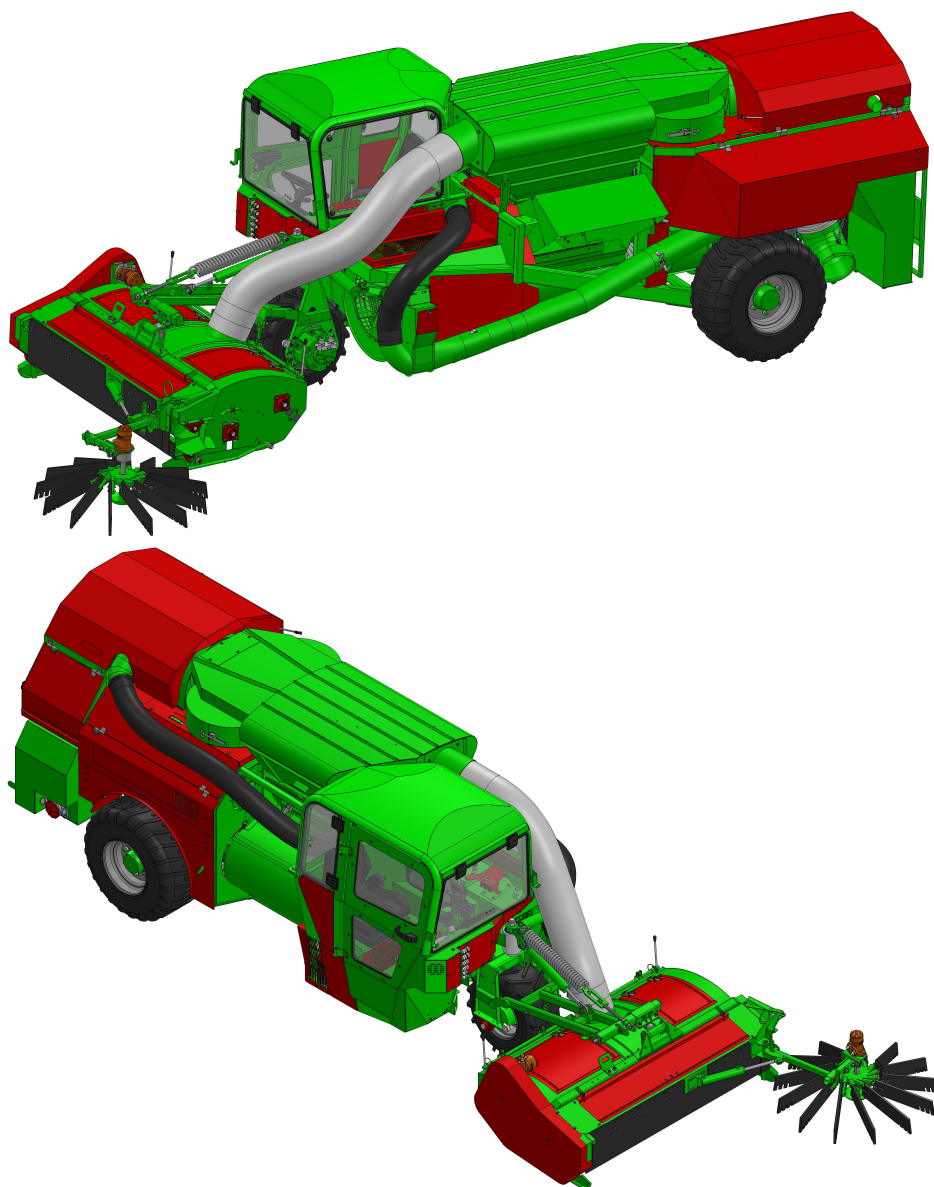


Figura 1.2: Modello 3D della raccoglitrice.

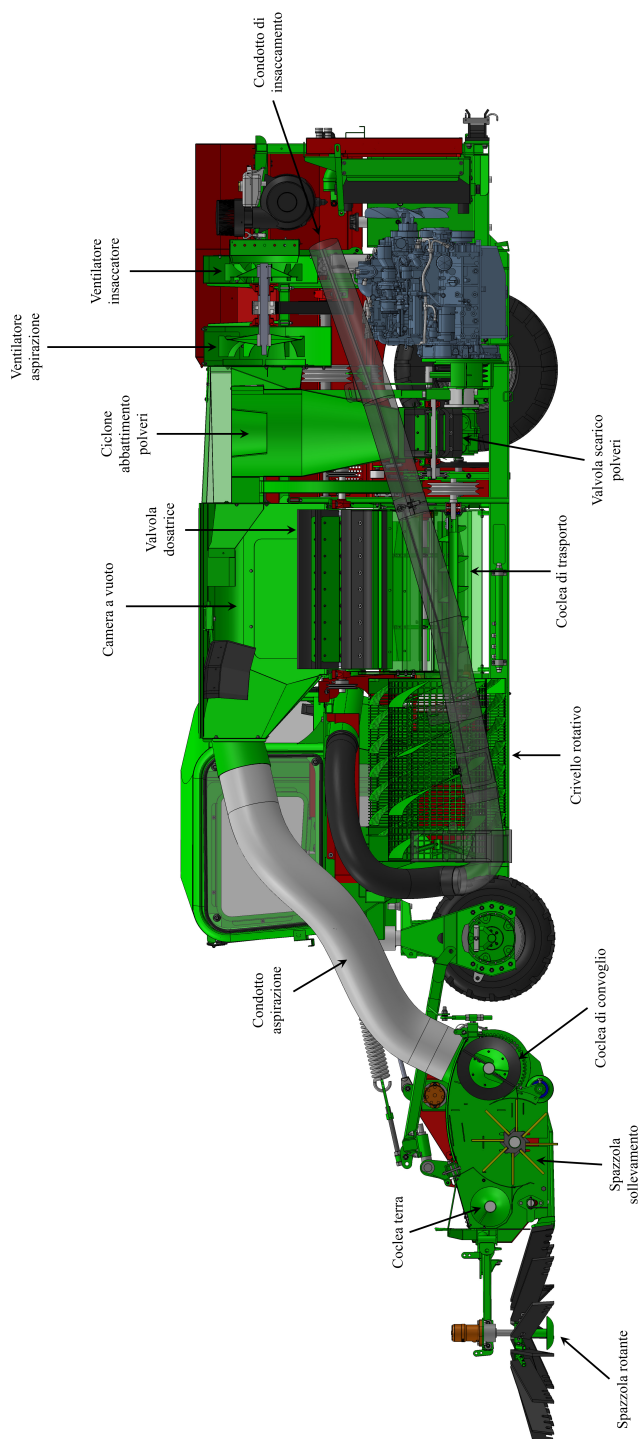


Figura 1.3: Sezione parziale del modello 3D della raccogliitrice.

Il percorso del prodotto

Di seguito, per comprendere il principio di funzionamento della macchina, si enuncia il percorso che il prodotto, gli elementi di scarto e l'aria intraprendono all'interno della macchina [16].

- Inizialmente il frutto, che giace a terra, viene convogliato verso il sistema di aspirazione, attraverso spazzole controrotanti, costituite da palette radiali in gomma.
- Una spazzola cilindrica, dotata di pale di gomma e inserita in un involucro in lamiera forata, ruotando in una direzione opposta alle ruote, solleva i frutti da terra.
- Gli scarti fini, come la terra, passano attraverso la griglia e vengono poi scaricati lateralmente al raccoglitore tramite una coclea.
- Il materiale sollevato dalle spazzole viene lanciato verso una camera cilindrica contenente due coclee, che possiedono un verso d'elica opposto, le quali hanno il compito di condurre la frutta verso il centro della testata.
- Le nocciole vengono aspirate, grazie a un ventilatore centrifugo, e inviate, mediante una condotta flessibile, verso la successiva camera di raccolta.
- Una valvola dosatrice, dotata di pale radiali in gomma, riveste il ruolo di tenuta e di scarico graduale del materiale raccolto (nocciole, ma anche sassi e rametti).
- All'uscita della valvola, una portata d'aria, generata da un altro ventilatore, permette l'espulsione del materiale di scarto più leggero (foglie e gusci vuoti).
- Una coclea trasporta le nocciole e i materiali pesanti verso il crivello. L'ultimo è un vaglio rotativo, che effettua la cernita del raccolto in base alle dimensioni, ed è costituito da uno o più cilindri coassiali in lamiera forata, al cui interno un'elicoide in lamiera trasporta i frutti in direzione assiale, separandoli dagli elementi che hanno dimensione differente (sassi, rametti, terra).

- Infine, il prodotto viene inviato pneumaticamente, mediante un secondo ventilatore centrifugo, in sacchi o più frequentemente in carrelli trainati.
- L'aria in uscita dalla camera di raccolta, aspirata dal primo ventilatore, passa attraverso il dispositivo per l'abbattimento delle polveri, costituito da due cicloni depolveratori.

Il sistema di trasmissione

Per meglio comprendere il funzionamento della macchina si riporta, in Fig. 1.4, lo schema di trasmissione che include tutti gli elementi propedeutici all'avanzamento e gli organi ausiliari per la raccolta. Il motore termico, unica sorgente di potenza, è collegato meccanicamente, attraverso un albero, a una pompa a pistoncini assiali a cilindrata variabile. Questa macchina oleodinamica permette di adeguare la cilindrata alle esigenze per regolare la portata di mandata dell'olio. L'uscita della pompa è collegata ai motori idraulici posteriori del veicolo, disposti in parallelo. All'uscita dei motori idraulici posteriori il flusso risultante viene convogliato all'aspirazione del motore idraulico anteriore. La cilindrata del motore anteriore, è stata scelta opportunamente, in funzione della portata d'olio elaborata e del raggio di lavoro delle ruote per allineare tutte le velocità periferiche evitando lo slittamento. All'uscita del motore idraulico anteriore il flusso viene indirizzato verso il serbatoio. Fra l'albero dei motori idraulici e le ruote si frappone un riduttore di velocità epicicloidale a due rapporti che dà la possibilità di utilizzare due gamme di velocità. L'albero motore, attraverso una puleggia e dei sistemi di trasmissione a cinghie trapezoidali, trasmette il moto di rotazione a diversi elementi:

- un ventilatore centrifugo adibito all'aspirazione del prodotto;
- un ventilatore centrifugo adibito all'insaccamento del prodotto;
- un ventilatore adibito alla rimozione degli scarti leggeri dal raccolto e alla defogliazione;
- la valvola dosatrice posta centralmente al veicolo sotto la camera a depressione;
- le valvole rotative di scarico delle polveri, separate dall'aria attraverso il ciclone depolveratore;

Il motore termico muove anche una pompa ad ingranaggi a cilindrata fissa che viene utilizzata per mandare in pressione l'olio nell'impianto frenante. Un'altra pompa dello stesso tipo viene utilizzata anche per inviare olio al motore eventualmente presente sul raccoglitore meccanico frontale RM170. Sull'albero del motore posto sul raccoglitore è calettato un ingranaggio dentato che, attraverso un sistema di catene, trasferisce il moto alle cocclee e alle spazzole installate nel raccoglitore.

1.2 I sistemi di acquisizione dati

1.2.1 Stato dell'arte sull'Agricoltura 4.0

L'elettronica ha caratterizzato lo sviluppo di ogni settore tecnologico negli ultimi decenni, facendo la sua comparsa a bordo delle macchine agricole nei primi anni '80. Nello stesso periodo storico vengono resi disponibili i *GPS* per uso pubblico. Si evolvono i sensori e i microprocessori, che diventano economicamente più accessibili. L'applicazione dell'elettronica si è gradualmente estesa a tutte le macchine agricole operatrici, assumendo il compito di regolazione automatica di numerose funzioni.

L'ottimizzazione funzionale, che deriva dall'adozione di sistemi automatici di controllo nelle macchine agricole moderne, permette di ottenere sia migliori capacità operative, sia lo svolgimento di un più elevato numero di operazioni, come la salvaguardia dell'integrità e della qualità del prodotto raccolto e lavorato. Inoltre, permette di ridurre l'impatto delle operazioni colturali sull'ambiente, mentre l'operatore addetto può lavorare in condizioni di minore affaticamento e maggiore sicurezza, delegando ai sistemi di controllo la regolazione continua dei singoli parametri di funzionamento della macchina.

A partire dagli anni 2000 vengono introdotte nuove tecnologie di comunicazione ed elaborazione dati che danno luogo alla rivoluzione *Industria 4.0*: emergono nuove tecnologie basate sui *big data*, *IoT*, si sviluppa il *cloud computing* e si intravedono le prime applicazioni di *IA* basate sul *machine learning* [6].

L'*Agricoltura 4.0*, si pone l'obiettivo di sfruttare le nuove tecnologie per aumentare la produttività dell'agricoltura e ridurre l'impatto ambientale [14]. Ai fini di integrare i concetti di *management* e automazione dei processi, occorre intervenire con tecnologie informatiche per raccogliere dati da fonti multiple in vista di un loro successivo utilizzo nell'ambito di decisioni riguardanti le attività produttive di campo. L'*Agricoltura 4.0* include il concetto di *Agricoltura di precisione*, che consiste nell'applicazione di tecnologie elettroniche, informatiche e pratiche agronomiche per gestire in modo automatico porzioni di terreno, secondo logiche sito-specifiche, allo scopo di migliorare l'efficienza e ridurre l'impatto ambientale della produzione agraria [8].

I *big data* prodotti nell'agricoltura di precisione possono essere sfruttati anche dai costruttori delle macchine, per capire come queste vengono

utilizzate, allo scopo di apportare migliorie al prodotto e per offrire servizi come la manutenzione predittiva. Il monitoraggio da remoto delle macchine agricole prende il nome di *telemetria* e, ormai, è una pratica diffusa su tutti i trattori e la maggior parte delle macchine agricole semoventi.

1.2.2 Struttura generale dei DAQ

I sistemi di acquisizione dati (DAQ) costituiscono le fondamenta della telemetria per i mezzi agricoli e permettono di monitorare i principali parametri di funzionamento della macchine.

La struttura generale di un DAQ adibito alle trattatrici è riportata in Fig. 1.5. Una conformazione più dettagliata di questi sistemi è rappresentata invece in Fig. 1.6, la cui consultazione risulterà utile per la comprensione dei capitoli successivi. I sensori hanno il compito di misurare una

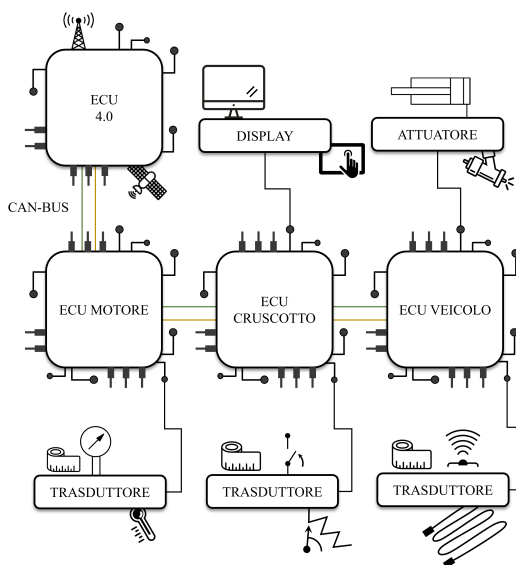


Figura 1.5: Componenti fondamentali di un sistema d'acquisizione dati per le macchine agricole.

variabile fisica, trasformandola in un segnale elettrico. Questo viene letto ed elaborato da una centralina elettronica dedicata (ECU). In genere nelle trattatrici sono presenti almeno 3 ECU (cruscotto, motore, veicolo). Ciascuna elabora il segnale dei sensori che vengono installati nella zona di competenza. Sui trattori moderni è presente una ECU che può

integrare al suo interno altri sensori, come il ricevitore GNSS, e una scheda di rete che è in grado di trasmettere i dati sul cloud. All'uscita delle centraline elettroniche troviamo i dispositivi di interfaccia utente come il display e gli attuatori che vengono comandati elettronicamente. Le centraline comunicano tra loro attraverso un bus di sistema che, con ogni probabilità, è di tipo CAN.

1.2.3 Sensori

I sensori sono dispositivi di misura che rilevano grandezze fisiche, convertendole in un segnale elettrico. In altre parole trasformano le manifestazioni energetiche connesse alla grandezza da misurare in un segnale analogico di tensione o corrente [10]. Un trasduttore si compone di tre elementi: un'interfaccia di ingresso, un sensore e un'interfaccia d'uscita. I trasduttori possono essere, a seconda del segnale emesso all'uscita, di due tipologie:

- analogici, se forniscono un segnale di tensione o corrente che varia in proporzione o in modo continuo al valore della grandezza fisica che rilevano;
- digitali, se forniscono una sequenza di tensioni, che possono assumere solo due valori predefiniti, di solito rappresentati attraverso i numeri 0 e 1 del sistema binario.

Si definisce caratteristica statica di un trasduttore, o funzione di trasferimento, la funzione che lega l'uscita dello stesso alla variabile misurata. Questa è una funzione del tipo $y = f(x)$, dove y è il segnale all'uscita del trasduttore e x è la variabile misurata.

I principali indici di qualità del sensore, che meritano approfondimento in questo trattato, sono [18]:

- la linearità è la misura di quanto l'uscita sia proporzionale linearmente con la variabile misurata. La non-linearità indica quanto la funzione di trasferimento si discosta da una legge lineare del tipo $y = ax + b$, con a e b opportuni coefficienti;
- la risoluzione, ovvero la minima variazione della grandezza da misurare che il trasduttore può percepire;

- l'errore di fuori zero o offset, cioè l'uscita del sensore quando la grandezza da misurare è nulla;
- la sensibilità, definita come la massima variazione dell'uscita rapportata alla massima variazione dell'ingresso;
- la precisione, che è la misura di quanto l'uscita è ripetibile su diversi campionamenti;
- l'accuratezza, ovvero la misura di quanto l'uscita si discosta da quella che dovrebbe essere l'uscita ideale.

1.2.4 Circuiti di condizionamento dei segnali

Il segnale emesso dai sensori talvolta necessita di essere trattato attraverso appositi circuiti di condizionamento. Il compito di questi è quello di adeguare e modificare il segnale per renderlo leggibile alle ECU. Osservando la figura 1.6, si può vedere che gli elementi di un circuito di trattamento possono essere [18] [11]:

- amplificatori operazionali;
- filtri;
- *multiplexer*;
- *sample and hold*;
- ADC e/o DAC.

Amplificatori. Inizialmente utilizzati nei calcolatori analogici, oggi gli amplificatori operazionali, indicati anche con la sigla Op-Amp (*Operational Amplifiers*), sono ampiamente applicati nelle tecniche di generazione, elaborazione e condizionamento dei segnali elettronici. Il nome deriva dal fatto che permettono sia di amplificare un segnale analogico di tensione, sia di svolgere operazioni di calcolo come la somma, la sottrazione, l'integrazione e la derivazione.

Filtri. Un filtro è caratterizzato da una funzione di trasferimento per la quale sono discriminati (filtrati) i segnali d'ingresso con frequenza non appartenente a un insieme prestabilito. Esistono filtri del tipo:

- passa basso, se attenuano i segnali con frequenza inferiore a una soglia specifica;
- passa alto, se attenuano i segnali con frequenza superiore ad una soglia specifica;
- passa banda, se attenuano i segnali al di fuori di un certo *range* di frequenze specifico;
- elimina banda, se attenuano i segnali entro un certo *range* di frequenze specifiche.

Nella classificazione, secondo il tipo di circuito, si distinguono:

- I filtri passivi, che sono realizzati con resistori, condensatori o induttori e sono usati alle alte frequenze.
- I filtri attivi, che sono realizzati con resistenze, condensatori e un dispositivo attivo, che generalmente è un amplificatore operazionale, usati fino a frequenze di alcune centinaia di kHz .

Analog multiplexer. Nel caso in cui vi siano più sensori o segnali analogici da trattare attraverso un circuito di condizionamento potrebbe essere necessario un selettore analogico o *analog multiplexer*. Questo viene utilizzato per consentire la selezione di una tra le sorgenti multiple dirette al convertitore analogico digitale. Il selettore agisce come un commutatore elettronico, aprendo la via al corrispondente ingresso analogico.

Sample and Hold. Il dispositivo *Sample and Hold* mantiene costante il segnale analogico diretto all'ADC nel corso del processo di conversione. Senza questa condizione, nei caso di segnali multipli e velocemente variabili, il processo di conversione risulterebbe falsato, dando luogo a risultati inaccurati o errati. Il processo si sviluppa in due fasi:

1. nella fase di campionamento *Sample* viene letto un valore del segnale da convertire;
2. nella fase di mantenimento *Hold* il valore letto viene mantenuto all'uscita del dispositivo fino a completa conversione.

ADC e DAC. Il Convertitore Analogico Digitale ADC (*Analog to Digital Converter*) converte i campioni della tensione analogica in numeri in formato binario; svolge la funzione opposta del DAC, ovvero converte un segnale di tensione analogica in una serie di valori discreti in formato numerico binario.

Il Convertitore Digitale Analogico DAC (*Digital to Analog Converter*) è un dispositivo di interfaccia tra mondo virtuale numerico e mondo fisico dal lato della catena di distribuzione dei dati. Il DAC accetta in ingresso un numero e fornisce all'uscita una tensione ad esso proporzionale.

1.2.5 ECU

La sezione di elaborazione dati è affidata alle centraline elettroniche, chiamate ECU ovvero *electronic control unit*. Dal punto di vista hardware la ECU si interfaccia tramite le proprie porte di ingresso con i segnali provenienti dalla catena di acquisizione e tramite le porte di uscita con i dispositivi della catena di distribuzione. Dal punto di vista software la sezione di elaborazione provvede a eseguire il programma che gestisce il complesso di lettura ed elaborazione dei dati, il controllo di tutti gli attuatori e la comunicazione con gli altri dispositivi. Le unità di controllo elettronico contengono al loro interno [18] [20]:

1. porte di input e output;
2. memorie;
3. processore CPU;
4. convertitori ADC e DAC;
5. un bus interno.

Porte di Input e Output. I microcomputer richiedono dispositivi di interfaccia che consentano loro di comunicare con altri sistemi. A tale scopo sono presenti porte di ingresso e uscita per monitorare o inviare segnali:

- porte analogiche (indicate con A.IN/OUT) per monitorare o inviare segnali di tensione e corrente;
- porte digitali (contrassegnate dalla sigla D.IN/OUT) per trasmettere o recepire segnali del tipo omonimo;

- porte per la trasmissione di segnali di potenza per comandare gli attuatori elettrici (POW.OUT.);
- porte di alimentazione (+,-,GND);
- porte per la comunicazione tramite altre periferiche attraverso bus (CAN-H, CAN-L);

Memorie. La memoria è la parte del sistema che memorizza sia le istruzioni, contenute nel programma, che deve svolgere il microprocessore, sia tutte le informazioni di cui il microprocessore necessita. Le memorie principali a semiconduttore sono di due tipi: ROM (*Read Only Memory*) e RAM (*Random Access Memory*). Nelle ROM le informazioni sono memorizzate in modo permanente e possono essere solo lette. Nelle RAM le informazioni possono essere o scritte o lette. Le RAM sono perciò memorie volatili, poiché tutte le informazioni contenute in esse vengono rimosse quando viene a mancare l'alimentazione del dispositivo. L'elemento base delle memorie a semiconduttore è la cella di memoria, dove è immagazzinato un bit; 8 celle formano una locazione di memoria, dove è immagazzinato un byte. Il numero di byte che può contenere un insieme di celle è detto capacità di memoria. Un insieme di celle di memoria è chiamato sistema di memoria ed ogni singola locazione di memoria è individuata assegnandole un codice binario detto indirizzo. I programmi memorizzati nella ROM sono talvolta chiamati firmware piuttosto che software poiché sono immutabili. Questo tipo di memoria consente al microcomputer di iniziare immediatamente ad eseguire il proprio programma non appena viene acceso. Un tipico microcomputer contiene sia memoria di tipo ROM che RAM.

CPU La CPU, acronimo di *Central Processing Unit*, o più comunemente processore, è un circuito integrato composto da migliaia di transistor e diodi su un chip di silicio, che ha il compito di eseguire istruzioni, calcoli ed elaborare dati. Al suo interno è presente:

- un'unità aritmetico-logica (ALU), che esegue le operazioni matematiche e logiche;

- un'unità di controllo (CU), che coordina il funzionamento degli altri componenti della CPU e gestisce l'esecuzione delle istruzioni del programma contenuto nella memoria ROM;

Tra le caratteristiche più importanti dei processori troviamo:

- la frequenza di *clock*, che indica la velocità con cui la CPU esegue le istruzioni, misurata in *GHz*.
- il numero di *core* che sono le unità di elaborazione indipendenti all'interno della CPU.

1.2.6 BUS di sistema e CAN-Bus

La complessità dei moderni sistemi di cablaggio è aumentata costantemente negli ultimi 40 anni circa e al giorno d'oggi il numero di sistemi controllati dall'elettronica è in continuo aumento. Una soluzione che permette di ridurre il peso e la complessità dei cablaggi consiste nell'utilizzare un bus dati comune, che consentirebbe anche la comunicazione reciproca tra i componenti. Paradossalmente, se i dati potessero essere trasmessi lungo un filo e resi disponibili a tutte le parti del veicolo, il cablaggio del veicolo potrebbe essere ridotto a soli tre fili. Questi fili sarebbero un'alimentazione, una connessione di terra e un filo di segnale.

La tecnologia o il processo di comunicazione tra più dispositivi elettronici mediante un unico canale o il sistema che consente la trasmissione, praticamente simultanea, di due o più messaggi attraverso la stessa linea è detta *multiplexing* o *multiplexing*. Ci sono diverse logiche che permettono la condivisione, senza equivoci e interferenze, dei segnali tra sorgenti multiple. Il *multiplexing* tipicamente adottato nei bus delle macchine agricole è a divisione di tempo TDM, ovvero ogni dispositivo ottiene a turno l'uso esclusivo del canale di trasmissione per un breve lasso di tempo.

Il CAN, sviluppato da Bosch nel 1986, è oggi lo standard di comunicazione nelle autovetture e nelle macchine agricole [20]. Il termine CAN, acronimo di *controller area network*, lascia intendere che le unità di controllo elettroniche, i sensori e gli attuatori sono collegati tra loro in un'unica rete ad anello e possono scambiare dati reciprocamente, perciò si dice che il CAN è un sistema di bus seriale con funzionalità *multicast*. Il bus funziona ad una velocità di trasmissione dati fino a 1 *Mbit/s*. Il

bus dati è fondamentalmente composto da due fili che collegano insieme le ECU: la linea *high controller area network* (H-CAN) e *low controller area network* (L-CAN). Le linee adottano velocità di trasmissione differenti in modo che i messaggi più urgenti possano essere trasmessi con più tempestività rispetto ai messaggi differibili. Le tensioni agenti sui fili sono specificate da una coppia di stati: lo stato dominante, associato ad un'alta tensione, e lo stato recessivo, associato ad una bassa tensione. Un segnale CAN è quindi costituito da una sequenza di cifre binarie, ciascuna delle quali rappresenta un bit. Il messaggio è composto da:

- una sequenza di bit di avvio;
- il nominativo o identificativo (ID) del mittente;
- alcuni bit di controllo;
- i bit dei dati;
- bit di controllo per il rilevamento degli errori;
- un segnale di conferma;
- una sequenza di bit di fine messaggio.

L'identificatore definisce il contenuto del messaggio e la sua priorità ed è univoco all'interno del rete. Attualmente, lo standard SAE J1939 della *Society of Automotive Engineers* è la normativa raccomandata per definire il protocollo di comunicazione CAN-Bus dei veicoli [13].

1.2.7 Microcontrollori Arduino

I microcontrollori Arduino (in Fig. 1.7 il modello più conosciuto) sono schede digitali *low cost* programmabili, utilizzate per realizzare i progetti elettronici più disparati: leggere sensori, effettuare calcoli, comandare attuatori e comunicare con altri dispositivi elettronici [3] [1]. Le schede integrano di base:

- un microprocessore;
- memorie di tipo flash e EEPROM;
- convertitori DAC e ADC;

- porte di I/O per l'alimentazione e per la connessione con le periferiche esterne, come sensori, moduli aggiuntivi, computer etc.

Arduino ha rapidamente guadagnato popolarità come piattaforma di prototipazione *open source* grazie alla sua flessibilità, facilità d'uso e vasta comunità di sviluppatori.

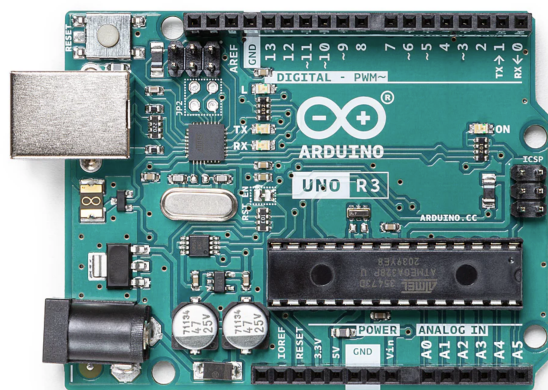


Figura 1.7: Il popolare modello di scheda digitale a microcontrollore Arduino Uno R3.

Tali schede sono progettate per essere accessibili anche da coloro che non hanno una vasta esperienza in elettronica o programmazione. La loro interfaccia *user friendly*, insieme alla disponibilità di un'ampia documentazione e tutorial online, rende facile iniziare a lavorarci. Grazie, inoltre, alla disponibilità di numerose librerie e moduli aggiuntivi, le schede Arduino possono essere facilmente interfacciate con una vasta gamma di sensori e dispositivi di misura. Ciò permette di acquisire dati da molteplici fonti e di integrarli in un unico sistema. La programmazione della scheda avviene tramite il software nativo Arduino IDE che utilizza un linguaggio di programmazione basato su C/C++. La popolarità delle schede in questione è giustificata anche dalla facile accessibilità economica.

1.3 Obiettivi

Attualmente la corilicoltura ha raggiunto un grado di meccanizzazione estremamente elevato, specialmente nei processi di raccolta. Le macchine raccoglitrice di nocciole, in particolar modo le semoventi, assicurano ottime capacità di raccolta, un'eccellente pulizia del prodotto, una riduzione sostanziale della manodopera e dei costi associati, oltre a fornire sicurezza e comfort nella conduzione grazie alle cabine di guida. Tuttavia, la diffusione delle tecnologie digitali dell'Agricoltura 4.0 non ha ancora interessato questo tipo di veicolo. Al giorno d'oggi, grazie all'applicazione dell'elettronica in agricoltura è possibile monitorare le macchine e le produzioni agricole e, con l'avvento dell'agricoltura di precisione, di ridurre i costi e gli sprechi dei processi produttivi. Inoltre, le tecnologie digitali sono diventate più accessibili economicamente e semplici da utilizzare: ne è indice la diffusione dei microcontrollori a basso costo, programmabili attraverso software *open source*, studiati per acquisire con estrema facilità i dati provenienti da sensori di vario genere.

Pertanto l'obiettivo di questo trattato è progettare e implementare un sistema di acquisizione dati *low cost* per monitorare le prestazioni di una semovente raccoglitrice di nocciole, la Cimina 300S, prodotta da FACMA. Nello specifico sono stati installati sensori per misurare la velocità di avanzamento, le pressioni nel circuito idraulico di trasmissione e frenatura, le temperature dell'olio idraulico, le portate d'aria nei condotti di ventilazione, i parametri motore come potenza, coppia, regime di giri e consumi, la posizione satellitare e la pendenza affrontata. Ciò può essere utile per:

- definire con precisione quali siano i parametri operativi della macchina durante le tipiche lavorazioni;
- stabilire le migliori condizioni operative per massimizzare l'efficienza dei processi di raccolta e minimizzare i costi e gli sprechi associati;
- abilitare un miglioramento tecnologico del mezzo per incrementarne l'efficienza;
- trovare gli strumenti, i sensori e i metodi di misura più efficaci per la realizzazione delle telemetria su veicoli di questo tipo.

La complessità dello studio richiede un approccio multidisciplinare che integri conoscenze di ingegneria, agronomia e informatica. In questo trattato saranno esplorate diverse aree di competenza: dalla programmazione *software* del sistema di acquisizione dati alla sua integrazione sulla semovente raccoglitrice, fino alla validazione dei risultati ottenuti in condizioni reali di utilizzo.

Capitolo 2

Materiali e metodi

Questo capitolo tratta dei metodi di progettazione e degli strumenti adottati per realizzare il sistema di acquisizione dati completo installato sulla semovente. In altre parole si affrontano sia gli aspetti *hardware* che *software* da tenere a mente nella realizzazione del DAQ. Inizialmente si esaminano nel dettaglio gli strumenti impiegati per la realizzazione del sistema di acquisizione dati, partendo dai sensori e arrivando infine ai microcontrollori, giustificandone la scelta. In seconda fase si analizzano i cablaggi, più precisamente i circuiti di condizionamento dei segnali e i collegamenti elettrici con i microcontrollori, soffermandosi anche sul processo di realizzazione e implementazione dell'impianto sul veicolo. Infine, ampio spazio è dedicato alla programmazione dei microcontrollori in Arduino IDE, per acquisire e salvare i dati rilevati dai sensori, e al processo di elaborazione dei dati ottenuti dalle prove in ambiente Matlab, per estrapolare le informazioni di maggior interesse.

2.1 Hardware

2.1.1 Sensori

Sono state identificate le principali variabili per valutare le *performances* delle semoventi aspiratrici. Da ciò è scaturito un piano per misurarle, che viene sintetizzato in tab. 2.1, riportando variabili misurate, tipologia e posizionamento di ogni sensore. La posizione di ogni sensore sul veicolo è anche rappresentata nelle Fig. 2.1 e 2.2.

Parametro	Sensore	Posizione
Posizione	GPS	Cabina
Velocità avanzamento	GPS, prossimità Hall	Cabina, ruote
Pendenza terreno	Giroscopio, accelerometro	Cabina, telaio
Potenza, coppia, consumo combustibile	CAN-BUS Data logger	Presa diagnostica
Pressione olio idraulico	Trasmettitori di pressione	Tubazioni idrauliche, mandata pompa ingresso motori idraulici
Temperatura olio idraulico	Termometro a resistenza	Ingresso e uscita radiatore
Portata d'aria nel circuito pneumatico	Sensori di pressione differenziale	Condotto di aspirazione e di insaccamento

Tabella 2.1: Sensori e metodi utilizzati per misurare i principali parametri prestazionali della semovente.

Nei successivi paragrafi si motiva la scelta dei sensori utilizzati, analizzando il principio di funzionamento e le specifiche tecniche degli stessi.

Temperatura del fluido idraulico

La misurazione della temperatura dell'olio di trasmissione idraulica potrebbe fornire informazioni utili sul rendimento della trasmissione idraulica e sull'efficienza di raffreddamento del radiatore. Per massimizzare l'efficienza delle macchine idrauliche è importante che la temperatura del fluido di lavoro si mantenga in un preciso *range* di valori, scongiurando surriscaldamenti che comporterebbero una variazione della viscosità e la degradazione del fluido stesso.

Per effettuare la misurazione della temperatura dell'olio si è scelto di adottare dei termometri al platino Pt100 di Autosen (Fig. 2.3). I sensori

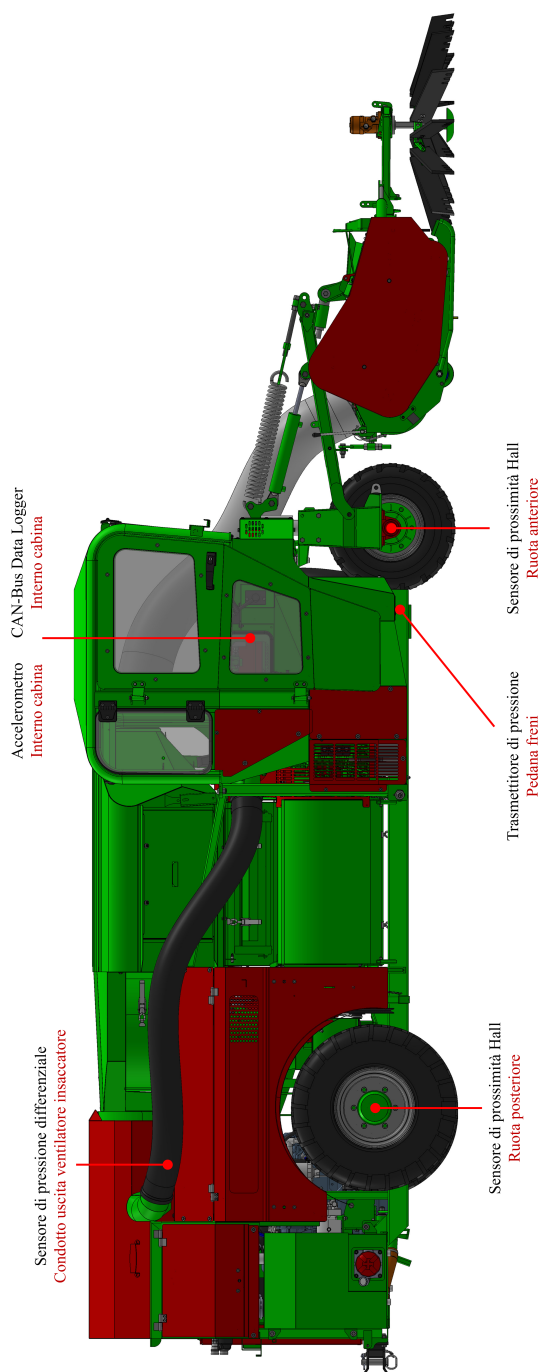


Figura 2.1: Posizione di alcuni sensori in vista laterale destra.

