POLITECNICO DI TORINO

Collegio di Ingegneria Meccanica, Aerospaziale, dell'Autoveicolo e della Produzione

Corso di Laurea Magistrale in Ingegneria Meccanica

Tesi di Laurea Magistrale

Sviluppo di sistemi integrati per la produzione

Relatore

prof. Federico Colombo

Candidato

Davide Battezzati s262798

Marzo 2021

Sommario

1. Introduzione ... 1
1.1 Analisi del problema ed obiettivi ... 1
1.2 Ruolo dei sensori nell'automazione ... 2
1.3 L'azienda PSA PIPES s.r.l. ... 2
1.4 Organizzazione del lavoro svolto ... 5
2. Sistema Contapezzi ... 7
2.1 Descrizione del principio di trasduzione ... 7
2.2 Acquisizione del segnale digitale ... 10
2.3 Cablaggio del circuito ... 12
2.4 Programma Raspberry ... 14
2.5 Invio dei dati ... 19
2.5.1 Cenni su XAMPP ... 19
2.5.2 Database e script in PHP ... 23
2.6 Risultati ... 27
3. Stazione di controllo diametri ... 31
3.1 Principio di trasduzione ... 31
3.1.1 Caratteristiche del micrometro laser KEYENCE IG-028 ... 32
3.1.2 Caratteristiche dell'amplificatore IG-1000 ... 38
3.2 Acquisizione del segnale analogico ... 39
3.2.1 Convertitore analogico-digitale ad approssimazioni successive ... 42
3.3 Realizzazione del sistema per misurazioni ... 47
3.4 Programma Fishino ... 51
3.5 Invio dei dati ... 58
3.6 Calibrazione del sistema di misurazione ... 59
4. Misurazioni ed analisi dei dati ... 65
4.1 Considerazioni sui valori fuori tolleranza ... 71
4.2 Controllo di processo statistico ... 73
4.2.1 Capacità di processo ... 74
4.3 Considerazioni sui risultati ... 78
5. Conclusioni ... 79
6. Bibliografia e sitografia ... 81
7. Appendice ... 82
8. Ringraziamenti ... 91
9. Allegati ... 92

1. Introduzione

1.1 Analisi del problema ed obiettivi

Il lavoro di tesi viene intrapreso a seguito dell'attività di tirocinio svolta presso PSA PIPES s.r.l., azienda nichelinese nel settore metalmeccanico specializzata nella piegatura e nella lavorazione dei tubi.

La necessità che porta allo sviluppo della tesi nasce dall'esigenza di rilevare dati dai processi produttivi mediante sensoristica applicata al fine di ottenere un controllo in tempo reale delle informazioni e di poter elaborare le statistiche ad esse associate.

In ambito industriale l'utilizzo dei sensori è fondamentale per molti aspetti: da una parte sono in grado di affiancare l'operatore nel regolare svolgimento dell'attività produttiva allo scopo di ridurre drasticamente gli errori commessi dalle imprecisioni umane; dall'altra permettono di monitorare costantemente le apparecchiature principali al fine di raccogliere e memorizzare i dati. In particolar modo l'analisi delle informazioni acquisite nel lungo periodo permette di avere gli strumenti necessari all'elaborazione di strategie più efficaci: una tra tutte è la "manutenzione predittiva". In quest'ottica l'intelligenza artificiale fornisce informazioni utili non solo per raggiungere l'eccellenza produttiva, ma anche avere risparmi sui costi di manutenzione non trascurabili.

A tal proposito il lavoro svolto è finalizzato all'ideazione di un sistema *contapezzi* ed alla progettazione e sperimentazione di un sistema adibito alle misurazioni di controllo grazie al quale si ha la possibilità di effettuare un'analisi statistica sui dati raccolti.

Nel primo caso l'esigenza nasce dall'impossibilità di avere un controllo in tempo reale del numero di pezzi che ciascuna macchina produce. Ad oggi il conteggio del numero dei pezzi prodotti è, nella buona parte dei casi, effettuato tramite contatori che rilevano il numero dei cicli di lavorazione delle macchine permettendone la visualizzazione su appositi display digitali. I dati acquisiti, però, non sono in alcun modo registrati e trasmessi. Un'altra limitazione del caso è sicuramente la gestione degli scarti: in particolare i contatori presenti non permettono di decrementare il conteggio in presenza di pezzo non conforme a seguito della lavorazione. La registrazione dei dati viene comunque effettuata dall'operatore che, una volta terminato il proprio turno sulla macchina, riporta manualmente il numero di pezzi e scarti prodotti che possono essere soggetti quindi alle inesattezze insorte durante la registrazione manuale. A fronte di queste problematiche si provvede alla realizzazione di un sistema che prevede l'utilizzo di un sensore fotoelettrico atto alla rilevazione di un oggetto e che, non solo permette di sopperire alle limitazioni già descritte, ma anche di poter calcolare indirettamente il tempo effettivo di lavoro della macchina e poter adottare così strategie di manutenzione apposite sulla base dei dati ricavati.

Per quanto riguarda il secondo caso di studio l'esigenza parte da ulteriori limitazioni: in particolare quelle che si verificano durante l'effettuazione delle misurazioni di controllo sui pezzi. Il controllo a campione dei diametri dei tubi in produzione avviene infatti tramite l'utilizzo di calibri a tampone passa-non passa ed è compito dell'operatore segnalare l'avvenuto controllo, che genericamente prevede un numero di cinque pezzi all'ora, tramite una registrazione cartacea. Per quanto riguarda invece i controlli, anch'essi a campione, che avvengono nella postazione adibita al controllo finale, l'operatore effettua le misure tramite calibro a corsoio e registra, sempre in modo cartaceo, solo il valore minimo e massimo

riscontrato sulle misurazioni di un certo numero di pezzi, segnalando eventuali pezzi fuori tolleranza. Ragion per cui è oggetto di studio un sistema che permetta, tramite l'utilizzo di un micrometro laser, di effettuare misurazioni in modo rapido, riducendo al minimo gli errori umani e, soprattutto, di avere una registrazione digitale dei controlli effettuati con le relative misure. Quest'ultimo aspetto è fondamentale per poter adottare strategie di prevenzione sui processi produttivi che permetterebbero di intervenire in tempo utile prima che si verifichino anomalie.

1.2 Ruolo dei sensori nell'automazione

Come già accennato la parte fondamentale necessaria al funzionamento di questi sistemi è rappresentata dai sensori. Essi costituiscono il primo anello della catena di misura e rappresentano dispositivi in grado di trasformare una grandezza fisica in un segnale elettrico. Grazie a questa capacità si ha la possibilità di sfruttare i vantaggi che garantiscono i segnali elettrici come la possibilità di estrarre informazioni a qualunque distanza e la possibilità di effettuare, per mezzo della conversione analogica-digitale, l'elaborazione dei segnali.

I sensori sono da sempre strumenti fondamentali nello sviluppo tecnologico delle attività dei settori industriali: lo sviluppo accelerato negli ultimi anni dell'elettronica, dell'informatica e dell'automazione dei processi produttivi ha permesso un'ulteriore affermazione in campo industriale di tali dispositivi. A tal proposito le azioni svolte all'interno del processo produttivo devono essere controllate: tale procedimento può essere affidato all'operatore che tramite i propri sensi osserva l'andamento del processo oppure da un apparato che lo esegue automaticamente tramite appunto l'utilizzo dei sensori. In ambito industriale il progressivo sviluppo dell'automazione di processo è scaturito principalmente da due fattori: l'aumento del costo della manodopera e la comparsa sul mercato di microprocessori con rapporti prezzo/prestazioni sempre più vantaggiosi.

Un sistema di misura, nel suo complesso, è costituito da una successione di diversi elementi: un sensore che estrae l'informazione dalla grandezza fisica, un sistema di condizionamento che principalmente amplifica il segnale proveniente dal sensore, un sistema di conversione che trasforma il segnale analogico in digitale ed un sistema di controllo che memorizza o elabora l'informazione numerica tramite un opportuno programma.

1.3 L'azienda PSA PIPES s.r.l.

L'azienda PSA PIPES s.r.l. nasce nel 2013 per far fronte ad una richiesta da parte di T.ERRE s.r.l. riguardo un grosso progetto alla cui base è coinvolta la lavorazione del tubo. La società muove i primi passi acquistando i macchinari dalla società Officine Mollo di Nichelino e fatturando già nei primi due anni 2 milioni di euro. Il core business aziendale si stabilisce sin da subito sulla piegatura e la lavorazione dei tubi in acciaio ed alluminio destinati principalmente al convogliamento di acqua, aria e benzina. Nella maggior parte dei casi i tubi vengono utilizzati all'interno di motori termici o congiungono il motore ad altri elementi all'interno di veicoli di grandi dimensioni. In piccoli casi si realizzano tubi per il sostegno degli specchietti ed in rare eccezioni tubi per il condizionamento. I diametri che sono coinvolti in queste applicazioni non superano i 42 mm. In Figura 1.1 è mostrata l'area produttiva dell'azienda.

Figura 1.1: Area produttiva aziendale

Le operazioni sul tubo vengono quasi tutte realizzate internamente all'azienda: viene effettuata la piegatura, la lavorazione delle estremità, la foratura, la puntatura, la brasatura ad induzione e le prove tenuta in acqua ed aria. Le uniche operazioni che vengono effettuate esternamente sono quelle inerenti ai trattamenti termici.

Il mercato di riferimento di PSA PIPES è quindi quello "automotive", in cui si cerca non di perseguire il contatto diretto con gli utilizzatori finali del servizio o prodotto fornito, ma di ricercare i propri clienti tra le aziende leaders fornitrici delle case automobilistiche in modo tale da potere affrontare al loro riparo eventuali problematiche (tecniche, commerciali, qualitative) che richiederebbero strutture aziendali molto più impegnative di quelle di cui può disporre l'azienda. Nello sviluppare i rapporti con i propri clienti e nell'investire su di essi, PSA PIPES s.r.l. privilegia quei clienti che pur internazionalizzando le loro attività conservano un forte radicamento sul territorio nazionale e che hanno principalmente l'estero come destinazione finale dei loro prodotti.

I reparti principali dell'azienda sono:

- Il reparto di piegatura in cui sono presenti le piega-tubi e rappresenta il reparto più importante dell'azienda. In Figura 1.2 è riportata una curvatubi idraulica che adempie alla lavorazione principale di piegatura del tubo.

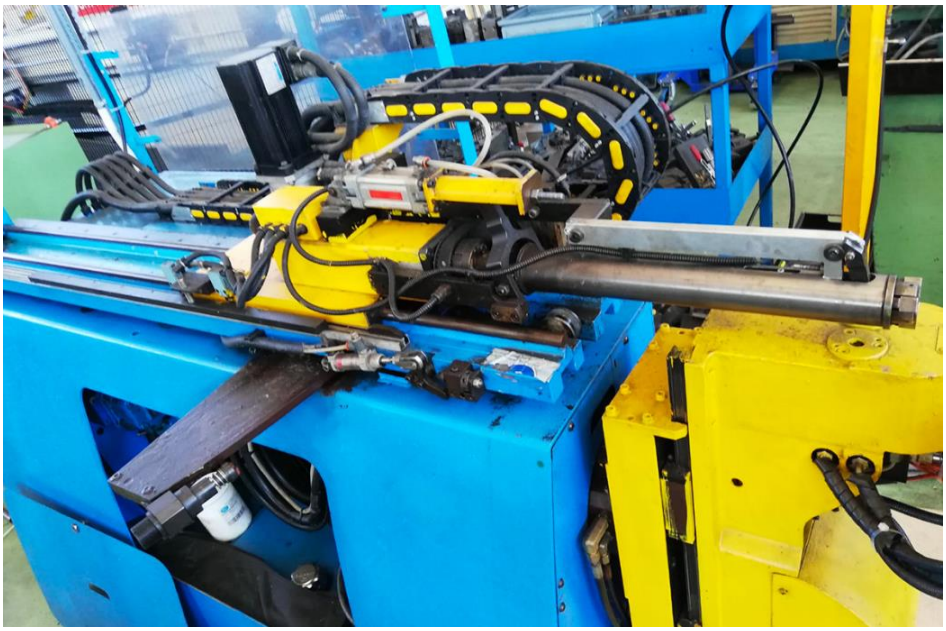


Figura 1.2: Curvatubi

- Il reparto di saldatura adibito ad operazioni di puntatura. In Figura 1.3 sono riportate le cabine in cui sono presenti i robot di saldatura.

Figura 1.3: Reparto di saldatura

- Il reparto di deformazione in cui sono presenti le deformatrici e le presse che lavorano le estremità del tubo;
- Il reparto di prova tenuta necessario a provare i tubi sia in acqua che in aria;
- Il centro di lavoro di fresatura e tornitura in grado di garantire una certa elasticità quando si deve intervenire per manutenzione ordinaria e straordinaria. Negli ultimi anni si è introdotta una stampante 3D per produrre piccole attrezzature di controllo come le "dime" permettendo di ridurre notevolmente i tempi ed i costi per queste realizzazioni.

- La sala metrologica che si occupa della qualità interna e gestisce i rapporti con i clienti ed i fornitori.

Nel corso degli anni l'azienda ha avuto un aumento progressivo del fatturato a partire dal primo anno, con un fatturato pari a 1,2 milioni e con una piccola perdita, fino ad arrivare ad un massimo di 5,5 milioni nel 2018. Negli ultimi due anni l'azienda chiude rispettivamente con un pareggio ed una perdita di bilancio determinati principalmente dal cambiamento radicale del mix di prodotto: nei primi anni si produceva infatti prodotti più redditizi e col passare del tempo si è creata una maggiore dispersione del prodotto scaturendo un aumento dei turni di lavoro e del personale.

Nonostante l'anno passato difficile in termini di utile, PSA PIPES s.r.l. rimane ancora una realtà produttiva grazie soprattutto al principale cliente T.ERRE, società in forte espansione nel settore automotive di tutto il mondo.

1.4 Organizzazione del lavoro svolto

Il lavoro viene suddiviso in tre parti.

Capitolo 2: ci si occupa dello studio di un sistema di conteggio dei pezzi che prevede l'applicazione di un sensore fotoelettrico. I dati acquisiti dal sensore, assieme al conteggio degli scarti prodotti, vengono inviati ad un Raspberry Pi 3 Model B+, ossia un controller a basso costo in grado di gestire egregiamente compiti dedicati di automazione. Fondamentale è la presenza di un modulo WiFi integrato che permette di inviare i dati ricevuti in ingresso, tramite un opportuno programma realizzato in linguaggio Python, al database che consente di avere uno *storage* di informazioni in completa modalità wireless.

L'accesso al database è gestito da alcuni script in linguaggio PHP responsabili anche della popolazione delle relative tabelle mediante l'aggiornamento dei contatori. Una volta realizzato il cablaggio e la parte software si pensa ad una futura applicazione in prossimità della curvatubi CRIPPA 1042E in cui l'implementazione dovrà essere oggetto di uno studio sicuramente più approfondito che possa garantire la completa automazione del processo. Importante è la possibilità di completare la postazione introducendo un'app su tablet realizzata con il software MIT App Inventor che permette all'operatore di inviare gli opportuni dati del processo ad inizio lavorazione e vedere su schermo il conteggio dei pezzi conformi e degli scarti.

Capitolo 3: si passa allo studio di un sistema adibito alle misurazioni di controllo tramite l'utilizzo di un micrometro laser. Le misure effettuate vengono inviate al database tramite un Fishino UNO Rev. 2 che è in grado di svolgere funzioni analoghe al Raspberry Pi, ma, a differenza di quest'ultimo, importante è la presenza di un convertitore analogico-digitale (ADC) che consente di trasformare il segnale analogico del micrometro e procedere all'elaborazione del dato tramite un opportuno programma scritto in linguaggio C++. Anche in questo caso l'accesso al database e l'inserimento delle misurazioni avvengono tramite appositi script in linguaggio PHP. La postazione di controllo è completata da un'app che permette prima di selezionare su tablet il codice del pezzo che si vuole sottoporre a misurazione, poi di controllare se la dimensione del pezzo è conforme alle tolleranze imposte.

Per verificare che il dato inviato al database non sia affetto da errori con ordini di grandezza comparabili a quelli imposti nelle tolleranze dimensionali dei pezzi, si procede ad una

calibrazione del Fishino, considerando gli errori di non linearità e ripetibilità del micrometro laser più che soddisfacenti per le applicazioni in esame.

Capitolo 4: in ultima analisi si procede alla sperimentazione del sistema di misura andando ad effettuare delle misurazioni sul diametro dei tubi di un lotto di 80 pezzi. I tubi presi in esame subiscono una deformazione assiale per la formazione di un'ogiva sull'estremità. A causa della lavorazione non perfettamente uniforme si decide di effettuare più misurazioni su un singolo tubo e ricavarne la media. Dall'analisi dei dati acquisiti si elaborano alcune statistiche per stabilire la conformità del processo di lavorazione che sta a monte della deformazione del tubo quindi evidenziando gli aspetti positivi e negativi della stazione di controllo dei diametri ideata.

Nella postazione sperimentale si ha a disposizione un calibro a tampone passa-non passa per verificare che i pezzi fuori tolleranza individuati dal micrometro laser siano effettivamente da scartare.

2. Sistema Contapezzi

Con il seguente caso di studio s'intende realizzare un sistema *contapezzi* che permetta di avere il controllo in tempo reale del numero di pezzi e di scarti prodotti. Per la parte di trasduzione si utilizza un sensore fotoelettrico in grado di generare un segnale elettrico ogni qual volta il suo raggio di luce intercetta un ostacolo. Il segnale prodotto viene inviato ad un controllore Raspberry Pi che, tramite l'esecuzione di un programma, legge il segnale di output del sensore e lo invia ad un database al fine di ottenere uno *storage* di dati. Oltre al segnale di presenza pezzo è inviato un segnale di presenza scarto tramite la pressione di un apposito pulsante.

2.1 Descrizione del principio di trasduzione

Prima di passare al funzionamento specifico del sensore è bene descrivere il comportamento di un generico sensore fotoelettrico. Il dispositivo è costituito da un emettitore ed un ricevitore. L'emettitore è in grado di convertire un segnale elettrico modulato in frequenza in un segnale luminoso che, attraverso un sistema ottico, viene mandato ad un ricevitore. L'emettitore ha inoltre a disposizione un oscillatore che ha la funzione di modulare il segnale elettrico di alimentazione con lo scopo di proteggerlo dalle interferenze della luce ambiente. Il ricevitore viene sintonizzato per ricevere unicamente le frequenze della luce modulata (frequenza dell'ordine dei kHz) ed è costituito da un *demodulatore/amplificatore* che trasforma il segnale ricevuto in un segnale di uscita di tipo ON/OFF. Se il segnale ottico ricevuto è al di fuori della frequenza dell'oscillatore, l'uscita è nulla.

In Figura 2.1 viene mostrato lo schema di funzionamento di un generico sensore di posizione fotoelettrico.