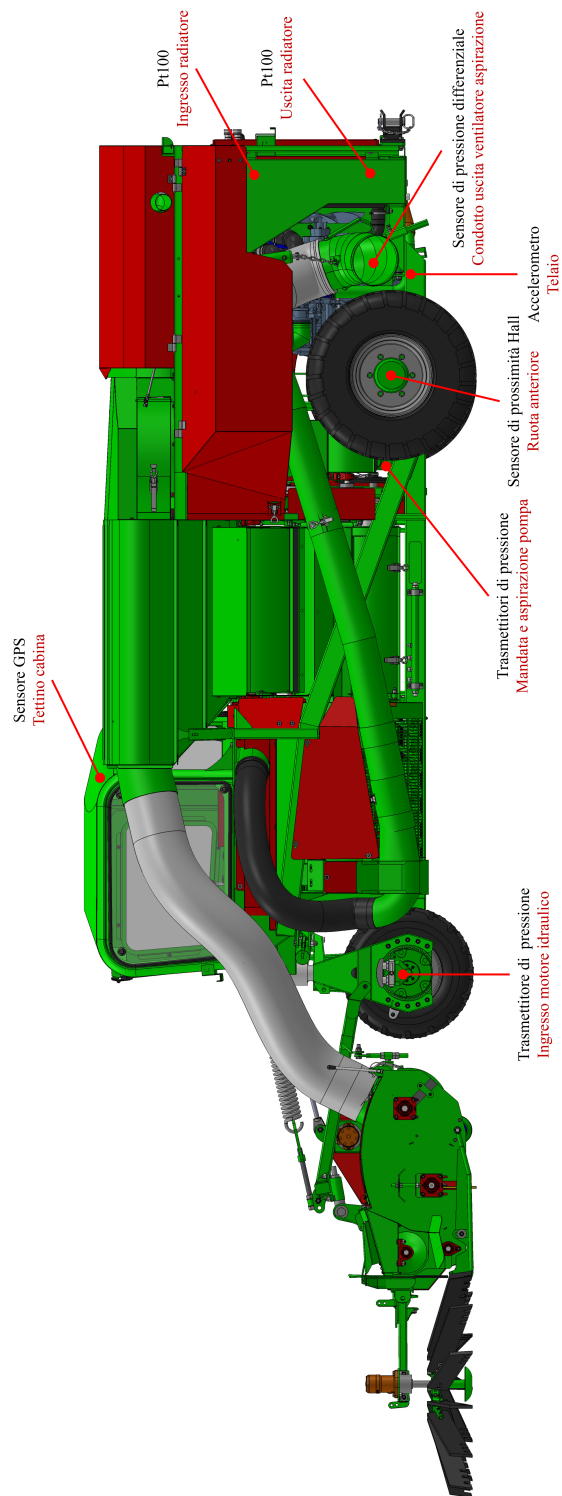


Figura 2.2: Posizione di alcuni sensori in vista laterale sinistra.

Pt100, noti anche come termoresistenze al platino, incorporano un materiale resistivo, la cui resistività elettrica varia in modo prevedibile, in funzione della temperatura dell’ambiente in cui si trovano [24]. All’aumentare della temperatura, infatti, la resistenza del platino aumenta secondo una relazione di tipo lineare. Il nome Pt100 deriva dal fatto che il trasduttore presenta a 0°C una resistenza pari a 100 Ω.



Figura 2.3: AT105 Sonda ad avvitamento Pt100.

Tale dispositivo, grazie al design compatto e alla presenza della filettatura di collegamento G1/4”, può essere facilmente integrato senza richiedere modifiche sostanziali sul sistema idraulico del veicolo. Due sensori di questo tipo sono stati installati uno all’ingresso del radiatore dell’olio e uno all’uscita dello stesso, direttamente avvitato nei raccordi delle condotte idrauliche. Le specifiche tecniche primarie del sensore sono riportate in Tab.2.2.

Pt100 AT105	
Campo di misura	−40 ± 200 °C
Accuratezza	Class A
	±0.06 Ω at 0 °C
	±0.13 Ω at 100 °C
Output	100 Ω at 0 °C
	138.5 Ω at 100 °C
Prezzo	23,14 €

Tabella 2.2: Specifiche tecniche del sensore di temperatura Pt100 AT105.

Pressione del fluido idraulico

La pressione dell’olio idraulico potrebbe fornire informazioni sul sistema di trasmissione idraulico per l’avanzamento; in particolare quali siano anzitutto le condizioni operative della pompa e dei motori idraulici,

sia in aspirazione che in mandata, nei diversi scenari di lavoro della macchina. È importante verificare che le pressioni si mantengano al disotto di certi valori soglia per mantenere un rendimento elevato delle macchine idrauliche e scongiurare picchi di pressione, che comporterebbero un'apertura prolungata delle valvole di massima o che minaccerebbero l'integrità strutturale del sistema. Inoltre, l'installazione dei sensori di pressione anche nel circuito frenante potrebbe essere utile per verificarne il corretto funzionamento, ai fini di una migliore sicurezza nella guida.

Per misurare la pressione dell'olio idraulico all'interno del circuito di trasmissione idrostatica della macchina si è scelto di utilizzare dei trasmettitori di pressione piezoresistivi. I materiali piezoresistivi si contraddistinguono per la variazione di resistenza che subiscono in risposta agli stress e alle deformazioni. Questi materiali vengono impiegati per la realizzazione dei cosiddetti estensimetri da incorporare all'interno di appositi diaframmi. Il diaframma, posto all'interno del sensore, se sottoposto alla pressione di un fluido, si deforma e provoca quindi una variazione di resistenza dell'estensimetro [10]. Ricordiamo che il circuito di trasmissione idraulica della semovente in esame incorpora una pompa a pistoncini assiali a cilindrata variabile, che aspira dal serbatoio e manda il fluido ai motori idraulici posteriori, posto l'uno rispetto all'altro in parallelo. Il motore idraulico anteriore, invece, è posto in serie ai motori idraulici posteriori.

La macchina poi è dotata di un circuito frenante negativo per incrementare la sicurezza. Ciò vuol dire che la macchina è tenuta frenata per mezzo di molle ed elementi elastici e l'azione della pressione idraulica nel circuito frenante permette di sbloccare i freni. Il funzionamento è, quindi, l'inverso dei circuiti di frenatura tradizionali che, invece, sfruttano l'azione della pressione idraulica per frenare il veicolo.

Si è scelto di installare i sensori di pressione:

- sulla bocca di aspirazione della pompa;
- all'uscita della pompa e, quindi, all'ingresso dei motori idraulici posteriori;
- all'ingresso del motore idraulico anteriore e, quindi, all'uscita dei motori idraulici posteriori;
- all'ingresso dei freni.

La filettatura G1/4M, di cui sono muniti i sensori scelti, permette un montaggio facile senza dover effettuare modifiche all’impianto. In Fig.



Figura 2.4: Trasmettitore di pressione AP031.

2.4 vi è la conformazione di uno di questi sensori, mentre in Tab. 2.3 si riportano le specifiche tecniche dei sensori impiegati.

	AP031	AP026	AP029	AP030
Campo m.	$-1 \div 1\ bar$	$0 \div 60\ bar$	$0 \div 400\ bar$	$0 \div 1000\ bar$
Accuratezza	$\pm 0.5\ \% FS$	$\pm 0.5\ \% FS$	$\pm 0.5\ \% FS$	$\pm 0.5\ \% FS$
Input	$7 \div 33\ V$	$7 \div 33\ V$	$7 \div 33\ V$	$7 \div 33\ V$
Output	linear	linear	linear	linear
	$4 \div 20\ mA$	$4 \div 20\ mA$	$4 \div 20\ mA$	$4 \div 20\ mA$
Prezzo	69,27 €	69,27 €	69,27 €	69,27 €

Tabella 2.3: Specifiche tecniche dei trasmettitori di pressione: AP031, AP026, AP029, AP030.

Pressione differenziale dell’aria

La semovente raccoglitrice di nocciole in esame è dotata di due ventilatori centrifughi: il primo permette l’aspirazione del prodotto da terra, mentre il secondo genera un flusso d’aria in grado di trasportare il frutto all’interno della macchina fino al carrello di stoccaggio. Le performance dei ventilatori, in termini di pressione e portata d’aria sono importanti per assicurare le funzionalità in aspirazione della macchina che deve garantire il più possibile che il prodotto venga aspirato senza rimanere a terra, sia in insaccamento, per far sì che il prodotto venga

correttamente trasportato all'interno della macchina e sui carrelli di raccolta evitando che si blocchi per una bassa portanza aerodinamica o che, al contrario, per l'eccessiva velocità si frantumi all'impatto con le pareti. Una portata d'aria eccessiva condurrebbe a un elevato consumo di energia. Perciò è fondamentale conoscere la velocità dell'aria per effettuare un corretto trasporto del prodotto e garantire una sufficiente capacità aspirante al fine di limitare le perdite di frutti che rimangono a terra.

Molto ardua è stata la scelta dei sensori per valutare le performance dei ventilatori in termini di portata d'aria erogata. Il sistema richiede, infatti, un tipo di misura non invasiva, in quanto all'interno delle condotte pneumatiche transita il prodotto raccolto, insieme a una notevole quantità di polveri e alcuni elementi di scarto come sassolini, rami e foglie. Perciò si è scelto di effettuare una stima indiretta della portata misurando la pressione statica all'interno delle condotte in punti specifici del circuito. Sappiamo, infatti, che le perdite di carico nei condotti dipendono dalla portata d'aria erogata con una legge quadratica del tipo:

$$\Delta p = kQ^2 \quad (2.1)$$

con k opportuno coefficiente di perdita in $pa \cdot s/m^3$.



Figura 2.5: Sensori di pressione differenziale dell'aria: a sinistra il modello MPX10DP, a destra il modello MPXV7002DP.

Per misurare la pressione statica si è scelto di adottare sensori di pressione dell'aria differenziale piezoresistivo. Il sensori di questo tipo misurano la differenza di pressione che c'è tra le due porte d'ingresso dell'aria. Il loro funzionamento è basato maggiormente su tre principi differenti: deflessione della membrana, variazione di resistenza e cavità differenziale. Sono sensori dotati di una membrana in silicone

deformabile alla differenza di pressione sulla quale è presente un sottile filo conduttore che subisce una variazione della propria resistenza elettrica in funzione della pressione applicata.

Sono stati impiegati due sensori di questo tipo, uno installato sul circuito pneumatico di aspirazione del prodotto e uno sul circuito pneumatico per l'insaccamento del prodotto. La differente conformazione dei circuiti e dei ventilatori impone l'utilizzo di due modelli di sensore differenti, in quanto cambia il campo di misura da tenere in considerazione. In Fig. 2.5 si trovano i due modelli utilizzati, mentre in Tab. 2.4 si può consultare la scheda tecnica degli stessi.

	MPX10DP	MPXV7002DP
Campo di misura	$0 \div 10 \text{ kPa}$	$-2 \div 2 \text{ kPa}$
Accuratezza	$\pm 1 \% FS$	$\pm 2.5 \% FS$
Input	$3 \div 6 \text{ V}$	$4.75 \div 5.25 \text{ V}$
Output	linear $20 \div 55 \text{ mV}$	linear $0.5 \div 4.5 \text{ V}$
Sensibilità	3.5 mV/kPa	1 V/kPa
Prezzo	14,24 €	22,00 €

Tabella 2.4: Specifiche tecniche dei sensori di pressione dell'aria: MPX10DP, MPXV7002DP.

Velocità di avanzamento

La velocità di avanzamento del mezzo ha influenza sui consumi, la capacità di raccolta e i tempi di transito su strada. Un'ottimizzazione della velocità consente di massimizzare il rendimento della raccolta, garantendo che il veicolo operi alla massima capacità senza sovraccaricare il sistema.

Per il monitoraggio della velocità di avanzamento del mezzo si è scelto di adottare dei sensori di prossimità, in particolare i sensori di prossimità ad effetto Hall, che sono rilevatori magnetici e funzionano sul fenomeno fisico che denomina lo stesso sensore. Dato un materiale semiconduttore attraversato da corrente elettrica, se esposto ad un campo magnetico si genera una differenza di potenziale ai suoi capi, che può essere misurata e convertita in un segnale elettrico proporzionale alla forza del campo magnetico [18]. Il trasduttore implementato sulla semovente FACMA

Cimina 300S è un sensore induttivo **AI908**, in Fig. 2.6, di classe standard acquistato da Autosen, le cui specifiche tecniche vengono esposte in Tab. 2.6.

Si è scelto di fissare il sensore al telaio in prossimità di una ruota dentata in lamiera, montata solidale alle ruote. Quando un dente si affaccia davanti al sensore l'uscita dello stesso cambia e quindi è possibile calcolare la velocità di rotazione in funzione del numero di impulsi percepiti dal sensore nell'unità di tempo. Tre sono i sensori implementati sulla semovente, uno per ruota.



Figura 2.6: Sensore induttivo AI908.

AI908	
Campo di misura	$0 \div 8\text{ mm}$
Input	$10 \div 30\text{ V}$
Output	Digital PNP $0 \div V_{in}$
Prezzo	13,89 €

Tabella 2.5: Specifiche tecniche del sensore di prossimità Hall AI908.

Accelerazioni e inclinazioni

I parametri operativi delle macchine agricole sono influenzati dall'ambiente in cui operano, in particolar modo dalla conformazione del terreno. La capacità di trazione, i consumi e lo sforzo della trasmissione vengono sicuramente influenzati dalle condizioni del terreno, in particolar modo dalla la pendenza. Inoltre, un terreno inclinato o irregolare può aumentare il rischio di ribaltamento del mezzo, soprattutto su terreni

collinari come quelli maggiormente adibiti alla raccolta delle nocciole. Monitorare l'inclinazione della macchina permette di individuare le zone del campo a rischio ribaltamento, fenomeno che mina la sicurezza del conducente.

Tra i sensori impiegati per misurare le accelerazioni e le inclinazioni troviamo gli accelerometri e i giroscopi. Gli accelerometri sfruttano il principio della forza inerziale per misurare l'accelerazione lineare rispetto a un sistema di riferimento inerziale. Nella configurazione tipica di un accelerometro una massa sospesa è connessa al telaio attraverso molle o elementi elastici e smorzatori di vibrazioni. Quando il telaio, solidale con il corpo vibrante, si muove, la massa tende a conservare il suo stato di quiete o di moto rettilineo uniforme, così che si innescano delle deformazioni degli elementi elastici proporzionali allo spostamento del telaio, che possono essere misurate tramite diversi metodi. Tra le varie tipologie di accelerometri maggiormente impiegati troviamo i dispositivi MEMS (Micro Electro Mechanical Systems), costituiti da uno o più elementi micro-meccanici sospesi all'interno del sensore. Questi componenti, chiamati masse sensibili, sono generalmente realizzati con strutture a trave, a molla o a pendolo.

L'accelerometro adottato per misurare l'inclinazione della semovente in esame è racchiuso nel modulo GY-521 venduto da AZ-Delivery, in Fig. 2.7, di cui la scheda tecnica viene esposta in Tab. 2.6. Il modulo è equipaggiato con il chip MPU-6050, che contiene un accelerometro e un giroscopio, entrambi a tre assi.

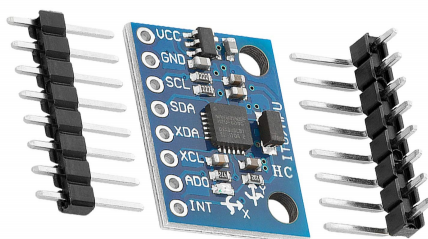


Figura 2.7: Accelerometro e Giroscopio GY-521 MPU-6050.

Si è deciso di implementare due di questi moduli: il primo è stato installato sul telaio in prossimità delle ruote posteriori, mentre il secondo è stato posizionato in cabina. I sensori sono stati racchiusi all'interno di

GY-521	
Campo di misura	$\pm 2, 4, 8, 16 g$
programmabile	$\pm 250, 500, 1000, 2000 \text{ deg/s}$
Input	$3 \div 5 V$
Output	Digital I2C
I2C Address	0x68 or 0x69
Bande programmabili	$5 \div 256 Hz$
filtro passa basso	
Prezzo	3,60 €

Tabella 2.6: *Specifiche tecniche del modulo accelerometro giroscopio GY-521.*

scatole in PLA e avvolti con un materiale in grado di smorzare le vibrazioni. Si è cercato di garantire il più possibile l'allineamento degli assi dei sensori con gli assi principali del veicolo.

Posizione

Conoscere la posizione del veicolo è fondamentale nei sistemi di telemetria delle macchine agricole, poiché permette di geolocalizzare le informazioni provenienti dai sensori e di adottare un approccio sito-specifico sulle future operazioni in campo. L'evoluzione della posizione nel tempo permette di tener traccia della porzione di campo lavorata e di effettuare una stima della capacità oraria di raccolta, un parametro fondamentale per l'agricoltore per pianificare le lavorazioni. Inoltre, se la posizione viene misurata con una certa accuratezza, si può stimare anche la velocità di avanzamento del veicolo.

La misura della posizione effettuata dalle antenne GNSS si basa sul principio della trilaterazione o triangolazione [8]. In particolare viene misurata la distanza dai satelliti sulla base del tempo di volo che un segnale radio impiega a raggiungere il ricevitore a terra. Per misurare questo tempo si usano orologi molto precisi: atomici a bordo dei satelliti, al quarzo nei ricevitori. Tutti i satelliti inviano simultaneamente a terra un segnale che contiene dei parametri di navigazione, a partire dai quali è possibile determinare la loro posizione nello spazio. Tre satelliti sono sufficienti ad individuare due punti nello spazio, di cui uno che giace sulla superficie terrestre. Ogni altro satellite che si trova nel campo

visivo del sensore incrementa l'accuratezza del risultato della trilaterazione e quindi della misura di posizione.

Per misurare la posizione del veicolo durante la raccolta si è optato per il modulo SparkFun GPS Breakout - NEO-M9N, con una GNSS Magnetic Mount Antenna, raffigurati l'uno di fianco all'altro in Fig. 2.8.

In Tab. 2.8 troviamo invece alcune specifiche tecniche del modulo GNSS.

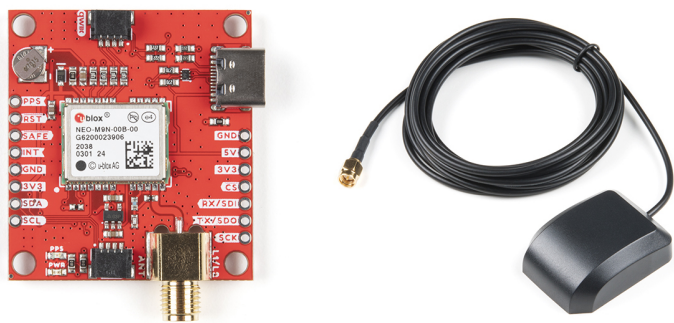


Figura 2.8: SparkFun Modulo GPS NEO-M9N con un antenna magnetica.

NEO-M9N + GNSS Antenna	
Costellazioni supportate	GPS/QZSS, GLONASS BEIDOU, GALILEO
Input	2.7 ÷ 3.6 V
Output	Digital I2C
Accuratezza posizione	~ 1.5 m
Accuratezza velocità	~ 0.05 m/s
Prezzo	68,95 + 12,83 €

Tabella 2.7: Specifiche tecniche del modulo NEO-M9N con antenna GPS.

2.1.2 Moduli aggiuntivi

Real Time Clock DS3231

Il modulo real time clock DS3231, in Fig. 2.9, è un dispositivo ampiamente utilizzato per tenere traccia del tempo reale nei progetti basati su Arduino e altri microcontrollori. La scelta di integrare questo dispositivo in tutte le schede utilizzate nel progetto è motivata dalla necessità di garantire una sincronizzazione temporale tra tutte le informazioni in uscita dalle schede, per agevolare un successivo confronto tra le variabili misurate. Il modulo contiene un oscillatore al cristallo compensato termicamente e una batteria removibile.

In Tab. 2.8 troviamo le specifiche tecniche salienti del dispositivo.

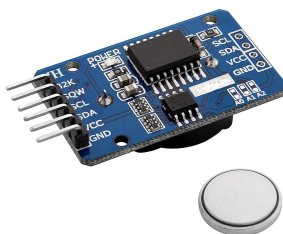


Figura 2.9: RTC DS3231.

RTC DS3231	
Unità misurate	YY/MM/DD/HH/mm/ss
Input	2.3 ÷ 5.5 V
Output	Digital I2C
I2C Address	0x57
Batteria	3V Button cell
Accuratezza	$\pm 2 \text{ ppm}$
Prezzo	3,20 €

Tabella 2.8: Specifiche tecniche del clock RTC DS3231.

SD Card Reader SPI

Il lettore di memoria SD, in Fig. 2.10, permette di archiviare i dati in uscita da Arduino.

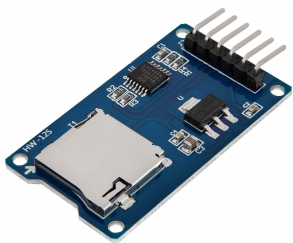


Figura 2.10: *Lettore SD SPI.*

Il modulo è stato implementato in ciascun dispositivo Arduino utilizzato nel progetto. Ogni scheda è stata programmata per scrivere i segnali in uscita dai sensori all’interno di un file di testo (.txt) salvato su scheda SD. Alcune specifiche tecniche del dispositivo sono presentate in Tab. 2.9.

SD Card Reader SPI	
Card supportate	FAT16/32 SD, micro SD card
Capacità SD	$\leq 16\text{ GB}$
Input	$3.3 \div 5\text{ V}$
Output	SPI Communication
Prezzo	2,00 €

Tabella 2.9: *Specifiche tecniche del lettore SD SPI.*

Display OLED I2C

È essenziale ricordare che, per visualizzare in tempo reale i segnali elaborati da una scheda di acquisizione dati, è necessario che quest’ultima sia connessa ad un computer. Pertanto la decisione di utilizzare un display è motivata dalla necessità di controllare il corretto funzionamento della scheda senza dover inserire un PC all’interno del veicolo. Nello specifico si è deciso di adottare un Display OLED, raffigurato in Fig. 2.11, le cui specifiche tecniche sono racchiuse in Tab. 2.10.

Gli schermi OLED, (*Organic Light Emitting Diodes*), sono schiere di LED impilati insieme in una matrice. A differenza degli schermi LCD tradizionali, i display OLED non richiedono retroilluminazione perché ogni pixel emette luce in modo indipendente quando alimentato. Lo

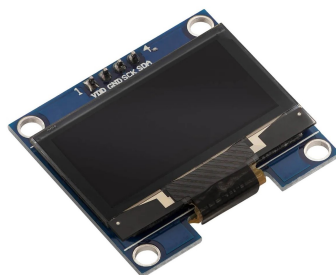


Figura 2.11: Display OLED I2C.

OLED I2C Display	
Input	3.3 ÷ 5 V
Output	Digital I2C
I2C Address	0x3C
Dimensioni schermo	0.96 inches
Risoluzione	128x32 pixels
Colore pixel	White
Prezzo	6,40 €

Tabella 2.10: Specifiche tecniche del display OLED I2C.

schermo ha un chip del driver chiamato SSD1306 con un'interfaccia I2C per la comunicazione con il microcontrollore principale.

Multiplexer TCA9548A

Il multiplexer TCA9548A, in Fig. 2.12, è un dispositivo che consente a un microcontrollore o a un'altra unità di controllo di comunicare con più dispositivi di tipo I2C utilizzando un singolo bus.

Quando più dispositivi I2C con lo stesso indirizzo sono collegati al bus principale, il TCA9548A risolve i conflitti di indirizzamento interrogando selettivamente un dispositivo alla volta. Infatti, questo dispone di molteplici canali di selezione, dando la possibilità di collegare diversi dispositivi I2C a un singolo bus principale. Per selezionare un canale specifico, il microcontrollore invia comandi di controllo all'indirizzo del multiplexer. In Tab. 2.11 si elencano le caratteristiche tecniche del modulo in questione.

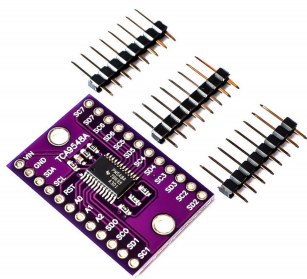


Figura 2.12: Multiplexer TCA9548A.

Multiplexer TCA9548A	
Input	1.8 ÷ 5 V
Output	Digital I2C
I2C Address	Aggiustabile da 0x70 a 0x77
Canali	8
Prezzo	4,00 €

Tabella 2.11: Specifiche tecniche del Multiplexer TCA9548A.

SparkFun CAN-Bus Shield

La centralina motore dei mezzi agricoli può fornire moltissime informazioni che derivano dai sensori collocati di serie sul motore termico. Questi sensori raccolgono informazioni sulle condizioni operative del motore stesso, sul combustibile, sulle condizioni dei gas dall’aspirazione allo scarico, sulle condizioni ambientali e sul sistema di raffreddamento. Nello specifico, per quanto riguarda l’analisi che stiamo effettuando, tra i parametri che si intende misurare, troviamo numero di giri, posizione del pedale d’accelerazione, coppia e potenza, portata di combustibile e condizioni dell’aria ambientale. Queste variabili sono fondamentali per capire come viene utilizzata la semovente dall’utente esterno, quali siano le condizioni di lavoro ambientali e i consumi associati alle singole operazioni effettuate.

Per riuscire a intercettare questi dati, che scorrono sulla linea CAN, si è deciso di adottare una SparkFun CAN-Bus Shield, in Fig.2.13, connessa ad una scheda digitale SparkFun RedBoard. La *shield* è progettata per essere facilmente collegata alla RedBoard tramite i pin a foro passante,

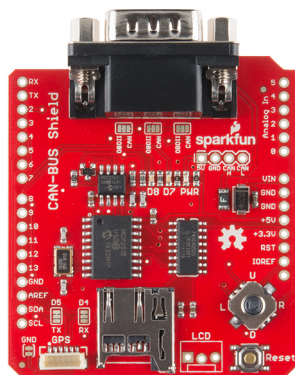


Figura 2.13: SparkFun CAN-Bus Shield.

consentendo una connessione stabile e affidabile. Alcune specifiche tecniche della scheda sono riportate in Tab.2.12.

CAN-Bus Shield	
CAN Controller	MPC2515
CAN Transceiver	MPC2551
Ingressi CAN	ODB-II CAN-H, CAN-L
Predisposizioni	Scheda SD Schermo LCD
Prezzo	26,60 €

Tabella 2.12: Specifiche tecniche della Shield CAN-BUS di SparkFun.

2.1.3 Microcontrollori

Arduino Mega 2560 Rev3

La scheda di acquisizione dati Arduino Mega 2560 Rev3, in Fig. 2.14, è stata impiegata per l'acquisizione dei dati provenienti da:

- 4 sensori di pressione dell'olio idraulico;
- 2 sensori di temperatura dell'olio idraulico;
- 2 sensori di pressione differenziale dell'aria;

- 2 accelerometri/giroscopi;

Le specifiche tecniche del microcontrollore in questione sono raccolte in Tab.2.13.

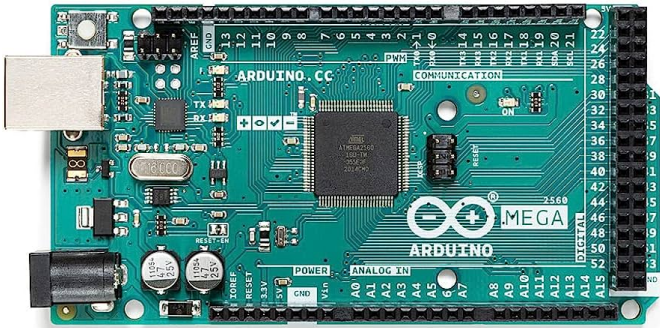


Figura 2.14: Arduino Mega 2560 Rev3.

Arduino Mega 2560 Rev3	
Tensione in ingresso	7 ÷ 12 V
Ingressi alimentazione	Jack 12V, Vin Pin, USB-B
Pin Analogici	16
Pin Digitali I/O	54, of witch 15 PWM
Processore	ATmega2560
Frequenza di Clock	16 MHz
Memoria Flash	256 kB
SRAM	8 kB
EEPROM	4 kB
Comunicazioni possibili	I2C, SPI, UART
Prezzo	46,00 €

Tabella 2.13: Specifiche tecniche della scheda Arduino Mega 2560 Rev3.

Arduino UNO R4 Minima

Si è deciso di utilizzare un Arduino UNO R4 Minima (Fig. 2.15) per campionare esclusivamente i sensori di prossimità Hall. Gli ultimi sono stati separati dagli altri sensori e collegati a una scheda di acquisizione dedicata al fine di massimizzare la frequenza di campionamento. Tale

scelta nasce dal desiderio di garantire una misura corretta dei segnali in uscita dal sensore Hall, per poter effettuare una stima altrettanto corretta della velocità. Infatti, affinché non si verifichi il fenomeno dell'*aliasing*, o sottocampionamento, la frequenza di campionamento deve essere almeno doppia rispetto alla frequenza con cui il segnale si presenta.

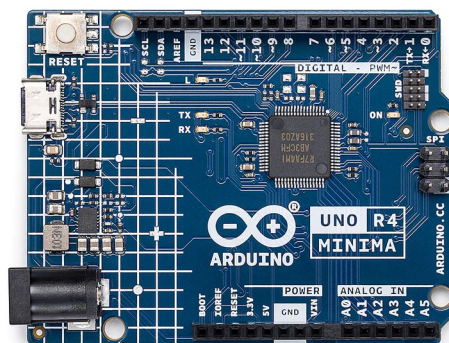


Figura 2.15: *Arduino UNO R4 Minima.*

In Tab. 2.14 sono elencate, oltre al prezzo, le caratteristiche *hardware* della scheda.

Arduino R4 Minima	
Tensione in ingresso	$6 \div 24\text{ V}$
Ingressi alimentazione	Jack 12V, Vin Pin, USB-C
Pin Analogici	6
Pin Digitali I/O	14, of witch 6 PWM
Processore	Arm Cortex-M4
Frequenza di Clock	48 MHz
Memoria Flash	256 kB
RAM	32 kB
Comunicazioni possibili	I2C, SPI, UART, CAN
Prezzo	18,00 €

Tabella 2.14: *Specifiche tecniche della scheda Arduino R4 Minima.*

SparkFun RedBoard

La SparkFun RedBoard, in Fig. 2.16, è stata utilizzata in duplice copia: sia per raccogliere i dati provenienti dalla centralina motore, sia per il sistema di localizzazione GNSS. Infatti, la scheda è studiata per offrire un collegamento semplificato con la CAN-BUS Shield, e il modulo NEO-M9N, prodotti sempre dalla stessa casa costruttrice. In Tab. 2.15 è

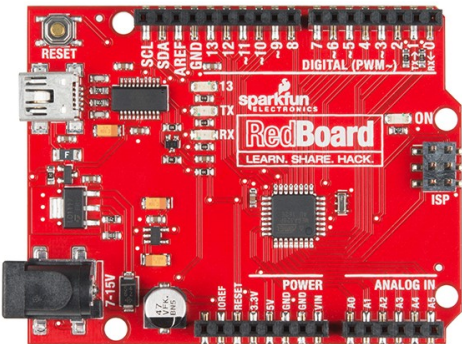


Figura 2.16: SparkFun RedBoard.

possibile consultare le specifiche tecniche della scheda di acquisizione.

Arduino R4 Minima	
Tensione in ingresso	7 ÷ 15 V
Ingressi alimentazione	Jack 12V, Vin Pin, Micro-USB
Pin Analogici	6
Pin Digitali I/O	14, of witch 6 PWM
Processore	ATmega328
Frequenza di Clock	16 MHz
Memoria Flash	32 kB
Comunicazioni possibili	I2C (Qwiic), SPI, UART, CAN
Prezzo	19,80 €

Tabella 2.15: Specifiche tecniche della scheda SparkFun RedBoard.

2.1.4 Assemblaggio dei componenti elettronici

All'interno dell'officina dell'azienda Facma S.r.l. la produzione dei collegamenti è avvenuta attraverso un processo accurato e meticoloso, eseguito interamente a mano.

Il primo passo nella realizzazione dei cablaggi è la progettazione del layout, dove vengono definiti i collegamenti elettrici necessari e il loro successivo posizionamento. Questa fase ha richiesto un'attenta analisi dei requisiti tecnici e funzionali del progetto, nonché una pianificazione accurata dei percorsi dei cavi lungo la semovente, al fine di garantire un'installazione efficiente e ordinata.

Una volta completata la progettazione, si è passati alla fase di preparazione dei materiali. In base alle specifiche del progetto, sono stati selezionati i cavi e i connettori più adatti alle esigenze di collegamento.

Successivamente, ci si è occupati dell'assemblaggio dei collegamenti seguendo scrupolosamente le istruzioni e le linee guida stabilite durante la fase di progettazione. Questo processo richiede grande precisione e attenzione ai dettagli, poiché un singolo errore di cablaggio potrebbe compromettere il funzionamento del sistema.

Durante l'assemblaggio, è stata prestata particolare cura alla gestione dei cavi, evitando grovigli e sovrapposizioni che potrebbero causare interferenze elettromagnetiche o danneggiare i componenti elettronici. I cavi sono stati passati attentamente attraverso dei corrugati elettrici per assicurare un isolamento ottimale, mentre i connettori saldati o crimpati con precisione per garantire una connessione stabile e affidabile.

Una volta completato l'assemblaggio e verificato il corretto funzionamento di tutti i collegamenti effettuati, questi sono stati posizionati accuratamente sulla macchina, prestando particolare attenzione a evitare interferenze con i diversi componenti del veicolo.

Di seguito l'elenco completo dei materiali impiegati per la realizzazione e l'assemblaggio dell'intero sistema di acquisizione:

- cavi elettrici di diametro 1.75 mm;
- cavetti jumper per Arduino;
- amplificatori operazionali TL072CP;
- resistenze elettriche di varia impedenza;
- pulsanti a 2 vie;

- diodi led;
- basette millefori per la creazione di circuiti saldati;
- connettori maschio-femmina a differenti numero di vie;
- corrugati elettrici di vario diametro;
- guaine termorestringenti di varie misure;
- stagno per saldatura;
- scatole in PLA realizzate a misura attraverso la stampa 3D, per la protezione dei componenti e dei circuiti elettronici;
- nastro adesivo;
- fascette stringicavo.

In Tab. 2.16 si elencano i costi complessivi dell'hardware e dei materiali impiegati nel DAQ. In Fig. 2.17 troviamo le tipologie di sensore utilizzate, mentre in Fig. 2.18, 2.20 e 2.19 troviamo gli stessi sensori predisposti con cavi e connettori pronti per essere installati sulla macchina.

Voce	Costo
Sensori	476,36 €
Microcontrollori	101,80 €
Moduli aggiuntivi	86.4 €
Materiali e componenti elettrici	~ 100 €
Totale	~ 764.56 €

Tabella 2.16: Costi complessivi per la realizzazione del DAQ.



Figura 2.17: Alcuni sensori impiegati nel sistema di acquisizione: trasmettitori di pressione, sensori di pressione differenziale, pt100, sensore Hall e accelerometro.



Figura 2.18: Sensori di pressione differenziale dell'aria muniti di cavi e connettori, pronti per l'installazione sulla semovente.

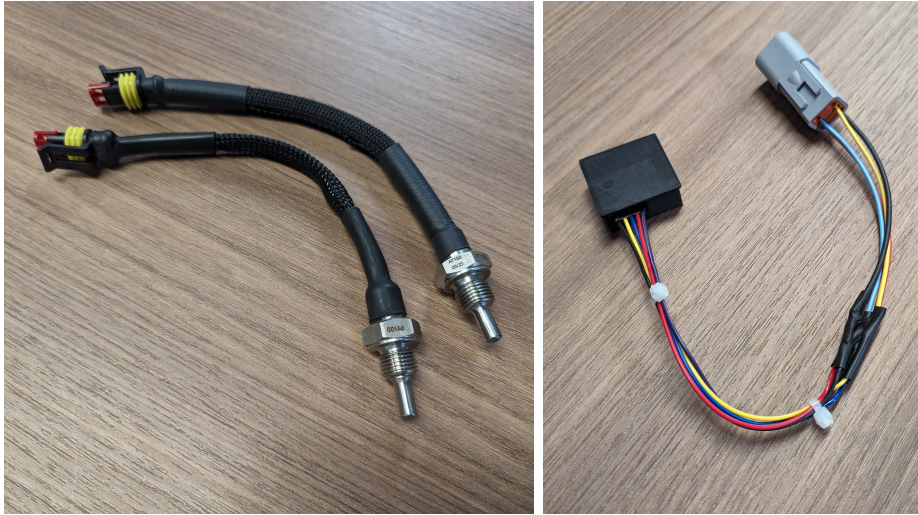


Figura 2.19: Sensore di temperatura Pt100, a sinistra, e modulo accelerometro/giroscopio GY-521, a destra, muniti di cavi e connettori, pronti per l'installazione sulla semovente.



Figura 2.20: Trasmettitori di pressione dell'olio muniti di cavi e connettori, pronti per l'installazione sulla semovente.

2.2 Cablaggi

La sezione corrente descrive i collegamenti elettrici adottati per la creazione del sistema di acquisizione dati. Per una maggiore chiarezza i circuiti elettrici delle diverse schede digitali vengono trattati separatamente.

Alcuni sensori prevedono un collegamento diretto al microcontrollore, mentre altri sensori, tipicamente analogici, necessitano di un condizionamento dei segnali, realizzato attraverso appositi circuiti elettronici.

2.2.1 Hall

Il sistema di misura della velocità angolare delle ruote è stato studiato per garantire una sufficiente accuratezza della misura alla massima velocità del veicolo. Il sensore Hall è stato posizionato davanti alle ruote, su cui è stata imbullonata una ruota dentata in lamiera. Il numero di denti della ruota è stato scelto in modo da garantire, in accordo con il teorema di Shannon, che la frequenza di acquisizione sia almeno pari al doppio della frequenza con cui un dente si presenta davanti al sensore alla massima velocità del veicolo. Il rispetto di questo evita che un dente possa passare davanti al sensore senza che esso lo rilevi, fenomeno che condurrebbe al calcolo di un valore errato della velocità angolare.

I sensori induttivi AI908 forniscono un segnale in uscita uguale alla tensione di alimentazione quando non rilevano un oggetto metallico, mentre producono un'uscita nulla quando individuano un conduttore metallico. Dal momento che i trasduttori sono stati alimentati con 12 V tramite la batteria del veicolo e la tensione massima che l'Arduino può leggere è di 5 V, è stato necessario generare una caduta di tensione all'uscita del sensore per adattarla alle specifiche della scheda. Per ottenere la caduta di tensione desiderata, è sufficiente realizzare un semplice circuito partitore di tensione, come quello rappresentato in Fig. 2.21. Quest'ultimo è un tipo di collegamento elettrico composto da due resistori collegati in serie tra una sorgente di alimentazione. Questo circuito suddivide la tensione totale applicata tra i due resistori in proporzione ai loro valori di impedenza. La tensione ai capi di uno dei resistori, detta tensione di uscita, è una frazione di quella totale e viene

determinata attraverso la seguente equazione:

$$V_{out} = V_s \frac{R_2}{R_1 + R_2} \quad (2.2)$$

Per registrare quindi la caduta di potenziale desiderata è stato necessario utilizzare le resistenze con i valori elencati nella Tab.2.17.

In Fig. 2.22 è rappresentato lo schema di cablaggio del sistema di acquisizione della velocità, costituito dai sensori Hall.

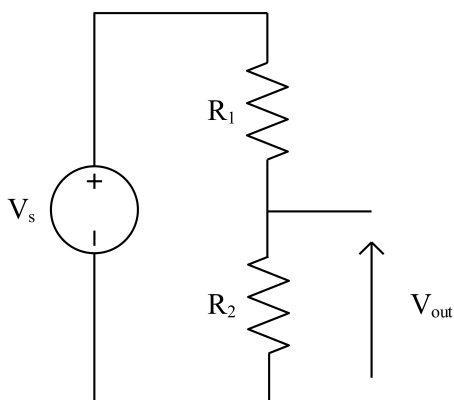


Figura 2.21: Schema di un partitore di tensione.

Sensore AI908	
R1	7 k Ω
R2	5 k Ω
V_s	12 V

Tabella 2.17: Resistenze impiegate nel circuito partitore di tensione per i sensori Hall.

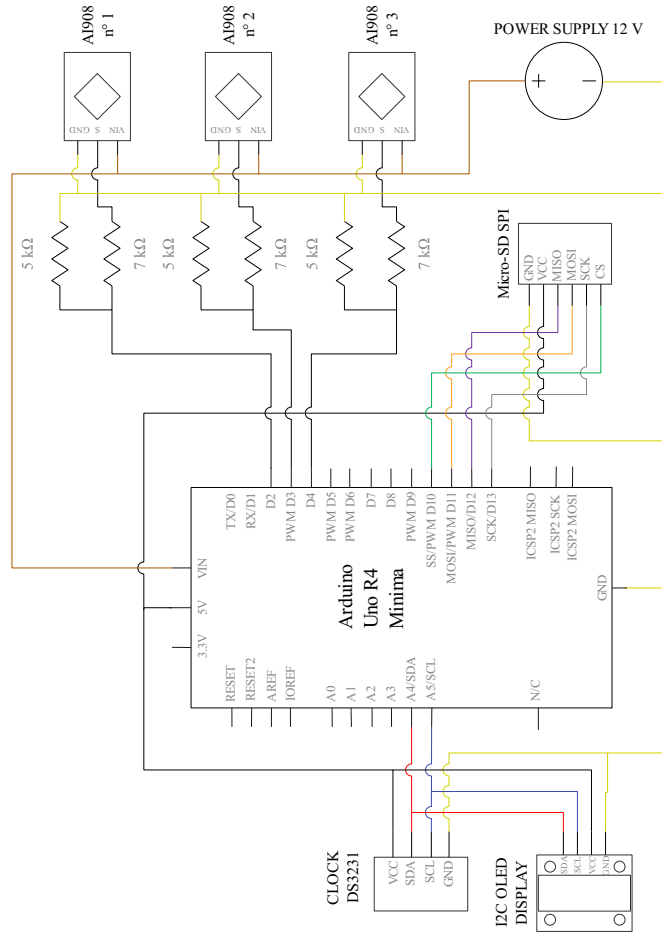


Figura 2.22: Collegamenti elettrici per il sistema di misura della velocità di avanzamento del mezzo.

2.2.2 Pressione, temperatura e accelerazione

Trasmettitori di pressione

I sensori di pressione dell'olio idraulico di Autosen presentano un'uscita analogica di 4 mA quando si trovano al valore più basso del campo di misura e di 20 mA quando sono al valore massimo del fondo scala.

Per trasformare l'uscita dei sensori in una caduta di potenziale, è stata inserita una resistenza in parallelo tra il pin del segnale e la messa a terra, come esposto in Fig. 2.23. Utilizzando la legge di Ohm è possibile calcolare la resistenza ottimale per massimizzare la sensibilità del sistema

di misurazione.

$$R = \frac{V}{I}$$

(2.3)

con V pari alla lettura di tensione massima che può effettuare l'Arduino e con I corrente massima in uscita dal sensore corrispondente al fondo scala.

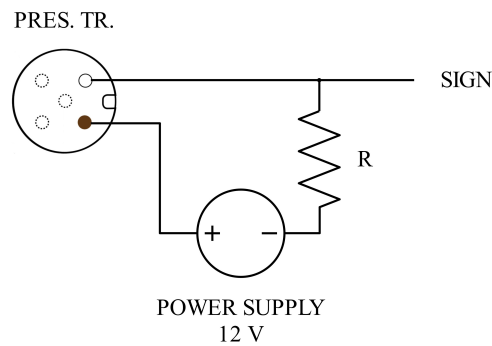


Figura 2.23: Schema di collegamento dei trasmettitori di pressione dell'olio.

	AP031	AP026	AP029	AP030
Campo di misura	-1÷1 bar	0÷60 bar	0÷400 bar	0÷1000 bar
Misurazione max.	1 bar	60 bar	400 bar	~ 400 bar
Uscita max.	20 mA	20 mA	20 mA	~ 10 mA
R	239.4 Ω	239.6 Ω	238.3 Ω	470.8 Ω

Tabella 2.18: Resistenze accoppiate ai sensori di pressione dell'olio idraulico.

Osservando i valori di resistenza scelti (Tab. 2.18), si noti come l'unica resistenza che si discosta dalle altre è quella accoppiata al sensore di pressione da 1000 bar. Questo perché, sebbene il campo di misura giunga a 1000 bar, le limitatrici dell'impianto idraulico sono tarate a 420 bar e quindi l'uscita massima del sensore, a quella pressione, sarebbe pari a circa 10 mA. Allora, per ottenere una caduta di tensione prossima ai 5 V, è necessario adottare una resistenza con valore doppio rispetto alle altre.

Pt100

Una termoresistenza Pt100 è un sensore di temperatura costituito da un resistore di platino (Pt) con un valore nominale di resistenza pari a $100\,\Omega$ a 0°C . La resistenza di una Pt100 varia linearmente con la temperatura, tuttavia, questa variazione può essere molto piccola, dell'ordine dei mV . Per sfruttare interamente questa variazione, la Pt100 è tipicamente collegata come resistenza variabile in un ponte di Wheatstone. Questo tipo di configurazione è particolarmente efficace in quanto il ponte è altamente sensibile alle variazioni di resistenza e tende ad amplificare leggermente anche quelle più piccole. Il ponte di Wheatstone, rappresentato in Fig. 2.24, è un circuito elettrico utilizzato per misurare variazioni di resistenza o resistori non noti.

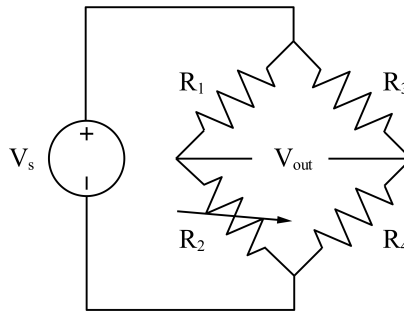


Figura 2.24: Ponte di Wheatstone.

È composto da quattro resistenze disposte in un quadrilatero con un lato collegato a una sorgente di tensione continua e l'altro a un galvanometro o a un altro dispositivo di misurazione.

L'equilibrio del ponte, cioè l'assenza di corrente attraverso il galvanometro, avviene quando il rapporto tra le resistenze è uguale:

$$\frac{R_1}{R_2} = \frac{R_3}{R_4} \quad (2.4)$$

Quando si desidera misurare la resistenza di una Pt100, essa può essere collegata a uno dei rami del ponte di Wheatstone. L'obiettivo quindi è scegliere le altre tre resistenze del ponte in modo da bilanciare la resistenza della Pt100 a una determinata temperatura di riferimento. In questo modo, quando la temperatura varia, anche la resistenza della

Pt100 varia, alterando il bilanciamento del ponte e generando un segnale di uscita proporzionale alla variazione di temperatura. La tensione in uscita dal ponte, pertanto, considerando R_2 pari alla resistenza della Pt100 R_{Pt100} , sarà:

$$V_{out} = V_s \cdot \left(\frac{R_{Pt100}}{R1 + R_{Pt100}} - \frac{R3}{R2 + R3} \right) \quad (2.5)$$

Per avere la massima sensibilità del ponte di Wheatstone le resistenze devono risultare uguali tra loro. Quindi, considerando che la resistenza della Pt100 a $0^\circ C$ è di 100Ω , si è cercato di replicare il ponte utilizzando le resistenze con un valore nominale di 100Ω . Qui di seguito (Tab.2.19) sono riportate le due configurazioni del ponte di Wheatstone ottenute, una per la Pt100 in ingresso al radiatore del fluido idraulico, indicata con $Pt100_{in}$, e una in uscita, indicata con $Pt100_{out}$.

	Pt100 _{in}	Pt100 _{out}
R1	97.68 Ω	97.89 Ω
R2	97.47 Ω	97.60 Ω
R3	97.50 Ω	97.65 Ω
V_{PW}	3.3 V	3.3 V

Tabella 2.19: Tensione d'alimentazione e resistenze adottate nei circuiti a ponte di Wheatstone.

Nonostante il ponte di Wheatstone sia altamente sensibile alle variazioni di resistenza, il segnale elettrico in uscita rimane comunque molto basso, dell'ordine di poche centinaia di mV , il che rende difficile apprezzare correttamente anche le più piccole variazioni di temperatura. Di conseguenza, sorge la necessità di inserire nel circuito un amplificatore operazionale capace di aumentare l'intensità del segnale proveniente dal ponte. Un amplificatore operazionale è un componente elettronico ampiamente utilizzato in vari circuiti elettronici per amplificare segnali e svolgere diverse operazioni matematiche come somma, sottrazione, integrazione e derivazione. È un dispositivo a stato solido con due ingressi e un'uscita, solitamente rappresentati come "+" (non invertente) e "-" (invertente). La caratteristica principale di un amplificatore è che tende ad amplificare la differenza di tensione tra i suoi ingressi, con un guadagno molto elevato e un'impedenza di ingresso molto alta.

L'amplificatore opera in modo che la tensione di uscita sia proporzionale alla differenza di tensione tra i suoi ingressi. Questa relazione è espressa dall'equazione fondamentale dell'amplificatore operazionale:

$$V_{out} = A(V_+ - V_-) \quad (2.6)$$

dove V_{out} è la tensione di uscita, V_+ quella all'ingresso non invertente, V_- quella all'ingresso invertente e A è il guadagno dell'amplificatore. Quest'ultimo è il parametro fondamentale di un amplificatore ed è definito come il rapporto tra la tensione di uscita e la differenza tra le tensioni in ingresso. Per decidere quindi il grado di amplificazione dello strumento, si va ad agire direttamente sul guadagno.

Considerando che l'uscita del ponte di Wheatstone non è altro che una differenza di tensione tra due segnali, per amplificare tale differenza è necessario un amplificatore operazionale in configurazione sottrattore. Nella configurazione sottrattore non invertente, due resistenze R_H e R_L vengono collegate tra l'ingresso non invertente V_+ e l'uscita V_{out} . La stessa coppia di resistenze inoltre collega l'ingresso invertente V_- e la massa.

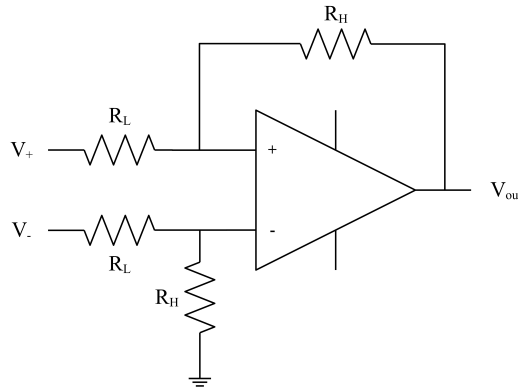


Figura 2.25: Circuito amplificatore operazionale non invertente sottrattore.

La funzione di trasferimento del sistema è determinata dalla seguente legge:

$$V_{out} = (V_+ - V_-) \frac{R_H}{R_L} \quad (2.7)$$

Si riportano ora, in Tab.2.20 i valori di R_H e R_L scelti per entrambi i circuiti di amplificazione delle due Pt100, opportunamente selezionati al fine di ottenere, a sospetta temperatura massima raggiungibile dal fluido idraulico, un segnale in uscita dall'amplificatore prossimo a 5 V.

	Pt100 _{in}	Pt100 _{out}
R_H	20.02 k Ω	20.09 k Ω
R_L	1.17 k Ω	1.18 k Ω
A	17.11	17.03
V_{amp}	5 V	5 V

Tabella 2.20: *Resistenze scelte per l'amplificazione dei segnali in uscita del ponte di Wheatstone.*

Sensori di pressione differenziale

Il sensore di pressione dell'aria differenziale MPX10DP presenta a 0 kPa a 25 °C un'uscita di circa 20 mV e a 10 kPa un'uscita di circa 55 mV, con una sensibilità di 3.5 mV/kPa.

Quindi capiamo che, anche in questa situazione, per rilevare le variazioni più piccole di pressione, è essenziale amplificare il segnale in uscita. Questo è stato realizzato attraverso una configurazione a doppio amplificatore (Fig. 2.26). Nel primo, configurato come sottrattore, viene amplificata la differenza dei segnali in uscita dal sensore, mentre nel secondo, in configurazione semplice non invertente, viene amplificato il segnale in uscita dal primo amplificatore.

Dopo aver effettuato diverse misurazioni con un anemometro all'interno del condotto del ventilatore di mandata, è stato osservato che la massima pressione registrabile in condizioni operative è di circa 3800 Pa. A questo valore di pressione, l'uscita del sensore si attesta intorno ai 35 mV. L'obiettivo è ottenere che l'uscita massima, venga amplificata fino a circa 5 V per massimizzare la sensibilità del sistema e minimizzare l'errore associato alla lettura dell'Arduino.

In questo modo si ottiene che il segnale in uscita del sensore necessita di un'amplificazione con un guadagno complessivo di 130. Pertanto si è scelto che la prima amplificazione deve avere un guadagno di 10, mentre la seconda di 13.

La funzione di trasferimento del primo circuito sottrattore è data

dall'Eq. 2.7, mentre quella relativa alla configurazione non invertente è espressa dalla seguente formula:

$$V_{out} = V_{in} \left(1 + \frac{R_{H2}}{R_{L2}} \right) \quad (2.8)$$

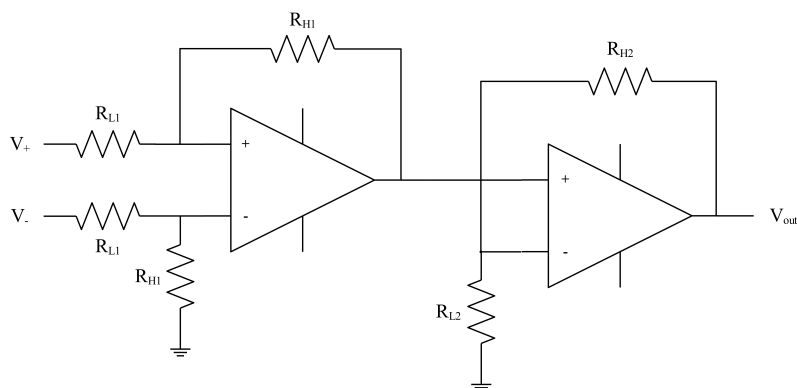


Figura 2.26: Configurazione a doppio amplificatore operazionale.

Qui di seguito (Tab.2.21) è presentata una tabella che elenca i valori delle resistenze selezionate per entrambe le configurazioni degli amplificatori. Per il sensore di pressione differenziale dell'aria posizionato

Sensore MPX10DP	
R_{L1}	10 k Ω
R_{H1}	100 k Ω
R_{L2}	10 k Ω
R_{H2}	120 k Ω
V_s	3.3 V
$V_{amp, 1}$	5 V
$V_{amp, 2}$	5 V

Tabella 2.21: Tensioni di alimentazione e resistenze adottate per il circuito di amplificazione doppia.

nel condotto all'uscita dal ventilatore di aspirazione, è sufficiente che sia collegato alla scheda di acquisizione e alimentato a 5 V, senza la

necessità di circuiti elettrici aggiuntivi, in quanto il segnale in uscita non necessita di amplificazioni. Infatti, dopo aver condotto diverse misurazioni con un anemometro, è stato confermato che il valore massimo rilevabile in tale condotta è di circa $1000 Pa$ che comporta un'uscita del sensore pari a $3.5 V$.

IMU

Gli accelerometri GY-521 utilizzati in questo progetto sono caratterizzati da una comunicazione di tipo I2C e hanno un indirizzo di trasmissione predefinito. Il dispositivo presenta una porta in ingresso, denominata AD0, che, a seconda del collegamento, consente di selezionare l'indirizzo tra due valori specifici: 0x68 e 0x69. Quando la porta indicata viene collegata a terra, l'indirizzo assume il valore 0x68; al contrario, se è collegata all'alimentazione, l'indirizzo diventa 0x69. Dato che il RTC DS3231 integrato nella scheda di acquisizione dati utilizza l'indirizzo 0x68, per evitare conflitti con i due accelerometri, la porta AD0 di questi ultimi è stata collegata a quella di alimentazione. La comunicazione I2C però, gestita dalle linee SDA e SCL, non consente di analizzare contemporaneamente due sensori che hanno in comune lo stesso indirizzo. Per risolvere questo problema, è stato implementato nel circuito un multiplexer, il quale divide le linee di comunicazione I2C e consente di interrogare uno alla volta entrambi gli accelerometri.

Schema dei cablaggi

Tutti i sensori descritti in questa sezione: trasmettitori di pressione, sensori di temperatura Pt100, sensori di pressione differenziale e accelerometri sono stati connessi ad una singola scheda Arduino Mega 2560. È presente anche un modulo Micro-SD SPI per il salvataggio dei dati, un Clock DS3231 per la registrazione della data e dell'ora e un Display OLED per la visualizzazione dei dati campionati. Come si può osservare dalla Fig. 2.27, la grande quantità di sensori e dispositivi interconnessi, insieme alla necessità di trattamento dei segnali, rendono il circuito in esame più complesso ed elaborato degli altri.

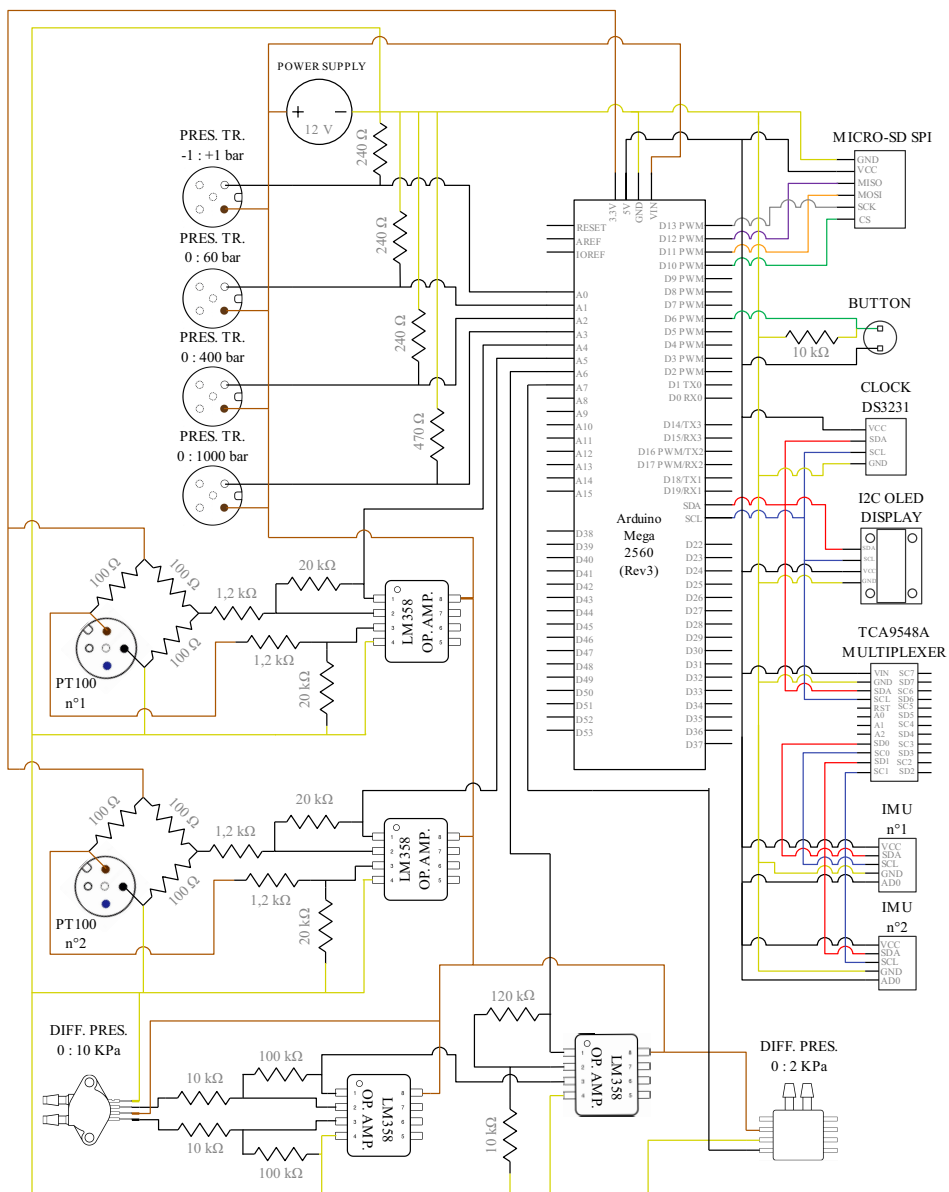


Figura 2.27: Collegamenti elettrici per il sistema di misura della pressione, della temperatura dell'olio, della portata d'aria e delle accelerazioni.

2.2.3 CAN-BUS

Il sistema di *data logging* per i segnali CAN-BUS è composto da un microcontrollore SparkFun RedBoard e una CAN-BUS *shield*, collegate attraverso dei pin saldati sulla seconda. La *shield* è direttamente connessa ai cavi CAN-H e CAN-L sulla presa diagnostica del veicolo. È presente anche il Clock DS3231 per la registrazione del tempo. Lo schema dei cablaggi, rappresentato in Fig. 2.28 appare quindi estremamente semplice rispetto agli altri.

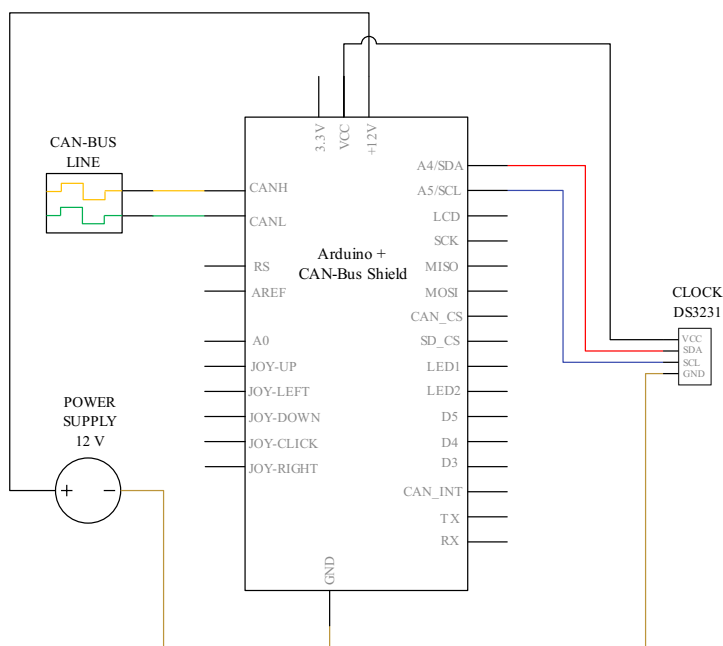


Figura 2.28: Collegamenti elettrici per il sistema di lettura e raccolta dei dati motore sulla linea CAN-BUS.

2.2.4 GPS

Il sistema di acquisizione della posizione GNSS risulta anch'esso molto basilare, essendo composto solamente da una scheda SparkFun RedBoard, un modulo Sparkfun M9N-NEO, un'antenna GNSS magnetica e un modulo Micro-SD SPI per il salvataggio dei dati. Le connessioni tra i componenti sopra citati sono delineate in Fig. 2.29.

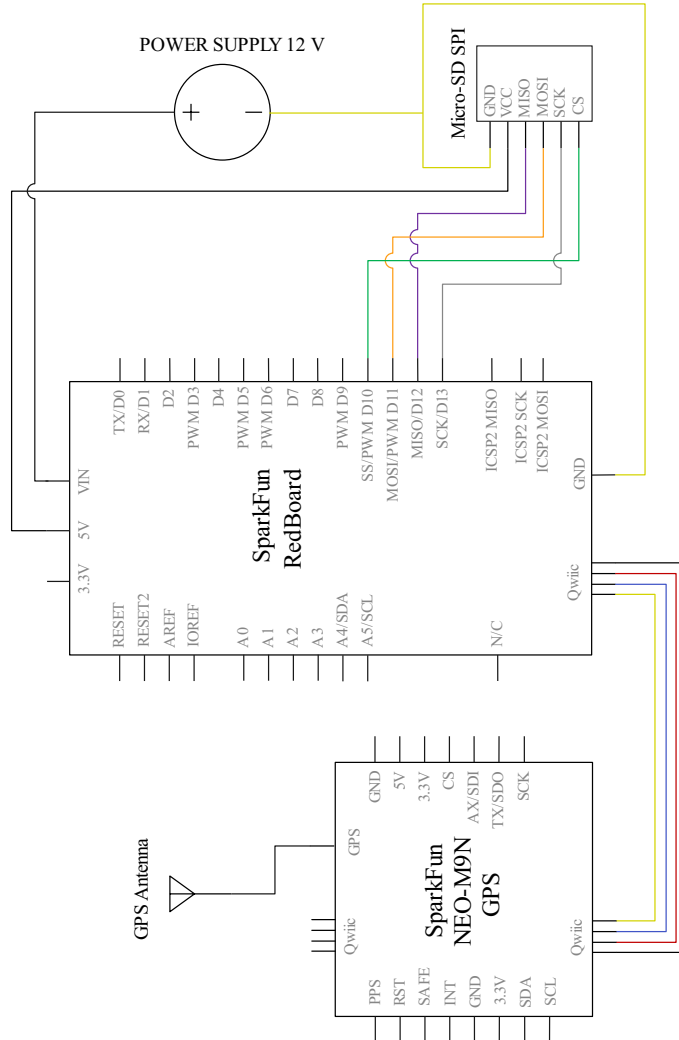


Figura 2.29: Collegamenti elettrici per il sistema di misura della posizione GNSS.

2.3 Programmazione

In questa sezione si tratta della programmazione dei microcontrollori utilizzati per acquisire, elaborare e salvare i dati provenienti dai sensori. Si riporterà sia, in linea più generale, il *workflow* logico implementato nei programmi, sia, in linea più dettagliata, il vero e proprio *sketch* di programmazione.

Il linguaggio adottato per la programmazione delle schede è quello di Arduino IDE, basato su C++ e tutti i programmi sono articolati in tre parti principali [1] [15]:

1. una parte dichiarativa in cui si importano tutte le librerie e si dichiarano tutte le variabili utilizzate all'interno del programma;
2. una seconda parte di *setup*, contenuta all'interno del `void setup`, in cui si inizializzano e si preparano tutti i dispositivi per l'acquisizione. Questa parte viene eseguita una volta all'accensione dei microcontrollori;
3. una parte di *loop* contenuta all'interno del `void loop`, in cui vengono letti, elaborati e salvati i segnali dei sensori. Questa parte viene eseguita iterativamente dal programma fino allo spegnimento della scheda.

In Fig. 2.30 è rappresentato il diagramma di flusso che accomuna i vari algoritmi di programmazione adottati per le 4 schede digitali. In linea di massima per tutte le schede di acquisizione si è scelto di adottare un simile algoritmo di programmazione, che si articola nelle seguenti fasi.

1. Inclusione delle librerie

Durante questa fase si importano le librerie propedeutiche alla programmazione dei componenti elettronici, sfruttando la sintassi `#include`. Queste aggiungono funzionalità in ambiente di programmazione ampliando i comandi disponibili per la stessa [15] e semplificando di molto la sintassi del codice. Le librerie utilizzate sono: la `Wire.h` per la comunicazione i2c; la `RTClib.h` per il real time clock; la `SPI.h` e la `SD.h` per la lettura delle schede SD; la `Adafruit_GFX.h` e la `Adafruit_SSD1306.h` per il display OLED; la `mcp_can.h` per comunicare con il CAN controller; e infine quella propedeutica alla comunicazione con il modulo GNSS di Sparkfun.

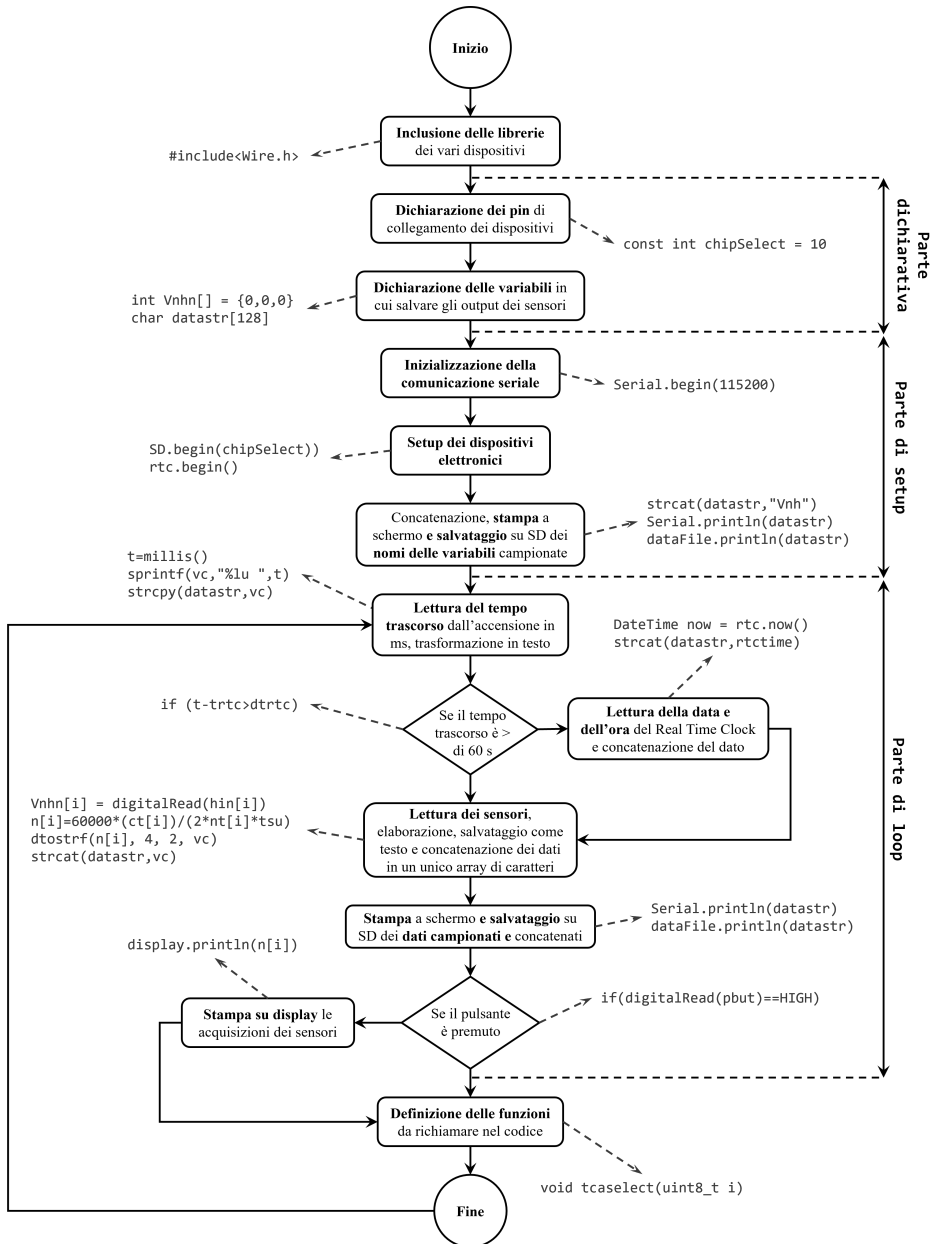


Figura 2.30: Diagramma di flusso dell'algoritmo di programmazione, adottato in ambiente Arduino IDE, delle schede elettroniche, con alcuni esempi di stringhe di codice.

2. Dichiarazione dei pin di collegamento

Si dichiarano poi i pin scelti per il collegamento dei dispositivi elettronici, che possono essere i vari sensori analogici, digitali o i moduli aggiuntivi come il lettore SD, il *clock*, il display e dispositivi più semplici come pulsanti e led. La sintassi per la dichiarazione prevede di specificare il tipo di variabile dichiarata, che, in questo caso, è **const unsigned int**, cioè costante, intera e positiva, seguita dal nominativo che si vuole assegnare uguagliato al numero del pin di collegamento di Arduino.