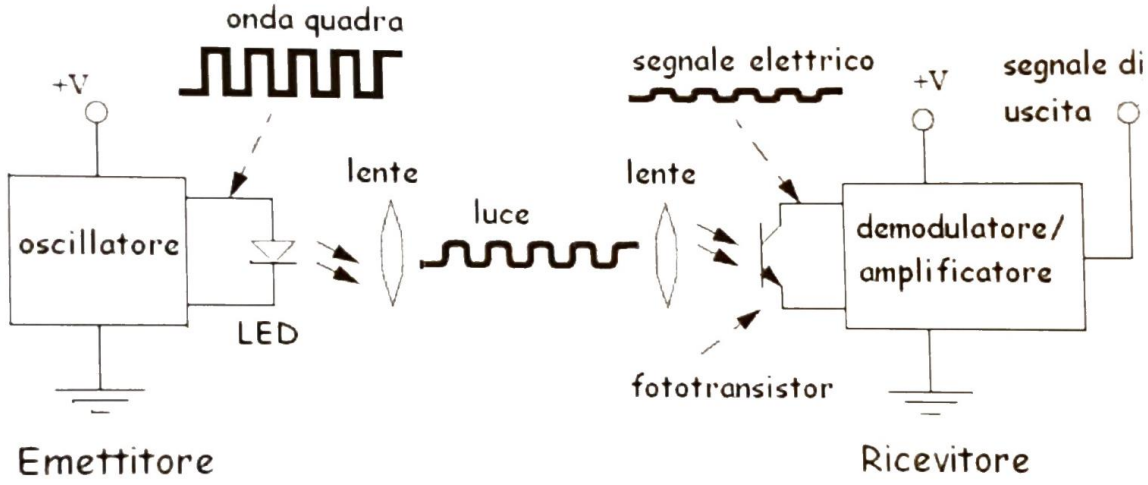


Figura 2.1: Schema di funzionamento di un sensore fotoelettrico (Figura tratta da "Automazione dei sistemi meccanici – Viktorov, Colombo)

La parte di trasduzione del caso di studio viene garantita dal sensore fotoelettrico modello KEYENCE PZ-G51P. Esso è un dispositivo sensibile alla luce che utilizza tecniche fotoelettriche per rilevare la presenza di un oggetto e, costituendo un sensore digitale, è in grado di generare un segnale di tipo ON/OFF. Viene classificato come dispositivo a sbarramento in quanto è costituito da un emettitore ed un ricevitore montati su due dispositivi separati che possono essere posti fino a 20 metri di distanza. In Figura 2.2 ed in Figura 2.3 sono mostrati rispettivamente l'immagine del sensore e le parti principali che costituiscono l'emettitore ed il ricevitore.

Figura 2.2: Sensore KEYENCE PZ-G51P

Specifiche tecniche del sensore KEYENCE PZ-G51P:

- Modalità di uscita: PNP;
- Massima distanza rilevata: $20\ m$;
- Sorgente di luce (LED): luce rossa;
- Uscita di controllo: collettore aperto 100 mA max (30 V max), tensione residua 1V max;
- Tensione di alimentazione: $10 \div 30\ VDC$;
- Consumi di corrente: emettitore 20 mA max, ricevitore 28 mA max.

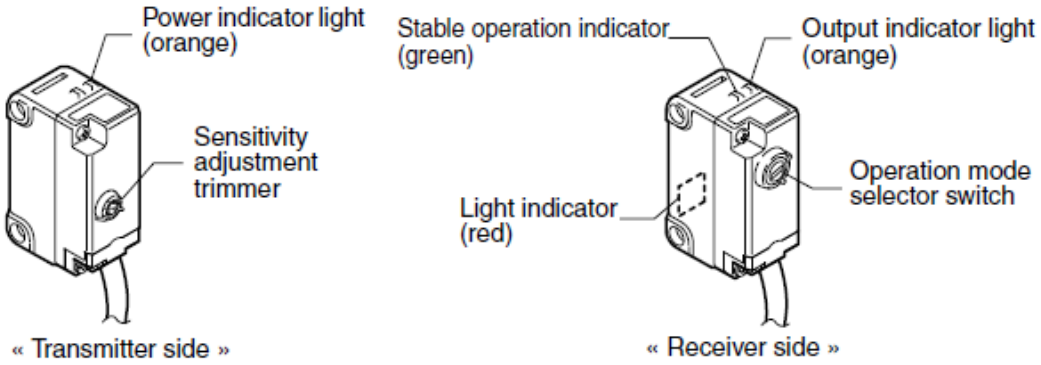


Figura 2.3: Parti principali del trasmettitore e del ricevitore del sensore KEYENCE PZ-G51P

Funzionamento del sensore KEYENCE PZ-G51P:

Per quanto concerne il funzionamento del dispositivo in questione, il ricevitore è costituito da un fototransistor che conduce quando riceve il raggio di luce prodotto dall'emettitore.

Una volta che il fascio luminoso viene interrotto a causa dell'interposizione di un ostacolo, lo stato di uscita del sensore cambia.

Da come si evince dai diagrammi di collegamento del circuito di I/O riportati in Figura 2.4 ed in Figura 2.5, il trasmettitore viene collegato all'alimentazione tramite il cavo marrone (si utilizza un alimentatore da 12 V) ed a GND (0 Volt) tramite il cavo blu.

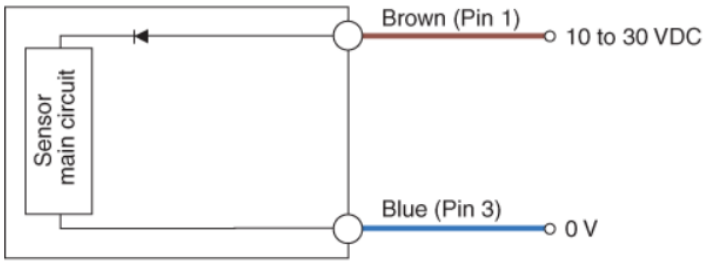


Figura 2.4: Diagramma di collegamento del circuito I/O lato trasmettitore

Il ricevitore, oltre a presentare i cablaggi dell'emettitore, ha a disposizione il cavo nero responsabile del segnale di *output* di tipo digitale quindi definito per istanti discreti di tempo e in grado di assumere un numero finito di valori discreti nel campo di variazione. In particolare il segnale di uscita è di tipo ON quando è presente l'oggetto tra emettitore e ricevitore; è di tipo OFF quando il fascio luminoso non è interrotto e quindi non è presente l'oggetto.

Quando emettitore e ricevitore sono alimentati si accende un LED arancione sul primo ed un LED verde sul secondo. L'interposizione dell'ostacolo tra i due dispositivi viene individuata dall'accensione di un LED arancione collocato sul ricevitore.

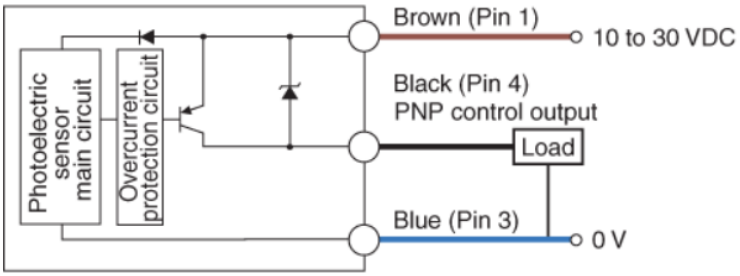


Figura 2.5: Diagramma di collegamento del circuito I/O lato ricevitore

Per avviare la commutazione del segnale viene impiegato un transistor bipolare che agisce da interruttore. La commutazione è di tipo PNP: il carico, che è rappresentato dal pin di input digitale del Raspberry Pi, è collegato all'uscita di commutazione ed a 0 V. In caso di commutazione di segnale del sensore, il transistor entra in saturazione e si comporta da interruttore chiuso. La corrente così scorre dall'alimentazione (12 V) tramite il transistor e fino a 0 V attraverso il carico con conseguente chiusura del circuito elettrico.

2.2 Acquisizione del segnale digitale

Il dispositivo utilizzato per ricevere in ingresso il segnale del sensore che individua la presenza di un ostacolo è un Raspberry Pi 3 Model B+ (modello mostrato in Figura 2.6).

Figura 2.6: Raspberry Pi 3 Model B+

Il Raspberry Pi 3 Model B+ è un "single-board" computer cioè un computer costruito su una scheda a circuito singolo costituita principalmente da un microprocessore, una memoria ed un'interfaccia GPIO. Si basa su un "system on a chip" che, nell'elettronica digitale, indica un circuito integrato in un solo chip che contiene, oltre al processore centrale, un chipset ed altri controller come quello per la memoria RAM e per la circuiteria di *input/output*.

Specifiche tecniche Raspberry Pi 3 Model B+:

- SoC Broadcom BCM2837B0, Cortex-A53 da 1,4 GHz e 64 bit
- Bluetooth 4.2
- Supporto PoE (con PoE HAT separato)
- Rete PXE potenziata
- 1 GB di memoria SDRAM LPDDR2
- Alimentazione cc di ingresso da 5 V/2,5 A (micro USB)
- LAN wireless da 2,4 GHz e 5 GHz IEEE 802.11.b/g/n/ac e Bluetooth 4.2/BLE
- Gigabit Ethernet su USB 2.0 (resa massima a 300 Mbps)
- 4 porte USB 2.0
- Header con GPIO a 40 pin
- Uscita video HDMI full-size
- Uscita audio stereo a 4 poli e porta video composito
- Porta CSI per il collegamento di una fotocamera Raspberry Pi
- Porta DSI per il collegamento di un display touchscreen Raspberry Pi
- Porta micro-SD per caricare il sistema operativo e conservare i dati
- Temperatura di funzionamento: da 0 °C a +50 °C
- Dimensioni: 120 mm x 75 mm x 34 mm
- Peso: 75 g

In Figura 2.7 vengono mostrati i principali componenti di cui è costituito il *single-board* computer.

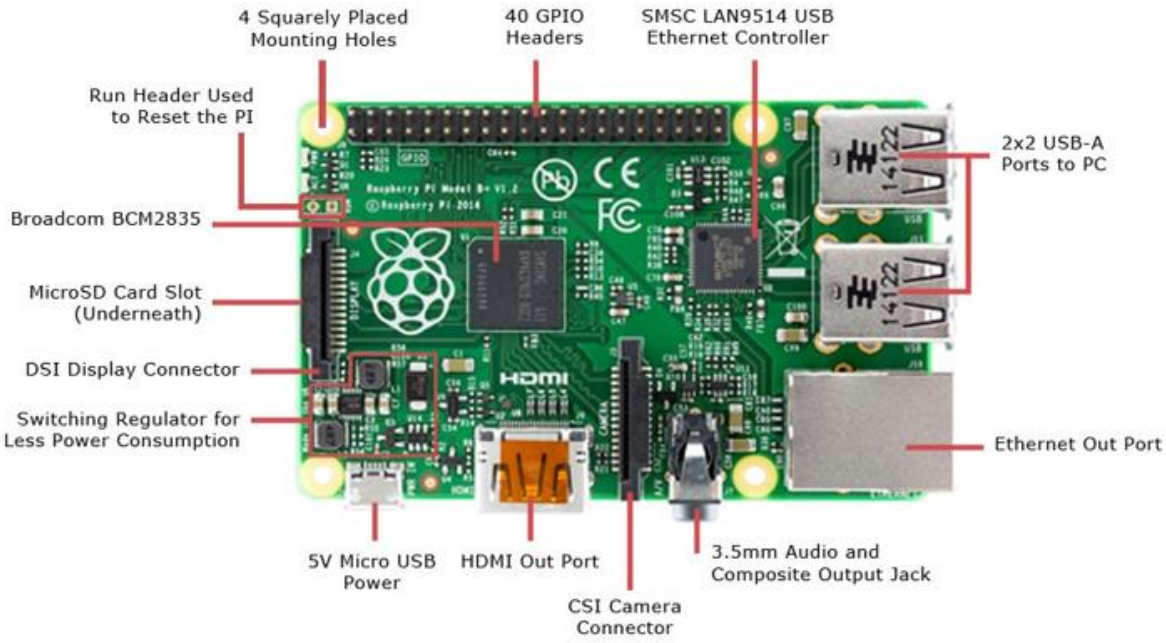


Figura 2.7: Componenti principali del Raspberry Pi 3 Model B+

Il sistema operativo utilizzato dal Raspberry è *Raspbian*, un sistema basato su Debian GNU/Linux. Il GPIO (General Purpose Input/Output) è un'interfaccia informatica hardware che consente al dispositivo di interagire con un'altra periferica. I pin presenti sulla GPIO possono agire come input per leggere i segnali digitali dalle altre parti del circuito, o output per controllare o segnalare agli altri dispositivi.

Il GPIO del Raspberry Pi 3 Model B+ è costituito da 40 pin disposti su due file. Sulla scheda sono presenti due pin di alimentazione a 5 V, due pin di alimentazione a 3,3 V ed otto pin di terra (GND, 0 V). I pin rimanenti sono tutti pin a 3,3 V per uso generico, quindi le uscite sono impostate a 3,3 V e gli ingressi tollerano un massimo di 3,3 V. In Figura 2.8 è riportato il *pinout* in cui vengono rappresentate le due file composte da 20 pin ciascuna.

Un pin configurato come output può essere impostato a livello alto (3,3 V) o basso (0 V). Un pin configurato come input può leggere solo un valore alto o basso escludendo la possibilità di ricevere in ingresso segnali analogici.

I pin GPIO possono essere utilizzati con una varietà di funzioni alternative, alcune sono disponibili su tutti i pin, altre su pin specifici.

PWM (pulse-width modulation):
- Software PWM disponibile su tutti i pin;
- Hardware PWM disponibile sul GPIO12, GPIO13, GPIO18 e GPIO19;

SPI:
- SPI0: MOSI (GPIO10), MISO (GPIO9), SCLK (GPIO11), CE0 (GPIO8), CE1 (GPIO7)
- SPI1: MOSI (GPIO20), MISO (GPIO19), SCLK (GPIO21), CE0 (GPIO18), CE1 (GPIO17), CE2 (GPIO16)

I2C:
- Data: (GPIO2) e Clock (GPIO3);
- EEPROM Data (GPIO0) ed EEPROM Clock (GPIO1);

Serial:

- TX (GPIO14) e RX (GPIO15).

Esistono due modi per indicare lo stesso GPIO: il sistema *BOARD* ed il sistema *BCM*.

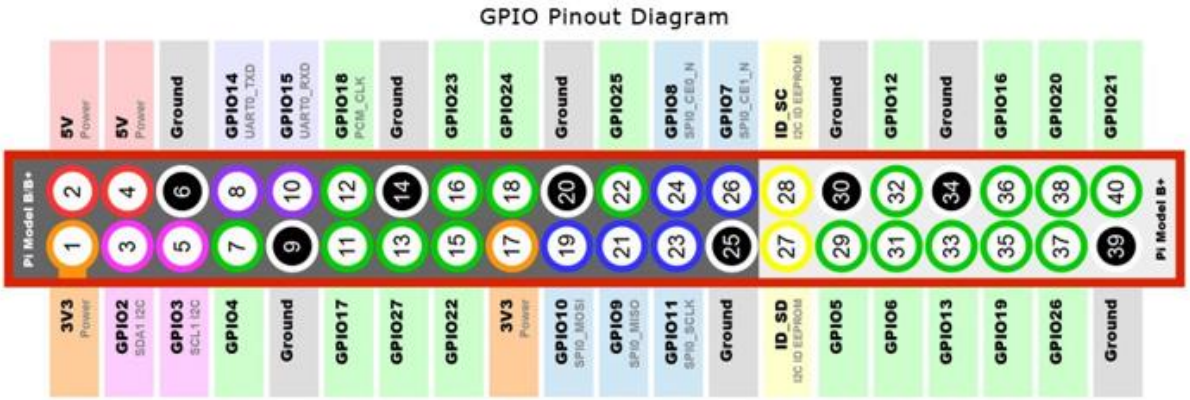


Figura 2.8: *Pinout* GPIO del Raspberry Pi 3 Model B+

Il sistema BOARD utilizza il numero del pin fisico da 1 a 40. Il sistema BCM, invece, utilizza il nome della linea diretta al microcontrollore ed ha quindi un modo differente per denominare i pin (ad esempio il pin in posizione 19 viene chiamato GPIO10).

2.3 Cablaggio del circuito

Il circuito elettrico che permette di ricevere il segnale dal sensore e svolgere le funzioni necessarie allo storage dei dati viene realizzato su *breadboard*. Come si può notare in Figura 2.9, in cui è riportato il circuito realizzato tramite il software FRITZING, sono presenti: il segnale di output del sensore, un partitore di tensione che permette di avere un valore di tensione adeguato all'ingresso del pin digitale, un pulsante di "scarto", un pulsante per l'azzeramento dei contatori, un LED verde, un LED rosso, due resistenze da $220\ \Omega$ e due resistenze di *pulldown* da $10\ k\Omega$. L'idea che sta alla base dell'applicazione consiste nell'associare il lampeggiamento del LED verde a seguito della ricezione del segnale del sensore e del LED rosso in seguito alla pressione di un pulsante atto a rilevare la presenza di un pezzo di scarto.

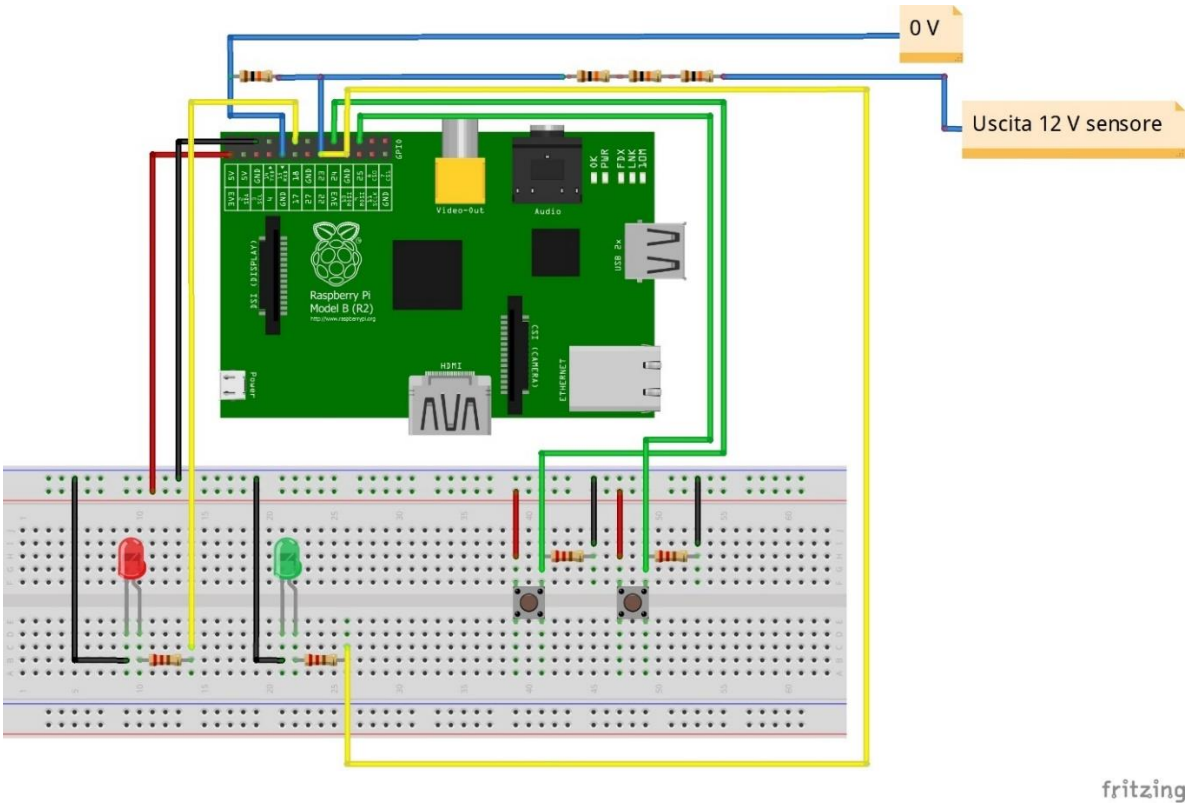


Figura 2.9: Circuito realizzato su breadboard con collegamento al Raspberry e segnale in uscita dal sensore

Il segnale di output del sensore fotoelettrico è ad una tensione pari a quella di alimentazione cioè 12 V. Essendo che i pin del Raspberry sono in grado di tollerare una tensione massima in ingresso pari a 3,3 V si decide di applicare un partitore di tensione allo scopo di ridurre il voltaggio considerando l'elevata impedenza dei pin del controllore. La relazione da cui si parte per ricavare il valore della resistenza da applicare è la seguente:

$$\frac{V_{out}}{V_{in}} = \frac{R_2}{R_1 + R_2} \tag{2.1}$$

dove:

- V_{out} è la tensione in uscita dal partitore che viene posta uguale a 3,3 V (valore massimo tollerabile dal Raspberry);

- V_{in} è la tensione in ingresso che corrisponde a quella del segnale in uscita dal sensore;

- R_2 è il valore di resistenza interposta tra il pin di ingresso e lo 0 V (sia quello del sensore che il GND del Raspberry) ed è posta uguale a 10 kΩ, cioè il valore usuale corrispondente alle resistenze di *pulldown*;

- R_1 è il valore incognito della resistenza interposta tra il segnale di uscita del sensore ed il pin di ingresso.