3. Dichiarazione delle variabili

Successivamente avviene la dichiarazione della variabili, che sono spazi allocati in memoria, contrassegnati da un preciso nominativo, da riservare per salvarvi valori numerici di diverso tipo o di testo [15].

In ogni caso si crea una variabile che indica il tempo trascorso in millisecondi, si creano delle variabili di testo in cui inserire la data e l'ora registrate dal *clock*, e la variabile di testo **datastr** che, ad ogni iterazione del void loop, verrà riempita di testo gradualmente, rappresentando l'intera riga di dati campionati, salvata su scheda SD e stampata a schermo. Troviamo in alcuni casi la dichiarazione delle variabili propedeutiche al funzionamento del display OLED, tra cui la durata dell'accensione e il pin a cui è collegato il pulsante a 2 vie che ne scatuisce l'accensione. Il resto delle istruzioni dichiarative riguarda sostanzialmente le variabili necessarie per salvare l'uscita dei sensori e le costanti utili per calcolare le grandezze misurate.

La sintassi per la dichiarazione prevede di specificare prima del nominativo della variabile il tipo della stessa: nel caso in esame sono stati utilizzati principalmente i tipi **char**, **int**, **float** che denotano variabili di testo (vettori di caratteri), di numeri interi e decimali con virgola. Alcune variabili sono di tipo **unsigned**, ovvero prive di segno, altre sono state dichiarate costanti **const** e ad alcune è stata allocata una memoria superiore (**long**). Sono stati utilizzati anche *array* di variabili, ovvero insiemi di variabili accessibili attraverso un indice. La stringa per la dichiarazione, in questo caso, prevede, accanto al nominativo, le parentesi quadre [], all'interno delle quali si inserisce il numero di elementi del vettore.

4. Inizializzazione della comunicazione seriale

Passando alle istruzioni che ricadono nel `void setup()`, anzitutto viene inizializzata la comunicazione con la porta seriale scegliendo la velocità di trasmissione dei dati in *bit/s* o *baud*, attraverso l'istruzione `Serial.begin()`.

5. Setup dei dispositivi elettronici

Poi si accendono, si inizializzano e si verificano i componenti elettronici quali schede SD, *clock* e display. La scheda SD viene inizializzata e viene stampato a schermo un messaggio che ne denota il corretto funzionamento. La parte di *setup* del *clock* contiene delle righe di codice che permettono di sincronizzare la data e l'ora dello stesso al terminale cui si connette Arduino. Il *setup* del display consiste nell'accensione dello stesso, nell'impostazione della dimensione del testo e del colore. Anche gli accelerometri possiedono una fase di *setup*, in cui vengono inviati messaggi in codice tramite la linea i2c che permettono di impostare il campo di misura sia dell'accelerometro che del giroscopio e le frequenze di taglio dei filtri passa basso integrati nei moduli. Anche l'*offset* o errore iniziale della IMU viene calcolato. Ove si utilizza il modulo mcp2515 per la lettura dei dati CAN-BUS è necessario un setup per inizializzare la scheda suddetta, scegliendo la frequenza e la velocità di comunicazione. Inoltre si impostano una serie di filtri che permettono di ricevere esclusivamente i dati provenienti dalle periferiche con gli ID specificati.

6. Stampa e salvataggio dei nomi delle variabili campionate

Per identificare correttamente e più facilmente le variabili numeriche salvate da Arduino nel file di testo occorre, all'accensione stampare l'elenco dei nomi delle colonne di variabili. Si utilizza, quindi, un nominativo di circa 2-4 lettere, seguito da un indice (se si hanno più sensori dello stesso tipo) e tra quadre l'unità di misura per identificare tutte le grandezze misurate e/o calcolate. Quindi in questa fase si concatenano i nomi delle colonne di dati che si intendono salvare all'interno dell'*array* di caratteri `datastr`, attraverso il comando `strcat()`: tra le variabili troviamo sempre il tempo di campionamento, la data, l'ora, i campionamenti dei vari sensori e eventualmente una variabile, chiamata `SDStatus`, che indica se il salvataggio su scheda SD sta avvenendo con successo. I

caratteri contenuti in `datastr` poi devono essere stampati a schermo e salvati nel file di testo all'interno della scheda SD avvalendosi rispettivamente delle istruzioni `Serial.println(datastr)` e `dataFile.println(datastr)`. La desinenza `\n` nei comandi suddetti manda il testo a capo.

7. Lettura del tempo trascorso

Il primo passo del *void loop* consiste nella stampa a schermo e nel salvataggio del tempo trascorso dall'accensione di Arduino, che si può determinare attraverso la funzione `millis()`.

La procedura di salvataggio di una variabile, che sia un'acquisizione di un sensore o altro, nel *void loop* si articola in due istruzioni: la formattazione e la trascrizione nella variabile di testo `vc` e la scrittura di quest'ultima nella stringa `datastr`. Utilizzando il comando `strcpy()` la stringa `datastr` viene sovrascritta e quindi pulita dal vecchio contenuto all'inizio del loop.

8. Lettura della data e dell'ora

Oltre al tempo in millisecondi viene salvata la data e l'ora corrente interrogando in primis il *real time clock* e poi, sempre con il procedimento sopra esposto, concatenando i dati nella stringa `datastr`. Questo processo non viene eseguito ad ogni iterazione del loop per evitare rallentamenti e velocizzare la frequenza di campionamento, bensì ogni minuto circa. Per far ciò le istruzioni di lettura sono state inserite in un ciclo `if` con condizione `(t - trtc > dtrtc)`.

9. Lettura dei sensori

Questa fase differisce a seconda del tipo di sensore che dev'essere letto. Per quanto riguarda i sensori collegati ai pin analogici e digitali, è necessario anzitutto effettuare una lettura del segnale avvalendosi dei comandi `analogRead` e `digitalRead`, associandoli a delle variabili. Il segnale campionato poi, non costituendo alcuna misura effettiva delle grandezze fisiche, va elaborato sulla base delle curve di taratura dei sensori per calcolare la misura nell'unità desiderata.

I sensori e i moduli che invece funzionano con la comunicazione `i2c` come gli accelerometri vengono interrogati mediante specifici comandi attraverso le stringhe `Wire.write()` e `Wire.read()`.

L'intercettazione e la lettura dei messaggi CAN invece viene eseguita sfruttando l'istruzione `CAN0.readMsgBuf()` offerta dalla libreria `mcp2551`.

Allo stesso modo il modulo GNSS offre la possibilità di ricavare la posizione e la velocità sfruttando i comandi `myGNSS.getLatitude`, `myGNSS.getLongitude` e `myGNSS.getSpeed` abilitati da una libreria specifica.

Ove ci sono più sensori della stessa tipologia, che richiedono una simile elaborazione, vengono sfruttati i cicli `for` per ripetere le istruzioni per ogni sensore e per salvare tutti i dati acquisiti dai sensori della stessa classe all'interno di un vettore. I cicli iterativi permettono di semplificare la sintassi del codice, rendendolo più compatto.

A prescindere da quale sia il metodo di lettura dei sensori, il procedimento per poi salvare questi dati rimane il medesimo. Ogni variabile numerica va a sovrascrivere il contenuto di `vc` tramite i comandi `sprintf()` o `dtostrf()`. La variabile `vc` a sua volta viene concatenata, insieme ad uno spazio, dentro la stringa `datastr` tramite l'istruzione `strcat()`.

10. Stampa e salvataggio dei dati

I dati contenuti nella stringa `datastr`, sotto forma di testo, possono essere ora stampati a schermo grazie alla funzione `Serial.print()` e salvati all'interno del file di testo nella scheda SD tramite il comando `dataFile.print()`, previa apertura del file e prima della chiusura dello stesso.

Prima del salvataggio viene segnalato il corretto o mancato funzionamento della scheda SD: qualora il collegamento con la periferica non funzionasse correttamente viene visualizzata così, da terminale, la dicitura `OFF` sotto la colonna chiamata `SDStatus`.

11. Visualizzazione dei dati su Display

Il corretto funzionamento dei sensori e della periferica di salvataggio può essere controllato attraverso un display o un semplice led di segnalazione. I dati su schermo possono essere visualizzati su richiesta dell'utente attraverso la pressione di un pulsante. La condizione che deve essere verificata sul programma per l'accensione dello schermo è la seguente: `if(digitalRead(pbut)==HIGH)`. Se questa condizione è verificata,

la stampa dei valori che si vogliono visualizzare a schermo avviene tramite il comando `display.print()`. Ove vi sono più sensori della stessa tipologia, le cui acquisizioni vengono salvate in vettori, si impiegano i cicli `for` per far scorrere l'indice del vettore e stampare a schermo i valori in sequenza. Ai valori campionati viene affiancata la variabile `SDStatus`, fondamentale per capire se il salvataggio dei dati su SD sta avvenendo con successo. Il comando `display.display` permette poi di aggiornare lo schermo e, quindi, di visualizzare tutti i valori stampati. Viene inserito un `delay` che permette di mantenere lo schermo acceso per la durata di 7 secondi; successivamente viene pulito lo schermo tramite l'istruzione `display.clearDisplay()` e aggiornato di nuovo.

Nel caso della scheda utilizzata per determinare la posizione non viene utilizzato un display per accertare il funzionamento del sistema, bensì un diodo LED. Se il salvataggio dei dati avviene correttamente su SD, quindi se è rispettata la condizione `if(datafile)`, allora si inseriscono le linee di codice `digitalWrite(ledpin, HIGH)` e `digitalWrite(ledpin, LOW)`, rispettivamente per accendere e spegnere il LED avvalendosi dei pin digitali di Arduino in grado di restituire all'uscita una tensione.

12. Definizione delle funzioni richiamate nel codice principale

Solo per la scheda connessa agli accelerometri si è usufruito delle funzioni, per snellire la parte principale del codice. Le funzioni permettono di eseguire, quando vengono richiamate nel codice, una serie di istruzioni semplici. Si è creata una funzione per il calcolo degli errori o *offset* iniziali degli accelerometri di tipo `void`, quindi senza nessuna variabile di output, chiamata `calculate_IMU_error`. La funzione effettua una serie di letture con gli accelerometri e con i giroscopi calcolandone il valor medio `AccError` e `GyroError`, che viene utilizzato poi come offset per correggere le misure degli angoli e delle velocità angolari del giroscopio.

La seconda funzione creata, chiamata `tcselect`, ha la funzione di inizializzare la comunicazione con il modulo multiplexer, necessario per interrogare entrambi gli accelerometri, e di selezionare il giusto canale per la comunicazione. Questa seconda funzione ovviamente, al contrario della prima citata, ha come variabile di input un indice

i che specifica il canale di trasmissione da aprire. Anche questa, essendo di tipo void, non restituisce alcun valore in uscita.

2.3.1 Hall

Si approfondisce di seguito il codice di programmazione della scheda Arduino R4 Minima, utilizzata per campionare i dati provenienti dai sensori di prossimità Hall.

```

1 // --- Inclusione delle librerie ---
2 #include <Wire.h>
3 #include <RTClib.h>
4 #include <SPI.h>
5 #include <SD.h>
6 #include <Adafruit_GFX.h>
7 #include <Adafruit_SSD1306.h>
8 // -----
9
10 // --- Dichiarazione delle variabili per la Micro-SD SPI ---
11 const int chipSelect = 10;
12 char SDstatus[8] = "";
13 // -----
14
15 // --- Dichiarazione delle variabili per il Real Time Clock ---
16 RTC_DS3231 rtc;
17 char rtcdate[16];
18 char rtctime[16];
19 const unsigned int dtrtc = 60000;
20 long unsigned int trtc = 0;
21 // -----
22
23 // --- Dichiarazione della variabile Elapsed Time ---
24 long unsigned int t = 0;
25 // -----
26
27 // --- Dichiarazione delle variabili di testo ---
28 char datastr[128];
29 char ic[1];
30 char vc[8];
31 // -----
32
33 // --- Dichiarazione delle variabili per i sensori Hall ---
34 int hin[] = {2,3,4};
35 int Vnhn[] = {0,0,0};
36 int Vnho[] = {0,0,0};
37 const int nt[] = {1,2,1};
38 float n[] = {0,0,0};
39 const int tsu = 3000;
40 unsigned int ct[] = {0,0,0};
41 unsigned int cti[] = {0,0,0};
42 unsigned int ta=0;
43 long unsigned int dt[] = {0,0,0};
44 long unsigned int th[] = {0,0,0};
45 long unsigned int tl[] = {0,0,0};
46 //-----
47
48 // --- Dichiarazione delle variabili per il Display OLED ---
49 #define SCREEN_WIDTH 128
50 #define SCREEN_HEIGHT 64
51 #define OLED_RESET 6
52 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
53 const int tdisplay = 7000;
54 const int pbut = 5;
55 // -----
56
57 void setup() {
58
59     // --- Inizializzazione della comunicazione seriale ---

```

```

60     Serial.begin(115200);
61     // -----
62
63     // --- Setup della Micro-SD SPI ---
64     if (!SD.begin(chipSelect)) {
65         Serial.println("Card failed or not present");
66     }
67     else {
68         Serial.println("Card initialized.");
69     }
70     // -----
71
72     // --- Setup del Real Time Clock ---
73     rtc.begin(); //Inizializzazione del real time clock
74     //rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
75     //rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
76     // -----
77
78     // --- Setup del Display OLED ---
79     if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
80         Serial.println(F("SSD1306 allocation failed"));
81     }
82     display.clearDisplay();
83     display.setTextSize(1);
84     display.setTextColor(WHITE, BLACK);
85     display.display();
86     // -----
87
88     // --- Stampa e salvataggio dei nomi delle colonne di dati campionati ---
89     strcpy(datastr, "t[ms] Date[YYYY/MM/DD] Time[HH:mm:ss] ");
90     for (int i=0; i<sizeof(hin)/sizeof(hin[0]); i++) {
91         sprintf(ic, "%u ", i);
92         strcat(datastr, "Vnh");
93         strcat(datastr, ic);
94         strcat(datastr, "[-]");
95     }
96     strcat(datastr, "SD status");
97     Serial.println(datastr);
98     File dataFile = SD.open("hall.txt", FILE_WRITE);
99     if (dataFile) {
100         dataFile.println(datastr);
101         dataFile.close();
102     }
103     // -----
104 }
105
106 void loop() {
107
108     // --- Lettura del tempo trascorso ---
109     t=millis();
110     sprintf(vc, "%lu ", t);
111     strcpy(datastr, vc);
112     // // -----
113
114     // --- Lettura della data e dell'ora ---
115     if (t-trtc>dtrtc) {
116         DateTime now = rtc.now();
117         sprintf(rtcdate, "%04d/%02d/%02d", now.year(), now.month(), now.day());
118         sprintf(rtctime, "%02d:%02d:%02d", now.hour(), now.minute(), now.second());
119         strcat(datastr, rtcdate);
120         strcat(datastr, " ");
121         strcat(datastr, rtctime);
122         trtc=t;
123     }
124     else {
125         strcat(datastr, "nan nan");
126     }
127     // -----
128
129     // ---- Lettura dei dati dei sensori ----
130     for (int i=0; i<sizeof(hin)/sizeof(hin[0]); i++) {
131         Vnhn[i] = digitalRead(hin[i]);
132         if (Vnhn[i]==1 & Vnho[i]==0) {

```



```

133             ct[i]=ct[i]+1;
134             th[i]=t;
135         }
136         if (Vnhn[i]==0 & Vnho[i]==1) {
137             ct[i]=ct[i]+1;
138             tl[i]=t;
139         }
140         Vnho[i]=Vnhn[i];
141     }
142     // -----
143
144     // --- Calcolo delle velocita di rotazione ---
145     if (millis()-ta>tsu){
146         for (int i=0; i<sizeof(hin)/sizeof(hin[0]); i++) {
147             n[i]=60000*(ct[i])/(2*nt[i]*tsu);
148             ct[i]=0;
149         }
150         ta=millis();
151     }
152     for (int i=0; i<sizeof(hin)/sizeof(hin[0]); i++) {
153         sprintf(vc,"%u ",Vnhn[i]);
154         strcat(datastr, " ");
155         strcat(datastr,vc);
156         dtostrf(n[i], 4, 2, vc);
157         strcat(datastr, " ");
158         strcat(datastr,vc);
159     }
160     // -----
161
162     // --- Stampa a schermo e salvataggio dei dati ---
163     File dataFile = SD.open("hall.txt", FILE_WRITE);
164     if (dataFile) {
165         strcpy(SDstatus, "ON");
166         strcat(datastr, " ON");
167         dataFile.println(datastr);
168         dataFile.close();
169     }
170     else {
171         strcpy(SDstatus, "OFF");
172         strcat(datastr, " OFF");
173     }
174     Serial.println(datastr);
175     // // -----
176
177     // --- Stampa sul Display OLED ---
178     if(digitalRead(pbut)==HIGH){
179         display.setCursor(0, 0);
180         for (int i=0; i<sizeof(hin)/sizeof(hin[0]); i++) {
181             display.print("Vn");
182             display.print(i);
183             display.print(" ");
184             display.print(Vnhn[i]);
185             display.print(" ");
186             display.print("n");
187             display.print(i);
188             display.print(" ");
189             display.println(n[i]);
190         }
191         display.print("SD Status ");
192         display.print(SDstatus);
193         display.display();
194         delay(tdisplay);
195         display.clearDisplay();
196         display.display();
197     }
198     // -----
199 }

```

2.3.2 Pressione, temperatura e accelerazione

Si riporta di seguito il codice di programmazione adottato per l'Arduino Mega 2560 Rev 3, connesso ai sensori di pressione dell'olio, di temperatura dell'olio, di pressione dell'aria e ai moduli IMU. Senza dubbio, vista la grande quantità di periferiche collegate, questo rappresenta il codice più prolisso.

```

1  // --- Inclusione delle librerie ---
2  #include <Wire.h>
3  #include <RTCLib.h>
4  #include <SPI.h>
5  #include <SD.h>
6  #include <Adafruit_GFX.h>
7  #include <Adafruit_SSD1306.h>
8  // -----
9
10 // --- Dichiarazione dell'indirizzo del Multiplexer ---
11 #define TCAADDR 0x70
12 // -----
13
14 // --- Dichiarazione delle variabili per la Micro-SD SPI ---
15 const int chipSelect = 53;
16 char SDbuffer[8] = "";
17 // -----
18
19 // --- Dichiarazione delle variabili per il Real Time Clock ---
20 RTC_DS3231 rtc;
21 char rtcdate[16];
22 char rtctime[16];
23 const unsigned int dtrtc = 60000;
24 long unsigned int trtc = 0;
25 // -----
26
27 // --- Dichiarazione della variabile Elapsed Time ---
28 long unsigned int t = 0;
29 // -----
30
31 // --- Dichiarazione delle variabili di testo ---
32 char datastr[512];
33 char ic[1];
34 char vc[8];
35 // -----
36
37 // --- Dichiarazione delle variabili per i trasmettitori di pressione ---
38 const int poin[] = {A0,A1,A2,A3};
39 const int pomin[] = {-1,0,0,0};
40 const int pomax[] = {1,60,400,1000};
41 float po[] = {0,0,0,0};
42 float Vpo[] = {0,0,0,0};
43 float Rpo[] = {239.4,239.6,238.3,470.8};
44 unsigned int Vpomin[] = {0,0,0,0};
45 unsigned int Vpomax[] = {0,0,0,0};
46 // -----
47
48 // --- Dichiarazione delle variabili per i sensori di temperatura Pt100 ---
49 const int Tin[] = {A4,A5};
50 float T[] = {0,0};
51 float VT[] = {0,0};
52 float Voutw[] = {0,0};
53 float RPt100[] = {0,0};
54 float GT[] = {17.254,17.154};
55 float AT[] = {9.094,8.992};
56 float BT[] = {95.856,96.461};
57 // -----
58
59 // --- Dichiarazione delle variabili per i sensori di pressione differenziale ---
60 const int pain[] = {A6,A7};
61 const int ptype[] = {2,10};

```

```

62 float Vpa[] = {0,0};
63 float pa[] = {0,0};
64 float Ap[] = {1.05,2.853};
65 float Bp[] = {-2807,-6395};
66 // -----
67
68 // --- Dichiarazione delle variabili per gli accelerometri ---
69 const int MPU = 0x69;
70 const int macin[] = {0,1};
71 float AccX[] = {0,0};
72 float AccY[] = {0,0};
73 float AccZ[] = {0,0};
74 float GyroX[] = {0,0};
75 float GyroY[] = {0,0};
76 float GyroZ[] = {0,0};
77 float AccAngX[] = {0,0};
78 float AccAngY[] = {0,0};
79 float GyroAngX[] = {0,0};
80 float GyroAngY[] = {0,0};
81 float elapsedTime[] = {0,0};
82 float currentTime[] = {0,0};
83 float previousTime[] = {0,0};
84 float AccErrorX[] = {0,0};
85 float AccErrorY[] = {0,0};
86 float GyroErrorX[] = {0,0};
87 float GyroErrorY[] = {0,0};
88 float GyroErrorZ[] = {0,0};
89 int conterr=0;
90
91 // -----
92
93 // --- Dichiarazione delle variabili per la tensione di alimentazione ---
94 const unsigned int V3inpin = A9;
95 const unsigned int V5inpin = A8;
96 float V3=0;
97 float V5=0;
98 // -----
99
100 // --- Dichiarazione delle variabili per il Display OLED ---
101 #define SCREEN_WIDTH 128
102 #define SCREEN_HEIGHT 64
103 #define OLED_RESET 4
104 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
105 const int tdisplay = 7000;
106 const int pbut = 26;
107 // -----
108
109 void setup() {
110
111     // --- Inizializzazione della comunicazione seriale ---
112     Serial.begin(115200);
113     // -----
114
115     // --- Setup della Micro-SD SPI ---
116     if (!SD.begin(chipSelect)) {
117         Serial.println("Card failed or not present");
118     }
119     else {
120         Serial.println("Card initialized.");
121     }
122     // -----
123
124     // --- Setup del Real Time Clock ---
125     rtc.begin(); //Inizializzazione del real time clock
126     //rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
127     //rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
128     // -----
129
130     // --- Setup degli accelerometri ---
131     Wire.begin();
132     for (int i=0; i<sizeof(macin)/sizeof(macin[0]); i++) {
133         twselect(macin[i]);
134         Wire.beginTransmission(MPU);

```

```

135         Wire.write(0x6B);
136         Wire.write(0x00);
137         Wire.endTransmission(true);
138         Wire.beginTransmission(MPU);
139         Wire.write(0x6B);
140         Wire.write(0x03);
141         Wire.endTransmission(true);
142         Wire.beginTransmission(MPU);
143         Wire.write(0x1B);
144         Wire.write(0x08);
145         Wire.endTransmission(true);
146         Wire.beginTransmission(MPU);
147         Wire.write(0x1C);
148         Wire.write(0x00);
149         Wire.endTransmission(true);
150         Wire.beginTransmission(MPU);
151         Wire.write(0x1A);
152         Wire.write(0x06);
153         Wire.endTransmission(true);
154         delay(20);
155     }
156     calculate_IMU_error();
157     delay(20);
158     // -----
159
160     // --- Setup del Display OLED ---
161     if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
162         Serial.println(F("SSD1306 allocation failed"));
163     }
164     display.clearDisplay();
165     display.setTextSize(1);
166     display.setTextColor(WHITE, BLACK);
167     display.display();
168     // -----
169
170     // --- Salvataggio dei nomi delle colonne di dati campionati ---
171     strcpy(datastr, "t[ms] Date[YYYY/MM/DD] Time[HH:mm:ss] ");
172     for (int i=0; i<sizeof(poin)/sizeof(poin[0]); i++) {
173         Vpomin[i] = 4 * Rpo[i];
174         Vpmax[i] = 20 * Rpo[i];
175         sprintf(ic, "%u ", i);
176         strcat(datastr, "Vpo[mV]");
177         strcat(datastr, ic);
178         strcat(datastr, "po[bar]");
179         strcat(datastr, ic);
180     }
181     for (int i=0; i<sizeof(Tin)/sizeof(Tin[0]); i++) {
182         sprintf(ic, "%u ", i);
183         strcat(datastr, "VT[mV]");
184         strcat(datastr, ic);
185         strcat(datastr, "T[° C]");
186         strcat(datastr, ic);
187     }
188     for (int i=0; i<sizeof(pain)/sizeof(pain[0]); i++) {
189         sprintf(ic, "%u ", i);
190         strcat(datastr, "Vpa[mV]");
191         strcat(datastr, ic);
192         strcat(datastr, "pa[Pa]");
193         strcat(datastr, ic);
194     }
195     for (int i=0; i<sizeof(macin)/sizeof(macin[0]); i++) {
196         sprintf(ic, "%u ", i);
197         strcat(datastr, "AccX[g]");
198         strcat(datastr, ic);
199         strcat(datastr, "AccY[g]");
200         strcat(datastr, ic);
201         strcat(datastr, "AccZ[g]");
202         strcat(datastr, ic);
203         strcat(datastr, "GyroX[deg]");
204         strcat(datastr, ic);
205         strcat(datastr, "GyroY[deg]");
206         strcat(datastr, ic);
207         strcat(datastr, "GyroZ[deg]");

```

```

208         strcat(datastr,ic);
209         strcat(datastr,"AccAngX[g]");
210         strcat(datastr,ic);
211         strcat(datastr,"AccAngY[deg]");
212         strcat(datastr,ic);
213         strcat(datastr,"GyroAngX[deg]");
214         strcat(datastr,ic);
215         strcat(datastr,"GyroAngY[deg]");
216         strcat(datastr,ic);
217     }
218     strcat(datastr,"V3[mV] ");
219     strcat(datastr,"V5[mV] ");
220     strcat(datastr,"SDStatus");
221     Serial.println(datastr);
222     File dataFile = SD.open("a.txt", FILE_WRITE);
223     if (dataFile) {
224         dataFile.println(datastr);
225         delay(50);
226         dataFile.close();
227     }
228     // -----
229
230     //--- Stampa e scrittura del tempo della data all'accensione ---
231     t=millis();
232     sprintf(vc,"%lu ",t);
233     strcpy(datastr,vc);
234     DateTime now = rtc.now();
235     sprintf(rtcdate,"%04d/%02d/%02d",now.year(),now.month(),now.day());
236     sprintf(rtctime,"%02d:%02d:%02d",now.hour(),now.minute(),now.second());
237     strcat(datastr,rtcdate);
238     strcat(datastr," ");
239     strcat(datastr,rtctime);
240     trtc=t;
241     Serial.println(datastr);
242     dataFile = SD.open("a.txt", FILE_WRITE);
243     if (dataFile) {
244         dataFile.println(datastr);
245         delay(50);
246         dataFile.close();
247     }
248     // -----
249 }
250
251 void loop() {
252
253     // --- Lettura del tempo trascorso ---
254     t=millis();
255     sprintf(vc,"%lu ",t);
256     strcpy(datastr,vc);
257     // -----
258
259     // --- Lettura della data e dell'ora ---
260     if (t-trtc>dtrtc) {
261         DateTime now = rtc.now();
262         sprintf(rtcdate,"%04d/%02d/%02d",now.year(),now.month(),now.day());
263         sprintf(rtctime,"%02d:%02d:%02d",now.hour(),now.minute(),now.second());
264         strcat(datastr,rtcdate);
265         strcat(datastr," ");
266         strcat(datastr,rtctime);
267         trtc=t;
268     }
269     else {
270         strcat(datastr, "nan nan");
271     }
272     // -----
273
274     // --- Lettura dei trasmettitori di pressione e calcolo delle pressioni ---
275     for (int i=0; i<sizeof(poin)/sizeof(poin[0]); i++) {
276         Vpo[i] = analogRead(poin[i]); //Lettura analogica
277         Vpo[i] = Vpo[i] * 5000/1024; //Trasformazione in mV
278         po[i] = pomin[i]+(pomax[i]-pomin[i])*(Vpo[i]-Vpomin[i])/(Vpomax[i]-Vpomin[i]);
279         dtostrf(Vpo[i], 4, 2, vc);
280

```

```

281         strcat(datastr, " ");
282         strcat(datastr, vc);
283         dtostrf(po[i], 3, 2, vc);
284         strcat(datastr, " ");
285         strcat(datastr, vc);
286     }
287     // -----
288
289     // --- Lettura delle Pt100 e calcolo delle temperature ---
290     for (int i=0; i<sizeof(Tin)/sizeof(Tin[0]); i++) {
291         VT[i] = analogRead(Tin[i]);
292         VT[i] = VT[i] * 5000/1024;
293         RPt100[i]=AT[i]*VT[i]/1000+BT[i];
294         T[i] = (RPt100[i]-100)/(100*3.9083*0.001);
295         dtostrf(VT[i], 4, 1, vc);
296         strcat(datastr, " ");
297         strcat(datastr, vc);
298         dtostrf(T[i], 3, 1, vc);
299         strcat(datastr, " ");
300         strcat(datastr, vc);
301     }
302     // -----
303
304     // --- Lettura dei sensori di pressione differenziale e calcolo delle pressioni ---
305     for (int i=0; i<sizeof(pain)/sizeof(pain[0]); i++) {
306         Vpa[i] = analogRead(pain[i]);
307         Vpa[i] = Vpa[i] * 5000/1024;
308         if (ptype[i]==10) {
309             pa[i] = Vpa[i]*Ap[i] + Bp[i];
310         }
311         else {
312             if (ptype[i]==2) {
313                 pa[i] = Vpa[i]*Ap[i] + Bp[i];
314             }
315         }
316         dtostrf(Vpa[i], 4, 1, vc);
317         strcat(datastr, " ");
318         strcat(datastr, vc);
319         dtostrf(pa[i], 4, 1, vc);
320         strcat(datastr, " ");
321         strcat(datastr, vc);
322     }
323     // -----
324
325     // --- Lettura degli accelerometri ---
326     for (int i=0; i<sizeof(macin)/sizeof(macin[0]); i++) {
327         tcaselect(macin[i]);
328         Wire.beginTransmission(MPU);
329         Wire.write(0x3B);
330         Wire.endTransmission(false);
331         Wire.requestFrom(MPU, 6, true);
332         AccX[i] = (Wire.read() << 8 | Wire.read()) / 16384.0;
333         AccY[i] = (Wire.read() << 8 | Wire.read()) / 16384.0;
334         AccZ[i] = (Wire.read() << 8 | Wire.read()) / 16384.0;
335         AccAngX[i] = (atan(AccY[i] / sqrt(pow(AccX[i], 2) +
336         pow(AccZ[i], 2))) * 180 / PI) - AccErrorX[i];
337         AccAngY[i] = (atan(-1 * AccX[i] / sqrt(pow(AccY[i], 2) +
338         pow(AccZ[i], 2))) * 180 / PI) - AccErrorY[i];
339         previousTime[i] = currentTime[i];
340         currentTime[i] = millis();
341         elapsedTime[i] = (currentTime[i] - previousTime[i]) / 1000;
342         Wire.beginTransmission(MPU);
343         Wire.write(0x43);
344         Wire.endTransmission(false);
345         Wire.requestFrom(MPU, 6, true);
346         GyroX[i] = (Wire.read() << 8 | Wire.read()) / 131.0;
347         GyroY[i] = (Wire.read() << 8 | Wire.read()) / 131.0;
348         GyroZ[i] = (Wire.read() << 8 | Wire.read()) / 131.0;
349         GyroX[i] = GyroX[i] - GyroErrorX[i];
350         GyroY[i] = GyroY[i] - GyroErrorY[i];
351         GyroZ[i] = GyroZ[i] - GyroErrorZ[i];
352         GyroAngX[i] = GyroAngX[i] + GyroX[i] * elapsedTime[i];
353         GyroAngY[i] = GyroAngY[i] + GyroY[i] * elapsedTime[i];

```

```

354         dtostrf(AccX[i], 2, 2, vc);
355         strcat(datastr, " ");
356         strcat(datastr,vc);
357         dtostrf(AccY[i], 2, 2, vc);
358         strcat(datastr, " ");
359         strcat(datastr,vc);
360         dtostrf(AccZ[i], 2, 2, vc);
361         strcat(datastr, " ");
362         strcat(datastr,vc);
363         dtostrf(GyroX[i], 3, 1, vc);
364         strcat(datastr, " ");
365         strcat(datastr,vc);
366         dtostrf(GyroY[i], 3, 1, vc);
367         strcat(datastr, " ");
368         strcat(datastr,vc);
369         dtostrf(GyroZ[i], 3, 1, vc);
370         strcat(datastr, " ");
371         strcat(datastr,vc);
372         dtostrf(AccAngX[i], 4, 1, vc);
373         strcat(datastr, " ");
374         strcat(datastr,vc);
375         dtostrf(AccAngY[i], 4, 1, vc);
376         strcat(datastr, " ");
377         strcat(datastr,vc);
378         dtostrf(GyroAngX[i], 4, 1, vc);
379         strcat(datastr, " ");
380         strcat(datastr,vc);
381         dtostrf(GyroAngY[i], 4, 1, vc);
382         strcat(datastr, " ");
383         strcat(datastr,vc);
384     }
385     // -----
386
387     // --- Lettura delle tensioni di alimentazione di Arduino ---
388     V3=analogRead(V3inpin);
389     V3=V3*5000/1024;
390     V5=analogRead(V5inpin);
391     V5=V5*5000/1024;
392     dtostrf(V3, 4, 1, vc);
393     strcat(datastr, " ");
394     strcat(datastr,vc);
395     dtostrf(V5, 4, 1, vc);
396     strcat(datastr, " ");
397     strcat(datastr,vc);
398     // -----
399
400     // --- Stampa e salvataggio di tutti i dati letti ---
401     File dataFile = SD.open("a.txt", FILE_WRITE);
402     if (dataFile) {
403         strcpy(SDstatus, "ON");
404         strcat(datastr, " ON");
405         dataFile.println(datastr);
406         delay(50);
407         dataFile.close();
408     }
409     else {
410         strcpy(SDstatus, "OFF");
411         strcat(datastr, " OFF");
412     }
413     Serial.println(datastr);
414     // -----
415
416     // --- Visualizzazione su richiesta dei dati su Display OLED ---
417     if(digitalRead(pbuts)==HIGH){
418         display.setCursor(0, 0);
419         display.print("po");
420         display.print(" ");
421         for (int i=0; i<sizeof(poin)/sizeof(poin[0]); i++) {
422             display.print(poin[i]);
423             display.print(" ");
424             if (i==1){
425                 display.println();
426                 display.print("po ");

```

```

427     }
428 }
429 display.println();
430 display.print("To");
431 display.print(" ");
432 for (int i=0; i<sizeof(Tin)/sizeof(Tin[0]); i++) {
433     display.print(T[i]);
434     display.print(" ");
435 }
436 display.println();
437 display.print("pa");
438 display.print(" ");
439 for (int i=0; i<sizeof(pain)/sizeof(pain[0]); i++) {
440     display.print(pa[i]);
441     display.print(" ");
442 }
443 display.println();
444 display.print("a ");
445 for (int i=0; i<sizeof(macin)/sizeof(macin[0]); i++) {
446     if (i==1){
447         display.println();
448         display.print("a ");
449     }
450     display.print(AccX[i]);
451     display.print(" ");
452     display.print(AccY[i]);
453     display.print(" ");
454     display.print(AccZ[i]);
455 }
456 display.println();
457 display.print("SD Status ");
458 display.print(SDstatus);
459 display.display();
460 delay(tdisplay);
461 display.clearDisplay();
462 display.display();
463 }
464 // -----
465
466 // --- Introduzione di un tempo di attesa ---
467 delay(32);
468 // -----
469 }
470
471 // --- Definizione della funzione di calcolo errori accelerometri ---
472 void calculate_IMU_error() {
473     for (int i=0; i<sizeof(macin)/sizeof(macin[0]); i++) {
474         tcaselect(macin[i]);
475         while (conterr < 200) {
476             Wire.beginTransmission(MPU);
477             Wire.write(0x3B);
478             Wire.endTransmission(false);
479             Wire.requestFrom(MPU, 6, true);
480             AccX[i] = (Wire.read() << 8 | Wire.read()) / 16384.0 ;
481             AccY[i] = (Wire.read() << 8 | Wire.read()) / 16384.0 ;
482             AccZ[i] = (Wire.read() << 8 | Wire.read()) / 16384.0 ;
483             AccErrorX[i] = AccErrorX[i] + ((atan((AccY[i]) /
484             sqrt(pow((AccX[i]), 2) + pow((AccZ[i]), 2))) * 180 / PI));
485             AccErrorY[i] = AccErrorY[i] + ((atan(-1 * (AccX[i]) /
486             sqrt(pow((AccY[i]), 2) + pow((AccZ[i]), 2))) * 180 / PI));
487             conterr++;
488         }
489         AccErrorX[i] = AccErrorX[i] / 200;
490         AccErrorY[i] = AccErrorY[i] / 200;
491         conterr = 0;
492         while (conterr < 200) {
493             Wire.beginTransmission(MPU);
494             Wire.write(0x43);
495             Wire.endTransmission(false);
496             Wire.requestFrom(MPU, 6, true);
497             GyroX[i] = Wire.read() << 8 | Wire.read();
498             GyroY[i] = Wire.read() << 8 | Wire.read();
499             GyroZ[i] = Wire.read() << 8 | Wire.read();

```



```

500         GyroErrorX[i] = GyroErrorX[i] + (GyroX[i] / 131.0);
501         GyroErrorY[i] = GyroErrorY[i] + (GyroY[i] / 131.0);
502         GyroErrorZ[i] = GyroErrorZ[i] + (GyroZ[i] / 131.0);
503         conterr++;
504     }
505     GyroErrorX[i] = GyroErrorX[i] / 200;
506     GyroErrorY[i] = GyroErrorY[i] / 200;
507     GyroErrorZ[i] = GyroErrorZ[i] / 200;
508     Serial.print("AccErrorX: ");
509     Serial.print(AccErrorX[i]);
510     Serial.print(" ");
511     Serial.print("AccErrorY: ");
512     Serial.print(AccErrorY[i]);
513     Serial.print(" ");
514     Serial.print("GyroErrorX: ");
515     Serial.print(GyroErrorX[i]);
516     Serial.print(" ");
517     Serial.print("GyroErrorY: ");
518     Serial.print(GyroErrorY[i]);
519     Serial.print(" ");
520     Serial.print("GyroErrorZ: ");
521     Serial.println(GyroErrorZ[i]);
522     conterr=0;
523 }
524 }
525 // -----
526
527 // --- Definizione della funzione di selezione canale Multiplexer ---
528 void tcasselect(uint8_t i) {
529     if (i > 7) return;
530     Wire.beginTransmission(TCAADDR);
531     Wire.write(1 << i);
532     Wire.endTransmission();
533 }
534 // -----

```

2.3.3 CAN-BUS

Più snello risulta il codice di programmazione della scheda SparkFun ReadBoard impiegata per acquisire i dati CAN-BUS, trascritto qui sotto.

```

1 // --- Inclusione delle librerie ---
2 #include <mcp_can.h>
3 #include <SPI.h>
4 #include <SD.h>
5 #include <RTCLib.h>
6 // -----
7
8 // --- Dichiarazione delle variabili dei messaggi CAN ---
9 long unsigned int id;
10 unsigned char len = 0;
11 unsigned char buf[8];
12 char msgString[128];
13 char msg[60];
14 const int nmsg = 10;
15 // -----
16
17 // --- Dichiarazione della variabili di tempo e data ---
18 long unsigned int t=0;
19 char date[16];
20 char time[16];
21 char tc[9];
22 RTC_DS3231 rtc;
23 // -----
24
25 // --- Dichiarazione del pin CS della Micro-SD SPI ---
26 const int chipSelect = 9;
27 // -----
28

```

```

29 // --- Definizione del contatore per il salvataggio del tempo ---
30 unsigned int c = 0;
31 const int cmax = 2000;
32 // -----
33
34 // --- Dichiarazione dei pin per il funzionamento del MCP2515 ---
35 #define CAN0_INT 2
36 MCP_CAN CAN0(10);
37 // -----
38
39 void setup() {
40
41     // --- Inizializzazione della comunicazione seriale ---
42     Serial.begin(115200);
43     // -----
44
45     // --- Introduzione di un tempo di attesa ---
46     delay(1000);
47     // -----
48
49     // --- Inizializzazione del CAN Controller MCP 2515 ---
50     if (CAN0.begin(MCP_STDEXT, CAN_250KBPS, MCP_16MHZ) == CAN_OK) {
51         Serial.println("MCP2515 Initialized Successfully!");
52     }
53     else {
54         Serial.println("Error Initializing MCP2515...");
55     }
56     // -----
57
58     // --- Impostazione dei filtri per i dati CAN ---
59     CAN0.init_Mask(0,1,0xFFFFFFFF);
60     CAN0.init_Mask(1,1,0xFFFFFFFF);
61     CAN0.init_Filt(2,1,0x0CF00400);
62     CAN0.init_Filt(3,1,0x0CF00300);
63     CAN0.init_Filt(4,1,0x18FEF200);
64     CAN0.init_Filt(5,1,0x18FEF500);
65     CAN0.setMode(MCP_NORMAL);
66     pinMode(CAN0_INT, INPUT);
67     // -----
68
69     // --- Introduzione di un tempo di attesa ---
70     delay(1000);
71     // -----
72
73     // --- Setup della scheda SD SPI ---
74     Serial.print("Initializing SD card...");
75     pinMode(chipSelect, OUTPUT);
76     if (!SD.begin(chipSelect)) {
77         Serial.println("Card failed, or not present");
78         return;
79     }
80     Serial.println("card initialized.");
81     // -----
82
83     // --- Setup del Real Time Clock ---
84     rtc.begin(); //Inizializzazione del real time clock
85     //rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
86     //rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
87     // -----
88
89     // --- Stampa e salvataggio delle data e dell'ora all'accensione ---
90     DateTime now = rtc.now();
91     sprintf(date, "%04d/%02d/%02d", now.year(), now.month(), now.day());
92     sprintf(time, "%02d:%02d:%02d", now.hour(), now.minute(), now.second());
93     Serial.print(date);
94     Serial.print(" ");
95     Serial.print(time);
96     t=millis();
97     Serial.print(" ");
98     Serial.println(t);
99     // -----
100
101     // --- Stampa e salvataggio dei nomi delle colonne di dati campionati ---

```

```

102     Serial.println("t[ms] Date[YYYY/MM/DD] Time[HH:mm:ss] ID Byte Data");
103     File dataFile = SD.open("candata.txt", FILE_WRITE);
104     if (dataFile) {
105         dataFile.print(date);
106         dataFile.print(" ");
107         dataFile.print(time);
108         dataFile.print(" ");
109         dataFile.println(t);
110         dataFile.println("t[ms] ID Byte Data");
111         dataFile.close();
112     }
113     // -----
114
115     // --- Introduzione di un tempo di attesa ---
116     delay(1000);
117     // -----
118
119 }
120
121 void loop() {
122
123     if (!digitalRead(CAN0_INT)) {
124         // --- Lettura del tempo e del Real Time Clock ---
125         c = c + 1;
126         t=millis();
127         sprintf(tc,"%lu ",t);
128         strcpy(msg,tc);
129         if (c > cmax - 1) {
130             DateTime now = rtc.now();
131             sprintf(date,"%04d/%02d/%02d",now.year(),now.month(),
132                 now.day());
133             sprintf(time,"%02d:%02d:%02d",now.hour(),now.minute(),
134                 now.second());
135             strcat(msg,date);
136             strcat(msg, " ");
137             strcat(msg,time);
138             strcat(msg, " ");
139             c = 0;
140         }
141         else {
142             strcat(msg, "nan nan ");
143         }
144         // -----
145
146         // --- Lettura del messaggio CAN ---
147         CAN0.readMsgBuf(&id, &len, buf);
148         sprintf(msgString, "%.8lX %ld", (id & 0xFFFFFFFF), len);
149         strcat(msg,msgString);
150         for (byte i = 0; i<len; i++) {
151             sprintf(msgString, " %.2X", buf[i]);
152             strcat(msg,msgString);
153         }
154         // -----
155
156         // --- Stampa e salvataggio del tempo e del messaggio ---
157         File dataFile = SD.open("candata.txt", FILE_WRITE);
158         if (dataFile) {
159             dataFile.println(msg);
160             dataFile.close();
161             Serial.println(msg);
162         }
163         // -----
164
165     }
166 }

```

2.3.4 GPS

Come ultimo è riportato il codice di programmazione della SparkFun RedBoard, utilizzata per la misura della posizione geografica tramite modulo GNSS.

```

1
2 // --- Inclusione delle librerie ---
3 #include <Wire.h>
4 #include <SPI.h>
5 #include <SD.h>
6 #include <SparkFun_u-blox_GNSS_Arduino_Library.h>
7 // -----
8
9 // --- Dichiarazione delle variabili data, ora, posizione e velocita' ---
10 SFE_UBLOX_GNSS myGNSS;
11 char gpsdate[16];
12 char gpstime[16];
13 long lat=0;
14 long lon=0;
15 long speed=0;
16 // -----
17
18 // --- Dichiarazione delle variabili per la Micro-SD SPI ---
19 const int chipSelect = 10;
20 char SDstatus[8] = "";
21 // -----
22
23 // ----- Dichiarazione della variabile tempo -----
24 long unsigned int t = 0;
25 // -----
26
27 // ----- Dichiarazione della variabili di testo -----
28 char datastr[128];
29 char ic[1];
30 char vc[12];
31 // -----
32
33 // --- Dichiarazione del pin del led di segnalazione ---
34 const int ledpin=9;
35 // -----
36
37 void setup() {
38
39     // --- Inizializzazione della comunicazione seriale ---
40     Serial.begin(115200);
41     // -----
42
43     // --- Setup della Miro-SD SPI ---
44     if (!SD.begin(chipSelect)) {
45         Serial.println("Card failed or not present");
46     }
47     else {
48         Serial.println("Card initialized.");
49     }
50     // -----
51
52     // --- Inizializzazione della comunicazione i2c ---
53     Wire.begin();
54     myGNSS.begin();
55     if (myGNSS.begin() == false) {
56         Serial.println(F("u-blox GNSS not detected at default I2C address. Please check wiring. Freezing."));
57         while (1);
58     }
59     // -----
60
61     // --- Stampa dei nomi delle colonne ---
62     strcpy(datastr,"t[ms] Date[YYYY/MM/DD] Time[HH:mm:ss] Lat[°] Lon[°] Speed[mm/s] SDStatus");
63     Serial.println(datastr);
64     // -----
65

```

```

66      // --- Scrittura di nomi delle colonne ---
67      File dataFile = SD.open("gps.txt", FILE_WRITE);
68      if (dataFile) {
69          dataFile.println(datastr);
70          dataFile.close();
71      }
72      // -----
73
74      // --- Impostazione del pin per il led ---
75      pinMode(ledpin, OUTPUT);
76      // -----
77  }
78
79  void loop() {
80
81      if (millis() - t > 1000){
82
83          // --- Lettura del tempo ---
84          t=millis();
85          sprintf(vc,"%lu ",t);
86          strcpy(datastr,vc);
87          // -----
88
89          // --- Lettura della data e dell'ora GNSS ---
90          sprintf(gpsdate,"%04d/%02d/%02d",myGNSS.getYear(),myGNSS.getMonth(),myGNSS.getDay());
91          sprintf(gpstime,"%02d:%02d:%02d",myGNSS.getHour(),myGNSS.getMinute(),myGNSS.getSecond());
92          strcat(datastr,gpsdate);
93          strcat(datastr," ");
94          strcat(datastr,gpstime);
95          // -----
96
97          // --- Lettura della latitudine, della longitudine e della velocita' ---
98          lat = myGNSS.getLatitude();
99          sprintf(vc,"%u ",lat);
100         strcat(datastr," ");
101         strcat(datastr,vc);
102         lon = myGNSS.getLongitude();
103         sprintf(vc,"%u ",lon);
104         strcat(datastr," ");
105         strcat(datastr,vc);
106         speed = myGNSS.getGroundSpeed();
107         sprintf(vc,"%u ",speed);
108         strcat(datastr," ");
109         strcat(datastr,vc);
110         // -----
111
112         // --- Scrittura dei dati e accensione del LED ---
113         File dataFile = SD.open("gps.txt", FILE_WRITE);
114         if (dataFile) {
115             strcpy(SDstatus, "ON");
116             strcat(datastr, " ON");
117             dataFile.println(datastr);
118             dataFile.close();
119             digitalWrite(ledpin, HIGH);
120             delay(200);
121             digitalWrite(ledpin, LOW);
122         }
123         else {
124             strcpy(SDstatus, "OFF");
125             strcat(datastr, " OFF");
126         }
127         // -----
128
129         // --- Stampa dei dati ---
130         Serial.println(datastr);
131         // -----
132     }
133 }

```

2.4 Elaborazione dei dati

Terminata la fase di acquisizione, si è in possesso di un'enorme mole di dati, contenuta all'interno dei file di testo, con una struttura simile a quella riportata qui sotto:

```
1 t[ms] Date[YYYY/MM/DD] Time[HH:mm:ss] Vpo[mV]0 po[bar]0 Vpo[mV]1 po[bar]1 Vpo[mV]2 ...
2 422337 2024/08/29 17:47:23 2626.95 -0.13 2680.66 26.96 1328.13 39.35 2026.37 19.03 ...
3 422557 nan nan 3359.38 0.25 2695.31 27.19 1279.30 34.23 2075.20 25.51 3125.0 62.6 ...
4 422778 nan nan 2695.31 -0.09 2709.96 27.42 1250.00 31.16 2050.78 22.27 3110.4 62.2 ...
5 422995 nan nan 3090.82 0.11 2675.78 26.88 1289.06 35.25 2031.25 19.68 3144.5 63.1 ...
6 423238 nan nan 3544.92 0.35 2695.31 27.19 1328.13 39.35 2045.90 21.62 3129.9 62.7 ...
7 423459 nan nan 2504.88 -0.19 2685.55 27.04 1352.54 41.91 2031.25 19.68 3085.9 61.6 ...
8 423677 nan nan 3305.66 0.23 2690.43 27.11 1362.30 42.94 2045.90 21.62 3081.1 61.5 ...
9 423916 nan nan 2719.73 -0.08 2690.43 27.11 1333.01 39.86 2026.37 19.03 3134.8 62.9 ...
10 424133 nan nan 3315.43 0.23 2690.43 27.11 1367.19 43.45 2036.13 20.33 3095.7 61.9 ...
11 424353 nan nan 2719.73 -0.08 2709.96 27.42 1352.54 41.91 2075.20 25.51 3120.1 62.5 ...
```

Gli identificativi in cima alle colonne sono i nomi delle variabili campionate, i cui valori numerici si trovano sotto di essi. Per ogni test eseguito vi sono 4 file di questo tipo, ciascuno proveniente da una scheda di acquisizione diversa. Quindi, la quantità di dati a disposizione è assai elevata poiché in un minuto di campionamento vengono salvati complessivamente 66300 dati, suddivisi come segue:

- 30000 da Arduino R4 Minima;
- 17000 da Arduino Mega 2560 Rev3;
- 19000 dalla SparkFun RedBoard con la CAN-BUS Shield;
- 300 dalla SparkFun RedBoard con il modulo GNSS.

In un giorno di raccolta, quindi, dalla durata di 8 ore i dati salvati ammonterebbero a 32000000 circa. È facile intuire, allora, che occorre una scrematura e una selezione accurata dei campionamenti, al fine di determinare quelli più esemplificativi per dedurre i parametri prestazionali della macchina in funzione delle condizioni di lavoro. Occorre tradurre i dati di maggiore importanza in una forma che sia più semplice da leggere e interpretare, avvalendosi di:

- grafici che mostrino l'andamento delle grandezze misurate durante le principali operazioni svolte dal veicolo o che mostrino l'influenza reciproca di alcune variabili;
- tabelle che riportino dei parametri operativi medi della macchina nelle più ricorrenti condizioni di lavoro.

L'elaborazione dei dati al fine di ottenere questa tipologia di *output* è stata svolta avvalendosi del software MATLAB.

In linea generale il diagramma di flusso dei processi di elaborazione è rappresentato in Fig. 2.31.

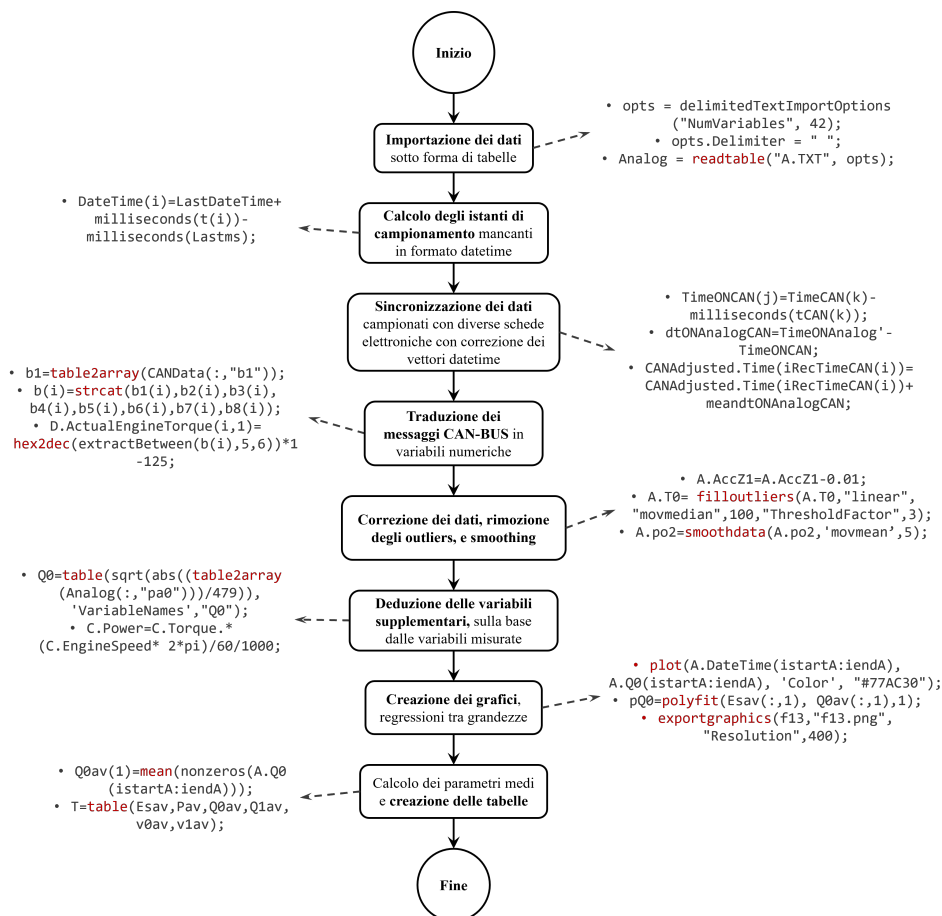


Figura 2.31: Diagramma di flusso dell'algoritmo di elaborazione e post-processing dei dati in ambiente MATLAB, con alcuni esempi di stringhe di codice.

Capitolo 3

Risultati

In questo capitolo sono illustrati nel dettaglio i risultati delle prove effettuate con la semovente, munita di tutti i sensori e dispositivi propedeutici al monitoraggio. Il capitolo in questione, così come sono stati suddivisi i test, si articola in 3 parti principali:

- l'analisi dei test effettuati sull'impianto di ventilazione, utili per determinare le prestazioni dei ventilatori, in termini di portate d'aria, che vanno rapportate ai consumi degli stessi;
- l'analisi dei test su strada, utile per caratterizzare il comportamento del veicolo, in termini di velocità, consumi energetici e prestazioni della trasmissione idrostatica, durante la marcia su strada asfaltata;
- l'analisi dei test in campo, utile a definire il comportamento della macchina nelle condizioni di utilizzo per cui è stata concepita, ovvero durante la raccolta delle nocciole in campo agricolo. In questa sezione si porrà l'attenzione su tutte le variabili campionate.

In Fig. 3.1 è possibile osservare la semovente all'opera durante i test in campo, effettuati il primo giorno di raccolta.



Figura 3.1: Semovente Cimina 300S in funzione durante il primo giorno di raccolta.

3.1 Test dell'impianto di ventilazione

Durante la prova è stato azionato il sistema pneumatico della macchina, con l'avviamento, attraverso una apposita frizione, del ventilatore centrifugo di aspirazione e di insaccamento del prodotto, del ventilatore tangenziale per la defogliazione e del crivello, o doppio vaglio rotativo, per la cernita. Attraverso una trasmissione di cinghie e pulegge, il moto rotativo viene trasferito dal motore a tutti questi organi, con un rapporto di trasmissione fisso.

Le prestazioni e gli assorbimenti dei dispositivi sono stati monitorati a differenti regimi di giri motore: circa 1200, 1400, 1600 e 1800 *rpm*. I parametri su cui ci si deve soffermare sono la pressione statica nei condotti, la portata d'aria e i consumi energetici in funzione del regime di rotazione del motore.

3.1.1 Condizioni ambientali

La grande quantità di sensori installati sul motore di serie permettono di estrapolare, dalla linea CAN-BUS, anche dati come la temperatura e la pressione ambientale. Come si può vedere in Fig. 3.2, queste due variabili durante il test ammontavano a 34.5°C per la temperatura e 96.5 kPa per la pressione atmosferica.

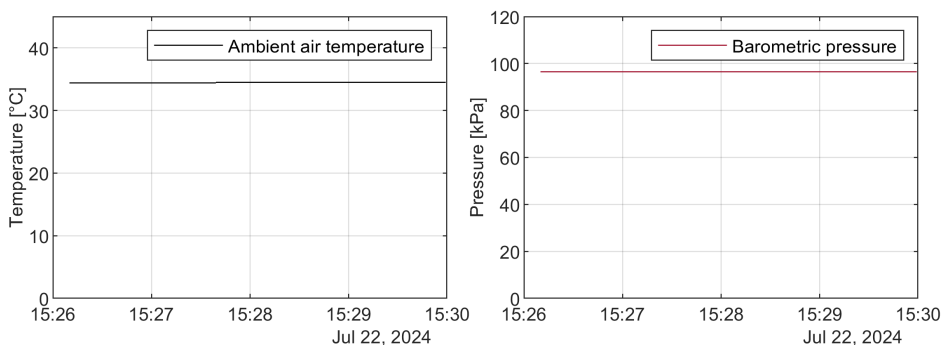


Figura 3.2: Condizioni ambientali durante il test dell'impianto di ventilazione: temperatura e pressione barometrica.

3.1.2 Parametri motore

Sempre dai dati motore, rappresentati in Fig. 3.3, è possibile individuare il momento in cui è stata attivata la frizione per l'accensione

dei ventilatori, avvenuta alle 15:25:48, che ha portato ad un picco sulla coppia assorbita, dovuto all'accelerazione degli organi rotanti.

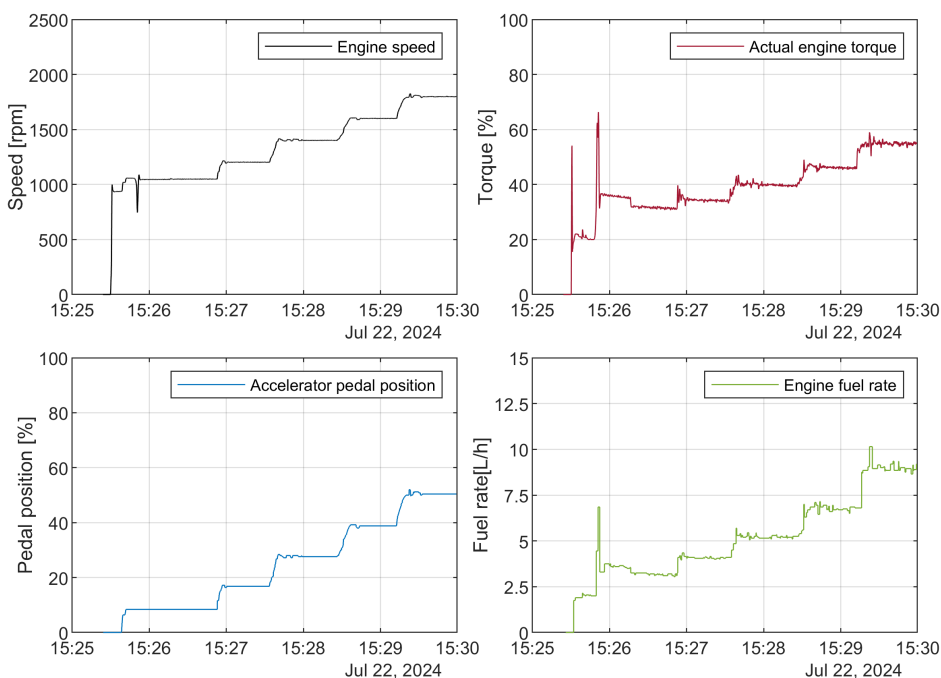


Figura 3.3: Parametri motore durante il test dell'impianto di ventilazione: velocità di rotazione, coppia relativa, posizione del pedale d'accelerazione e portata di combustibile.

Come si può vedere dalla posizione del pedale acceleratore e dai giri, il motore è stato accelerato a step di circa 200 rpm.

L'andamento della coppia e del consumo di combustibile ricalca quello del numero di giri, rimarcando ovviamente che il consumo energetico delle macchine sia proporzionale a quest'ultimo.

3.1.3 Prestazioni dei ventilatori

Così come il consumo energetico anche le portate d'aria calcolate, presentate in Fig. 3.4, incrementano a step all'aumentare del numero di giri. Di conseguenza anche le pressioni statiche misurate presentano un simile andamento, in quanto le perdite di carico dipendono dal quadrato della portata d'aria.

In Tab. 3.1 si riportano i consumi medi di potenza e le portate d'aria erogate dal ventilatore di aspirazione e da quello di insaccamento in

funzione del numero di giri impostato durante i test.

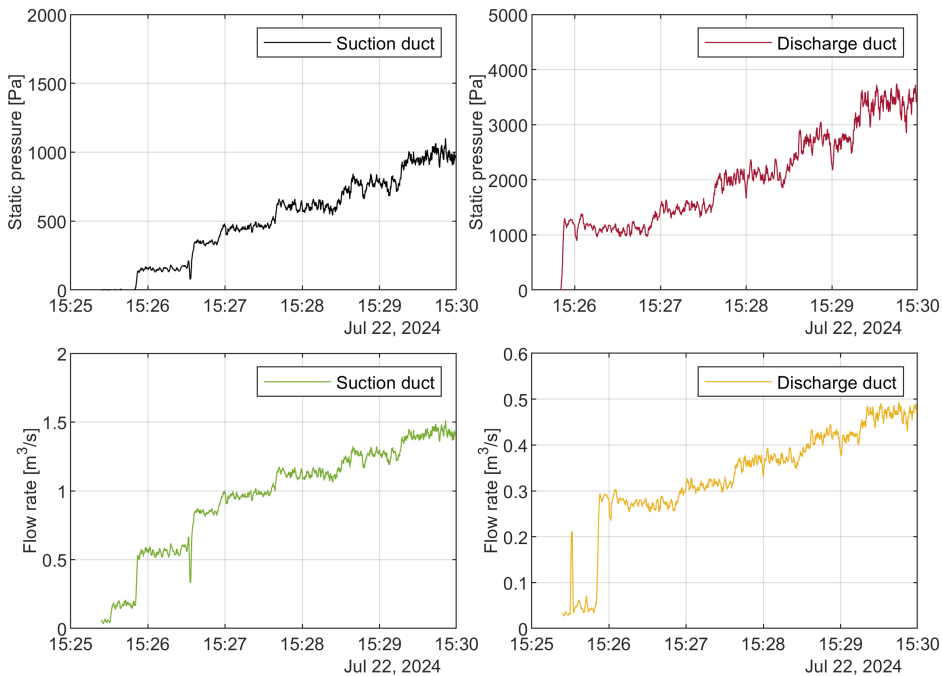


Figura 3.4: Prestazioni dei ventilatori durante il test: pressione statica e portata d'aria nei condotti di aspirazione e insaccamento.

Conoscendo i diametri del condotto di aspirazione del prodotto e del condotto di mandata, è possibile ricondursi alla velocità media dell'aria, che viene riportata, in funzione del numero di giri, in Tab.3.2.

Soffermandoci sui valori della velocità si può notare che, sebbene le portate erogate dai ventilatori siano molto diverse, le differenze tra le velocità dell'aria nei condotti si assottigliano per via dei loro differenti diametri. Ciò è necessario per garantire una simile portanza aerodinamica sul prodotto in aspirazione e mandata. Si può vedere comunque che la macchina è stata studiata per garantire una capacità di aspirazione maggiore, in quanto la portata Q_s a 1800 rpm risulta circa 3 volte maggiore della portata Q_d .

Graficando i valori della Tab.3.1 in Fig. 3.5 e 3.6, è possibile notare che la portata d'aria erogata dai ventilatori risulta linearmente dipendente dal numero di giri. Nello specifico il fit lineare dei dati è espresso adeguatamente dalle seguenti relazioni:

$$Q_s = 0.00074 \cdot n + 0.079 \quad (3.1)$$

$$Q_d = 0.00026 \cdot n + 0.00060 \quad (3.2)$$

con n velocità di rotazione del motore espressa in rpm, Q_s portata erogata dal ventilatore per l'aspirazione del prodotto, Q_d portata erogata dal ventilatore per l'insaccamento del prodotto.

n [rpm]	P [kW]	Q_s [m ³ /s]	Q_d [m ³ /s]
1203 ± 2	12.1 ± 0.2	0.97 ± 0.02	0.31 ± 0.01
1402 ± 4	16.4 ± 0.2	1.12 ± 0.02	0.37 ± 0.01
1604 ± 17	21.9 ± 1.0	1.26 ± 0.03	0.42 ± 0.02
1800 ± 6	29.0 ± 0.4	1.42 ± 0.03	0.47 ± 0.02

Tabella 3.1: Portate d'aria medie Q_s e Q_d erogate rispettivamente dal ventilatore per l'aspirazione e per l'insaccamento e potenza media P erogata dal motore per ogni velocità di rotazione n adottata durante i test.

n [rpm]	v_s [m/s]	v_d [m/s]
1203 ± 2	19.8 ± 0.4	15.5 ± 0.4
1402 ± 4	22.9 ± 0.5	18.3 ± 0.6
1604 ± 17	25.7 ± 0.6	20.8 ± 0.6
1800 ± 6	28.9 ± 0.6	23.3 ± 0.6

Tabella 3.2: Velocità medie dell'aria v_s e v_d rispettivamente nel condotto di aspirazione e nel condotto di scarico per ogni velocità di rotazione n adottata durante i test.

Non si può affermare lo stesso per il consumo di potenza. Infatti la relazione che intercorre tra questa variabile e la velocità di rotazione del motore non risulta affatto lineare, ma quadratica. Una valida equazione di regressione risulta:

$$P_c = 1.78 \cdot 10^{-5} \cdot n^2 - 0.0255 \cdot n + 16.9 \quad (3.3)$$

con P_c potenza consumata in kW .

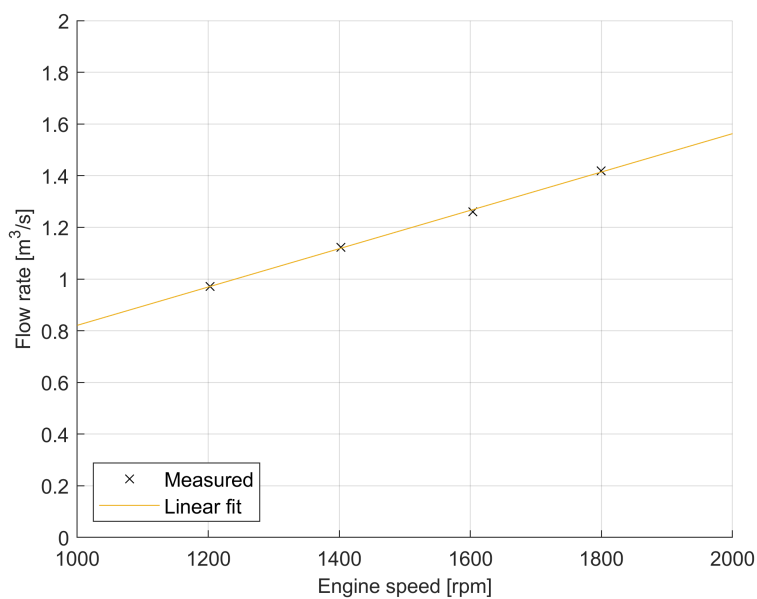


Figura 3.5: Fit lineare tra portata erogata dal ventilatore per l'aspirazione e numero di giri motore.

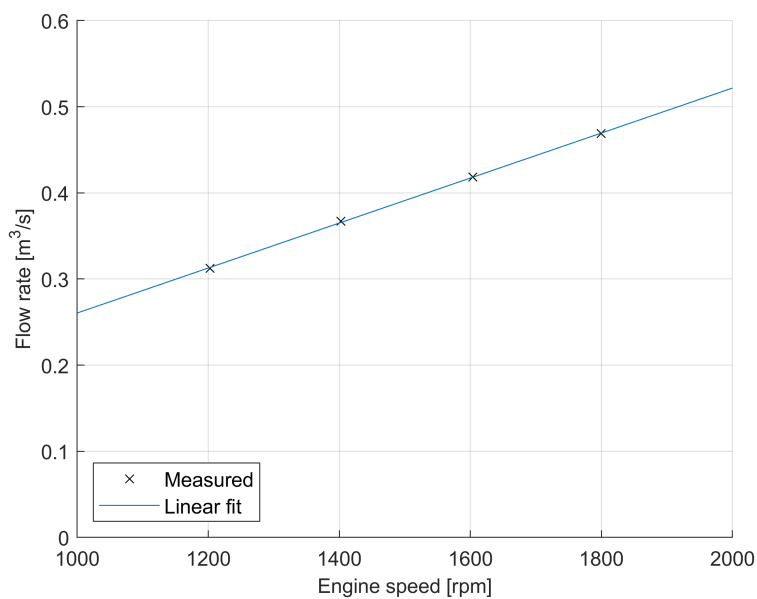


Figura 3.6: Fit lineare tra portata erogata dal ventilatore per l'insaccamento e numero di giri motore.

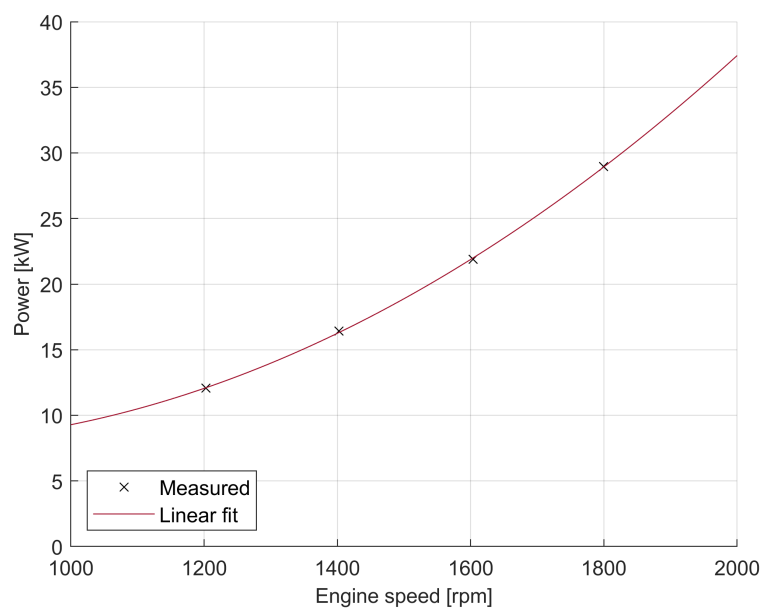


Figura 3.7: Fit quadratico tra giri motore e potenza consumata dal sistema di aspirazione e cernita.

3.2 Test su strada: massima velocità

Durante la prova, la macchina affronta un tragitto pressoché rettilineo di circa 300 m di lunghezza, caratterizzato da una pendenza media del 6%. La tratta affrontata è stata praticata in entrambi i versi di percorrenza per ben 2 volte e il veicolo è stato spinto al massimo della velocità, sia in salita che in discesa.

L'intento è quello di misurare la massima velocità del veicolo e vedere come si comportano la trasmissione idrostatica e il motore termico in tali condizioni di marcia su strada asfaltata.