Il valore di resistenza da applicare è quindi pari a 26,363 kΩ; di conseguenza si decide di applicare tre resistenze in serie da 10 kΩ ciascuna, facendo corrispondere un valore di V_{out} pari a 3 V. Una volta realizzato il circuito si nota, tramite la misura con un multimetro che, per ragioni non ben chiare, il trasformatore che alimenta il sensore fotoelettrico eroga una tensione di qualche Volt superiore ai 12 V indicati sul trasformatore. Si rileva comunque

sperimentalmente che la tensione in ingresso al Raspberry è attorno ai 3,3 V ragion per cui si mantiene il valore delle resistenze come precedentemente calcolato.

Per quanto riguarda la restante parte del circuito realizzato su *breadboard*, le resistenze da $220\ \Omega$ vengono interposte tra i pin di uscita digitale del Raspberry (in cui la tensione erogata in uscita è pari a $3{,}3\ V$) ed i LED che per essere accesi necessitano di una tensione pari a circa $2\ V$ed una corrente che può variare attorno ai $10\ mA$. Tramite l'applicazione della legge di Ohm si ricava il valore della resistenza da utilizzare per non bruciare il LED:

$$R = \frac{3{,}3 - 2}{0{,}01} = 130\ \Omega \tag{2.2}$$

Si sceglie quindi una resistenza da $220\ \Omega$ per rimanere dalla parte della sicurezza. Si evidenzia inoltre che i LED sono dotati di polarità e quindi hanno un terminale positivo ed uno negativo. Il terminale negativo si riconosce facilmente poiché è quello più corto tra i due e, da quella parte, il bordo del LED presenta una smussatura.

Le resistenze da $10\ \text{k}\Omega$, utilizzate per i pulsanti presenti, sono resistenze dette di *pulldown*, che, assieme a quelle di *pullup*, sono molto comuni quando si utilizzano i microcontrollori o qualsiasi altro dispositivo digitale. Il motivo per cui si inseriscono queste resistenze nei circuiti deriva dal fatto che i pin di ingresso dei dispositivi digitali presentano impedenze piuttosto alte permettendo anche a correnti minime di poterne cambiarne lo stato. Questo significa che tali pin sono soggetti a captare qualsiasi disturbo elettromagnetico o elettrostatico e quindi a commutare le loro uscite in modo del tutto imprevisto ed incontrollato. Un pin di ingresso che si trova in questo stato si definisce in stato di "floating" (fluttuante) e permette a campi elettrici e magnetici esterni di indurre tensioni (anche piccole) tali da farne commutare il circuito interno. Ne deriva la necessità di porre sempre questi pin ad un adeguato livello, che può essere alto o basso. Condizioni intermedie al di fuori dei margini di sicurezza portano a stati di incertezza che possono dare origine a commutazioni indesiderate e mal funzionamento apparentemente difficile da diagnosticare. Quindi:

- il livello basso (0) è definito da una tensione di ingresso minore di $0{,}8\ V$;
- il livello alto (1) è una tensione di ingresso maggiore di $2\ V$;
- Il campo di tensioni tra $0{,}8\ V$ e $2\ V$ è dichiarato come indefinito ovvero, se applicato all'ingresso, darà un'uscita non determinabile, casuale, rappresentando una condizione da evitare assolutamente.

A fronte di queste considerazioni si interpone una resistenza di *pulldown* tra il pin di ingresso e il GND. Quando l'interruttore è aperto, il pin di ingresso legge la tensione sul GND cioè 0 V. Quando l'interruttore si chiude, l'ingresso viene portato al valore della tensione di alimentazione pari a 3,3 V. La resistenza serve quindi a portare il livello logico basso (0) all'ingresso del pin quando l'interruttore è aperto.

2.4 Programma Raspberry

Una volta realizzato il circuito su breadboard si procede alla realizzazione della parte software attraverso la quale vengono memorizzati i valori in input ed inviati al database. Il linguaggio di programmazione utilizzato per la realizzazione del codice è Python, introdotto tramite l'ambiente di sviluppo "Thonny Python IDE" (Figura 2.11) presente all'interno del sistema operativo Raspbian (schermata principale rappresentata in Figura 2.10).

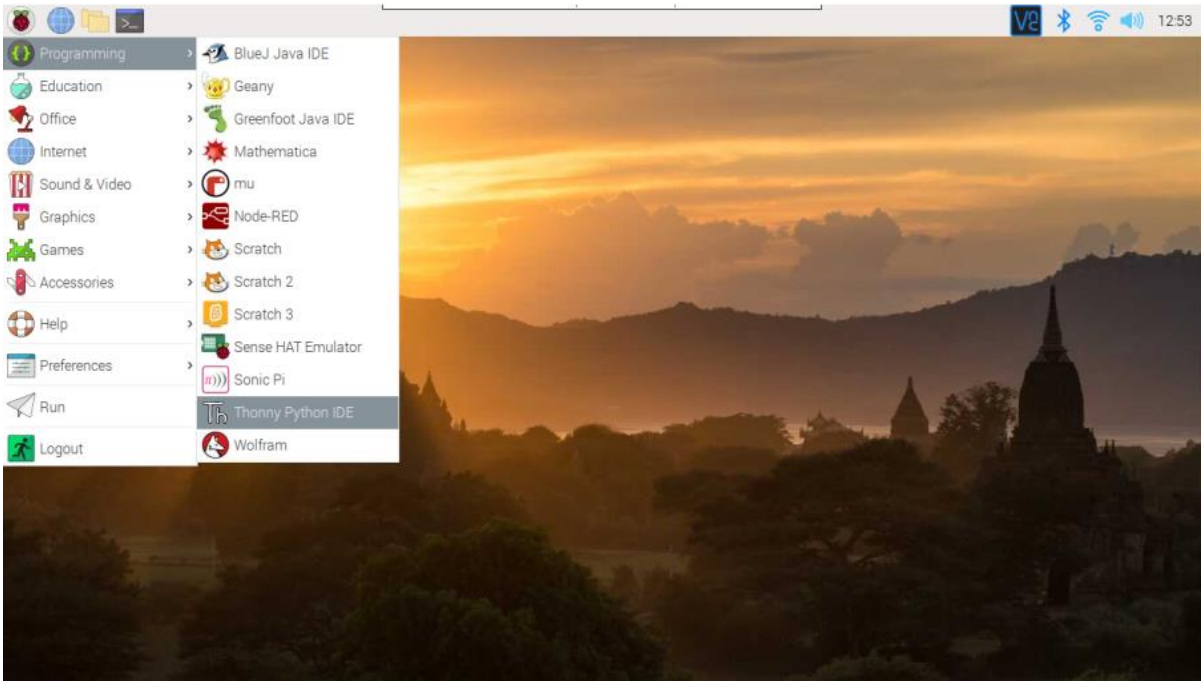


Figura 2.10: Schermata principale di Raspian con ambiente di sviluppo

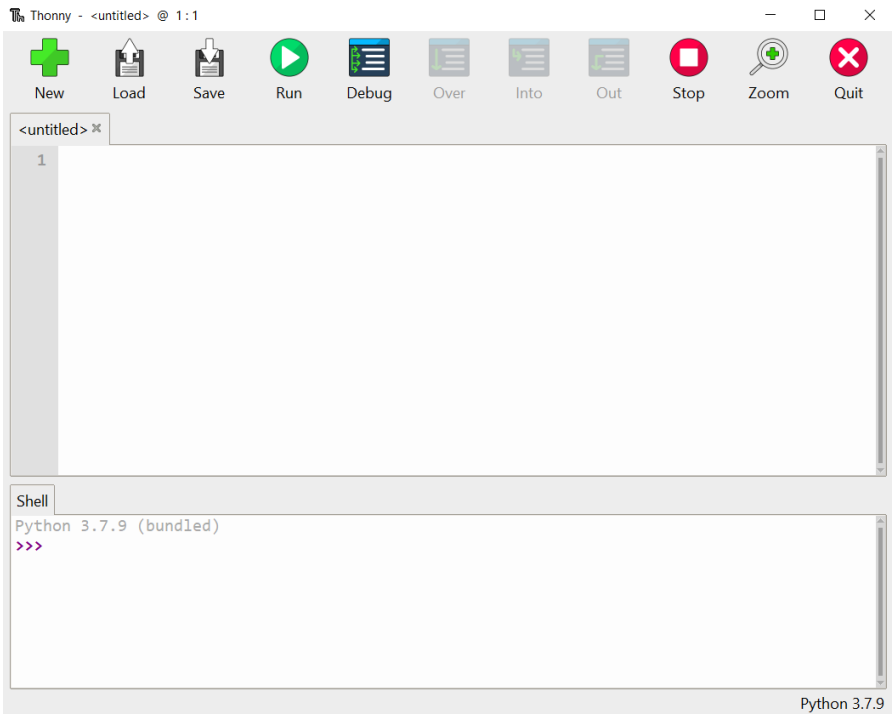


Figura 2.11: Ambiente di sviluppo Thonny Python IDE

Il linguaggio Python prevede l'uso dell'indentazione che, a differenza di altri linguaggi, evita l'utilizzo di parentesi graffe per delimitare blocchi di codice. Lo script viene riportato di seguito (si nota la presenza del *backslash* come carattere speciale che permette al codice di eseguire un'istruzione presente su più righe ed è introdotto per rendere più chiara la lettura del codice).

```
import urllib.request
import RPi.GPIO as GPIO
import time
```

Si utilizza la funzione `import` per caricare le librerie di funzioni esterne che vengono utilizzate all'interno del programma. In Python tali librerie sono dette "moduli". Il modulo `urllib.request` definisce le funzioni che permettono di aprire l'URL. Importando il modulo `RPi.GPIO` si possono utilizzare i pin del Raspberry sia come input che come output. La sintassi `import as` è utilizzata per rinominare ed abbreviare il modulo in `GPIO` che risulta più semplice da richiamare nel programma. Il modulo `time` infine fornisce varie funzioni relative al tempo.

```
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(23, GPIO.IN)
GPIO.setup(16, GPIO.IN)
GPIO.setup(24, GPIO.IN)
GPIO.setup(17, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(22, GPIO.OUT, initial=GPIO.LOW)
```

Tramite `GPIO.setmode(GPIO.BCM)` si indica che la numerazione dei pin non segue quella fisica del Raspberry, bensì la numerazione *Broadcom* riportata sul *pinout*. Con `GPIO.setup` si impostano i pin di ingresso e di uscita utilizzati del microcontrollore. Nello specifico i pin di input 23, 16 e 24 ricevono rispettivamente il segnale in uscita dal sensore, il segnale inviato a seguito della pressione del pulsante di presenza scarto e il segnale generato dalla pressione del pulsante associato all'inizio della lavorazione su un nuovo codice (si precisa che con il termine "codice" si intende un pezzo con una determinata lavorazione che viene appunto identificato da un codice alfanumerico). I pin di output 17 e 22 erogano la tensione per i due LED presenti nel circuito. Il pin 17 è responsabile del lampeggiamento del LED verde associato alla presenza del pezzo; il pin 22 è utilizzato per il lampeggiamento del LED rosso a seguito della pressione del pulsante "di scarto". I pin di uscita sono inoltre inizializzati al livello logico basso tramite `initial=GPIO.LOW`.

```
count_1 = 0
scarto_1 = 0
```

La variabile `count_1` è responsabile del conteggio effettivo dei pezzi; la variabile `scarto_1` si occupa invece di memorizzare il numero di scarti prodotti.

```
while True:
    pin1 = GPIO.input(23)
    pin2 = GPIO.input(16)
    pin0 = GPIO.input(24)
```

Il corpo centrale del codice viene inserito in un loop infinito definito all'interno di `while True`. Si assegnano i valori dei pin di input a tre variabili denominate `pin0`, `pin1` e `pin2`.

```
    if(pin0):
        count_1 = 0
```

```
        scarto_1 = 0
```

Il codice presente nel primo *if* permette di azzerare le variabili di conteggio dei pezzi prodotti e degli scarti una volta premuto il pulsante associato alla variabile *pin0* che di conseguenza si porta a livello logico alto. Questa operazione viene effettuata dall'operatore ad inizio lavorazione ed ogni volta che la macchina è attrezzata per la lavorazione di un nuovo codice. Prima di iniziare la lavorazione, quindi, l'operatore invia tramite tablet, con l'apposita app, al database la nuova riga in tabella le cui colonne *count* e *scarto* verranno aggiornate con i successivi conteggi.

```
    if(pin1):
        GPIO.wait_for_edge(23, GPIO.FALLING)
        count_1 = count_1 + 1
        print("contatore:")
        print(count_1)
        count= "%d" %count_1
        with urllib.request.urlopen("http://192.168.1.53:8080/\
        phpmyadmin/prod/aggiorna.php?count="+count+"")as response:
            html = response.read()
        with urllib.request.urlopen("http://192.168.1.53:8080/\
        phpmyadmin/prod/datafine.php") as response:
            html = response.read()
        GPIO.output(17, GPIO.HIGH)
        time.sleep(1)
        GPIO.output(17, GPIO.LOW)
        time.sleep(1)
    time.sleep(.1)
```

Quando il livello logico del pin 23 (associato alla variabile *pin1*) è alto, ovvero quando il pin riceve il segnale di uscita dal sensore, il codice entra nel secondo *if*. L'istruzione `GPIO.wait_for_edge(23, GPIO.FALLING)` permette, grazie all'importazione del modulo GPIO, di attendere il fronte di discesa del segnale derivante dal sensore e associato alla presenza di un oggetto. Di conseguenza le righe di codice successive, all'interno dello stesso *if*, vengono eseguite solamente dopo che il pezzo viene tolto dalla traiettoria del fascio di luce permettendo che, anche in caso di sosta prolungata di un pezzo davanti al sensore, il contatore incrementi di una sola unità.

La variabile `count_1` associata quindi al conteggio dei pezzi aumenta ed il valore corrispondente viene stampato sulla *shell*.

A questo punto avviene la parte più importante adibita all'invio del dato al database. La funzione `urllib.request.urlopen` apre quindi l'URL `http://192.168.1.53:8080/phpmyadmin/prod/aggiorna.php?count="+count+"` in cui si nota:

- il protocollo HTTP per l'accesso al server;
- l'host che identifica il server su cui risiede la risorsa (192.168.1.53);
- la porta del servizio di rete al quale inoltrare la richiesta (8080);

- il percorso del file in linguaggio PHP `aggiorna.php`;
- la *query string* separata dal "?" che consente di passare al server il parametro che ha come valore quello all'interno della variabile `count`.

Il protocollo HTTP si basa su richieste e risposte: il client effettua la richiesta ed il server invia la risposta. La funzione `urllib.request` rispecchia questo concetto tramite la creazione di un *request object* che rappresenta la richiesta effettuata e restituendo un *response object* che viene letto tramite l'istruzione `response.read()`.

Allo stesso modo viene passato il file in PHP `datafine.php` che non presenta alcuna *query string* all'interno dell'URL.

Sempre all'interno dell'*if* si ha il lampeggiamento del LED verde: si utilizza l'istruzione `GPIO.output(17, GPIO.HIGH)` per impostare il pin di uscita a livello alto e l'istruzione `GPIO.output(17, GPIO.LOW)` per impostare lo stesso pin a livello basso. Il tempo di ritardo dell'esecuzione del programma viene introdotto tramite la funzione `time.sleep` all'interno della quale viene inserto il tempo espresso in secondi. Quest'ultima funzione è utilizzata sia per gestire i tempi di accensione e spegnimento del LED sia per gestire il tempo di rilevamento del pezzo dal sensore.

```
if(pin2):
    scarto_1 = scarto_1 + 1
    print("scarto:")
    print(scarto_1)
    count_1 = count_1 - 1
    count= "%d" %count_1
    print("contatore:")
    print(count_1)
    scarto = "%d" %scarto_1
    with urllib.request.urlopen("http://192.168.1.53:8080/\
    phpmyadmin/prod/aggiornascarto.php?\
    count="+count+"&scarto="+scarto+"") as response:
        html = response.read()
    GPIO.output(22, GPIO.HIGH)
    time.sleep(1)
    GPIO.output(22, GPIO.LOW)
    time.sleep(1)
time.sleep(.2)
```

Il programma entra nel terzo *if* ogni volta che viene rilevato uno scarto e quindi viene premuto il pulsante associato alla variabile `pin2`. La variabile `scarto_1` viene incrementata, mentre quella di *contapezzo* `count_1` viene decrementata. Questo passaggio è implementato nel codice in quanto il sensore rileva la presenza del pezzo appena finisce il ciclo di lavorazione incrementando così la variabile di conteggio a prescindere che il pezzo sia stato lavorato correttamente dalla macchina o meno. Dopo che il pezzo viene prelevato e controllato dall'operatore si stabilisce se è stato prodotto un pezzo conforme o

di scarto: nel caso si verifichi quest'ultima opzione, l'operatore, tramite la pressione di un pulsante, incrementa il contatore di scarto decrementando quella di conteggio dei pezzi.

Analogamente a come avviene all'interno del precedente *if* viene aperto l'indirizzo che contiene il file `aggiornascarto.php`. A differenza dei casi precedenti, però, la *query string* è costituita da due parametri che, separati da un "*&*", passano al server i relativi valori associati alle variabili `count` e `scarto`. Si ha inoltre il lampeggiamento di un LED rosso identificativo della presenza dello scarto.

2.5 Invio dei dati

Prima di procedere alla spiegazione dei file PHP, introdotti nello script in Python e che permettono di inviare i dati al database, è bene fare una breve descrizione del server aziendale.

2.5.1 Cenni su XAMPP

XAMPP è una piattaforma software open source che rende la creazione di web server molto semplice ed è un acronimo che sta per:

- X: Cross-Platform, indica che è una multi-piattaforma funzionante su ambienti Linux, Mac e Windows.
- A: Apache;
- M: MySQL;
- P: PHP;
- P: Perl.

"Apache httpd server" è un server web ovvero un sistema informatico che si occupa di rispondere a richieste di risorse usando il protocollo HTTP (acronimo che sta per HyperText Transfer Protocol) che utilizza la porta TCP 8080. Il Web Server Apache presenta un'architettura modulare in cui ad ogni richiesta del client vengono svolte funzioni specifiche da ogni modulo. Le principali fasi di cui è composto un ciclo sono:

- Translation: traduce la richiesta del client;
- Access control: controlla le richieste in base ai criteri di autorizzazione;
- MIME Type: identifica il tipo di contenuto e decide quali moduli possono contribuire a servire la richiesta;
- Response: invia la risposta al client e attiva eventuali procedure;
- Logging: tiene traccia di tutto ciò che è stato fatto.

Alcuni dei contributi fondamentali che offrono i sistemi informatici sono la memorizzazione e la conservazione duratura dei dati i quali possono essere recuperati in modo efficiente e sottoposti a molteplici analisi. Questo ruolo viene interpretato molto bene da un prodotto software completo, efficiente ed affidabile: MySQL (logo riportato in Figura 2.12).