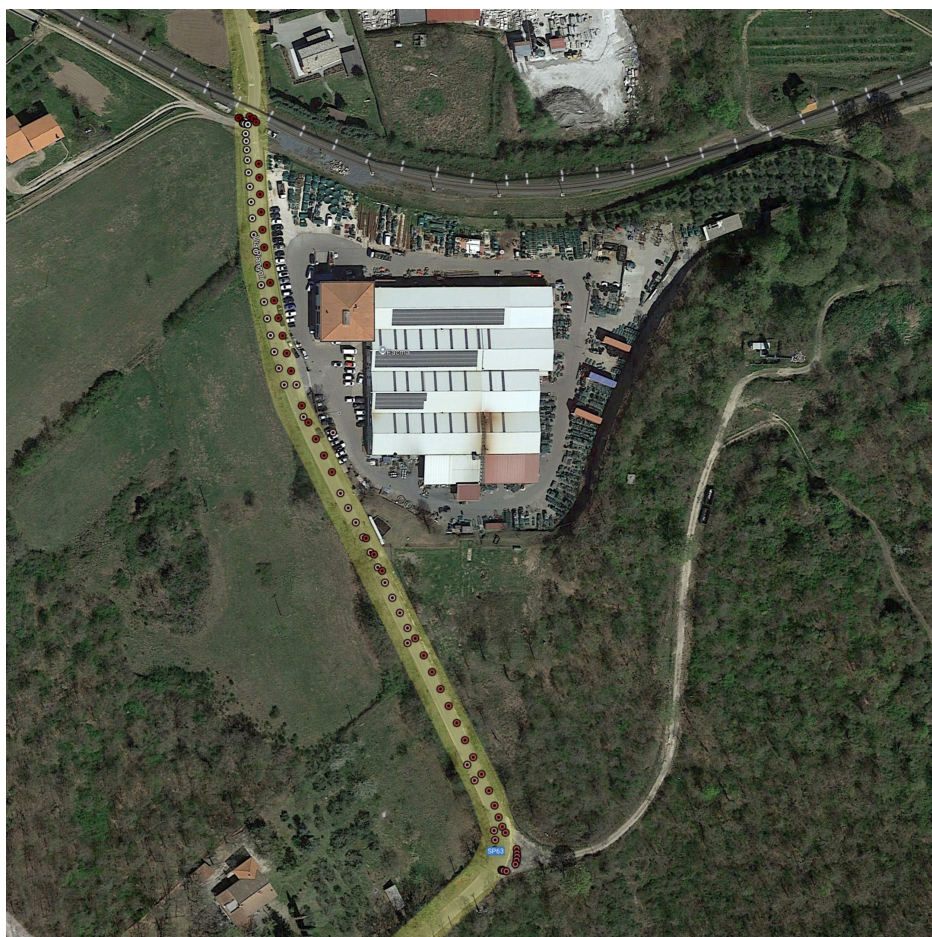


Figura 3.8: Tragitto di salita e discesa percorso dal veicolo 2 volte durante il test della massima velocità su strada.

3.2.1 Posizione

Osservando la Fig. 3.8, si capisce bene quale fosse il tragitto compiuto dalla macchina durante questo test.

La colorazione dei punti sulla mappa è stata effettuata sulla variabile tempo di campionamento: al trascorrere di questo il colore passa dal bianco al rosso.

L'intensità del segnale GPS ha garantito una misura accurata della posizione, ai fini di una corretta geolocalizzazione su strada, permettendo anche un calcolo accurato della velocità di avanzamento, come vedremo nella sezione 3.2.2.

3.2.2 Velocità

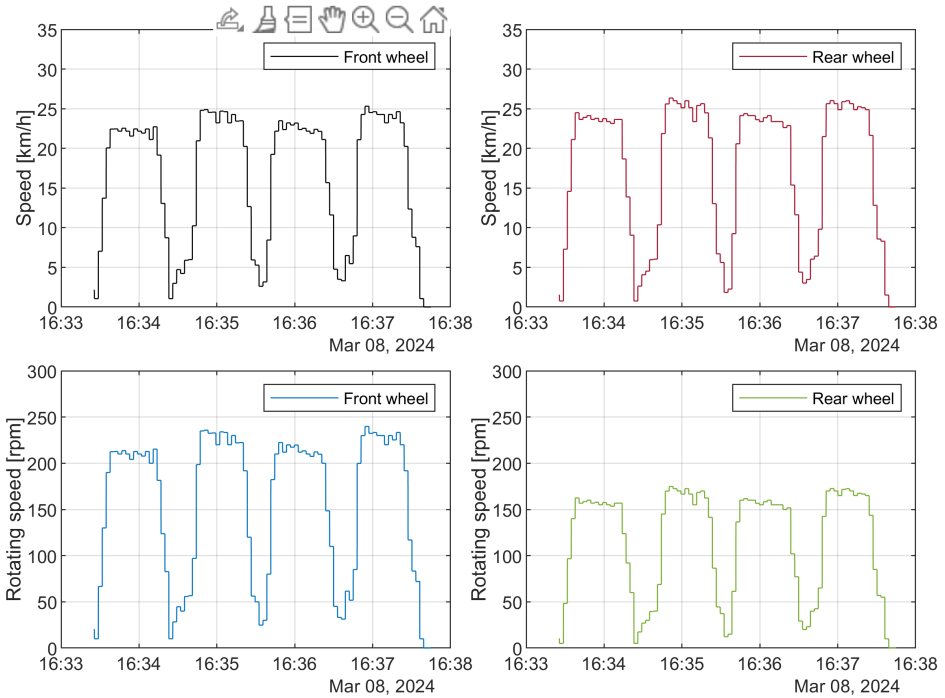


Figura 3.9: Velocità periferica e rotativa alle ruote anteriori e posteriori durante il test della massima velocità su strada.

In Fig. 3.9 si riporta l'andamento della velocità di rotazione e della velocità periferica delle ruote calcolate grazie ai sensori Hall.

Da notare che le velocità di rotazione sono diverse tra ruote posteriori e

anteriori per via del loro diverso diametro. Tuttavia, le differenze tra le velocità periferiche calcolate risultano minime, tra gli 1 e i 2 km/h, e potrebbero essere imputabili al valore utilizzato di raggio di lavoro delle ruote per il calcolo dei parametri, oltre che ai diversi slittamenti. Ovviamente la velocità massima del veicolo risulta minore durante la percorrenza della salita. In generale si può affermare che la velocità massima del veicolo si attesta sui 25 km/h. È interessante riportare, in Fig. 3.10, il confronto tra le velocità calcolate in funzione dei dati provenienti dai sensori Hall e dai dati di posizione, ricavati dal sensore GPS.



Figura 3.10: Velocità del veicolo media misurata con i sensori Hall e GPS a confronto durante il test della massima velocità su strada.

Il transitorio mostra una precisa corrispondenza tra le variabili riportate. Infatti la discrepanza assoluta ammonta a 1.5 km/h in media alle velocità massime. In percentuale questo errore risulta essere pari al 6% circa. Ciò dimostra affidabilità nella misura della velocità con entrambi i metodi, quando il segnale GPS è sufficientemente intenso.

3.2.3 Parametri motore

Dall'analisi dei parametri motore, Fig. 3.11, si nota che questo è stato accelerato fino ai 2700rpm per effettuare le tratte.

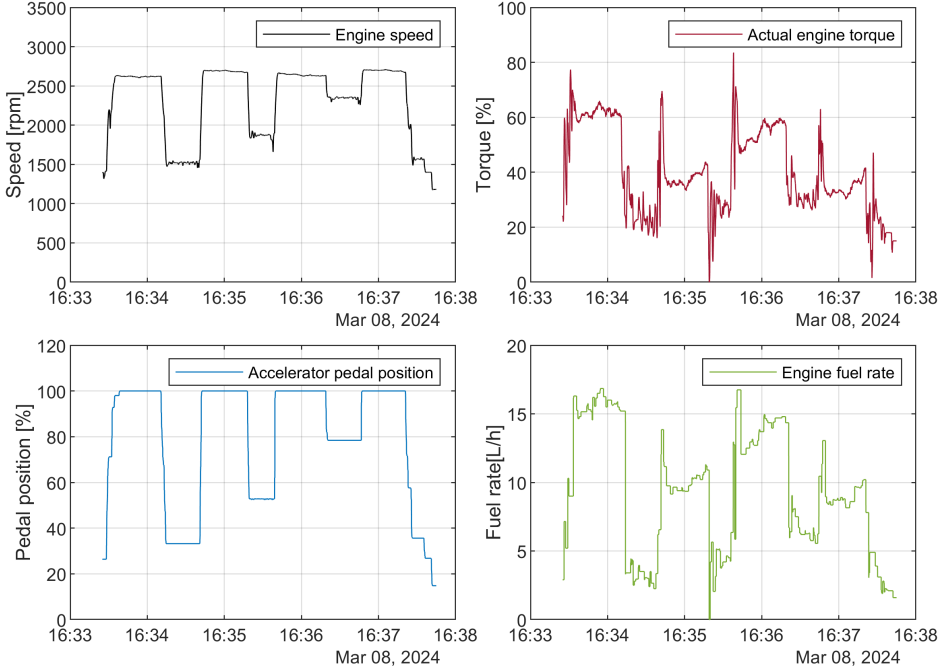


Figura 3.11: Parametri motore durante il test della massima velocità su strada: velocità di rotazione, coppia relativa, posizione del pedale d'accelerazione e portata di combustibile.

Questo regime corrisponde al massimo della corsa del pedale dell'acceleratore.

Osservando la coppia relativa e il consumo di combustibile, si nota che quando il veicolo deve accelerare, il motore risente delle forze inerziali e quindi i valori di coppia incrementano sensibilmente, presentando dei picchi locali. I valori di coppia si stabilizzano quando la macchina raggiunge la velocità d'equilibrio.

È possibile apprezzare come cambiano i consumi in funzione della tratta che il veicolo percorre: naturalmente se si trova in salita questi valori aumentano per vincere i valori della componente parallela della forza peso, viceversa si riducono. Si può vedere che, ovviamente, il consumo di combustibile è strettamente correlato agli andamenti della coppia.

In Fig. 3.12 si riporta l'andamento della potenza erogata dal motore e

della pendenza media della strada, calcolata mediante la funzione "Show Elevation Profile" di Google Earth Pro, che risulta pari al 6%, corrispondente a $3,4^\circ$ di inclinazione.

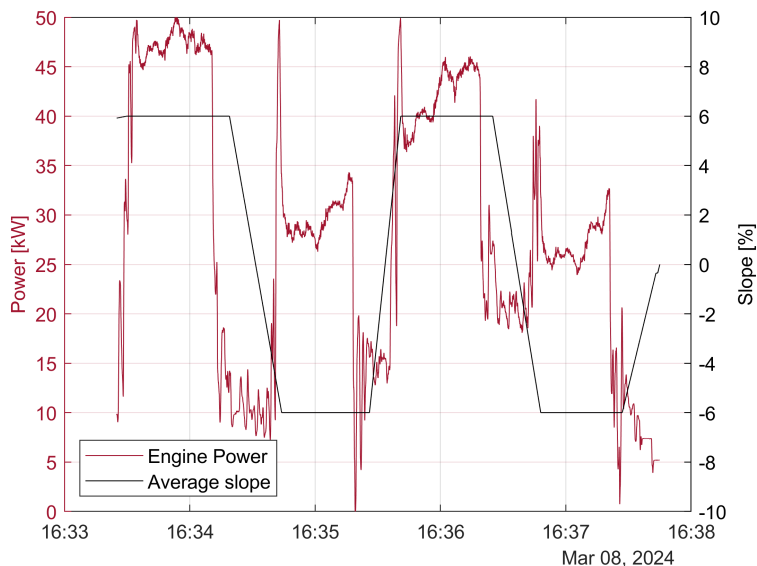


Figura 3.12: Potenza erogata dal motore e pendenza del terreno a confronto durante il test della massima velocità su strada.

Anche la potenza ricalca l'andamento della coppia e del consumo di combustibile, visto che i giri motore vengono mantenuti costanti durante i tratti di transito. Si riscontrano valori medi di $44.9 \pm 1.6 kW$ durante le tratte di salita e di $23.8 \pm 1.4 kW$ durante le tratte di discesa.

Potrebbe essere utile dedurre l'incremento approssimativo di potenza dovuto all'aumento di un grado percentuale di pendenza alla massima velocità su strada asfaltata, che risulta pari a $1.75 \pm 0.18 kW$.

3.2.4 Pressioni nel circuito idraulico

Dall'analisi delle pressioni nel circuito idraulico, Fig. 3.13, si possono determinare le condizioni di lavoro della pompa e dei motori idraulici, oltre che del sistema frenante. La pressione all'aspirazione della pompa presenta delle oscillazioni più o meno marcate a seconda se il veicolo stia accelerando o proceda a velocità costante. È utile calcolare il valor medio durante questa rilevazione, che si attesta sui $-0.0482 bar$. Da tenere in considerazione anche la depressione massima che si riscontra alla fine del

transitorio, pari a -0.18bar .

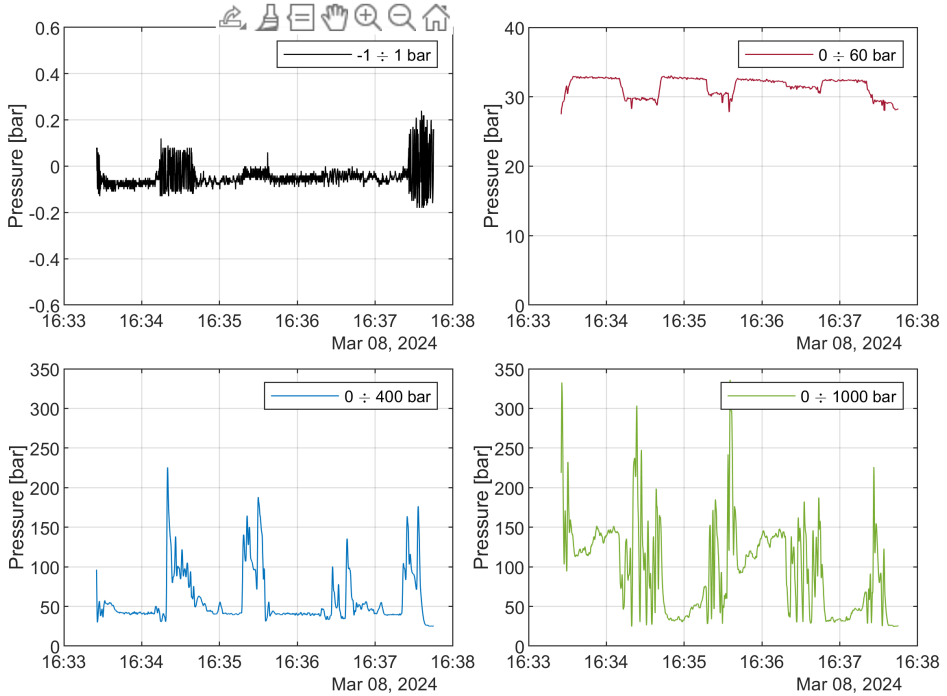


Figura 3.13: Pressioni idrauliche durante il test della massima velocità su strada: all'aspirazione della pompa ($-1 \div +1 \text{ bar}$), nel circuito frenante ($0 \div 40 \text{ bar}$), all'ingresso del motore idraulico anteriore ($0 \div 400 \text{ bar}$), alla mandata della pompa ($0 \div 1000 \text{ bar}$).

La pressione nel circuito frenante rimane pressoché costante durante tutta la prova, con un valor medio pari a $31.5 \pm 1.3\text{bar}$, mostrando dei piccoli incrementi in funzione del numero di giri del motore. Questo perché il circuito frenante è separato da quello di trasmissione e viene mantenuto in pressione da una pompa ad ingranaggi esterni a cilindrata fissa, anch'essa direttamente collegata al motore come la pompa per la trasmissione.

La pressione di mandata della pompa presenta un andamento fluttuante durante le fasi di accelerazione e decelerazione del veicolo, con picchi che possono arrivare anche a 300bar , valore che comunque si trova al di sotto della pressione di taratura delle limitatrici pari a 420bar . Il transitorio risulta più o meno costante quando la coppia motore si stabilizza e si riscontrano pressioni medie di $129.5 \pm 14.1\text{bar}$ durante la marcia in salita

e di $82.6 \pm 15.9bar$ durante la marcia in discesa. Questi valori suggeriscono un aumento medio della pressione pari a $7.83 \pm 1.3bar$ per un aumento di un grado percentuale di pendenza.

Non si può dire lo stesso della pressione all'ingresso del motore idraulico anteriore, che si mantiene più bassa anche durante le tratte in salita. I picchi di pressione, in questo caso, raggiungono massimo $200bar$ e si presentano in fase di decelerazione del veicolo. Quando la velocità è costante non si riscontra più un calo delle pressioni marcato tra le tratte in salita ($40.6 \pm 4.7bar \pm$) e in discesa ($40.3 \pm 4.6bar$). Da questa osservazione si può dedurre che, a contribuire maggiormente alle capacità di trazione della macchina siano effettivamente le ruote posteriori.

3.3 Test su strada: tragitto d'esempio

In questa sezione si mostrano le variabili campionate durante un tipico transito su strada asfaltata, avvenuto per trasferire il veicolo dalla rimessa ad un campo di raccolta.

L'intento è quello di monitorare il veicolo in toto durante le più comuni condizioni di marcia su strada asfaltata, soffermandosi sulle variabili velocità, consumi, pendenze, pressioni e temperature dell'olio idraulico. Per comodità e per una maggior chiarezza si riporteranno i grafici delle variabili campionate in un intervallo di 10 minuti. La scarsa intensità del segnale GPS non ha permesso, tuttavia, di determinare la posizione del veicolo.

Infine il test ha consentito di valutare i consumi medi e le pressioni medie nel circuito idraulico durante le condizioni stazionarie di marcia del veicolo su strada pressoché pianeggiante, quando il veicolo viaggia a 18 km/h a 2000 rpm .

3.3.1 Condizioni ambientali

In Fig. 3.14 si riportano la temperatura dell'aria ambientale e la pressione barometrica campionate dai sensori collocati sul motore.

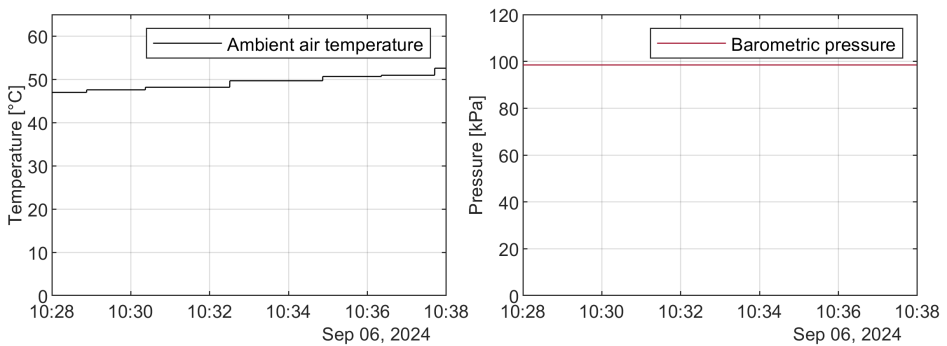


Figura 3.14: Condizioni ambientali durante il tragitto su strada: temperatura e pressione barometrica.

Si nota come la temperatura dell'aria in prossimità del motore termico è in crescita per via del surriscaldamento della zona, superando anche i 50°C .

3.3.2 Velocità

Dalla Fig. 3.15, si evince che il veicolo percorre il primo tratto di strada accelerando e decelerando diverse volte, mentre in un secondo momento la velocità si stabilizza a circa 18 km/h . Il veicolo non viene spinto, in questo caso, alla sua massima velocità, che corrisponde a 25 km/h , sulla base dei test fatti in precedenza.

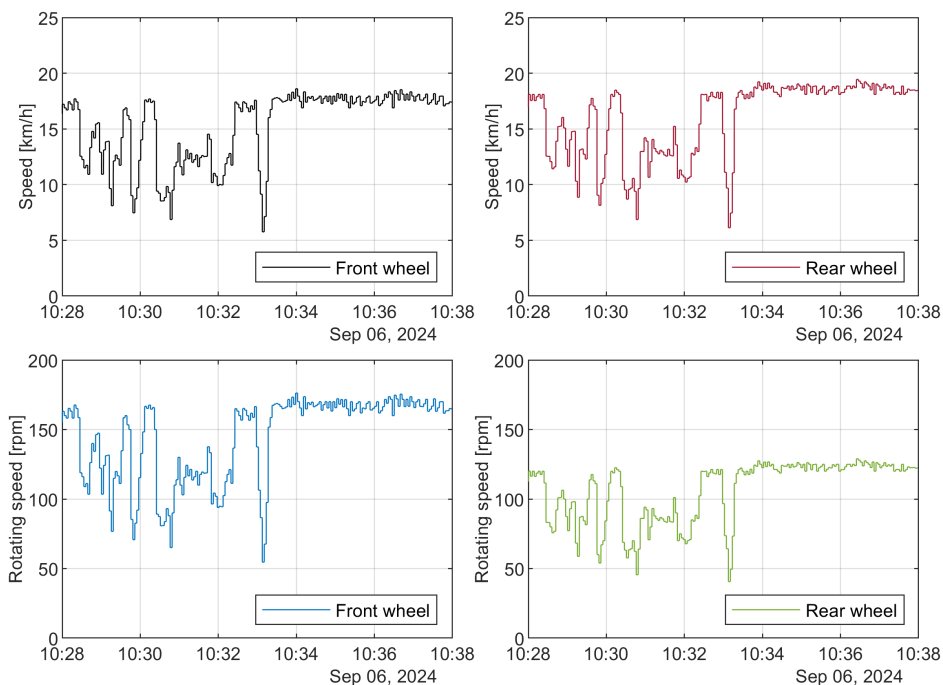


Figura 3.15: Velocità periferica e rotativa alle ruote anteriori e posteriori durante il tragitto su strada.

3.3.3 Pendenza

In questo esperimento sono stati impiegati gli accelerometri, installati sul telaio del veicolo, per valutare le accelerazioni, le vibrazioni e gli angoli di marcia durante il tragitto su strada. Si rappresentano, in Fig. 3.16, i valori delle accelerazioni, opportunamente filtrate a monte e a valle del campionamento, misurate dall'accelerometro posto in cabina. Dalle accelerazioni è stato possibile dedurre le pendenze del terreno lungo la direzione longitudinale X e trasversale Y , raffigurate in Fig. 3.17.

È sorprendente osservare come sia stato possibile evidenziare il *trend* delle due variabili eliminando il rumore e l'effetto delle vibrazioni innescate dal veicolo in marcia.

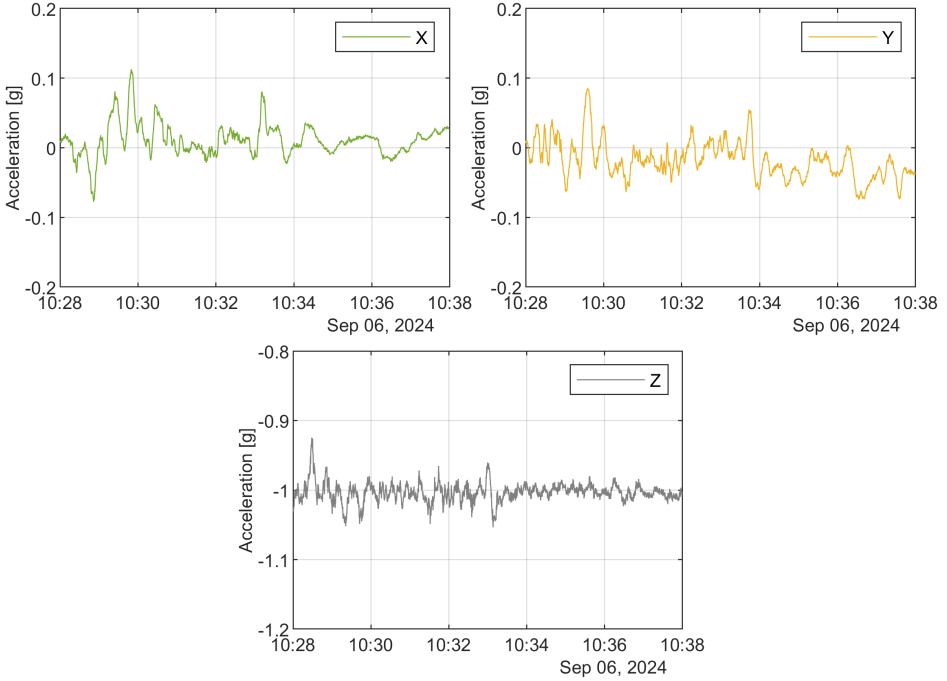


Figura 3.16: Accelerazioni misurate lungo la direzione longitudinale (X), trasversale (Y) e verticale (Z) durante il tragitto su strada.

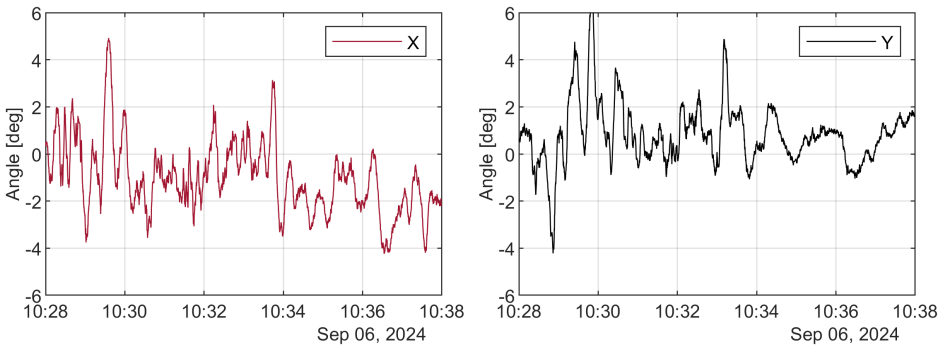


Figura 3.17: Angoli di pendenza misurati attorno all'asse longitudinale (X), quindi lungo la direzione trasversale del veicolo, e attorno all'asse trasversale (Y), quindi lungo la direzione dell'avanzamento, durante il tragitto su strada.

In generale la pendenza della strada lungo la direzione dell'avanzamento, rappresentata dall'angolo attorno all'asse trasversale Y, presenta un andamento molto variabile da angoli di $+6^\circ$ a angoli di -4° che corrispondono a pendenze del $+10\%$ e del -7% rispettivamente.

3.3.4 Parametri motore

I parametri motore, in Fig. 3.18, evidenziano che il numero di giri è stato mantenuto pressoché costante per tutto il percorso a circa 2000 rpm , corrispondenti al 60% della corsa massima dell'acceleratore.

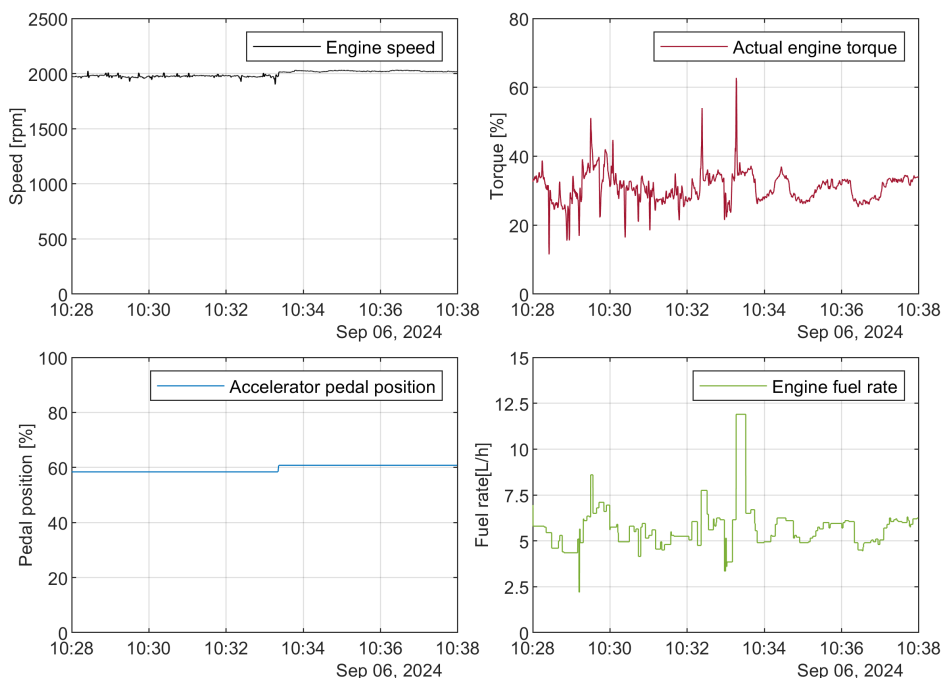


Figura 3.18: Parametri motore durante il tragitto su strada: velocità di rotazione, coppia relativa, posizione del pedale d'accelerazione e portata di combustibile.

Come osservato nella Sez. 3.2, la coppia presenta dei picchi in corrispondenza delle accelerazioni del mezzo e viene influenzata dalla pendenza del terreno. Per osservare meglio questa correlazione si riporta, in Fig. 3.19, il valore della potenza consumata e della pendenza lungo la direzione dell'avanzamento, quindi valutata con l'angolo attorno all'asse trasversale (Y), in quanto è la più determinante per i consumi. È immediatamente chiaro che, in questo intervallo di tempo, quando il

motore gira a velocità costante e risulta costante anche la velocità del mezzo, l'unica variabile a influenzare la potenza consumata è la pendenza attorno all'asse Y. Questa stretta correlazione sottolinea come siano valide le misure effettuate dall'accelerometro, che ha permesso di misurare variazioni accurate di pendenza anche in un range così ridotto da $+2^\circ$ a -1° , e quanto sia stata efficace la filtrazione dei segnali effettuata, che ha permesso di eliminare l'effetto delle vibrazioni.

Il diagramma del consumo di combustibile mostra ovviamente un andamento simile a quello della potenza, con valori influenzati sia dalla velocità di avanzamento che dalla pendenza.

In Tab. 3.4 infine si riportano i valori medi dei parametri motore, quando la macchina opera in transito su strada pianeggiante in regime pressoché stazionario.

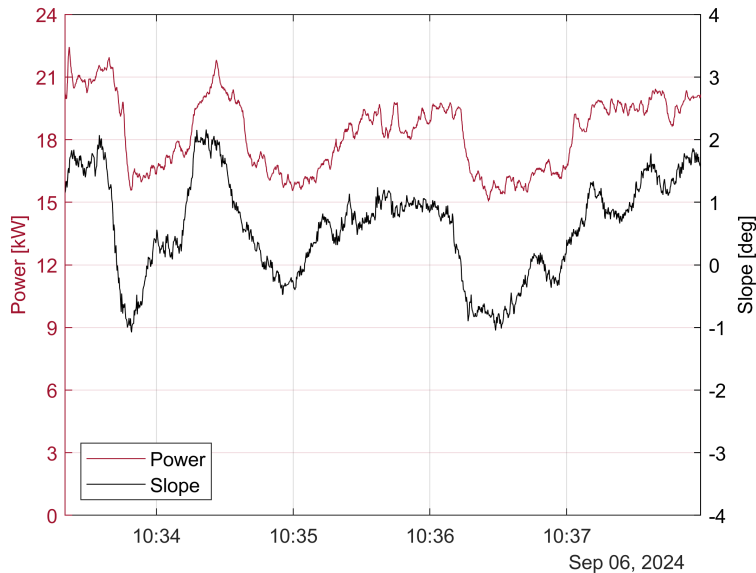


Figura 3.19: Potenza erogata dal motore e pendenza del terreno a confronto durante il tragitto su strada.

3.3.5 Pressioni nel circuito idraulico

In Fig. 3.20 si mostrano le pressioni dell'olio nel circuito di trasmissione e frenatura durante il tragitto su strada.

Si può vedere come la pressione nel circuito frenante rimanga costante e pari a 28 bar leggermente superiore alla pressione di taratura delle

limitatrici.

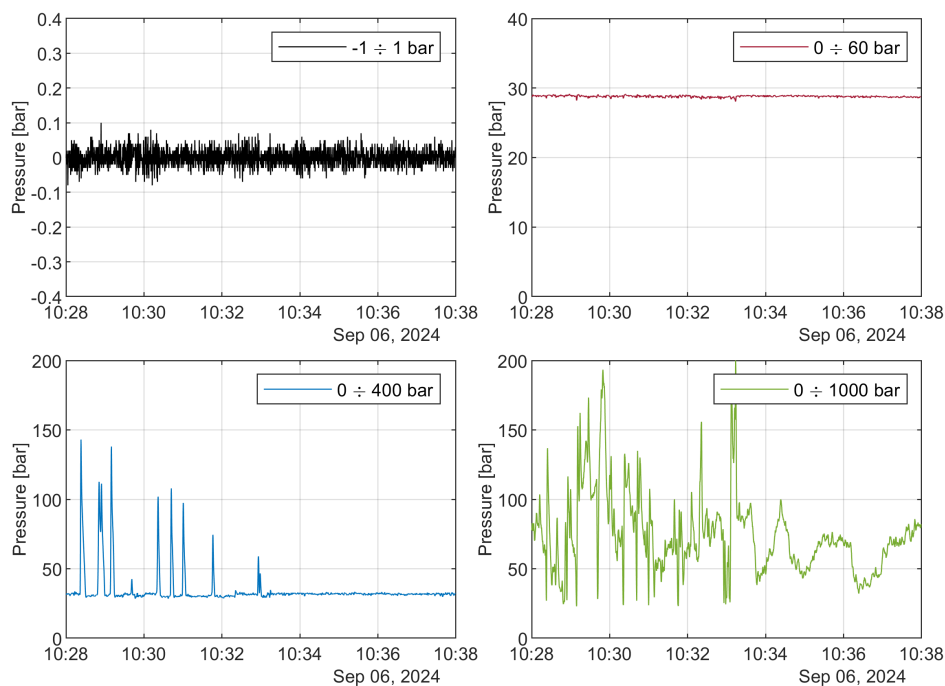


Figura 3.20: Pressioni idrauliche durante il test della massima velocità su strada: all'aspirazione della pompa ($-1 \div +1$ bar), nel circuito frenante ($0 \div 40$ bar), all'ingresso del motore idraulico anteriore ($0 \div 400$ bar), alla mandata della pompa ($0 \div 1000$ bar).

Si possono notare continue oscillazioni della pressione relativa all'aspirazione della pompa, che in media si mantiene prossima allo 0, ovvero, in termini assoluti, pari al valore della pressione atmosferica, con una deviazione standard pari a 0.02 bar.

Per capir meglio da quali fattori dipendono, invece, la pressione di mandata della pompa e la pressione in aspirazione al motore idraulico anteriore conviene osservare la Fig. 3.22 che rappresenta queste grandezze in comparazione alla velocità del veicolo. Quando il veicolo accelera la pressione di mandata della pompa presenta dei picchi notevoli che arrivano anche a 190 bar, per via della presenza delle forze inerziali del veicolo. La pressione in aspirazione al motore idraulico anteriore viceversa aumenta quando il veicolo decelera. Ciò si accompagna al fatto che il salto di pressione durante alcuni istanti della decelerazione del veicolo ai motori idraulici posteriori è positivo e non più negativo.

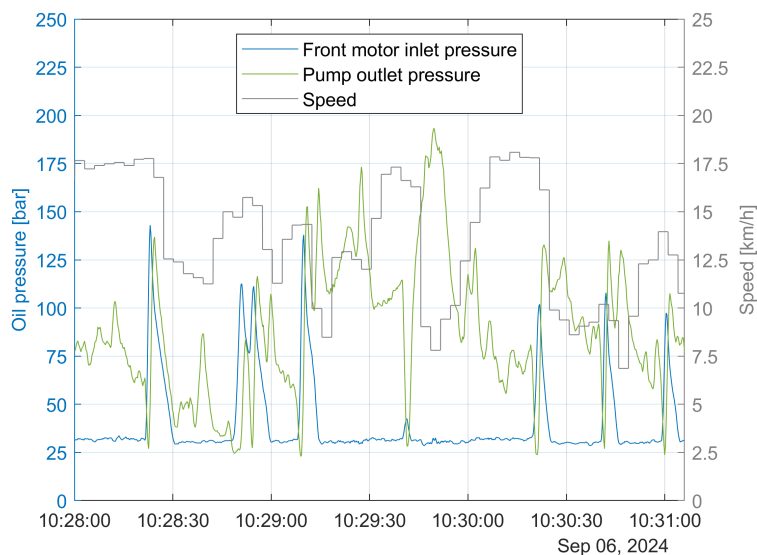


Figura 3.21: Pressione all'ingresso dei motori idraulici posteriori e anteriore a confronto con la velocità di avanzamento durante il tragitto su strada.

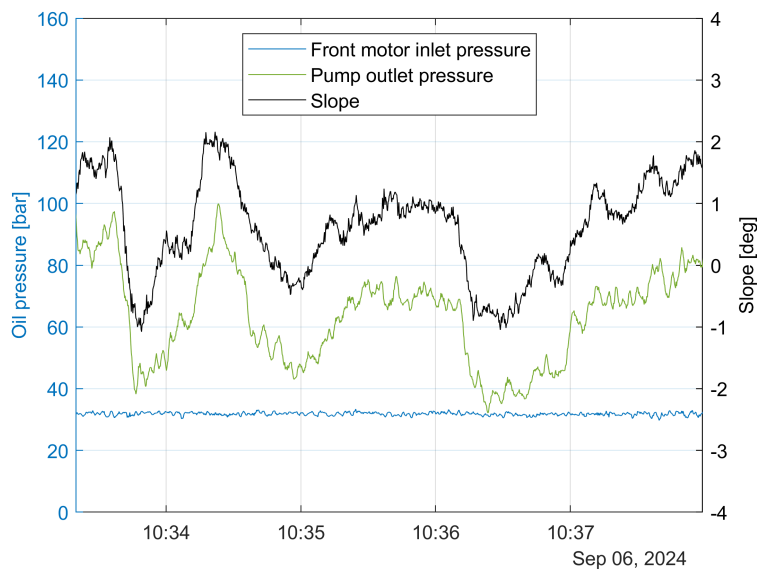


Figura 3.22: Pressione all'ingresso dei motori idraulici posteriori e anteriore a confronto con la pendenza della strada lungo la direzione dell'avanzamento durante il tragitto.

In altri termini i motori si comportano come "pompe", agendo come se fossero un freno alla velocità della macchina.

In secondo luogo, consultando la Fig. 3.22, si nota l’influenza della pendenza sulla pressione di mandata della pompa. Infatti la pressione di mandata della pompa incrementa proporzionalmente alla coppia resistente alle ruote, che risulta maggiore per pendenze positive.

Infine si riportano, in Tab. 3.4 i valori medi delle pressioni di cui sopra, quando la macchina opera in transito su strada pianeggiante in regime pressoché stazionario.

3.3.6 Temperature dell’olio idraulico

Si rappresenta ora, in Fig. 3.23, l’andamento delle temperature dell’olio in ingresso e in uscita dal radiatore durante il tragitto su strada.

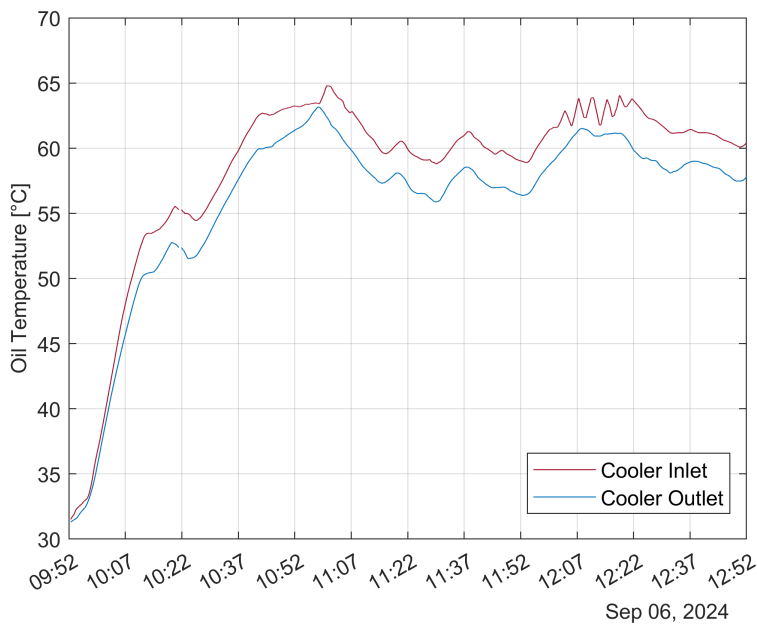


Figura 3.23: Temperature dell’olio idraulico all’ingresso e all’uscita del radiatore durante il tragitto su strada.

È opportuno graficare le misure durante un intervallo di tempo più ampio, che ricopre ben 3 ore di funzionamento della macchina, al fine di evidenziare, per intero, il *trend* di crescita della temperatura.

Le temperature inizialmente si equivalgono e risultano prossime alla

temperatura ambientale campionata dai sensori sul motore. Si presuppone infatti che l'olio sia in equilibrio termico con l'ambiente. Una volta iniziata la marcia si registra un'impennata della temperatura, che giunge a regime dopo circa 1h di viaggio. La differenza tra le temperature in ingresso e uscita si amplifica quando l'olio si scalda e giunge anch'essa ad un valore di regime pari a 5°C . L'olio raggiunge all'interno del circuito a regime temperature comprese tra i 60° e i 65°C .

3.3.7 Parametri di lavoro tipici su strada

I test su strada hanno permesso di individuare i valori medi, in Tab. 3.4, delle variabili di maggior interesse quando la macchina transita su strada asfaltata grossomodo pianeggiante, in regime stazionario, a velocità e giri costanti.

<i>Condizioni di guida</i>		
n [rpm]	v [km/h]	θ_Y [deg]
2043 ± 3	18.4 ± 0.2	0.158 ± 0.435

<i>Potenza e consumo di combustibile</i>	
P [kW]	F_c [L/h]
2043 ± 3	18.4 ± 0.2

<i>Pressioni nel circuito idraulico</i>	
p₄₀₀ [bar]	p₁₀₀₀ [bar]
2043 ± 3	18.4 ± 0.2

Tabella 3.3: Consumi di potenza P e di combustibile F_c del veicolo e pressioni nel circuito idraulico di trasmissione alla mandata della pompa p_{1000} e all'ingresso dei motori idraulici posteriori p_{400} su strada in condizioni pressoché stazionarie di giri n , velocità v e pendenza θ_Y .

3.4 Test in campo: primo giorno di raccolta

I risultati delle prove in campo sono senza dubbio i più importanti, in quanto la macchina esplica tutte le funzioni per cui è stata concepita.

In questo test la semovente è stata monitorata durante il primo giorno di raccolta del 2024. Il campo di lavoro, situato a Bomarzo (VT), caratterizzato da un terreno inerbito e asciutto, presenta una topografia prevalentemente collinare, con una modesta variabilità di pendenza.

Durante la raccolta la macchina transita sui filari della coltivazione con tutti gli organi di raccolta e cernita in funzione (raccogliatore meccanico, ventilatori, crivello). Il veicolo compie un percorso composto da tratti rettilinei seguiti da manovre di sterzo a fine campo per intraprendere il filare successivo. In questo frangente, tutte le variabili campionate, ad inclusione della portata d'aria dei ventilatori, meritano interesse.

Per semplicità si mostra l'andamento dei parametri durante 10 minuti di attività, tra le 14:20 e le 14:30.

3.4.1 Posizione

In Fig. 3.24 si mostrano le coordinate in mappa del veicolo salvate dal sistema di acquisizione della posizione nella prima mattina di raccolta, dalle 9:40 alle 10:40.

Si può individuare facilmente la porzione di campo in cui la macchina ha effettuato la raccolta, informazione utile per avere un'idea dell'area lavorata in un certo intervallo di tempo. Tuttavia, l'inaccuratezza delle coordinate, dovuta a una bassa intensità del segnale GNSS, non hanno permesso di individuare bene i tragitti tra i filari, quantomeno di calcolare la velocità del mezzo con una sufficiente precisione.

È esplicativa la Fig. 3.25, che rappresenta le coordinate della macchina in mappa in comparazione al tragitto effettivamente percorso, individuato dalle frecce. Le posizioni campionate non permettono di individuare sempre con precisione il filare su cui lavora la macchina: sembra che la macchina salti da un filare all'altro, soprattutto nella porzione inferiore del campo, in cui probabilmente il segnale risultava più debole.

A maggior ragione i valori della velocità del veicolo calcolati basandosi sui dati GNSS risultano errati, perciò non saranno nemmeno considerati.



Figura 3.24: Posizioni registrate della macchina, nel primo giorno di raccolta, dalle 9:40 alle 10:40, colorate in funzione del tempo di acquisizione.



Figura 3.25: Posizioni registrate della macchina, nel primo giorno di raccolta, dalle 10:12 alle 10:24, comparate al tragitto realmente percorso.

3.4.2 Velocità

È importante conoscere la variabile che maggiormente inficia la capacità oraria di raccolta: la velocità di avanzamento, che viene riportata, insieme alle velocità di rotazione delle ruote, in Fig. 3.26.

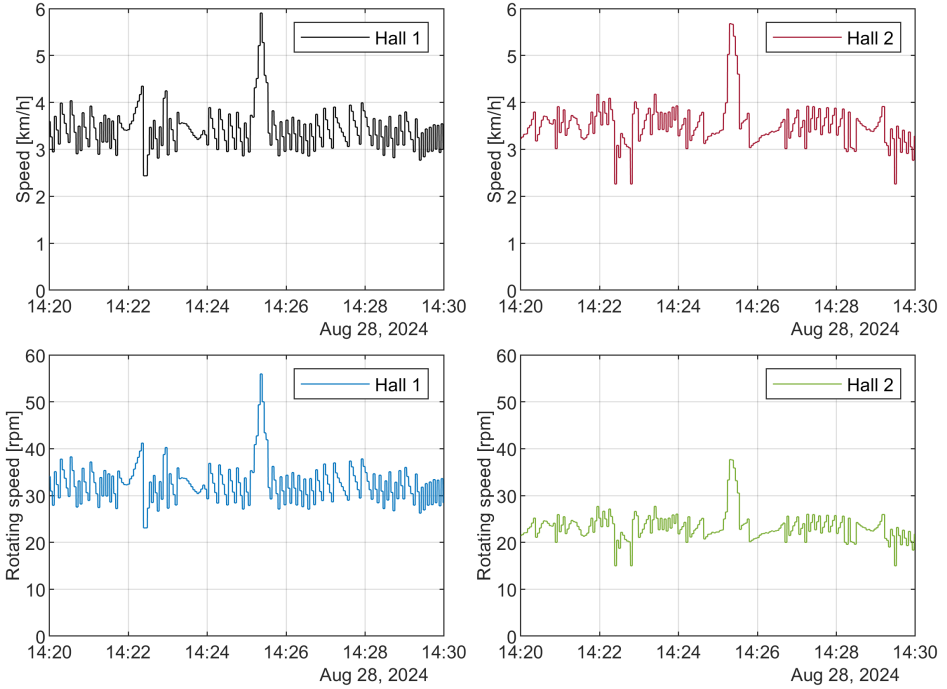


Figura 3.26: Velocità periferica e rotativa alle ruote anteriori e posteriori durante una tratta del primo giorno di raccolta.

In questo frangente la velocità di avanzamento si mantiene grossomodo costante ed è caratterizzata da un valor medio di $3,5 \pm 0,4 \text{ km/h}$.

Ancora una volta si vuole sottolineare che, se la velocità periferica risulta la medesima tra ruote anteriori e posteriori, a meno degli errori di misura e degli slittamenti, la velocità di rotazione della ruota anteriore è sicuramente maggiore delle ruote posteriori in misura del rapporto tra i raggi delle ruote.

3.4.3 Parametri motore

Osservando i parametri motore durante i 10 minuti esemplificativi della raccolta, si nota che la macchina è mantenuta in media a $1757 \pm 5.4 \text{ rpm}$, corrispondente al 48.4% della corsa massima

dell'acceleratore.

Coppia a consumi presentano un andamento qualitativo analogo, che come vedremo è influenzato dalla pendenza del terreno, con una media di consumo di combustibile pari a $10.3 \pm 0.8 L/h$.

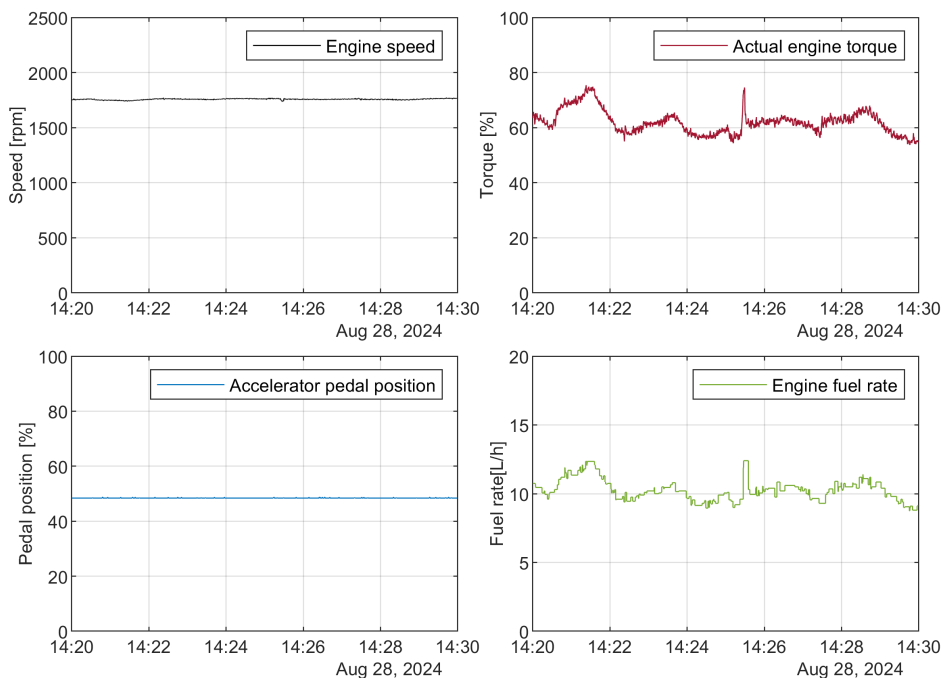


Figura 3.27: Parametri motore durante una tratta del primo giorno di raccolta: velocità di rotazione, coppia relativa, posizione del pedale d'accelerazione e portata di combustibile.

3.4.4 Pendenza

In Fig. 3.28 sono raffigurate le accelerazioni lungo i tre assi dell'accelerometro, corrispondenti agli assi principali del veicolo: l'asse longitudinale lungo la direzione dell'avanzamento X, l'asse trasversale Y e l'asse verticale Z.

Dalle accelerazioni derivano le pendenze del terreno calcolate come angoli rispetto all'asse longitudinale del veicolo X, quindi lungo la direzione trasversale e rispetto all'asse trasversale Y, quindi lungo la direzione longitudinale. La pendenza del terreno presenta un *trend* molto variabile, con valori oscillanti tra $+10^\circ$ e -10° lungo la direzione dell'avanzamento corrispondenti a pendenze del 18% in modulo.

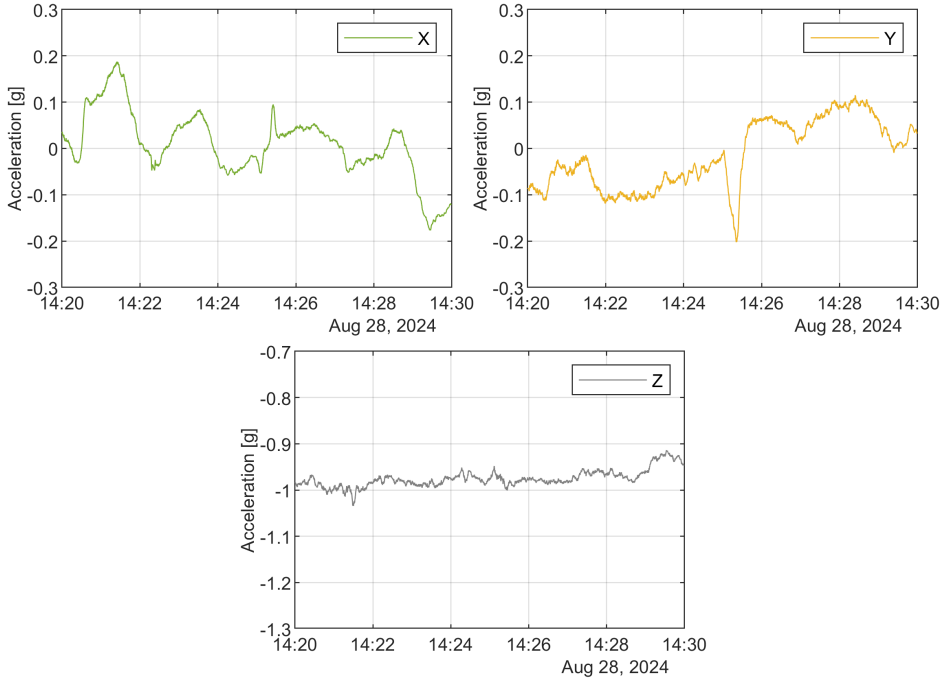


Figura 3.28: Accelerazioni misurate lungo la direzione longitudinale (X), trasversale (Y) e verticale (Z), durante una tratta del primo giorno di raccolta.

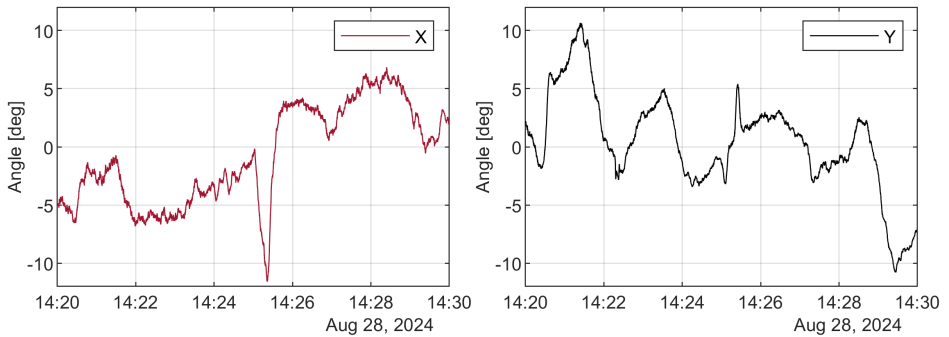


Figura 3.29: Angoli di pendenza misurati attorno all'asse longitudinale (X), quindi lungo la direzione trasversale del veicolo, e attorno all'asse trasversale (Y), quindi lungo la direzione dell'avanzamento, durante una tratta del primo giorno di raccolta.

Se si mettono a confronto la potenza consumata con la pendenza attorno all'asse Y, in Fig. 3.30, è facile individuare una stretta

proporzionalità tra le due grandezze. La potenza oscilla fra circa 28 e 38 kW . Su questa base è stato possibile effettuare una stima dell'incremento di consumo di potenza dovuto all'aumento di un punto percentuale di pendenza, che risulta pari a $0.205 \pm 0.053 kW$.

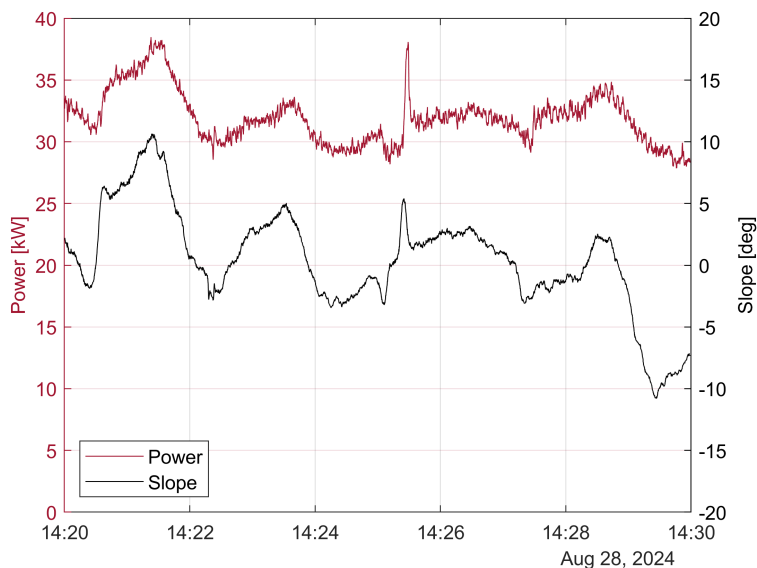


Figura 3.30: Potenza erogata dal motore e pendenza del terreno a confronto durante una tratta del primo giorno di raccolta.

3.4.5 Pressioni nel circuito idraulico

Osservando la Fig. 3.31, possiamo subito notare che le oscillazioni di pressione all'ingresso della pompa, quando è attivo sia il sistema idraulico dei servizi, sia il sistema d'aspirazione, risultano molto più marcate: si vede una banda di valori compresi tra i 0.25 e i $-0.2 bar$ di pressione relativa.

La pressione nel circuito frenante rimane costante, pari a 28 bar , in quanto i giri rimangono costanti.

La pressione in uscita dalla pompa presenta un andamento analogo a quello della coppia e della pendenza.

Infine la pressione in aspirazione al motore idraulico anteriore presenta un andamento grossomodo costante, con alcuni incrementi quando le pendenze assumono valori negativi.

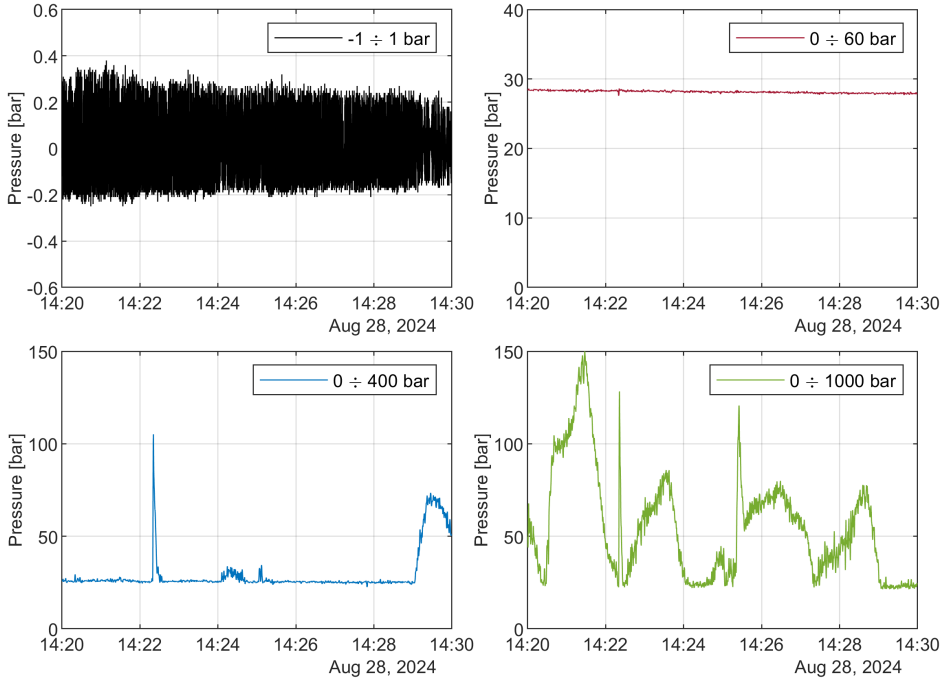


Figura 3.31: Pressioni idrauliche durante una tratta del primo giorno di raccolta: all'aspirazione della pompa ($-1 \div +1$ bar), nel circuito frenante ($0 \div 40$ bar), all'ingresso del motore idraulico anteriore ($0 \div 400$ bar), alla mandata della pompa ($0 \div 1000$ bar).

3.4.6 Prestazioni dei ventilatori

In Fig. 3.32 è possibile osservare l'andamento delle pressioni statiche misurate nei condotti di aspirazione e insaccamento del prodotto e le portate d'aria nei condotti, calcolate conoscendo le curve caratteristiche delle perdite di carico nel circuito.

Essendo i giri costanti, anche le pressioni e le portate d'aria seguono lo stesso tipo di andamento, in quando i ventilatori operano con un rapporto di moltiplicazione fisso rispetto ai giri motore.

In media si riscontra una portata di $1.41 \pm 0.04 m^3/s$ nei condotti di aspirazione e di $0.461 \pm 0.010 Pa$ nei condotti di insaccamento. Questi sono valori assolutamente compatibili con quelli misurati nei test della Sez. 3.1.

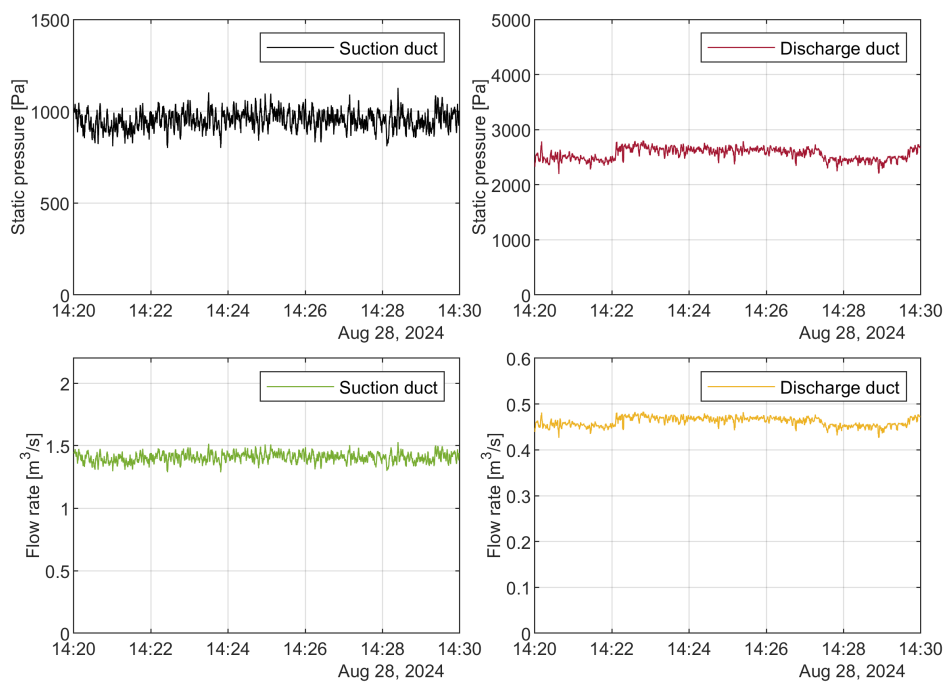


Figura 3.32: Prestazioni dei ventilatori durante una tratta del primo giorno di raccolta: pressione statica e portata d'aria nei condotti di aspirazione e insaccamento.

3.5 Test in campo: secondo giorno di raccolta

A conferma dei dati precedenti è utile riportare anche alcuni dati campionati dai sensori durante il secondo giorno di raccolta, il 29/08/2024. La macchina ha lavorato sempre sullo stesso terreno, rappresentato in Fig. 3.24. Nello specifico si analizzano la maggior parte dei dati raccolti durante i 12 minuti di attività trascorsi dalle 17:42 alle 17:54. Anche in questo intervallo temporale la macchina transitava sui filari della coltivazione a velocità di avanzamento e giri motore piuttosto costanti, con tutti gli organi di raccolta e cernita in funzione (raccoglitore meccanico, ventilatori, crivello).

3.5.1 Condizioni ambientali

In Fig. 3.33 si riporta il decorso della temperatura dell'aria ambientale e la pressione barometrica campionate dai sensori collocati sul motore, stavolta durante tutta la giornata di raccolta, dalle 08:00 alle 18:00.

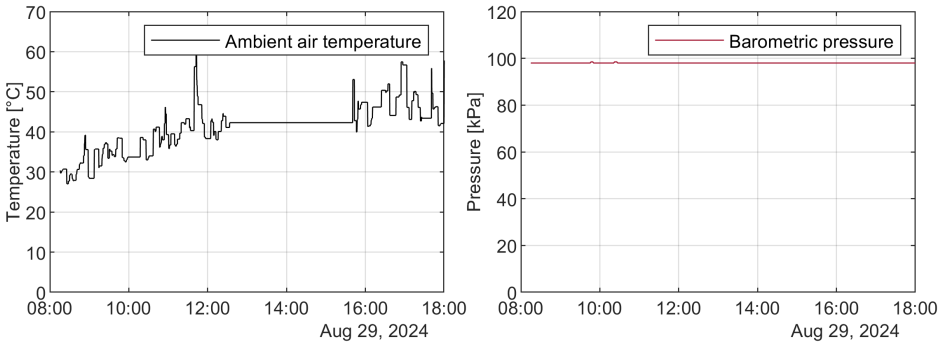


Figura 3.33: Condizioni ambientali durante una tratta del secondo giorno di raccolta: temperatura e pressione barometrica.

Si può notare come il riscaldamento del motore possa alterare le temperature circostanti la zona di campionamento del sensore, che raggiungono in alcuni momenti valori di $50 \div 60^{\circ}\text{C}$.

La pressione barometrica, dipendendo principalmente dall'altitudine, non subisce variazioni e si mantiene pari a 98 kPa .

3.5.2 Velocità

Osservando la Fig. 3.34, si nota che, anche in questo caso, la velocità del mezzo viene mantenuta costante con valori che non si discostano affatto da quelli già visti nella Sez. 3.4.

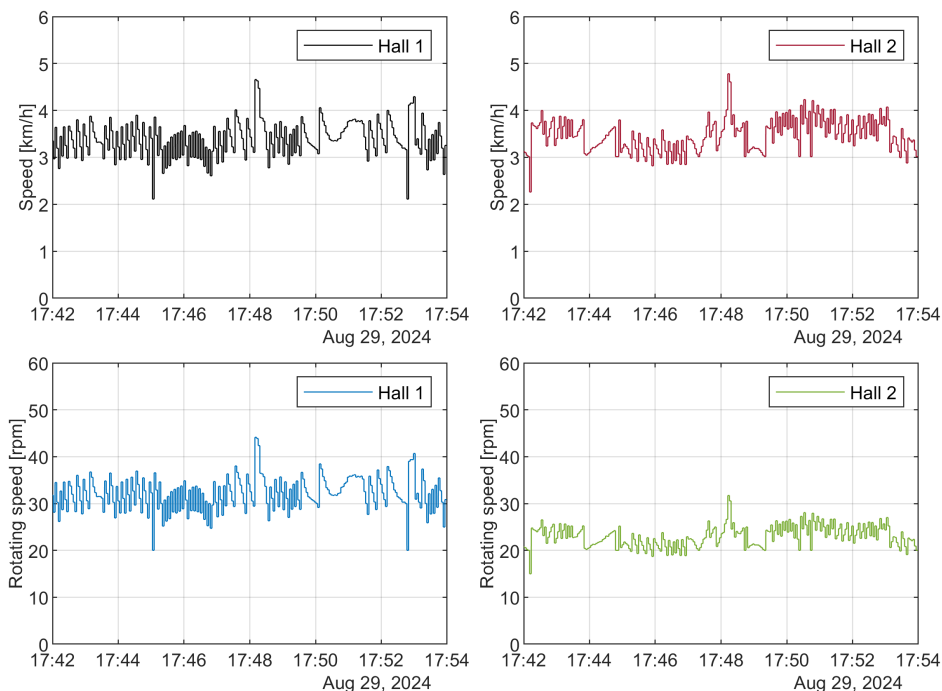


Figura 3.34: Velocità periferica e rotativa alle ruote anteriori e posteriori durante una tratta del secondo giorno di raccolta.

3.5.3 Parametri motore

Si riportano in Fig. 3.35 i parametri motore. La piccola differenza che si nota, rispetto al caso precedente, è la velocità di rotazione del motore, che stavolta viene mantenuto a un regime leggermente più basso di circa 50 rpm rispetto al caso del primo giorno di raccolta analizzato nella Sez.3.4. Probabilmente perché l'operatore ha preso coscienza che le capacità di aspirazione della macchina sono sufficienti a questo regime (1700 rpm), che infatti risulta quello raccomandato dalla ditta costruttrice. Riguardo la coppia e i consumi possono essere effettuate le stesse considerazioni fatte in Sez. 3.1.

Un interessante studio che si potrebbe fare è considerare, nel bilancio

energetico, il consumo associato al sistema di aspirazione, dedotto nella Sez. 3.1, per calcolare il consumo associato al solo sistema di avanzamento e servizi ausiliari, quali spazzole e sistema di raccolta. Dal fit eseguito in Fig. 3.7 si può dedurre che, al regime di giri considerato, la potenza consumata dai ventilatori risulta pari a 25.48 kW .

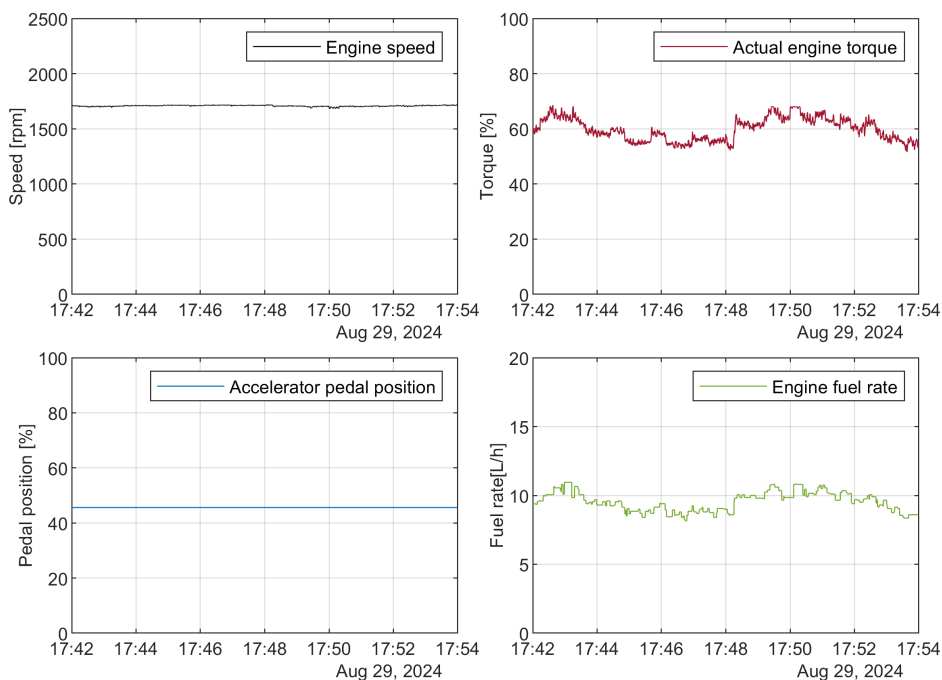


Figura 3.35: Parametri motore durante una tratta del secondo giorno di raccolta: velocità di rotazione, coppia relativa, posizione del pedale d'accelerazione e portata di combustibile.

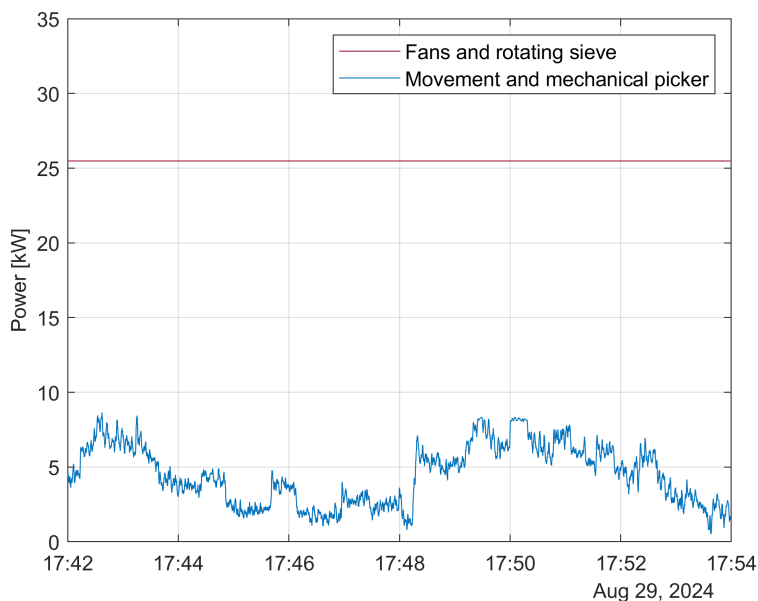


Figura 3.36: Potenza consumata dall'avanzamento e dai ventilatori a confronto durante una tratta del secondo giorno di raccolta.

Se si assume che questa potenza rimanga costante e si sottrae alla potenza complessivamente erogata si trova, in Fig. 3.36, la potenza richiesta per l'avanzamento e per l'utilizzo dei servizi (spazzole e sistema di raccolta nel raccoglitore meccanico).

La potenza media consumata per l'avanzamento e l'azionamento dei servizi risulta pari a $4.49 \pm 1.97 kW$, che costituisce solamente il 15% della potenza complessiva. La restante parte dei consumi, ben l'85%, è dovuta all'azionamento dei ventilatori.

Il miglioramento dei ventilatori potrebbe costituire, quindi, il punto di partenza più solido per l'efficientamento energetico della macchina.

3.5.4 Pendenza

Analizzando l'andamento della pendenza del terreno lungo la direzione dell'avanzamento, che in Fig. 3.29 viene rappresentata come angolo rispetto all'asse Y, si noti come la parte lavorata in campo presentava un terreno meno scosceso, ma più pianeggiante rispetto al caso in Sez. 3.4, in quanto stavolta i minimi e i massimi angoli passano da 6° a -6° rispettivamente.

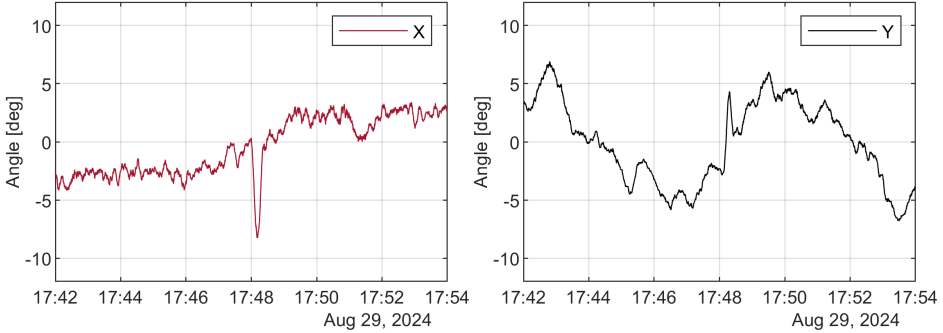


Figura 3.37: Angoli di pendenza misurati attorno all'asse longitudinale (X), quindi lungo la direzione trasversale del veicolo, e attorno all'asse trasversale (Y), quindi lungo la direzione dell'avanzamento, durante una tratta del primo giorno di raccolta.