Figura 2.12: Logo MySQL

I concetti centrali in tema di memorizzazione dei dati sono il database e il DBMS (Database Management System). Un database indica un sistema di file finalizzato a memorizzare informazioni la cui struttura interna deve rispettare una certa architettura di immagazzinamento dei dati per poterne permettere, non solo il corretto salvataggio, ma anche agevolarne il recupero.

Un DBMS è un servizio software realizzato in genere come server in esecuzione continua in grado di gestire uno o più database. I programmi che devono interagire con il database non possono farlo direttamente, ma devono dialogare con il DBMS, che rappresenta l'unica entità che può accedere fisicamente alle informazioni. Il DBMS è quindi il componente che si occupa di tutte le politiche di accesso, gestione, sicurezza ed ottimizzazione dei database.

Alcuni dei DBMS più diffusi sono i RDBMS (Relational DBMS), governati dalla teoria relazionale. MySQL stesso è un sistema open source di gestione di database relazionali. Il funzionamento di MySQL si base quindi su due concetti fondamentali:

- Database relazionale;
- Modello client-server.

Il database relazionale è costituito da tabelle, ognuna delle quali è composta da righe identificate da un codice univoco denominato "chiave". Le tabelle che compongono il database non sono del tutto indipendenti tra loro ma relazionate da legami logici. Se si cercasse infatti di raggruppare i dati in un unico grande recipiente si avrebbero problemi come:

- Dati duplicati;
- Nessuna organizzazione.

Ciascuna tabella ha una colonna chiamata *id* contenente l'identificativo univoco che corrisponde alla chiave primaria che permette di riconoscere una riga senza possibilità di confusione.

Tramite il linguaggio SQL (Structured Query Language) si indica al DBMS quali operazioni svolgere sui database che gestisce. Si può attivare qualsiasi tipo di operazione, sia sui dati, sia sulla struttura interna del database, sebbene le principali (e più frequenti) operazioni ricadano in una delle seguenti quattro tipologie: inserimento, lettura, modifica e cancellazione di dati tipicamente indicate con l'acronimo CRUD (Create-Read-Update-Delete).

Tra le caratteristiche di MySQL si evidenzia:

- l'alta efficienza nonostante le moli di dati affidate;

- l'alta capacità di integrazione con i principali linguaggi di programmazione e ambienti di sviluppo.

Oltre ad essere un sistema di database relazionale, MySQL utilizza anche un modello client-server. Il server è il luogo in cui risiedono effettivamente i dati ai quali è possibile accedere solamente richiedendoli. Utilizzando il linguaggio SQL (Structured Query Language) il client invia una richiesta al server del database per ricevere i dati di cui ha bisogno. Per amministrare i database MySQL si ricorre ad un'interfaccia web denominata phpMyAdmin: uno strumento *web-based* molto diffuso che rappresenta un progetto open source realizzato in PHP.

Tramite phpMyAdmin è possibile visualizzare impostazioni e dati di un database MySQL e quindi è possibile creare, modificare, cancellare tabelle (o righe delle stesse) ed effettuare esportazione di dati. In Figura 2.13 si riporta la schermata di accesso in cui immettere le credenziali ed in Figura 2.14 l'interfaccia di phpMyAdmin.

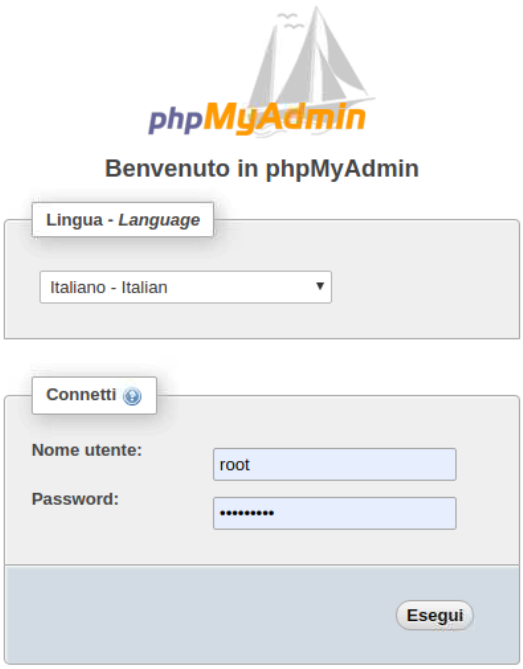


Figura 2.13: Schermata di login per phpMyadmin

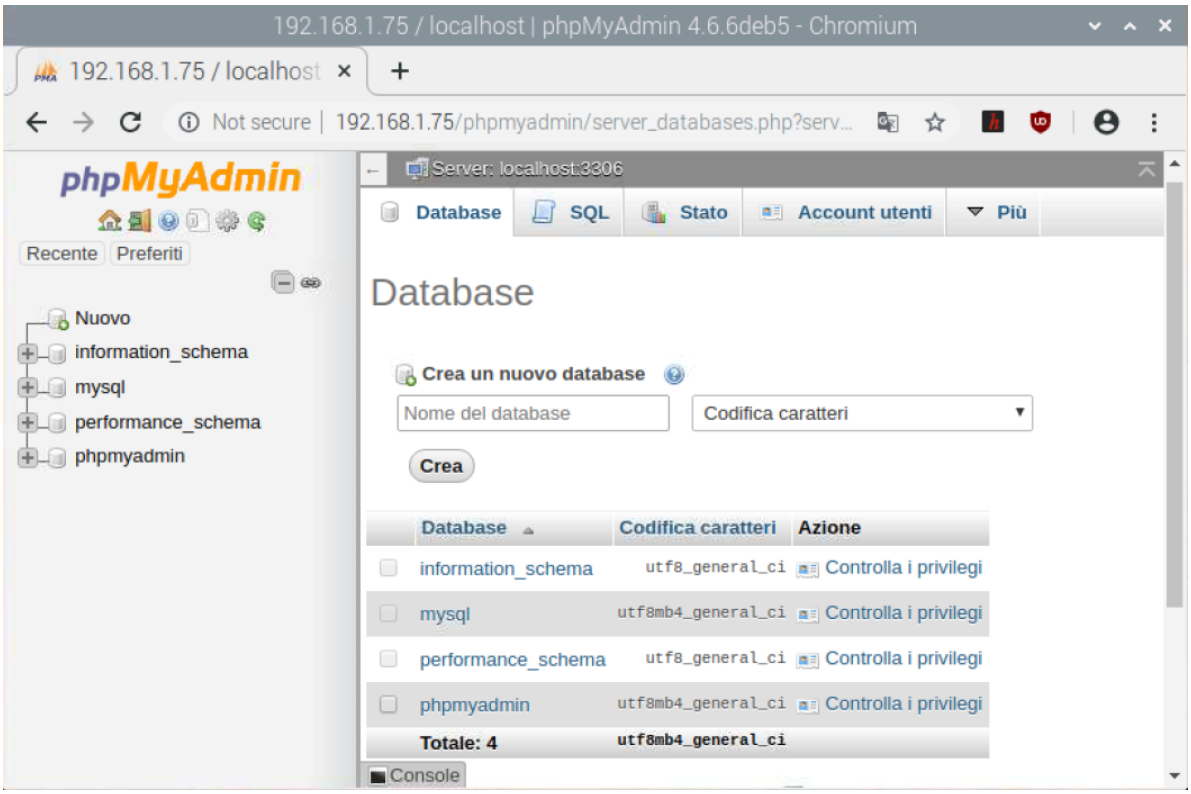


Figura 2.14: Schermata principale di phpMyadmin

Ogni operazione eseguita sul Web coinvolge un client ed un server: il client è un dispositivo (ad esempio un browser) che effettua una richiesta ad un server remoto che, attraverso un linguaggio come il PHP, interpreta la richiesta ed invia una risposta al client che a sua volta fornisce all'utente: nel caso di browser, il client riceve una pagina HTML che mostra all'utente.

PHP è un linguaggio che risiede in un server in remoto e che, in fase di esecuzione, interpreta le informazioni ricevute da un client grazie al Web Server e le elabora restituendo un risultato al client stesso. Semplificando il più possibile, le azioni che vengono compiute sono le seguenti:

- il browser (client) effettua una richiesta al server remoto tramite il protocollo http;
- il Web server riceve la richiesta e la interpreta verificando che i dati inviati siano corretti;
- il Web server restituisce una risposta al client che potrà essere diversa a seconda del fatto che la risorsa richiesta sia disponibile o meno.

Quello di cui si occupa il PHP è appunto elaborare i dati veicolati tramite una richiesta http e fornire un'adeguata risposta al client. Il passaggio delle informazioni viene riassunto in Figura 2.15.

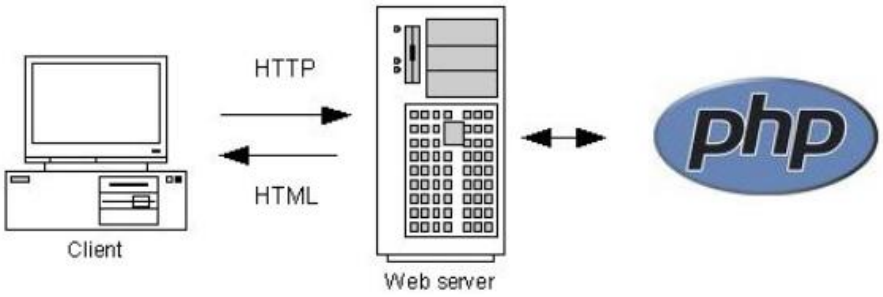


Figura 2.15: Richiesta client/server

La specifica HTTP definisce 9 tipi di metodi: i più diffusi sono il metodo GET ed il metodo POST. GET è il metodo con cui vengono richieste la maggior parte delle informazioni ad un Web server. Tali richieste vengono veicolate tramite *query string*, cioè la parte di un URL che contiene dei parametri da passare in input ad un'applicazione. Il metodo POST invece consente di inviare dati ad un server senza mostrarli in *query string*. Esistono quindi meccanismi che permettono all'utente, oltre di richiedere una pagina ad un web server, anche di specificare determinati parametri che sono utilizzati dallo script PHP per determinare quali contenuti la pagina deve mostrare. Uno dei sistemi utilizzato per passare dati ad uno script è appunto il metodo GET.

Il metodo GET in pratica consiste nell'accodare i dati all'indirizzo della pagina richiesta facendo seguire il nome della pagina da un punto interrogativo e dalle coppie *nome/valore* dei dati di interesse. Nome e valore sono separati da un segno di uguale. In caso di presenza di più coppie s'interpongono delle '&'. La stringa che si trova dopo il punto interrogativo, contenente nomi e valori dei parametri, viene detta *query string*. I valori contenuti nella *query string* vengono memorizzati dal PHP nell'array $_GET che è un array super-globale.

2.5.2 Database e script in PHP

Prima di analizzare i file in PHP che vengono passati all'interno del codice, si procede alla definizione del database e delle relative tabelle all'interno dell'ambiente phpMyAdmin. Per prima cosa si crea il database denominato "produzione". Ogni database deve contenere delle tabelle in cui vengono memorizzati i dati. A tal proposito si crea la tabella "prod" definendo anche il numero di campi, che in questo caso è 5. Una volta creata la tabella, phpMyAdmin offre la possibilità di definire le voci per ciascuna colonna inserita. Nella Figura 2.16 si riporta a titolo di esempio la tabella senza attributi.

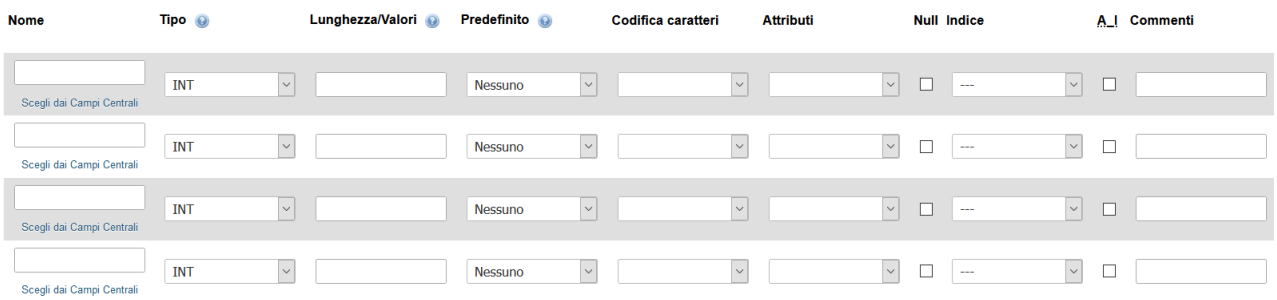


Figura 2.16: Campi della tabella in phpMyAdmin senza attributi

Si nota la presenza delle seguenti voci:

- Nome: indica il nome attribuito alla colonna della tabella;

- Tipo: indica il tipo di dati salvato nella colonna. Permette ad esempio di definire i dati come numeri interi, virgola mobile, inserire indicazioni temporali o date;

- Lunghezza/Valori: formattazione opzionale che permette di assegnare una lunghezza massima ai valori presenti;

- Predefinito: consente di definire un valore standard per una colonna, che viene inserito automaticamente ogni qual volta una registrazione non contiene alcun valore per la relativa colonna;

- Codifica caratteri: permette di definire un tipo di carattere specifico che può variare dalle impostazioni globali del database;

- Attributi: permette di specificare attributi opzionali. Ad esempio "signed" e "unsigned" indicano rispettivamente che un certo tipo di dato può contenere numeri anche negativi o solo positivi;

- Indice: definisce le regole per l'indicizzazione;

- A_I: tale abbreviazione, che sta per "AUTO_INCREMENT", se spuntata, permette di conteggiare un valore automaticamente;

- Commenti: tale campo consente di inserire un commento nella colonna.

Per una maggiore comprensione si riporta in Figura 2.17 le varie scelte che vengono effettuate per le voci all'interno della tabella *prod*.

	#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito	Commenti	Extra	Azione
☐	1	**id** 🔑	int(11)			No	*Nessuno*		AUTO_INCREMENT	Modifica Elimina Più
☐	2	**codice**	varchar(30)	latin1_swedish_ci		No	*Nessuno*			Modifica Elimina Più
☐	3	**macchina**	varchar(30)	latin1_swedish_ci		No	*Nessuno*			Modifica Elimina Più
☐	4	**data_avvio**	datetime			No	current_timestamp()			Modifica Elimina Più
☐	5	**count**	int(11)			Sì	*NULL*			Modifica Elimina Più
☐	6	**data_fine**	timestamp			No	current_timestamp()			Modifica Elimina Più
☐	7	**scarto**	int(11)			No	*Nessuno*			Modifica Elimina Più
☐	8	**fase**	varchar(30)	latin1_swedish_ci		No	*Nessuno*			Modifica Elimina Più
☐	9	**descr_fase**	varchar(100)	latin1_swedish_ci		No	*Nessuno*			Modifica Elimina Più

Figura 2.17: Campi della tabella "prod" in phpMyAdmin

Le colonne introdotte sono:

- id: numero di identificazione che definisce in modo univoco ciascuna riga di dati memorizzata. I valori assunti da tale colonna sono "int" (abbreviazione di "integer", che indica un numero intero). Alla voce "index" si sceglie "primary" in modo da identificare tale colonna come chiave primaria della tabella. Gli *id* per ciascuna registrazione vengono inoltre generati automaticamente con numeri progressivi tramite la spunta su "A_I".

- codice: indica il codice alfanumerico identificativo del pezzo sottoposto a lavorazione. Il tipo di dato con cui è definita questa colonna è un "varchar", cioè una sequenza di caratteri di lunghezza variabile (lunghezza massima impostata a 30 caratteri).

- macchina: indica il numero identificativo della macchina di produzione definito anch'esso come *varchar* con lunghezza massima pari a 30.

- data_avvio: corrisponde alla data inviata ad inizio ciclo produttivo per un determinato codice di produzione. Il tipo di dato della colonna viene definito come *datetime* in cui l'indicazione della data è del tipo "YYYY-MM-DD HH:MM:SS" dove YYYY-MM-DD indica l'anno, il mese ed il giorno, mentre HH:MM:SS indica l'ora, i minuti ed i secondi.

- count: identifica la variabile per il conteggio dei pezzi che si aggiorna ad ogni incremento. Il tipo di dato utilizzato è un *int*.

- data_fine: è la data identificativa del pezzo prodotto e, come per la colonna *count*, si aggiorna ad ogni pezzo contato. Come per data_avvio il tipo utilizzato è un *datetime*.

- scarto: identifica il contatore dei pezzi di scarto ed è definito come numero intero (*int*).

- fase: è un *varchar* che rappresenta il numero della fase di lavorazione che avviene sul pezzo.

- descr_fase: indica una eventuale descrizione aggiuntiva della fase

Per quanto riguarda le voci *codice, macchina, data_avvio, fase* e *descr_fase* si utilizza un'apposita app che permette di inviare, a inizio lavorazione, i dati relativi a ciascuna voce. Per popolare le voci *count, data_fine* e *scarto* si utilizzano invece i file in PHP passati tramite URL nello script in Python descritto nel paragrafo precedente. Per poter colloquiare con MySQL si utilizza l'estensione MySQLi (MySQL improved) che, a differenza di altre estensioni, permette di comunicare solamente con il DBMS MySQL.

Il file *aggiorna.php* si occupa di passare il valore della variabile *count* che memorizza il conteggio dei pezzi. Lo script in PHP è il seguente:

```
1. <?php
2. $connect =
mysqli_connect("localhost" , "root" , "" , "produzione");
3. mysqli_query($connect , "UPDATE prod SET count =
'" . $_GET [ "count" ]. "' WHERE id = (SELECT MAX(id) FROM prod
WHERE macchina = 076)" );
4. ?>
```

Si commentano nel dettaglio le righe di codice:

Riga 1 e *4*: contengono rispettivamente i tag di apertura e chiusura del linguaggio. L'interprete PHP esegue solo il codice contenuto tra questi limiti.

Riga 2: effettua la connessione al database. I parametri che vengono passati prevedono il "localhost" che identifica l'host del server; "root" e " " identificano rispettivamente nome utente e password (stringa vuota) per accedere a phpMyAdmin ed infine "produzione" che il nome del database a cui si vuole accedere.

Riga 3: tramite questa stringa viene selezionata la tabella "prod" e viene aggiornato il valore già esistente della colonna "count" (per la prima registrazione è presente il valore di default) tramite l'utilizzo dell'array super-globale `$_GET`. Per spiegare le condizioni introdotte dai "WHERE" è bene introdurre la logica con cui lavora la macchina e quindi l'operatore.

La macchina 076, a cui viene si pensa di affiancare il sensore laser, può effettuare lavorazioni su differenti codici. Ogni qual volta il codice da lavorare cambia, la macchina viene attrezzata e viene caricato l'opportuno programma di lavorazione. L'operatore, di

conseguenza, effettua un primo invio dei dati tramite tablet, in cui è presente l'applicazione che permette di inviare la stringa contenente il codice da lavorare. Quindi una volta che il pezzo viene contato dal sensore, l'aggiornamento della colonna "count" avviene selezionando il massimo valore di "id" che corrisponde all'ultima riga inserita in tabella. Così facendo viene aggiornata la stessa riga fino a quando non viene inviato dall'operatore un nuovo codice che rappresenterà una nuova riga da aggiornare.

Il file *datafine.php* si occupa dell'invio della data di fine della singola lavorazione che coincide con quella di rilevamento del pezzo. Lo script in PHP viene riportato di seguito.

```
1. <?php
2. $connect=mysqli_connect("localhost","root","","produzione");
3. $today = date("Y-m-d H:i:s");
4. mysqli_query($connect,"UPDATE prod SET data_fine='$today' WHERE
 id=(SELECT MAX(id) FROM prod WHERE macchina = 076) ");
5. ?>
```

Vengono commentate le righe del codice:

Riga 1 e *5*: contengono i tag di apertura e chiusura del linguaggio.

Riga 2: effettua la connessione al database.

Riga 3: si definisce la variabile `$today` che memorizza la data corrente. Quest'ultima è introdotta tramite la funzione `date()` all'interno della quale viene passato il formato voluto: "Y" indica l'anno nel formato a quattro cifre; "m" indica il mese; "d" indica il giorno; "H" indica le ore; "i" indica i minuti ed infine "s" indica i secondi.

Riga 4: viene aggiornata la colonna "data_fine" con il valore all'interno della variabile `$today`. Il metodo con cui viene aggiornato è lo stesso visto per la memorizzazione di "count" dello script "aggiorna.php".

L'ultimo file che viene passato nello script in Python per l'invio dei dati è *aggiornascarto.php*. Questo file permette di aggiornare il dato memorizzato dal contatore degli scarti ed allo stesso tempo aggiorna quello relativo al contatore di pezzi "buoni". Infatti il segnale proveniente dal sensore permette di incrementare l'opportuna variabile di conteggio a prescindere che il pezzo sia conforme o meno. Solo in un secondo momento, quando l'operatore si accorge che il pezzo è da scartare, viene incrementato lo scarto e quindi decrementato il contatore dei pezzi buoni.

Il codice in PHP è il seguente:

```
1. <?php
2. $connect=mysqli_connect("localhost","root","","produzione");
3. mysqli_query($connect,"UPDATE prod SET count='".$_GET["count"].
"',scarto='".$_GET["scarto"]."' WHERE id = (SELECT MAX(id)
FROM prod WHERE macchina = 076)");
4. ?>
```

Di seguito si commentano le righe dello script.

Riga 1 e *4*: contengono i tag di apertura e chiusura del linguaggio;

Riga 2: effettua la connessione al database;

Riga 3: effettua l'aggiornamento dei dati precedentemente descritti, quindi appartenenti alla colonna "count" e "scarto" in cui sono presenti le stesse condizioni già viste prime.

2.6 Risultati

In Figura 2.18, 2.19 e 2.20 vengono mostrate le configurazioni assunte dal sistema *contapezzi* realizzato su breadboard. In Figura 2.18 si è nel caso in cui, con trasmettitore e ricevitore allineati, non si ha la rilevazione dell'oggetto e non si preme il pulsante associato allo scarto: i LED sono spenti. In Figura 2.19 si ha la rilevazione dell'oggetto che, grazie all'introduzione del codice del rilevamento del fronte di discesa del segnale, permette di incrementare il contatore e di inviare il dato solo una volta che esce dal fascio di luce del sensore: si ha quindi un lampeggiamento del LED verde. In Figura 2.20 si simula la pressione del pulsante da parte dell'operatore a seguito dell'individuazione di un pezzo di scarto: si ha l'accensione e lo spegnimento del LED rosso.

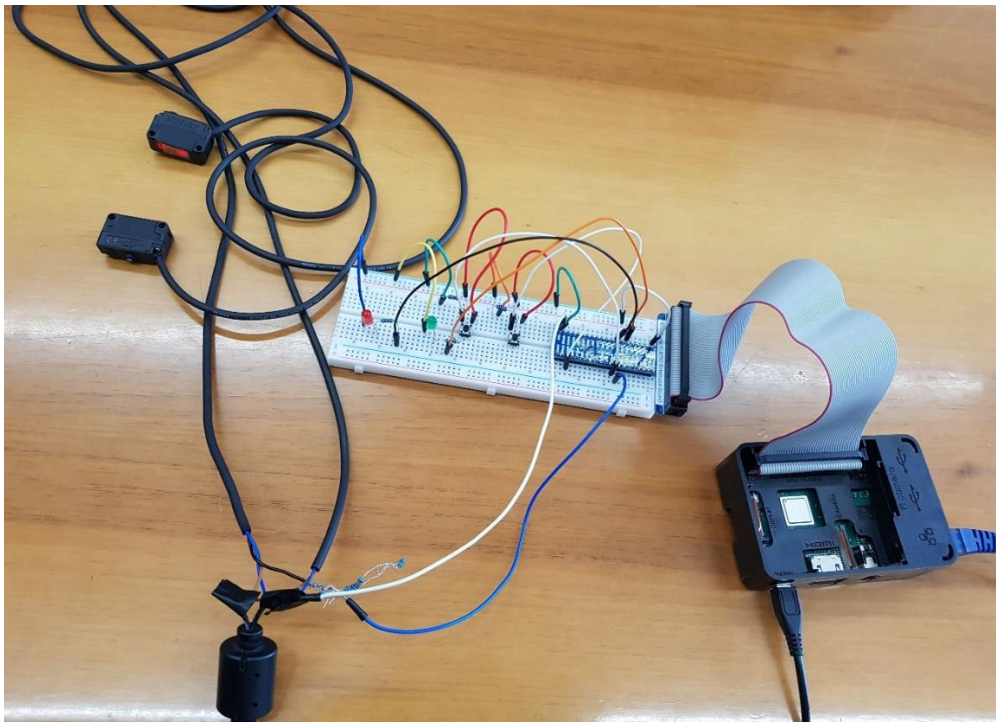


Figura 2.18: Sistema contapezzi realizzato su breadboard in configurazione neutra

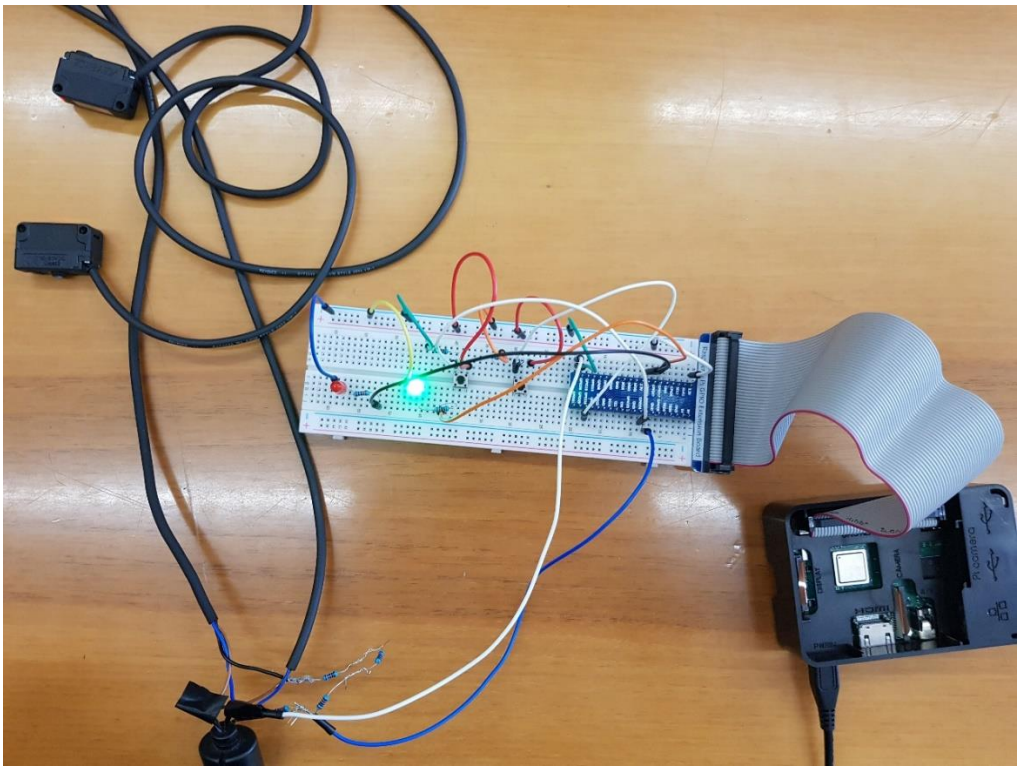


Figura 2.19: Sistema contapezzi realizzato su breadboard in presenza di rilevamento dell'oggetto

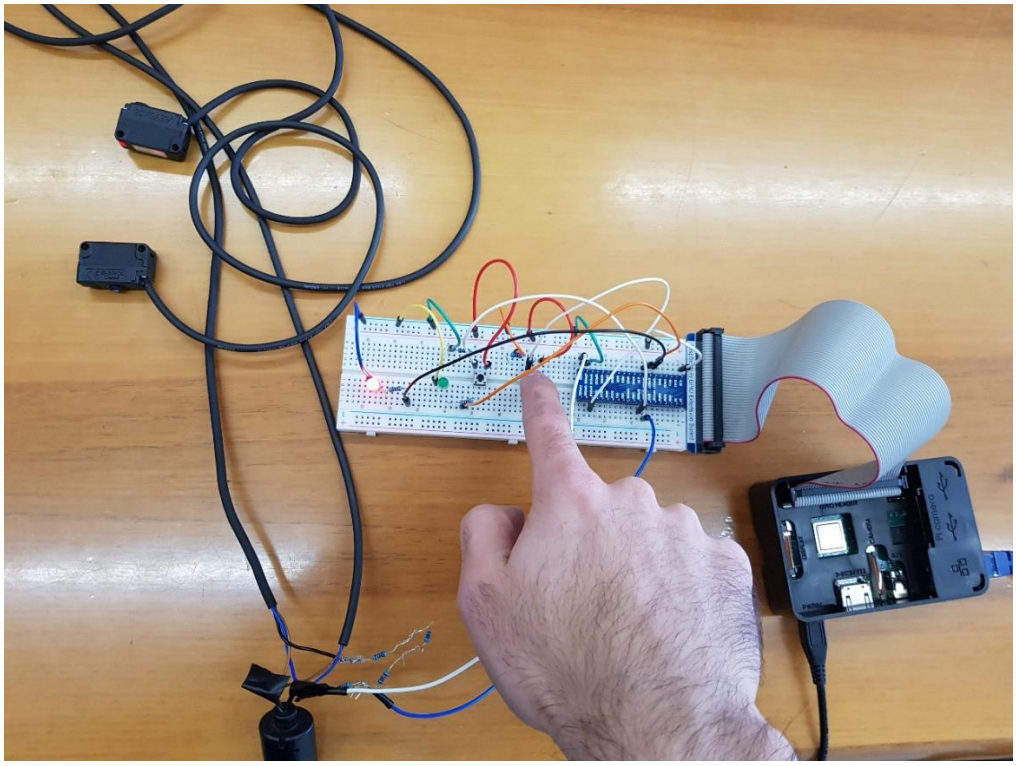


Figura 2.20: Sistema contapezzi realizzato su breadboard in presenza dello scarto

In Figura 2.21 viene riportato un esempio di come avviene la registrazione dei dati all'interno della tabella *prod* del database *produzione*: in particolare ad inizio lavorazione su un nuovo pezzo l'operatore, tramite app, invia all'interno del database la riga della tabella

contenente i valori opportuni di *codice*, *macchina*, *data_avvio*, *fase* e *descr_fase*. Quindi ad ogni ciclo di lavorazione viene aggiornata in modo automatizzato la variabile *count* e di *data_fine*. In caso di presenza di uno scarto l'operatore invia tramite la pressione del pulsante il valore presente nell'omonima colonna in tabella decrementando quindi il valore di *count*.

Grazie alla presenza del tablet l'operatore può vedere a schermo in tempo reale i valori del contatore dei pezzi buoni e degli scarti.

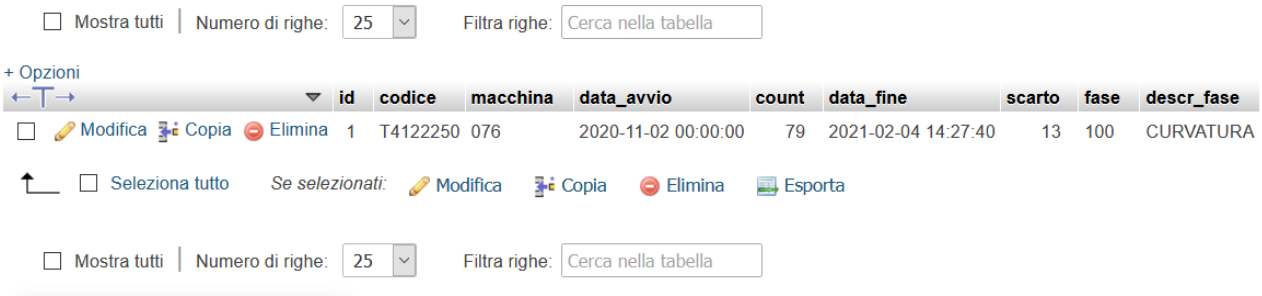


Figura 2.21: Immagazzinamento dei dati di conteggio pezzi e scarti

Lo studio e la realizzazione del sistema di conteggio dei pezzi ha lo scopo di poter essere oggetto di sperimentazione il prima possibile. La macchina su cui si ha intenzione di effettuare l'applicazione è la curvatubi tipo CRIPPA 1042E (mostrata in Figura 2.22) che, per le dinamiche delle lavorazioni attuali effettuate dall'azienda, piega giornalmente un numero di tubi considerevole.

Figura 2.22: Curvatubi CRIPPA 1042E

Si tratta di una macchina a controllo numerico per tubi con diametri fino a 42 mm e con spessori fino a 2 mm. Il movimento a crociera della testa permette di realizzare un ciclo completamente automatico che garantisce un'ottima precisione ed una maggior produttività. I movimenti della macchina, che avvengono su guide a ricircolo di rulli garantendo usure inferiori rispetto alle guide tradizionali, sono governati da 10 assi controllati azionati azionati da motori con tecnologia *Full Electric* e gestiti dal *CNC Siemens 840D sl*.

Caratteristiche tecniche Curvatubi CRIPPA 1042E:

Senso di curvatura: Destro e sinistro

Materiale: Acciaio

Max Diametro del tubo: Fino a Ø42x2mm

Modello CNC: Siemens 840D sl

Numero assi CNC: 10

Raggio Max Curvatura: 150 mm

Lunghezza utile Max: 1600/3300 mm

Velocità di curvatura: 240°/sec

Precisione curvatura: +/- 0,05°

Velocità asse Y: 85 m/min

Precisione di avanzamento: +/-0,05 mm

Potenza installata: 20 kW

Peso della macchina: 3600 kg

Dimensioni di ingombro (H/W/D): 1870/2200/4850 mm

Per il caso applicativo, non oggetto del lavoro di tesi, si dovrà pensare ad un'elaborazione del ciclo macchina che permetta, a fine lavorazione, di far cadere il tubo piegato su un nastro trasportatore e poter essere individuato dal sensore fotoelettrico.

3. Stazione di controllo diametri

Il secondo oggetto di studio è un sistema di misurazione adibito al controllo dei diametri esterni dei tubi. Lo scopo è permettere che vengano effettuati controlli a campione in modo rapido ed allo stesso tempo che si possa disporre di uno *storage* di misurazioni che possano essere oggetto di analisi.

Il sistema è studiato per poter essere affiancato non solo ai già presenti controlli effettuati tramite calibri a tampone durante il ciclo produttivo dei pezzi, ma anche alle misurazioni tramite calibri a corsoio per lo più presenti nelle postazioni finali di controllo.

L'introduzione di un micrometro laser è molto interessante principalmente per due aspetti: riduce al minimo gli errori che gli operatori possono commettere durante le misurazioni effettuate tramite i calibri a corsoio e permette, tramite il sistema a cui è collegato, di memorizzare in un database tutte le misurazioni effettuate. Fondamentalmente il condizionatore del micrometro laser invia un segnale analogico che viene elaborato da un Fishino che, tramite un programma scritto in C++, registra le misurazioni in un database.

3.1 Principio di trasduzione

La struttura di un generico trasmettitore, riportato in Figura 3.1, è costituita da:

- Un sensore, che trasforma tramite una legge fisica la variabile di processo in ingresso in un'altra grandezza fisica misurabile;
- Un trasduttore, che converte la grandezza fisica misurabile in un segnale in uscita di natura elettrica;
- Un condizionatore (amplificatore), che converte la variabile in uscita dal trasduttore in un segnale normalizzato. Esso svolge non solo la funzione di normalizzazione del segnale ma anche di amplificazione (il segnale di uscita dal trasduttore ha tipicamente valori molto bassi ed inferiori ai $100\,mV$), compensazione dei disturbi, linearizzazione, variazione del guadagno, variazione dello zero ed alimentazione del trasduttore.

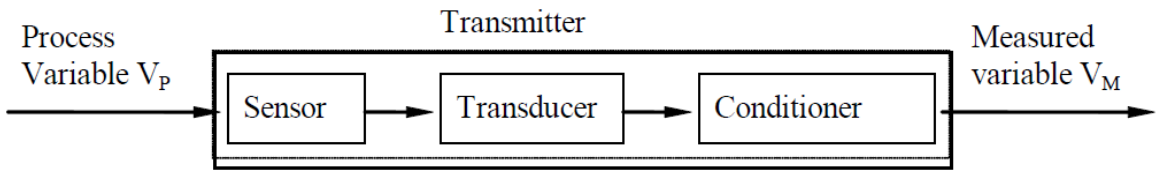


Figura 3.1: Struttura trasmettitore (Figura tratta da slide "Servosystems structure" – Massimo Sorli)

Lo strumento di misura utilizzato per la stazione di controllo dei diametri è un micrometro laser multifunzione costituito da un sensore laser a sbarramento KEYENCE IG-028, che comprende un emettitore ed un ricevitore (Figura 3.2), ed un amplificatore KEYENCE IG-1000 (Figura 3.3).

Figura 3.2: Micrometro laser KEYENCE IG-028

Figura 3.3: Amplificatore IG-1000 del micrometro laser

Le possibili modalità di misurazione disponibili dal micrometro sono:

- Modalità di controllo bordo o posizionamento;
- Modalità di misurazione diametri esterni o larghezze;
- Modalità di misurazione diametri interni o aperture;
- Modalità bordo vetro.

Nel caso in esame viene impostata la modalità di misurazione per diametri esterni poiché l'applicazione è prevista per il controllo delle lavorazioni che portano una variazione del diametro esterno del tubo. La versatilità dello strumento di misura permette comunque di cambiare la modalità di misura tramite l'apposito amplificatore in modo molto semplice.

3.1.1 Caratteristiche del micrometro laser KEYENCE IG-028

Una delle caratteristiche peculiari del sensore IG-028 è la presenza di un laser a più lunghezza d'onda che permette di avere un fascio luminoso con una distribuzione di intensità più uniforme rispetto ai tradizionali sensori laser che, a causa di una singola lunghezza d'onda, producono un'intensità del fascio luminoso irregolare. In Figura 3.4 sono rappresentati le parti fondamentali del trasmettitore e del ricevitore responsabili del funzionamento del micrometro.

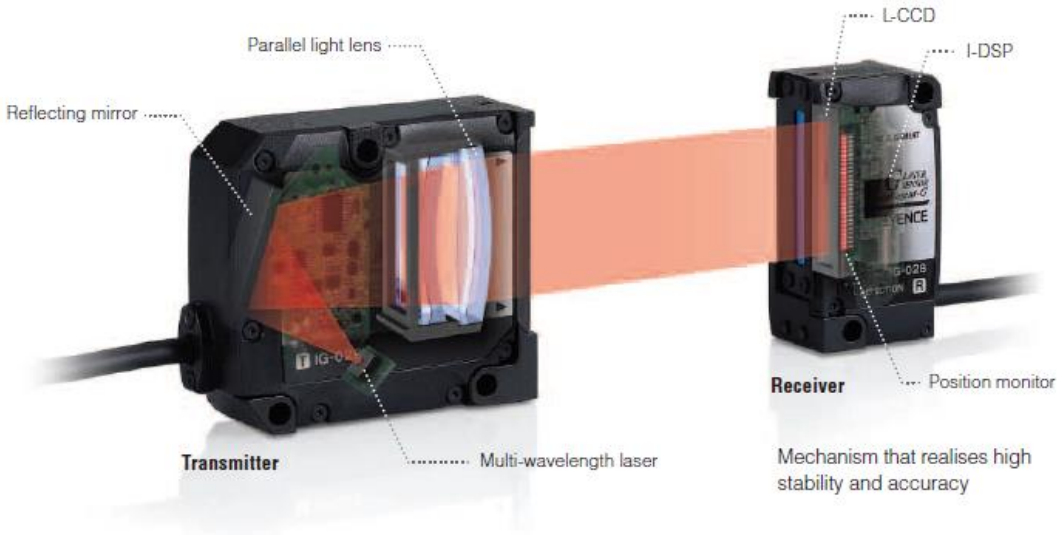


Figura 3.4: Principio di funzionamento del micrometro laser KEYENCE IG-028

Il principio di funzionamento del dispositivo è basato sulla tecnologia L-CCD (Linearised-Charge Coupled Device, cioè dispositivo ad accoppiamento di carica linearizzato). L'immagine digitale acquisita è data dalla matrice di numeri ottenuti tramite la trasduzione in valori numerici della quantità di energia che è stata catturata dal sensore sotto forma di carica elettrica. Il sensore CCD è infatti detto a trasferimento di carica poiché i pixel da cui è composto hanno la capacità di trasferire la carica in modo sequenziale tra loro.