3.5.5 Pressioni nel circuito idraulico

Osservando la Fig. 3.38, ancora una volta la pressione relativa all'ingresso della pompa presenta oscillazioni attorno allo 0 sia in positivo che in negativo, con una deviazione standard pari a 0.206 bar.

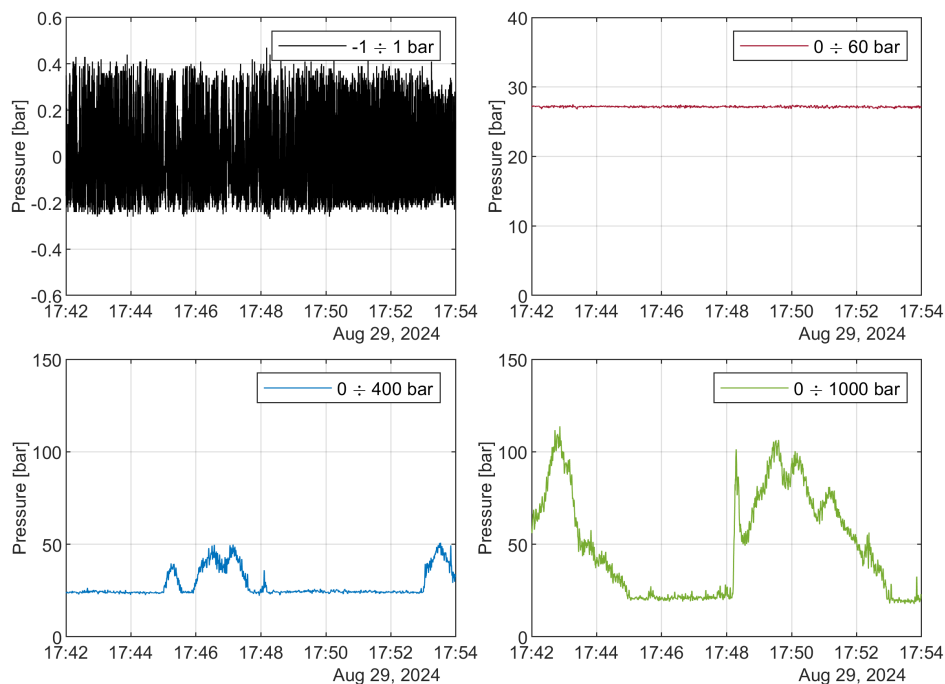


Figura 3.38: Pressioni idrauliche durante una tratta del secondo giorno di raccolta: all'aspirazione della pompa ($-1 \div +1$ bar), nel circuito frenante ($0 \div 40$ bar), all'ingresso del motore idraulico anteriore ($0 \div 400$ bar), alla mandata della pompa ($0 \div 1000$ bar).

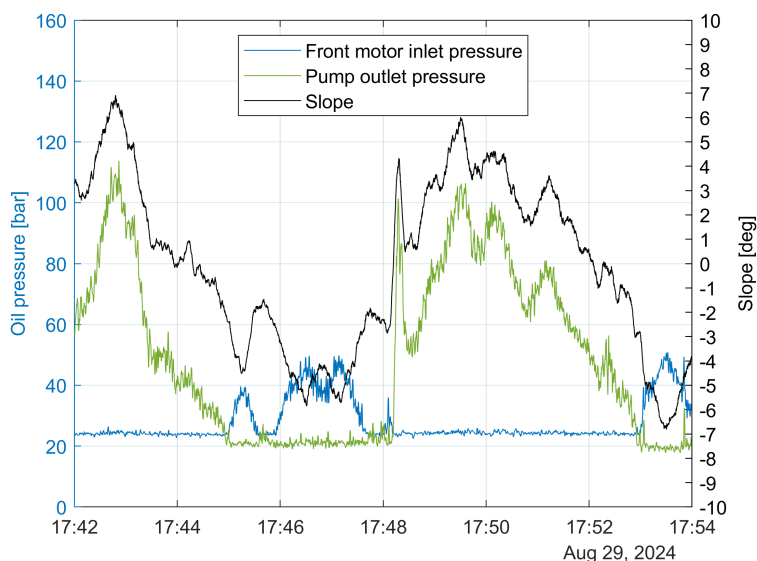


Figura 3.39: Pressioni nel circuito idraulico e pendenza del terreno a confronto durante una tratta del secondo giorno di raccolta.

Per trarre ulteriori informazioni utili dall'andamento delle pressioni si riporta l'andamento di queste all'ingresso dei motori idraulici posteriori e anteriore in sovrapposizione all'andamento della pendenza.

Come si può notare dal confronto, quando la pendenza scende al disotto dei -3° , la pressione di mandata assume un valore costante di circa 20 *bar*. L'andamento della pressione nel resto del transitorio risulta proporzionale all'angolo di pendenza. Proprio al disotto della stessa soglia dei -3° di pendenza si registra un aumento della pressione all'ingresso del motore idraulico anteriore in misura proporzionale al calo delle pendenza. Ciò vuol dire che in queste condizioni i motori idraulici posteriori presentano un salto di pressione positivo e si comportano come "pompe", agendo come un freno all'avanzamento del veicolo.

È stato calcolato l'incremento medio della pressione di mandata dovuto all'aumento di un punto percentuale della pendenza, che risulta pari a $3.82 \pm 1.98 \text{ bar}$.

3.5.6 Prestazioni dei ventilatori

In Fig. 3.40 è possibile vedere l'andamento delle pressioni statiche e delle portate d'aria misurate nei condotti di aspirazione e insaccamento del prodotto. Anche adesso, la costanza delle pressioni è sancita dalla costanza del numero di giri motore, che però, risultando leggermente più basso, giustifica un piccolo calo delle portate misurate rispetto a quelli della Sez.3.4.

Si ribadisce che rimane una netta corrispondenza tra i valori misurati e quelli deducibili dal grafico in Fig. 3.5.

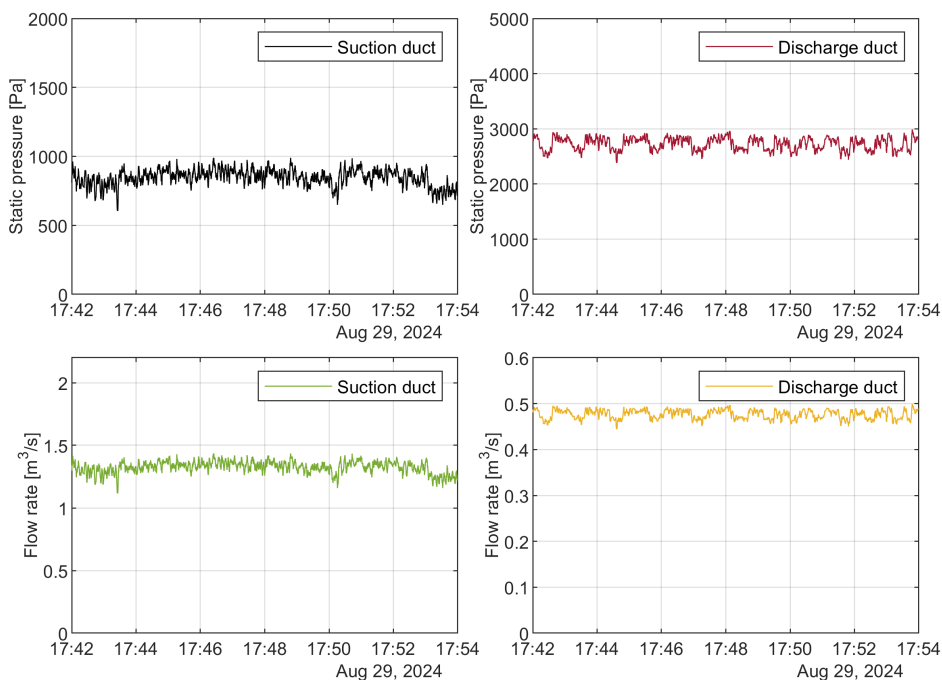


Figura 3.40: Prestazioni dei ventilatori durante una tratta del secondo giorno di raccolta: pressione statica e portata d'aria nei condotti di aspirazione e insaccamento.

3.5.7 Temperature dell'olio idraulico

Si mostrano ora gli andamenti delle temperature dell'olio in ingresso e in uscita dal radiatore durante la seconda giornata di raccolta (Fig. 3.41). La temperatura è stata graficata in un intervallo di tempo più ampio, corrispondente all'intera mattinata di raccolta, per riuscire ad apprezzare in modo adeguato il suo decorso.

Anche in questo caso, le temperature inizialmente risultano allineate alla temperatura ambientale campionata dai sensori sul motore, in quanto si presuppone che l'olio sia in equilibrio termico con l'ambiente.

Una volta incominciata la raccolta si registra un incremento delle temperature più moderato rispetto a quello che si registra quando la macchina è in viaggio a maggiori velocità su strada (Sez. 3.3). Infatti, se durante il transito su strada l'olio raggiungeva i 60°C dopo soli 45 min , durante la raccolta l'olio impiega ben 65 min a superare questo valore. Questo a causa delle differenti portate d'olio nel circuito, che, ad alte velocità del mezzo, comportano maggiori perdite di carico e dissipazioni

termiche.

Si può notare che la macchina è stata tenuta spenta dalle 10:00 alle 10:15.

La differenza tra le temperature campionate in ingresso e in uscita al radiatore, anche ora, sembra aggirarsi intorno ai 5°C .

In generale l'olio raggiunge all'interno del circuito a regime temperature comprese tra i 60° e i 65°C .

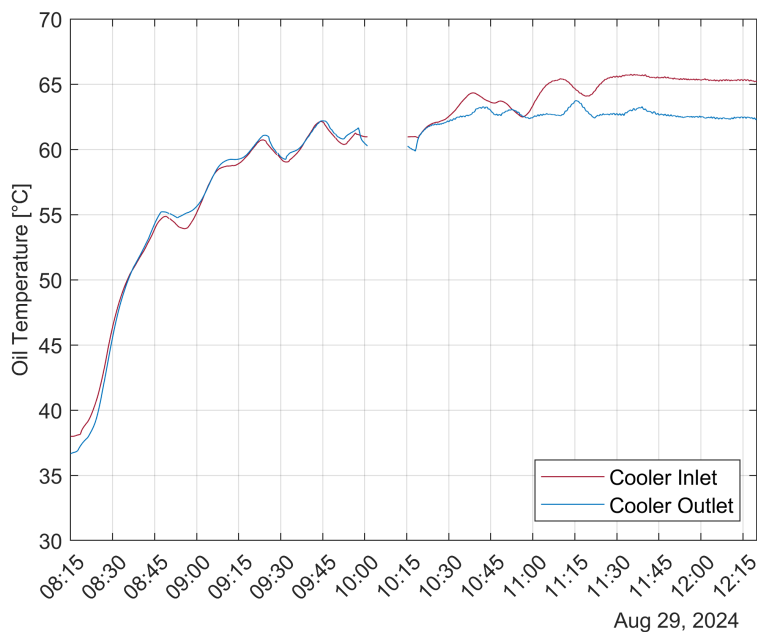


Figura 3.41: Temperature dell'olio idraulico all'ingresso e all'uscita del radiatore durante la seconda giornata di raccolta.

3.6 Parametri tipici durante la raccolta

I test su campo hanno permesso di individuare i valori medi, in Tab. 3.4, delle variabili di maggior interesse quando la macchina è in raccolta su un terreno in media pianeggiante con modesta variabilità di pendenza, in regime stazionario a velocità e giri costanti.

<i>Condizioni di guida</i>			
n [rpm]	v [km/h]	θ_Y [deg]	
1708 ± 6	3.43 ± 0.29	-0.05 ± 3.52	

<i>Potenza e consumo di combustibile</i>			
P [kW]	P_f [kW]	P_m [kW]	F_c [L/h]
30.0 ± 2.0	25.5 ± 0.4	4.49 ± 1.97	9.54 ± 0.70

<i>Pressioni nel circuito idraulico</i>	
p_{400} [bar]	p_{1000} [bar]
27.2 ± 0.1	49.7 ± 27.8

<i>Prestazioni dei ventilatori</i>			
Q_s [m ³ /s]	Q_d [m ³ /s]	v_s [m/s]	v_d [m/s]
1.32 ± 0.05	0.476 ± 0.010	27.0 ± 1.0	23.7 ± 0.5

Tabella 3.4: Parametri medi operativi della macchina in raccolta su terreno agricolo in condizioni pressoché stazionarie di giri n , velocità v e pendenza θ_Y : consumi di potenza complessivi P associati ai ventilatori P_f e al moto e servizi ausiliari P_m ; consumi di combustibile F_c ; pressioni nel circuito idraulico di trasmissione alla mandata della pompa p_{1000} e all'ingresso dei motori idraulici posteriori p_{400} ; prestazioni dei ventilatori in termini di portate erogate per aspirare il prodotto Q_s e insaccarlo Q_d ; velocità dell'aria media nei condotti di aspirazione v_s e insaccamento v_d .

Capitolo 4

Conclusioni

Questo studio ha permesso di monitorare le prestazioni di una semovente raccoglitrice di nocciole, la Cimina 300S, prodotta da FACMA, attraverso l'implementazione di un sistema di acquisizione dati, basato su sensori e microcontrollori a basso costo. I risultati dei campionamenti hanno fornito utilissime informazioni sui parametri di funzionamento della macchina, che fino ad oggi risultavano sconosciuti: pressioni nel circuito idraulico di trasmissione e frenatura, temperature dell'olio, portate d'aria nei condotti di ventilazione, potenza, coppia e consumi di combustibile. Inoltre è stata descritta l'influenza che le condizioni di impiego della macchina, come giri motore, velocità di avanzamento, e pendenza del terreno hanno sulle altre variabili campionate.

Lo studio corrente ha permesso di:

- stabilire quali siano le migliori condizioni di lavoro per massimizzare l'efficienza delle operazioni di raccolta e minimizzare i costi e gli sprechi associati;
- fornire dati operativi sul funzionamento dei vari sistemi, come impianto di ventilazione e trasmissione idrostatica, utili per pianificare un miglioramento tecnologico o un efficientamento del mezzo;
- trovare gli strumenti, i sensori e i metodi di misura più efficaci da implementare al fine di mettere a punto un sistema per la telemetria installato di serie sui veicoli.

L'approccio *low cost* adottato per la progettazione dell'impianto di acquisizione dati ha dimostrato che è possibile ottenere risultati significativi anche con risorse economiche limitate, rendendo il monitoraggio delle prestazioni dei veicoli agricoli accessibile anche per piccole e medie aziende.

La conclusione di questo progetto apre la strada a numerosi sviluppi futuri, che vengono elencati di seguito.

- Anzitutto si potrebbero integrare altri sensori nel sistema di acquisizione dati per estrapolare ulteriori informazioni utili come:
 - un sensore di pressione dell'olio per rilevare la pressione nella camera di controllo della cilindrata della pompa, utile per determinare la cilindrata stessa;
 - i sensori di portata dell'olio idraulico, che, anche se non rispettano propriamente il vincolo *low cost*, installati nel circuito di trasmissione, sarebbero utili per monitorare l'efficienza delle macchine idrauliche e dell'intera trasmissione;
 - sensori per monitorare la velocità di avanzamento, così da poter stimare lo slittamento delle ruote;
 - sensori per caratterizzare le condizioni di sicurezza e comfort del conducente come esposizione alle vibrazioni, alle polveri sottili e al rumore.
- Ulteriori test potrebbero essere effettuati sulla macchina per caratterizzarne il funzionamento in condizioni differenti. Per esempio, sarebbe interessante monitorare la raccolta in un terreno con diverse proprietà fisiche, per vedere come l'umidità, la durezza, la tessitura e il livello di inerbimento possano influenzare le prestazioni della semovente.
- Sebbene si tratti sempre di DAQ, meritano un trattamento a parte i sistemi di mappatura delle produzioni. Questi sistemi sono utilizzati nell'agricoltura di precisione per mappare le proprie produzioni agricole, misurando le quantità di prodotto raccolto e geolocalizzando l'informazione. Attualmente due studi sono stati effettuati per valutare i migliori metodi di misura per quantificare le nocciole raccolte [5] [22]. Tuttavia, deve ancora avvenire l'implementazione dei sistemi di mappatura sulle semoventi, che costituiranno parte integrante dei sistemi di acquisizione dati.

Elenco delle figure

1.1	Raccoglitrice semovente FACMA Cimina 300S, munita di raccoglitore meccanico RM170 e cabina di guida.	10
1.2	Modello 3D della raccoglitrice.	12
1.3	Sezione parziale del modello 3D della raccoglitrice.	13
1.4	Schema di trasmissione per l'avanzamento e per gli organi di raccolta e cernita della Cimina 300S.	16
1.5	Componenti fondamentali di un sistema d'acquisizione dati per le macchine agricole.	19
1.6	Struttura più dettagliata di un sistema d'acquisizione dati per le macchine agricole.	20
1.7	Il popolare modello di scheda digitale a microcontrollore Arduino Uno R3.	28
2.1	Posizione di alcuni sensori in vista laterale destra.	33
2.2	Posizione di alcuni sensori in vista laterale sinistra.	34
2.3	AT105 Sonda ad avvitamento Pt100.	35
2.4	Trasmettitore di pressione AP031.	37
2.5	Sensori di pressione differenziale dell'aria: a sinistra il modello MPX10DP, a destra il modello MPXV7002DP. . .	38
2.6	Sensore induttivo AI908.	40
2.7	Accelerometro e Giroscopio GY-521 MPU-6050.	41
2.8	SparkFun Modulo GPS NEO-M9N con un antenna magnetica. .	43
2.9	RTC DS3231.	44
2.10	Lettore SD SPI.	45
2.11	Display OLED I2C.	46
2.12	Multiplexer TCA9548A.	47
2.13	SparkFun CAN-Bus Shield.	48
2.14	Arduino Mega 2560 Rev3.	49

2.15	Arduino UNO R4 Minima.	50
2.16	SparkFun RedBoard.	51
2.17	Alcuni sensori impiegati nel sistema di acquisizione: trasmettitori di pressione, sensori di pressione differenziale, pt100, sensore Hall e accelerometro.	54
2.18	Sensori di pressione differenziale dell'aria muniti di cavi e connettori, pronti per l'installazione sulla semovente.	54
2.19	Sensore di temperatura Pt100, a sinistra, e modulo accelerometro/giroscopio GY-521, a destra, muniti di cavi e connettori, pronti per l'installazione sulla semovente.	55
2.20	Trasmettitori di pressione dell'olio muniti di cavi e connettori, pronti per l'installazione sulla semovente.	55
2.21	Schema di un partitore di tensione.	57
2.22	Collegamenti elettrici per il sistema di misura della velocità di avanzamento del mezzo.	58
2.23	Schema di collegamento dei trasmettitori di pressione dell'olio.	59
2.24	Ponte di Wheatstone.	60
2.25	Circuito amplificatore operativo non invertente sottrattore.	62
2.26	Configurazione a doppio amplificatore operativo.	64
2.27	Collegamenti elettrici per il sistema di misura della pressione, della temperatura dell'olio, della portata d'aria e delle accelerazioni.	66
2.28	Collegamenti elettrici per il sistema di lettura e raccolta dei dati motore sulla linea CAN-BUS.	67
2.29	Collegamenti elettrici per il sistema di misura della posizione GNSS.	68
2.30	Diagramma di flusso dell'algoritmo di programmazione, adottato in ambiente Arduino IDE, delle schede elettroniche, con alcuni esempi di stringhe di codice.	70
2.31	Diagramma di flusso dell'algoritmo di elaborazione e post-processing dei dati in ambiente MATLAB, con alcuni esempi di stringhe di codice.	92
3.1	Semovente Cimina 300S in funzione durante il primo giorno di raccolta.	94
3.2	Condizioni ambientali durante il test dell'impianto di ventilazione: temperatura e pressione barometrica.	95

3.3	Parametri motore durante il test dell'impianto di ventilazione: velocità di rotazione, coppia relativa, posizione del pedale d'accelerazione e portata di combustibile.	96
3.4	Prestazioni dei ventilatori durante il test: pressione statica e portata d'aria nei condotti di aspirazione e insaccamento.	97
3.5	Fit lineare tra portata erogata dal ventilatore per l'aspirazione e numero di giri motore.	99
3.6	Fit lineare tra portata erogata dal ventilatore per l'insaccamento e numero di giri motore.	99
3.7	Fit quadratico tra giri motore e potenza consumata dal sistema di aspirazione e cernita.	100
3.8	Tragitto di salita e discesa percorso dal veicolo 2 volte durante il test della massima velocità su strada.	101
3.9	Velocità periferica e rotativa alle ruote anteriori e posteriori durante il test della massima velocità su strada.	102
3.10	Velocità del veicolo media misurata con i sensori Hall e GPS a confronto durante il test della massima velocità su strada.	103
3.11	Parametri motore durante il test della massima velocità su strada: velocità di rotazione, coppia relativa, posizione del pedale d'accelerazione e portata di combustibile.	104
3.12	Potenza erogata dal motore e pendenza del terreno a confronto durante il test della massima velocità su strada.	105
3.13	Pressioni idrauliche durante il test della massima velocità su strada: all'aspirazione della pompa ($-1 \div +1 \text{ bar}$), nel circuito frenante ($0 \div 40 \text{ bar}$), all'ingresso del motore idraulico anteriore ($0 \div 400 \text{ bar}$), alla mandata della pompa ($0 \div 1000 \text{ bar}$).	106
3.14	Condizioni ambientali durante il tragitto su strada: temperatura e pressione barometrica.	108
3.15	Velocità periferica e rotativa alle ruote anteriori e posteriori durante il tragitto su strada.	109
3.16	Accelerazioni misurate lungo la direzione longitudinale (X), trasversale (Y) e verticale (Z) durante il tragitto su strada.	110
3.17	Angoli di pendenza misurati attorno all'asse longitudinale (X), quindi lungo la direzione trasversale del veicolo, e attorno all'asse trasversale (Y), quindi lungo la direzione dell'avanzamento, durante il tragitto su strada.	110

3.18 Parametri motore durante il tragitto su strada: velocità di rotazione, coppia relativa, posizione del pedale d'accelerazione e portata di combustibile.	111
3.19 Potenza erogata dal motore e pendenza del terreno a confronto durante il tragitto su strada.	112
3.20 Pressioni idrauliche durante il test della massima velocità su strada: all'aspirazione della pompa ($-1 \div +1 \text{ bar}$), nel circuito frenante ($0 \div 40 \text{ bar}$), all'ingresso del motore idraulico anteriore ($0 \div 400 \text{ bar}$), alla mandata della pompa ($0 \div 1000 \text{ bar}$).	113
3.21 Pressione all'ingresso dei motori idraulici posteriori e anteriore a confronto con la velocità di avanzamento durante il tragitto su strada.	114
3.22 Pressione all'ingresso dei motori idraulici posteriori e anteriore a confronto con la pendenza della strada lungo la direzione dell'avanzamento durante il tragitto.	114
3.23 Temperature dell'olio idraulico all'ingresso e all'uscita del radiatore durante il tragitto su strada.	115
3.24 Posizioni registrate della macchina, nel primo giorno di raccolta, dalle 9:40 alle 10:40, colorate in funzione del tempo di acquisizione.	118
3.25 Posizioni registrate della macchina, nel primo giorno di raccolta, dalle 10:12 alle 10:24, comparate al tragitto realmente percorso.	119
3.26 Velocità periferica e rotativa alle ruote anteriori e posteriori durante una tratta del primo giorno di raccolta.	120
3.27 Parametri motore durante una tratta del primo giorno di raccolta: velocità di rotazione, coppia relativa, posizione del pedale d'accelerazione e portata di combustibile.	121
3.28 Accelerazioni misurate lungo la direzione longitudinale (X), trasversale (Y) e verticale (Z), durante una tratta del primo giorno di raccolta.	122
3.29 Angoli di pendenza misurati attorno all'asse longitudinale (X), quindi lungo la direzione trasversale del veicolo, e attorno all'asse trasversale (Y), quindi lungo la direzione dell'avanzamento, durante una tratta del primo giorno di raccolta.	122

3.30	Potenza erogata dal motore e pendenza del terreno a confronto durante una tratta del primo giorno di raccolta. .	123
3.31	Pressioni idrauliche durante una tratta del primo giorno di raccolta: all'aspirazione della pompa ($-1 \div +1 \text{ bar}$), nel circuito frenante ($0 \div 40 \text{ bar}$), all'ingresso del motore idraulico anteriore ($0 \div 400 \text{ bar}$), alla mandata della pompa ($0 \div 1000 \text{ bar}$).	124
3.32	Prestazioni dei ventilatori durante una tratta del primo giorno di raccolta: pressione statica e portata d'aria nei condotti di aspirazione e insaccamento.	125
3.33	Condizioni ambientali durante una tratta del secondo giorno di raccolta: temperatura e pressione barometrica. . .	126
3.34	Velocità periferica e rotativa alle ruote anteriori e posteriori durante una tratta del secondo giorno di raccolta.	127
3.35	Parametri motore durante una tratta del secondo giorno di raccolta: velocità di rotazione, coppia relativa, posizione del pedale d'accelerazione e portata di combustibile.	129
3.36	Potenza consumata dall'avanzamento e dai ventilatori a confronto durante una tratta del secondo giorno di raccolta.	129
3.37	Angoli di pendenza misurati attorno all'asse longitudinale (X), quindi lungo la direzione trasversale del veicolo, e attorno all'asse trasversale (Y), quindi lungo la direzione dell'avanzamento, durante una tratta del primo giorno di raccolta.	130
3.38	Pressioni idrauliche durante una tratta del secondo giorno di raccolta: all'aspirazione della pompa ($-1 \div +1 \text{ bar}$), nel circuito frenante ($0 \div 40 \text{ bar}$), all'ingresso del motore idraulico anteriore ($0 \div 400 \text{ bar}$), alla mandata della pompa ($0 \div 1000 \text{ bar}$).	131
3.39	Pressioni nel circuito idraulico e pendenza del terreno a confronto durante una tratta del secondo giorno di raccolta.	131
3.40	Prestazioni dei ventilatori durante una tratta del secondo giorno di raccolta: pressione statica e portata d'aria nei condotti di aspirazione e insaccamento.	133
3.41	Temperature dell'olio idraulico all'ingresso e all'uscita del radiatore durante la seconda giornata di raccolta.	134

Elenco delle tabelle

2.1	Sensori e metodi utilizzati per misurare i principali parametri prestazionali della semovente.	32
2.2	Specifiche tecniche del sensore di temperatura Pt100 AT105.	35
2.3	Specifiche tecniche dei trasmettitori di pressione: AP031, AP026, AP029, AP030.	37
2.4	Specifiche tecniche dei sensori di pressione dell'aria: MPX10DP, MPXV7002DP.	39
2.5	Specifiche tecniche del sensore di prossimità Hall AI908.	40
2.6	Specifiche tecniche del modulo accelerometro giroscopio GY-521.	42
2.7	Specifiche tecniche del modulo NEO-M9N con antenna GPS.	43
2.8	Specifiche tecniche del clock RTC DS3231.	44
2.9	Specifiche tecniche del lettore SD SPI.	45
2.10	Specifiche tecniche del display OLED I2C.	46
2.11	Specifiche tecniche del Multiplexer TCA9548A.	47
2.12	Specifiche tecniche della Shield CAN-BUS di SparkFun.	48
2.13	Specifiche tecniche della scheda Arduino Mega 2560 Rev3.	49
2.14	Specifiche tecniche della scheda Arduino R4 Minima.	50
2.15	Specifiche tecniche della scheda SparkFun RedBoard.	51
2.16	Costi complessivi per la realizzazione del DAQ.	53
2.17	Resistenze impiegate nel circuito partitore di tensione per i sensori Hall.	57
2.18	Resistenze accoppiate ai sensori di pressione dell'olio idraulico.	59
2.19	Tensione d'alimentazione e resistenze adottate nei circuiti a ponte di Wheatstone.	61
2.20	Resistenze scelte per l'amplificazione dei segnali in uscita del ponte di Wheatstone.	63

2.21	Tensioni di alimentazione e resistenze adottate per il circuito di amplificazione doppia.	64
3.1	Portate d'aria medie Q_s e Q_d erogate rispettivamente dal ventilatore per l'aspirazione e per l'insaccamento e potenza media P erogata dal motore per ogni velocità di rotazione n adottata durante i test.	98
3.2	Velocità medie dell'aria v_s e v_d rispettivamente nel condotto di aspirazione e nel condotto di scarico per ogni velocità di rotazione n adottata durante i test.	98
3.3	Consumi di potenza P e di combustibile F_c del veicolo e pressioni nel circuito idraulico di trasmissione alla mandata della pompa p_{1000} e all'ingresso dei motori idraulici posteriori p_{400} su strada in condizioni pressoché stazionarie di giri n , velocità v e pendenza θ_Y	116
3.4	Parametri medi operativi della macchina in raccolta su terreno agricolo in condizioni pressoché stazionarie di giri n , velocità v e pendenza θ_Y : consumi di potenza complessivi P associati ai ventilatori P_f e al moto e servizi ausiliari P_m ; consumi di combustibile F_c ; pressioni nel circuito idraulico di trasmissione alla mandata della pompa p_{1000} e all'ingresso dei motori idraulici posteriori p_{400} ; prestazioni dei ventilatori in termini di portate erogate per aspirare il prodotto Q_s e insaccarlo Q_d ; velocità dell'aria media nei condotti di aspirazione v_s e insaccamento v_d	135

Bibliografia

- [1] *Arduino - Language Reference*. <https://docs.arduino.cc/language-reference/>. Accessed: 2024-10-31.
- [2] *FACMA - Construction of agricultural machinery*. <https://www.facma.it/index.asp?lang=eng>. Accessed: 2024-10-31.
- [3] Aliverti, P.: *Il manuale di Arduino: Guida completa*. Made for Makers. Edizioni LSWR, 2016.
- [4] Assettati, L.: *Recenti evoluzioni nella meccanizzazione della raccolta del nocciolo*. tesi di laurea, Università degli studi della Tuscia - Laurea Triennale in Scienze Agrarie e Ambientali, 2020.
- [5] Assettati, L., P. Rossi, G. Rigon, L. Bianchini, D. Monarca, M. Cecchini e R. Alemanno: *Development of a low-cost hazelnut counting system*. 2024.
- [6] Bertoglio, R., C. Corbo, F. Renga e M. Matteucci: *The Digital Agricultural Revolution: A Bibliometric Analysis Literature Review*. IEEE Access, 2021.
- [7] Biondi, P.: *Meccanica agraria. Le macchine agricole*. Terra e l'uomo. UTET, 1999.
- [8] Casa, R.: *Agricoltura di precisione. Metodi e tecnologie per migliorare l'efficienza e la sostenibilità dei sistemi colturali*. Università e Formazione. Edagricole, 2017.
- [9] forestali, M. delle politiche agricole alimentari e:
PIANO DEL SETTORE CORILICOLO 2022-2025.
<https://terraevita.edagricole.it/frutticoltura/piano-corilicolo-tracciato-percorso-rilanciare-nocciola/>.
Accessed: 2024-10-31.

- [10] Fraden, J.: *Handbook of Modern Sensors: Physics, Designs, and Applications*. Springer New York, 2010.
- [11] Gaetano Conte, E.I.: *Elettronica, Elettrotecnica e Automazione*. Hoepli, openschool ed., 2017.
- [12] Gattamelata, D., G. Rigon, D. Monarca, M. Cecchini, P. Rossi, R. Alemanno e L. Assettati: *Virtual and physical prototyping of a driver's cab for a self-propelled agricultural machine*. 2024.
- [13] International, S.: *SAE J1939 Standard*. <https://www.sae.org/standards/development/ground-vehicle/sae-j1939-standards-collection-on-the-web>. Accessed: 2024-10-31.
- [14] Lazzari, M. e F. Mazzetto: *Meccanica e meccanizzazione dei processi produttivi agricoli*. REDA, 2 ed., 2016.
- [15] Margolis, M.: *Arduino Cookbook: Recipes to Begin, Expand, and Enhance Your Projects*. O'Reilly, 2nd ed., 2011.
- [16] Monarca, D., R. Alemanno, G. Rigon e P. Rossi: *Hazelnut Harvesting Machines: Recent Advances and New Trends*. Nel Ferro, V., G. Giordano, S. Orlando, M. Vallone, G. Cascone e S.M.C. Porto (cur.): *AIIA 2022: Biosystems Engineering Towards the Green Deal*, pp. 763–773, Cham, 2023. Springer International Publishing.
- [17] Monarca, D., M. Cecchini, M. Guerrieri, M. Santi, R. Bedini e A. Colantoni: *SAFETY AND HEALTH OF WORKERS: EXPOSURE TO DUST, NOISE AND VIBRATIONS*. Nel VII *International Congress on Hazelnut*, pp. 437–442, 2009.
- [18] Ortolani, G. e E. Venturi: *Manuale di elettrotecnica, elettronica e automazione*. Hoepli, 2 ed., 2017.
- [19] Pagano, M., M. Biocca, M. Cecchini, A. Colantoni, F. R., G. Sperandio, M. Fedrizzi, P. Gallo, G. M. e D. Pochi: *Dust emissions with a new hazelnut mechanical harvesting prototype*. Nel *International Conference Rural Health RAGUSA SHWA: September 8-11, Lodi, "Safety Health and Welfare in Agriculture, Agro-food and Forestry"*, set. 2015.

-
- [20] Ribbens, W.: *Understanding Automotive Electronics: An Engineering Perspective*. Butterworth-Heinemann, 2017.
- [21] Rigon, G.: *Progettazione e verifica di una cabina per semovente agricola*. tesi di laurea, Università degli Studi di Modena e Reggio Emilia - Laurea Magistrale in Ingegneria Meccanica, 2019.
- [22] Rigon, G., T. Del Signore, L. Assettati, E. Cioccolo, L. Bianchini, D. Monarca, M. Cecchini, R. Alemanno e P. Rossi: *Volume measurement of hazelnuts harvest through ultra-sonic sensors*. 2024.
- [23] Um, D.: *Solid Modeling and Applications: Rapid Prototyping, CAD and CAE Theory*. Springer International Publishing, 2018.
- [24] Webster, J. e H. Eren: *Measurement, Instrumentation, and Sensors Handbook: Spatial, Mechanical, Thermal, and Radiation Measurement*. CRC Press, 2 ed., 2014.

Ringraziamenti

Anzitutto vorrei ringraziare papà Gabriele e mamma Silvana per dimostrare affetto e orgoglio nei miei confronti e per appoggiarmi nelle più importanti scelte vitali.

Poi ringrazio Silvia, la mia ragazza, che da più di 8 anni mi regala amore e affetto, sempre fiera di me e capace di addolcire la mia personalità.

Desidero ringraziare nonna Mariassunta e zia Annalisa per il loro continuo sostegno, per manifestare orgoglio nei miei confronti e per l'aiuto prezioso che mi hanno dato nella scrittura di questo lavoro.

Un sentito ringraziamento va al mio tutor di dottorato Danilo, nonché relatore della tesi, per avermi offerto la possibilità di intraprendere il percorso di dottorato che mi ha formato molto a livello professionale.

Desidero esprimere la mia gratitudine a Giacomo, collega e grande amico da moltissimi anni, pronto sempre ad aiutarmi e ad appoggiare le mie decisioni. Grazie al suo aiuto è stato possibile svolgere questo immenso studio, oggetto della sua tesi magistrale.

Ringrazio il collega di lavoro Leonardo, ormai amico, con cui ho collaborato nella maggior parte degli studi di ricerca intrapresi nel percorso di dottorato e condiviso la passione per questo filone di ricerca.

Un grande ringraziamento va a FACMA, che ha finanziato il corso di dottorato e che mi ha dato i mezzi per compiere moltissime ricerche e progetti soddisfacenti, arricchendo le mie competenze.

Vorrei ringraziare anche i colleghi ricercatori dell'Università, con cui ho condiviso bellissime esperienze, anche al di fuori del mondo lavorativo.

Infine ringrazio i miei amici, sempre pronti a donarmi consiglio, ilarità e spensieratezza, per la fiducia che ripongono in me e per considerarmi un punto di riferimento.