Il sensore lineare CCD (mostrato in Figura 3.5) è costituito da una piastrina in silicio sulla quale vi è disposta un'unica riga di pixel sui quali incide l'energia catturata. La trasformazione dell'energia in valore numerico viene effettuata tramite una fase di elaborazione.

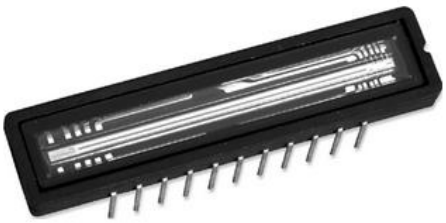


Figura 3.5: Sensore lineare CCD

Lo schema del sensore lineare (Figura 3.6) è costituito da un vettore di pixel e da un convertitore A/D. Il sensore acquisisce la carica e si comporta come un registro a scorrimento nella direzione del convertitore. L'uscita dell'ADC viene codificata ottenendo il risultato digitale.

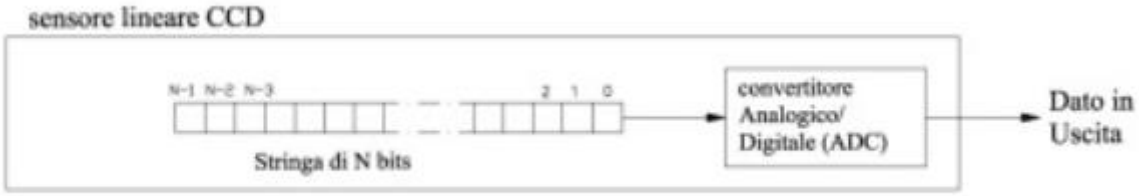


Figura 3.6: Schema del sensore lineare CCD

Per quanto riguarda il processo del trasferimento di carica vengono mostrati in Figura 3.7 i passaggi con cui avviene il procedimento considerando due pixel consecutivi i ed i_{+1}.

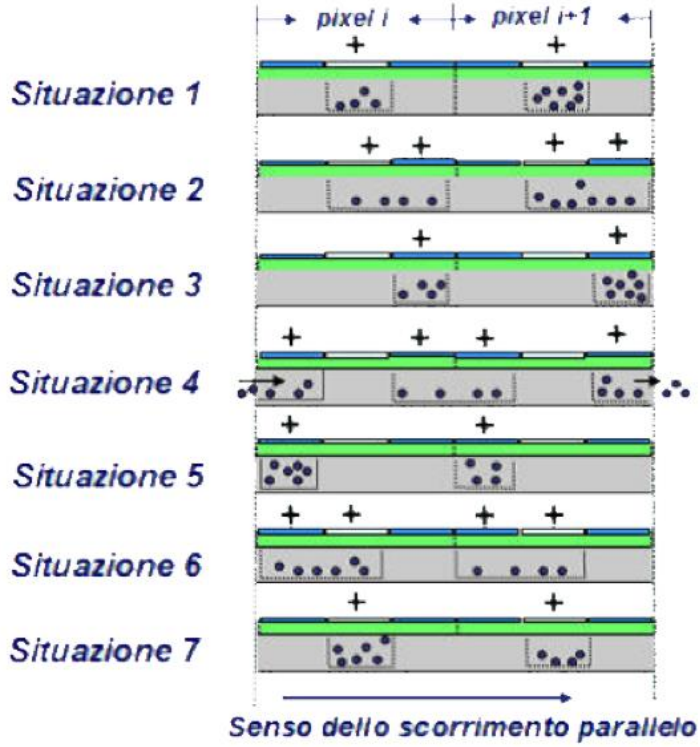


Figura 3.7: Principio del trasferimento di carica del sensore lineare CCD

- Situazione 1: nei due pixel gli elettrodi centrali sono polarizzati positivamente e confinano gli elettroni nelle regioni sottostanti.
- Situazione 2: nei due pixel vengono polarizzati positivamente i due elettrodi di destra e si allargano così le regioni occupate dagli elettroni.
- Situazione 3: viene tolta la tensione agli elettrodi centrali e le regioni si restringono alle sole zone di destra.
- Situazione 4: gli elettrodi di sinistra vengono polarizzati positivamente e nuovamente si allargano le regioni occupate dagli elettroni.
- Situazione 5: viene tolta la tensione agli elettrodi di destra e le regioni si restringono alle sole zone di sinistra.
- Situazione 6: gli elettrodi centrali vengono polarizzati positivamente e si allargano le regioni occupate.

- Situazione 7: viene tolta la tensione agli elettrodi di sinistra, gli elettroni si raccolgono nuovamente nelle regioni centrali e così si conclude il trasferimento della carica tra i due pixel.

In Figura 3.8 viene riportato come appare il vettore lineare CCD sul ricevitore che, una volta interposto l'oggetto, vede un fascio luminoso interrotto.

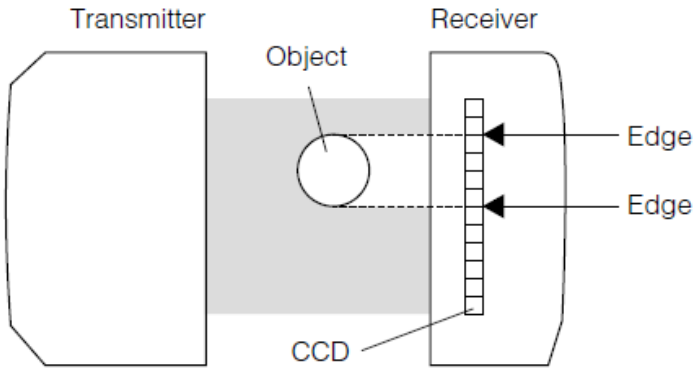


Figura 3.8: Funzionamento sensore lineare CCD quando viene interposto un oggetto

La sensibilità di misurazione può essere regolata in base alla quantità di luce ricevuta e, a tal proposito, si può agire impostando due parametri: il livello di binarizzazione e il valore del filtro. Il valore d'impostazione per giudicare il passaggio o il bloccaggio della luce in base alla quantità di luce ricevuta dal CCD viene detto "livello di binarizzazione" (mostrato in Figura 3.9). Partendo da una condizione in cui il passaggio di luce è pari al 100%, ossia in assenza dell'oggetto, se la quantità di luce ricevuta scende al di sotto del livello di binarizzazione, l'uscita viene giudicata bloccata. Per oggetti opachi come il metallo il valore predefinito consigliato del livello di binarizzazione è del 25% rispetto al valore in assenza di ostacolo. S'intuisce come per rilevare oggetti trasparenti sia necessario aumentare il livello di binarizzazione.

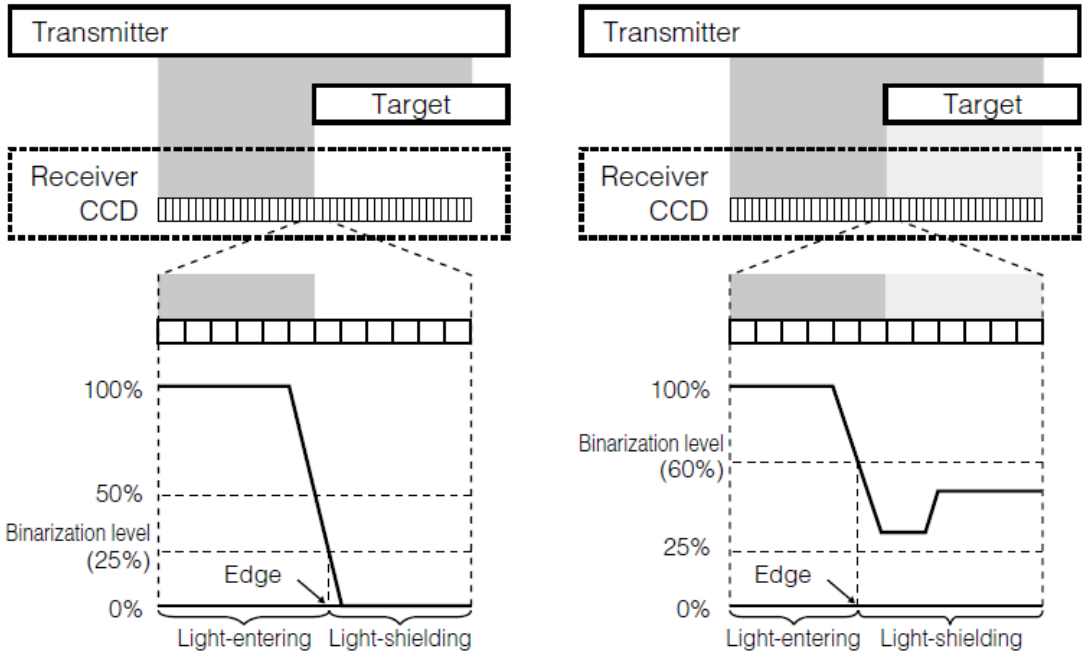


Figura 3.9: Livello di binarizzazione del sensore CCD, a sinistra per target opaco, a destra per target trasparente

Il valore del filtro, invece, agisce sul numero di pixel del CCD: quando il valore viene incrementato la misura è meno influenzata dalle impurezze che si depositano sul vetro del trasmettitore o del ricevitore; tuttavia quando il valore viene decrementato le misure degli oggetti di dimensioni più piccole sono più stabili. In modalità standard il valore del filtro è impostato pari a 9: questo implica che l'oggetto, interposto tra trasmettitore e ricevitore, permette di essere rilevato solamente se blocca il fascio luminoso per un numero di pixel maggiore o uguale a 9. Viceversa la parte di passaggio di luce è giudicata non bloccata solo se l'oggetto permette di avere un fascio luminoso lungo lo stesso numero di pixel.

In Figura 3.10 viene riportato un esempio con filtro impostato pari a 5: nell'immagine a sinistra si nota come l'oggetto, interposto tra trasmettitore e ricevitore e che blocca la luce per un numero di pixel pari a 4, non è rilevato e la luce viene giudicata non bloccata nell'intero range. Nell'immagine a destra, invece, l'oggetto blocca il fascio luminoso per un numero di pixel pari a 5 e viene rilevato.

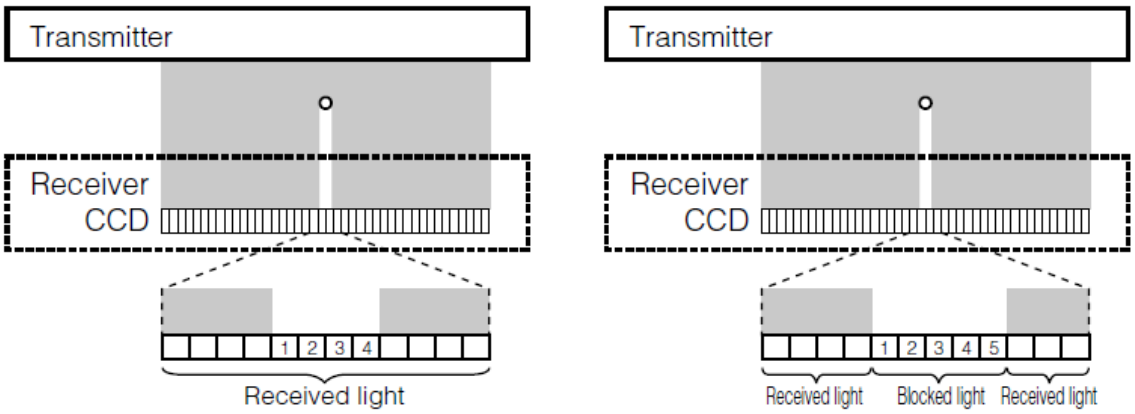


Figura 3.10: Valore del filtro del sensore lineare CCD impostato a 5. A sinistra l'oggetto non è rilevato, a destra l'oggetto è rilevato

Il campo di misurazione del micrometro laser è pari a $28\,mm$ e trasmettitore e ricevitore possono avere una distanza di installazione compresa tra 0 e $1500\,mm$. Tali parametri vanno ben oltre le condizioni di misura del caso applicativo in cui il sistema di misura è pensato per la rilevazione di tubi con diametri inferiori a 28 mm ed una distanza tra i due dispositivi di pochi centimetri. Dalle specifiche tecniche del dispositivo si evince inoltre un errore di linearità pari a $\pm 0{,}1\ \%\ F.S.$ $(\pm 28\ \mu m)$ ed una ripetibilità di $5\ \mu m$ per una distanza di posizionamento di $100\,mm$. In Figura 3.11 vengono mostrate le dimensioni del trasmettitore e del ricevitore necessarie per il dimensionamento della stazione di controllo.

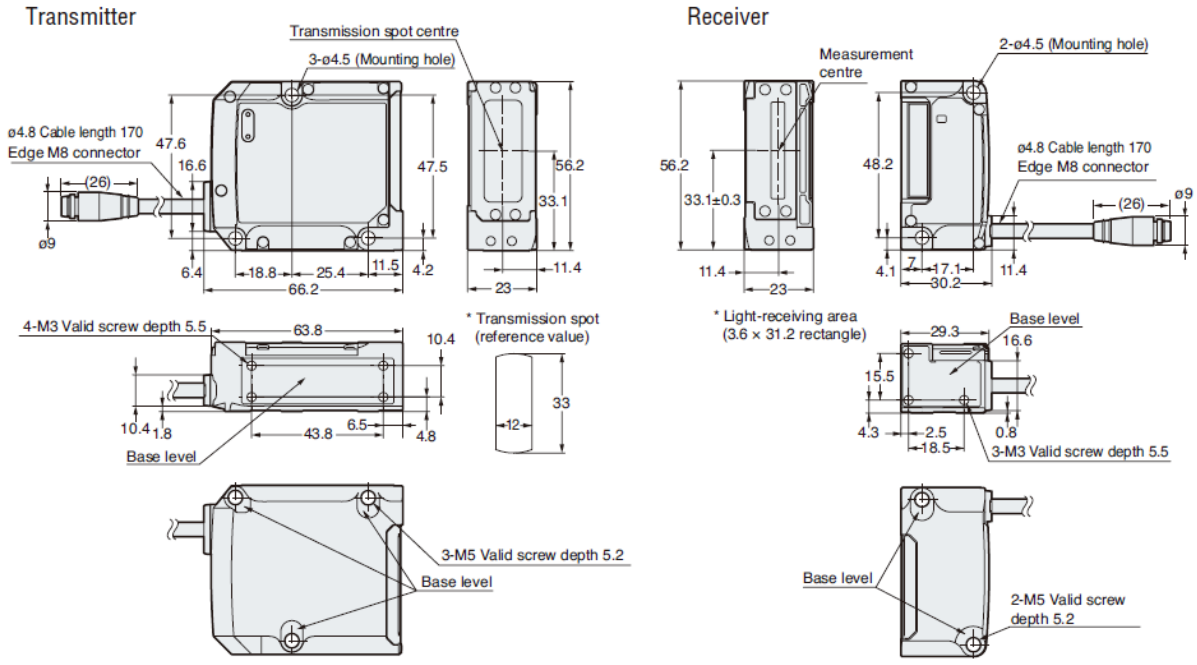


Figura 3.11: Ingombri e quote dei fori del trasmettitore e del ricevitore

3.1.2 Caratteristiche dell'amplificatore IG-1000

L'amplificatore ha a disposizione numerosi cavi elettrici ciascuno dei quali svolge una specifica funzione ed è distinguibile da un colore differente. In Figura 3.12 vengono mostrati i cavi presenti con le relative funzioni.

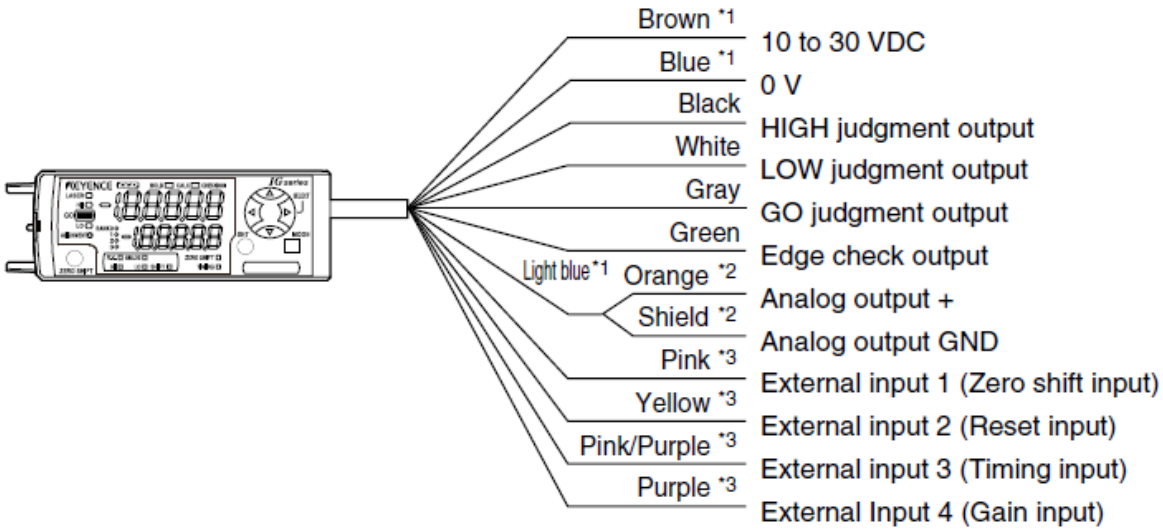


Figura 3.12: Cavi dell'amplificatore KEYENCE IG-1000

Una delle funzioni principali del condizionatore è sostanzialmente quella di alimentare trasmettitore e ricevitore ai quali è collegato tramite opportuni connettori. A tal proposito l'amplificatore dispone di:

- un cavo marrone che viene collegato all'alimentazione di 12 V in corrente continua;
- un cavo blu che è connesso a 0 V.

Per quanto riguarda le uscite del condizionatore si individuano tre uscite digitali di tipo PNP o NPN, che, associate ai valori di tolleranza impostati, consentono di generare segnali ON/OFF a seconda che l'oggetto sia all'interno del campo di tolleranza o meno. Nello specifico il condizionatore ha a disposizione:

- un cavo nero che rappresenta l'uscita HIGH. Questa uscita digitale è a livello alto quando la dimensione dell'oggetto sottoposto a misura supera il limite superiore impostato;
- un cavo bianco che identifica l'uscita LOW. Al contrario di quella precedente, tale uscita è a livello alto se la dimensione dell'oggetto è al di sotto del limite inferiore impostato;
- un cavo grigio che rappresenta l'uscita GO. L'uscita è a livello alto quando l'oggetto è in tolleranza ed è quindi compreso tra i valori limiti.

Queste tipologie di uscita non vengono utilizzate nell'applicazione in esame in quanto si dovrebbe ricorrere a impostare manualmente i valori di tolleranza per ogni codice in produzione rendendo il processo poco automatizzato.

Per il caso di studio si utilizza l'uscita di tipo analogico del condizionatore che viene individuata da:

- un cavo arancione che rappresenta l'uscita analogica impostata nel range $0 \div 5\ V$.
- un cavo schermato che rappresenta l'uscita analogica da collegare a GND.

In Figura 3.13 sono mostrati gli ingombri dell'amplificatore necessari per il dimensionamento della stazione di controllo.

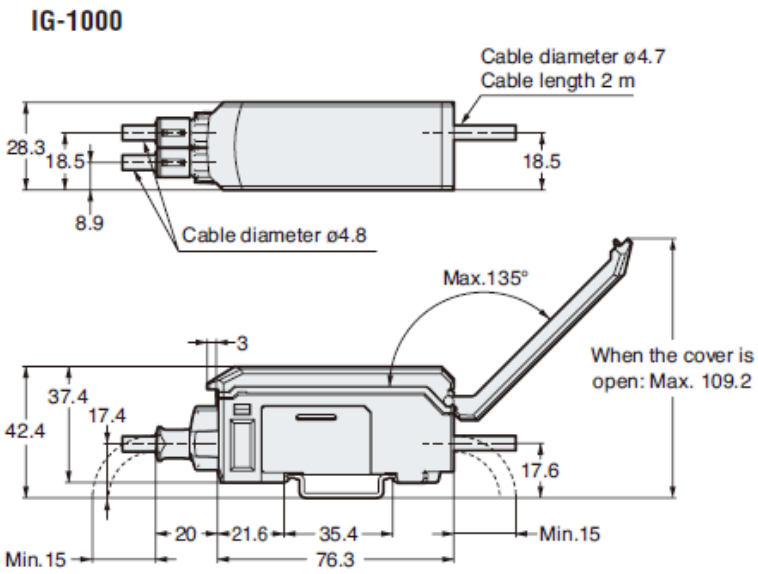


Figura 3.13: Ingombri dell'amplificatore

3.2 Acquisizione del segnale analogico

Il dispositivo che riceve il segnale normalizzato $0 \div 5\,V$ del sensore è un Fishino UNO Rev.2 (Figura 3.14). La scheda di acquisizione è rappresentata dall'interfaccia tra il segnale inviato in input al Fishino ed il microprocessore.

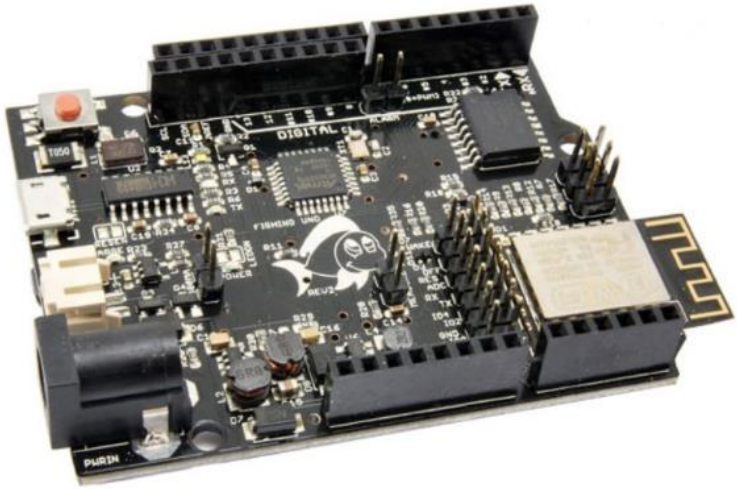


Figura 3.14: Fishino UNO Rev.2

Fishino UNO Rev.2 è una scheda basata sull'ATmega328, che è lo stesso microcontrollore di Arduino UNO, ed è dotato di connettività WiFi, real time clock e di lettore microSD.

Per il seguente caso di studio il Fishino viene utilizzato per le stesse funzionalità del Raspberry precedentemente descritto, ma grazie alla presenza del convertitore analogico-digitale viene scelto per ricevere il segnale dal micrometro laser.

Il *pinout* viene rappresentato in Figura 3.15.

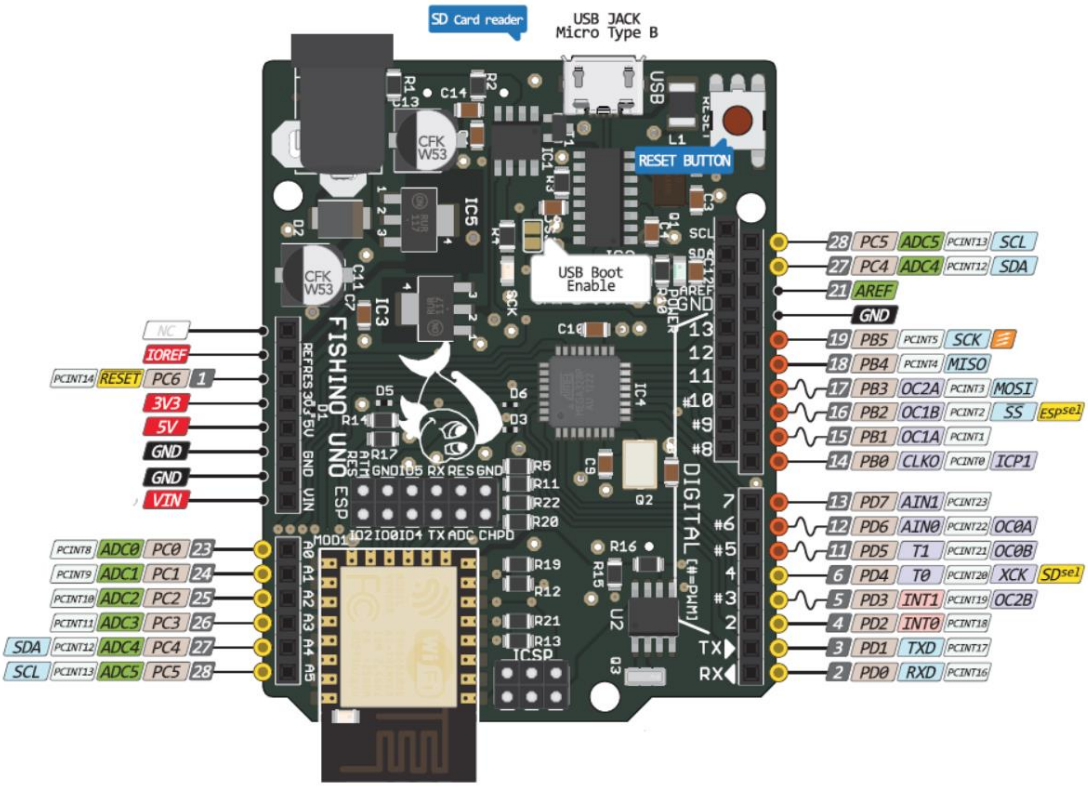


Figura 3.15: *Pinout* Fishino UNO Rev.2

Specifiche tecniche del Fishino UNO Rev. 2

Alimentazione:

- 5 V tramite connettore USB;
- 5 V sull'ingresso +5V;
- Da 7V a 15V sull'ingresso Vin;
- Da 7 V a 15 V sul plug di alimentazione;
- Batteria LiPo a singola cella

Livelli digitali:

- 5 V

Controller:

- ATmega 328p-au SMD a 8 bit

Clock:

- 16 MHz

Memoria:

- 32 KBytes Flash;

- 2 KBytes RAM;
- 1 KBytes EEPROM

Porte disponibili:

- 14 digitali di cui 6 in PWM;
- 6 analogiche

Interfacce disponibili:

- 1 x SPI;
- 1 x i2C

Moduli aggiuntivi sulla scheda:

- Wi-Fi;
- Connettore per schede microSD;
- Modulo RTC

Peso: 25 g

Dimensioni 76,5 x 53,5 x 14 mm

Per quanto riguarda l'alimentazione del Fishino occorre specificare meglio le varie modalità disponibili per comprendere le scelte effettuate nel caso applicativo:

• Porta USB: a questa presa arrivano i 5 V provenienti da una porta USB del computer. Tale soluzione si rivela non soddisfacente per diverse ragioni. Innanzitutto la tensione operativa, ovvero quella di funzionamento, non corrisponde ai 5 V teorici bensì ad un valore più basso (ciò può essere verificato andando a misurare la tensione tra i pin del 5 V e GND tramite un multimetro). Questo limite comporta che la funzione *analogread()* (definita nel paragrafo in cui è descritto nel dettaglio il programma) sia influenzata da una tensione diversa da 5 V e quindi si debba introdurre una compensazione con un opportuno coefficiente moltiplicativo all'interno dello sketch. Un altro svantaggio, sicuramente più incisivo, dell'alimentazione tramite USB deriva dalla presenza di fluttuazioni nella tensione di alimentazione che si ripercuotono sui valori di tensione letti tramite l'ADC.
• Presa Jack Japan: a questa presa si può collegare un alimentatore esterno con una tensione che può variare tra 6 V e 20 V. Il range di tensione consigliato dalla casa produttrice è tra 7 V e 12 V. Una tensione pari a 6 V potrebbe non garantire una corretta stabilizzazione del segnale in quanto bisogna tenere conto della caduta di tensione del diodo di protezione posto in serie all'ingresso del regolatore. Si evidenzia che la presenza di tale diodo serve a preservare la scheda in caso di inversione della polarità sullo spinotto. Una tensione al di sopra dei 12 V provocherebbe invece un surriscaldamento eccessivo del regolatore. Anche in questo caso si rileva una tensione operativa diversa da quella teorica (a differenza dell'alimentazione USB si registra una tensione tra il pin 5 V e GND che è poco al di sopra del valore teorico pari a 5 V). Un vantaggio sicuramente importante che consente di avere questa tipologia di alimentazione è l'assenza di fluttuazioni della tensione di alimentazione.
• Presa Vin: questo pin ha una duplice funzione. Può essere utilizzato come ingresso per un'alimentazione esterna, non protetta da inversioni di polarità ed in cui il range di tensione consigliata è tra i 7 V ed i 12 V; può essere utilizzato come uscita per alimentare carichi che richiedono una tensione maggiore di 5 V. Come nel caso precedente si ha l'eliminazione delle fluttuazioni della tensione di alimentazione.
• Presa 5 V: tale pin può essere utilizzato per alimentare direttamente il Fishino tramite una fonte esterna di 5 V stabilizzati. Anche in questo caso non c'è alcuna forma di protezione per l'inversione della polarità e si ha l'eliminazione delle fluttuazioni.

A seguito delle considerazioni fatte si decide di alimentare il Fishino tramite la presa Jack, fornendo una tensione pari a 12 V, principalmente per due motivi:

- una maggiore sicurezza grazie alla presenza del diodo di protezione che permette di non danneggiare irreparabilmente la scheda in caso di inversione della polarità;
- eliminare il problema delle fluttuazioni nella tensione di alimentazione che si ripercuote sui valori di tensione letti dal Fishino tramite l'ADC. In realtà tale vantaggio viene meno nell'applicazione sperimentale in quanto, per questioni ragionevoli, si ritiene opportuno adottare un'unica alimentazione sia per il controllore che per il micrometro laser. Questo comporta che il Fishino veda delle fluttuazioni di tensione derivanti dalla corrente assorbita da parte del sensore. Come si vede nel paragrafo successivo in cui è descritto il programma del Fishino vengono implementate delle righe di codice al fine di eliminare le fluttuazioni tramite una media mobile. In fase sperimentale si stabilirà se tali accortezze siano sufficienti ad avere dei valori di misurazione comunque accettabili o si debba intervenire con delle differenti alimentazioni per il Fishino e per il micrometro.

La calibrazione della tensione letta dal pin analogico viene fatta via software e verrà descritta nel paragrafo 3.4.

3.2.1 Convertitore analogico-digitale ad approssimazioni successive

Come anticipato nei paragrafi precedenti il segnale in uscita dal sensore di misura è un segnale di tipo analogico. Tali segnali, per poter essere trattati dai sistemi di acquisizione, devono essere convertiti in segnali digitali. In Figura 3.16 vengono rappresentati i principali componenti che costituiscono la catena di trasduzione e quella di acquisizione dei dati in un caso generico.

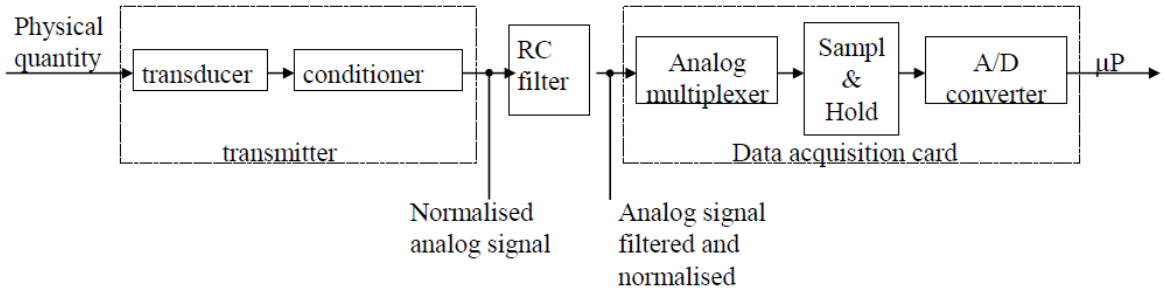


Figura 3.16: Struttura della catena di acquisizione dati (Figura tratta da slide "Servosystems structure" – Massimo Sorli)

La conversione del segnale avviene tramite quattro operazioni distinte: il campionamento, il mantenimento, la quantizzazione e la codifica.

Il campionamento è l'operazione che trasforma il segnale continuo in un segnale discreto e viene effettuata dal *sampler*. L'operazione di mantenimento del segnale campionato è effettuata invece dall'*holder*. Queste due operazioni sono raggruppate all'interno del dispositivo denominato "Sample & Hold" (mostrato in Figura 3.17)

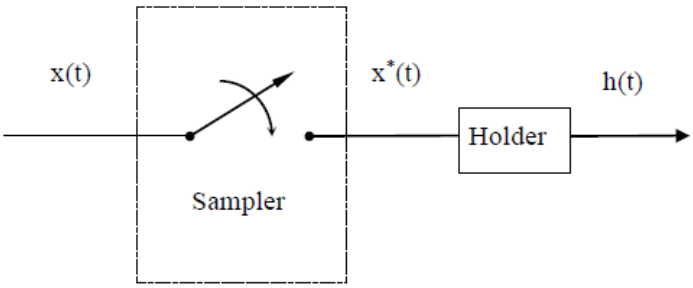


Figura 3.17: Sample & Hold (Figura tratta da slide "Servosystems structure" – Massimo Sorli)

Il segnale analogico in ingresso è quindi campionato in un treno di impulsi modulati in ampiezza e mantenuti costanti nel periodo di tempo T. In Figura 3.18 è rappresentato il segnale analogico di partenza; in Figura 3.19 viene rappresentata la discretizzazione nel tempo ed in Figura 3.20 il mantenimento del segnale negli intervalli di tempo.

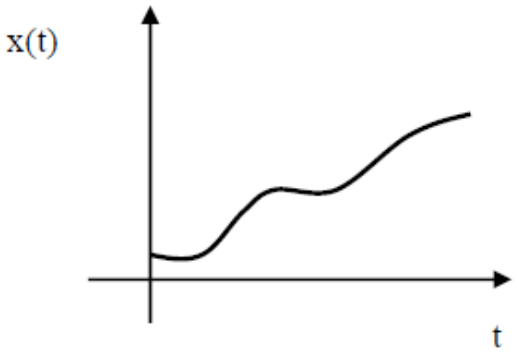


Figura 3.18: Segnale analogico (Figura tratta da slide "Servosystems structure" – Massimo Sorli)

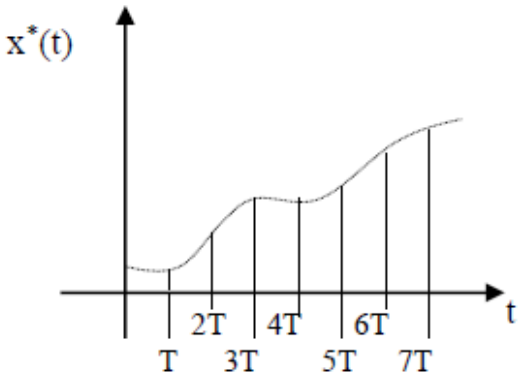


Figura 3.19: Segnale discretizzato nel tempo dal Sampler (Figura tratta da slide "Servosystems structure" – Massimo Sorli)

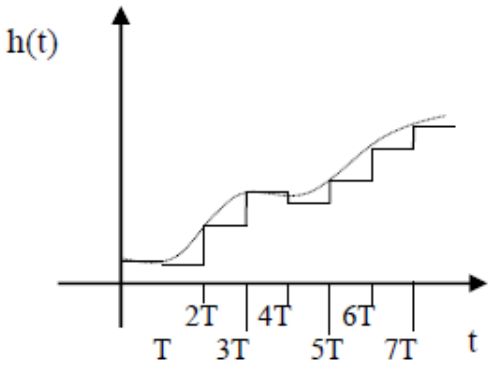


Figura 3.20: Segnale mantenuto nell'intervallo di tempo dall'holder (Figura tratta da slide "Servosystems structure" – Massimo Sorli)

Il processo di quantizzazione e codifica avviene all'interno del convertitore analogico-digitale. In Figura 3.21 viene mostrato un esempio di segnale variabile in un range di $0 - 10\ V$ campionato da un sistema di acquisizione dati a 3 bit.

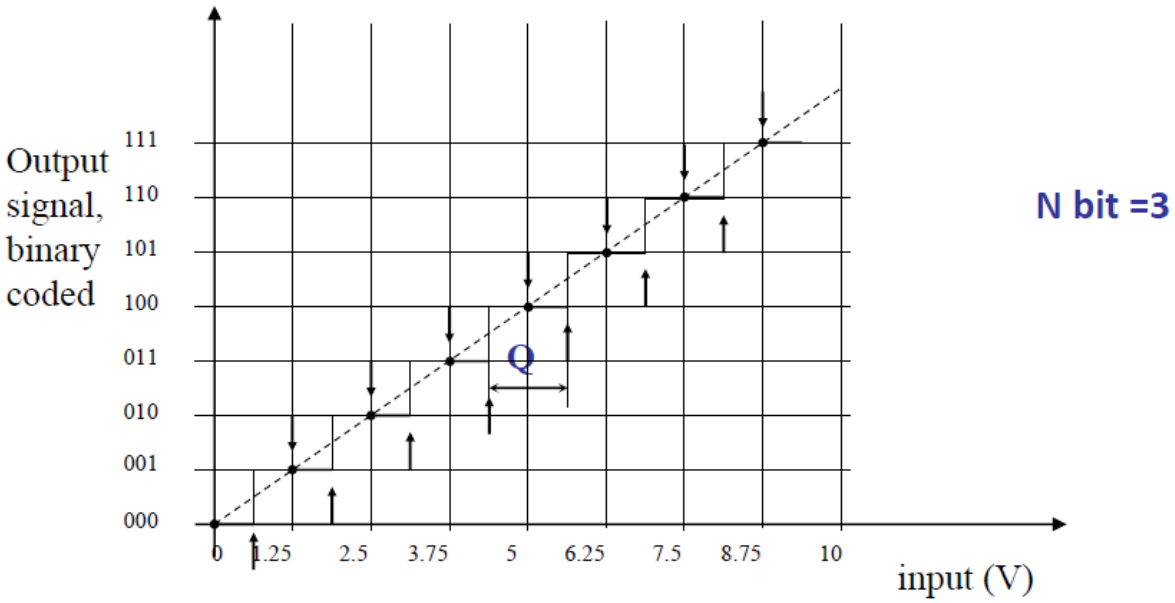


Figura 3.21: Quantizzazione e codifica per un sistema di acquisizione a 3 bit (Figura tratta da slide "Servosystems structure" – Massimo Sorli)

L'intervallo $0 - 10\ V$ riportato sull'asse delle ascisse è quindi suddiviso in 2^3 intervalli che stabiliscono i valori in ampiezza espressi in Volt determinati dal sistema di acquisizione dati e vengono convertiti in numeri binari riportati sull'asse delle ordinate.

Il microcontrollore ATmega328 del Fishino è a 10 bit ed è costituito da un ADC ad approssimazioni successive (Figura 3.22) composto da:

- un Sample & Hold che acquisisce la tensione V_{IN} del segnale in ingresso;

- un comparatore di tensione che confronta la tensione di ingresso con il segnale in uscita da un convertitore digitale-analogico e trasmette il risultato ad un registro ad approssimazioni successive;
- un SAR cioè un registro ad approssimazioni successive;
- un DAC cioè un convertitore digitale-analogico.

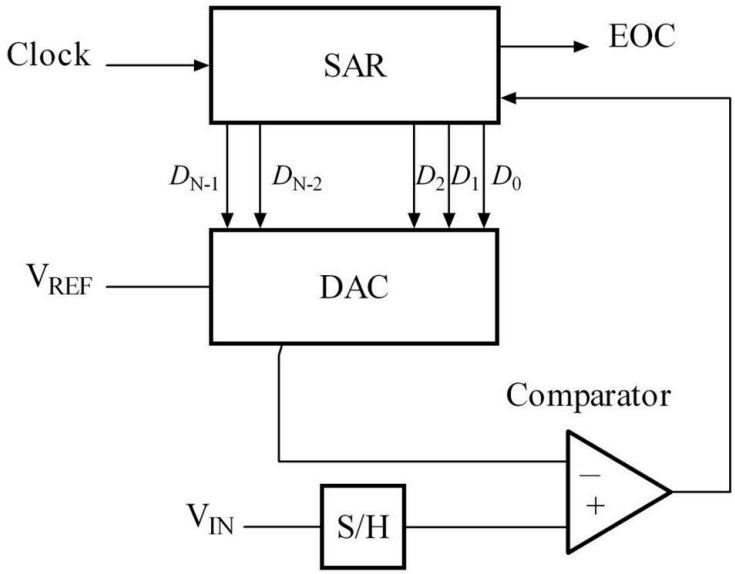


Figura 3.22: ADC ad approssimazioni successive

La tensione in ingresso viene acquisita da un S&H e l'uscita è inviata ad un comparatore che riceve come altro ingresso la tensione fornita dal convertitore digitale-analogico. Il DAC riceve in ingresso un certo numero di bit restituendo in uscita una serie di gradini in tensione proporzionali al dato binario in ingresso. L'avviamento della conversione del segnale è dato da un segnale generato attivando un bit nel registro di controllo dopo il tempo di acquisizione del S&H. Il registro SAR viene inizializzato al valore in codice binario pari a 1000000000 (considerando il convertitore a 10 bit). Si nota quindi che il valore di inizializzazione è ottenuto ponendo il bit più significativo (MSB cioè il bit $N-1$) ad 1 ed i restanti bit a 0. Questo numero equivale a metà del range del convertitore ossia è pari a

$$1024/2 = 1 \times 2^{N-1} = 1 \times 2^9 = 512 \tag{3.1}$$

e, al primo colpo di clock, comanda il DAC che fornisce un primo livello analogico V_{DAC} pari a metà V_{REF}. Questo valore analogico viene poi confrontato con il valore di ingresso V_{IN} dal comparatore e si hanno due casi:

- $V_{IN} > V_{DAC}$: l'uscita del comparatore va al valore alto, il MSB del registro resta pari a 1 e il bit $N-2$ viene posto a 1;
- $V_{IN} < V_{DAC}$: l'uscita del comparatore va al valore basso, il MSB del registro passa al valore 0 e il bit $N-2$ viene posto a 1.

L'uscita del comparatore viene inviata al SAR e si ripete la procedura: al successivo colpo di clock si confronta la nuova tensione in uscita dal DAC con la V_{IN} che deve rimanere per tutto il tempo di conversione. Quindi se:

- $V_{IN} > V_{DAC}$: l'uscita del comparatore va al valore alto, il bit $N-2$ resta pari a 1 e il bit $N-3$ viene posto a 1;

- $V_{IN} < V_{DAC}$: l'uscita del comparatore va al valore basso, il bit $N-2$ passa al valore 0 e il bit $N-3$ viene posto a 1.

E così via per tutti i bit. In altre parole la tensione V_{IN} è comparata con una frazione sempre più piccola della tensione V_{REF} e ad ogni step il corrispondente bit del registro SAR viene adeguato al risultato della comparazione.

Il flow chart in Figura 3.23 riassume il comportamento del convertitore ad approssimazioni successive.

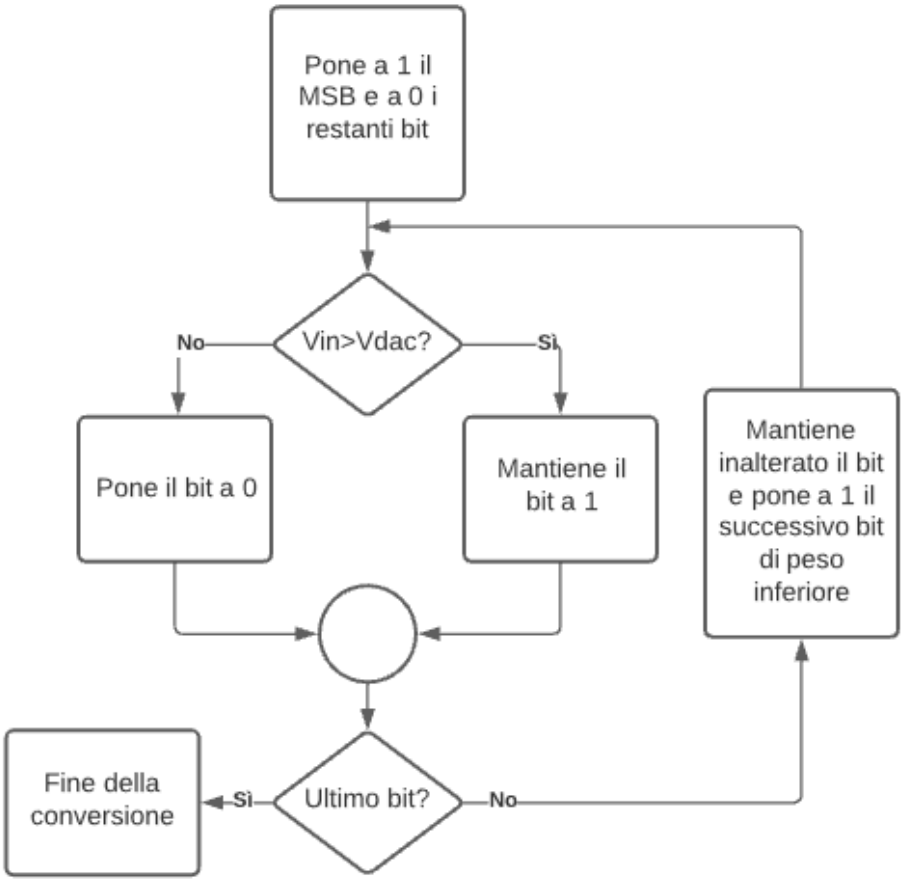


Figura 3.23: Flow chart ADC ad approssimazioni successive

Si ha così la serie di "approssimazioni successive" tra l'uscita del DAC e la tensione da misurare: quanto maggiore è il numero di bit N, tanto migliore sarà l'approssimazione tra il valore analogico e quello digitalizzato. La sequenza di operazioni termina quando il numero di colpi di clock ha raggiunto la comparazione relativa al bit meno significativo (LSB). Ne deriva che la conversione richiede al minimo un numero di impulsi di clock pari al numero dei bit da convertire. L' evento viene indicato da un flag EOC (*End Of Conversion*) e il gruppo di N bit collezionati nel registro costituisce il risultato binario della conversione.

Il convertitore analogico-digitale dell'Atmega328 accetta tensioni in ingresso comprese tra $0\,V$ e V_{REF}. Nel Fishino UNO, in condizioni di default, $V_{REF} = 5V$. Viene quindi definito intervallo di quantizzazione il valore:

$$Q = \frac{FSR}{2^N} = \frac{5}{2^{10}} = 4{,}883\ mV \tag{3.1}$$

che corrisponde all'errore massimo di risoluzione (a fronte di una variazione pari a Q della tensione in ingresso, l'uscita dal convertitore cambia di 1 bit).

Come si è descritto in precedenza l'acquisizione del segnale non avviene in modo continuo, bensì in modo periodico ad istanti di campionamento distanziati di un periodo T. La frequenza di campionamento viene quindi definita come:

$$f_s = \frac{1}{T} \tag{3.2}$$

e la pulsazione di campionamento come:

$$\omega_s = 2\pi f_s \tag{3.3}$$

Per il teorema del campionamento di Nyquist la frequenza di campionamento dev'essere superiore ad un valore minimo per essere in grado di ricostruire il segnale originario a partire da quello campionato. In particolare:

$$\omega_S > 2\omega_1 \tag{3.4}$$

dove ω_1 è la massima pulsazione del segnale continuo di ingresso. Per quanto riguarda il microcontrollore Atmega328 del Fishino la frequenza di default del clock del convertitore analogico-digitale è pari a $125\ kHz$ e, sapendo che ciascuna conversione richiede 13 cicli di clock, la frequenza di campionamento è pari a:

$$f_s = \frac{125}{13} = 9{,}615\ kHz \tag{3.5}$$

cioè un periodo T di campionamento di:

$$T = \frac{1}{9615} = 104\ \mu s \tag{3.6}$$

Per l'acquisizione del segnale proveniente dal micrometro laser si ritiene che la frequenza di campionamento rispetta ampiamente il teorema di Nyquist in quanto la massima frequenza del segnale è poco elevata e il sistema di acquisizione riesce a riprodurre il segnale sia nel contenuto in frequenza che nella forma.

3.3 Realizzazione del sistema per misurazioni

Per la realizzazione dei cablaggi viene in questa prima fase realizzato il circuito elettrico su breadboard. In Figura 3.24 viene mostrato il circuito tramite il software "Fritzing" (si specifica che si è utilizzato un FISHINO UNO Rev.2 differentemente dall'Arduino Uno mostrato in Figura 3.24).

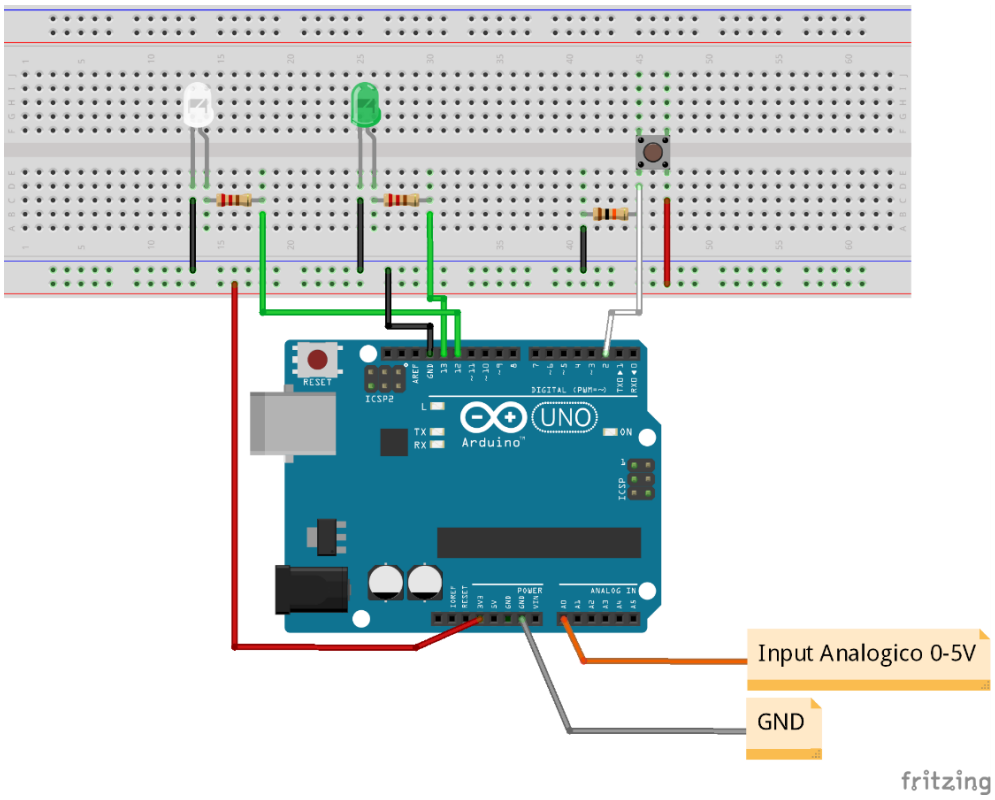


Figura 3.24: Circuito realizzato su breadboard con connessioni al Fishino e segnale analogico in uscita dal micrometro

Il circuito prevede che ci siano due fili elettrici che mettono in comunicazione il micrometro con il Fishino: il primo è il segnale analogico da 0 a 5 Volt uscente dal condizionatore e collegato al pin analogico A0; il secondo è il segnale analogico di GND connesso al corrispettivo pin del controllore. Si passa quindi al cablaggio del pulsante per l'invio della misurazione che è connesso da un lato ai 3,3V del Fishino e dall'altro a GND tramite la resistenza di pull-down da 10kΩ. Sono presenti inoltre due LED: quello verde collegato al pin 7 utilizzato per segnalare l'invio del dato; quello bianco connesso al pin 8 utilizzato per segnalare la connessione del Fishino alla rete. Entrambi i LED sono collegati alle uscite digitali tramite resistenze da 220Ω e hanno il polo negativo connesso a GND.

Una volta verificato il funzionamento del circuito tramite il programma caricato sul Fishino (che verrà riportato e spiegato nel paragrafo successivo) si passa alla realizzazione del "case" necessario al contenimento del Fishino con i relativi cablaggi e del micrometro laser. Il contenitore è prima realizzato mediante il software Solidworks (in Figura 3.25 e 3.26 è mostrato insieme all'assemblaggio di tutti i componenti) e poi convertito nell'apposito CAM per poter essere realizzato tramite stampante 3D GEEETECH.

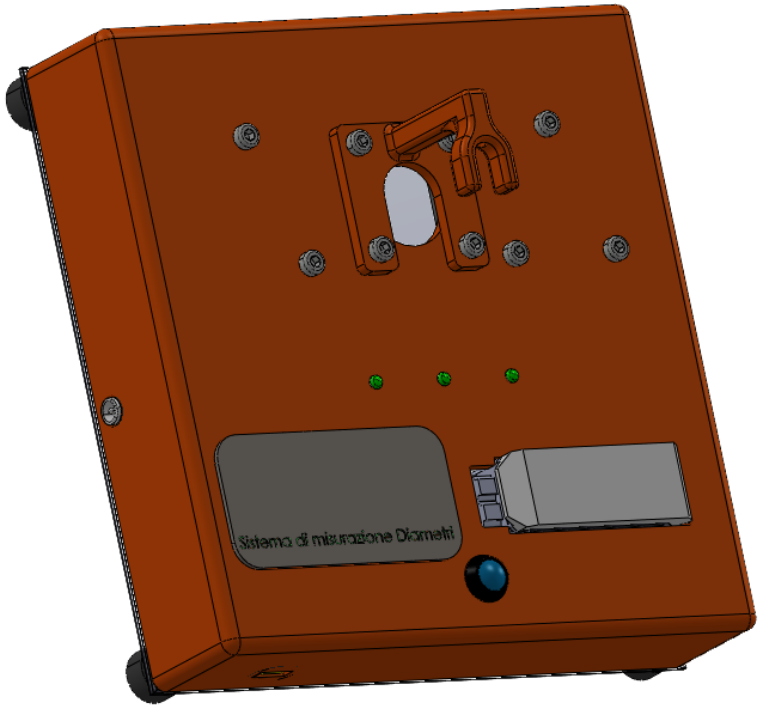


Figura 3.25: Stazione di controllo diametri realizzata su Solidworks con vista anteriore

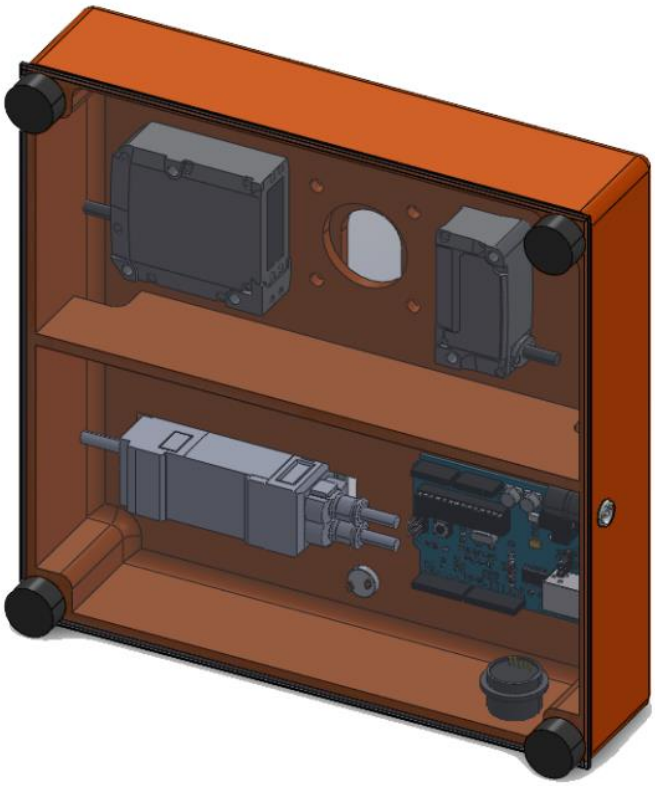


Figura 3.26: Stazione di controllo diametri realizzata su Solidworks con vista posteriore

Come si può notare vi sono due parti principali entrambe realizzate in PLA colore Silk Copper: la prima è il vero e proprio "case" mentre la seconda è il supporto necessario al sostenimento del pezzo da misurare ed è fissato al contenitore tramite collegamenti filettati.

Si utilizzano viti M5 a testa cilindrica con cava esagonale sia per i collegamenti del supporto sia per quelli del sensore al contenitore. Sulla parte superiore del contenitore sono realizzati, oltre ai fori per le viti, i fori per i led, il foro per il pulsante, la cava per il condizionatore che è fissato ad incastro. Nelle parti laterali è presente il foro per connettere la presa Jack di alimentazione del Fishino e del sensore e la cava per la presa USB per connettere il PC in caso di una modifica dello sketch caricato sul controllore. La parte inferiore prevede 4 fori filettati per i piedini di sostegno rivestiti in gomma.

Il pulsante utilizzato (mostrato in Figura 3.27) per l'invio al database della misura effettuata è un pulsante in metallo con contatti NC/NO/C ed è dotato di un polo positivo ed uno negativo per un'eventuale alimentazione a 12 V del LED integrato.

Figura 3.27: Pulsante in metallo con LED integrato utilizzato in configurazione NO

I contatti utilizzati per l'applicazione in questione sono il NO, collegato all'alimentazione di 3,3 V del Fishino, e il C da cui partono il cavo connesso al pin digitale e quello connesso a GND tramite la resistenza di pull-down. Tutte le giunzioni necessarie per la realizzazione del circuito vengono effettuate tramite saldatura a stagno ed isolate mediante l'applicazione di guaine termo-restringenti.

In Figura 3.28 sono riportate le foto dopo aver terminato i cablaggi e prima di chiudere il *case*.

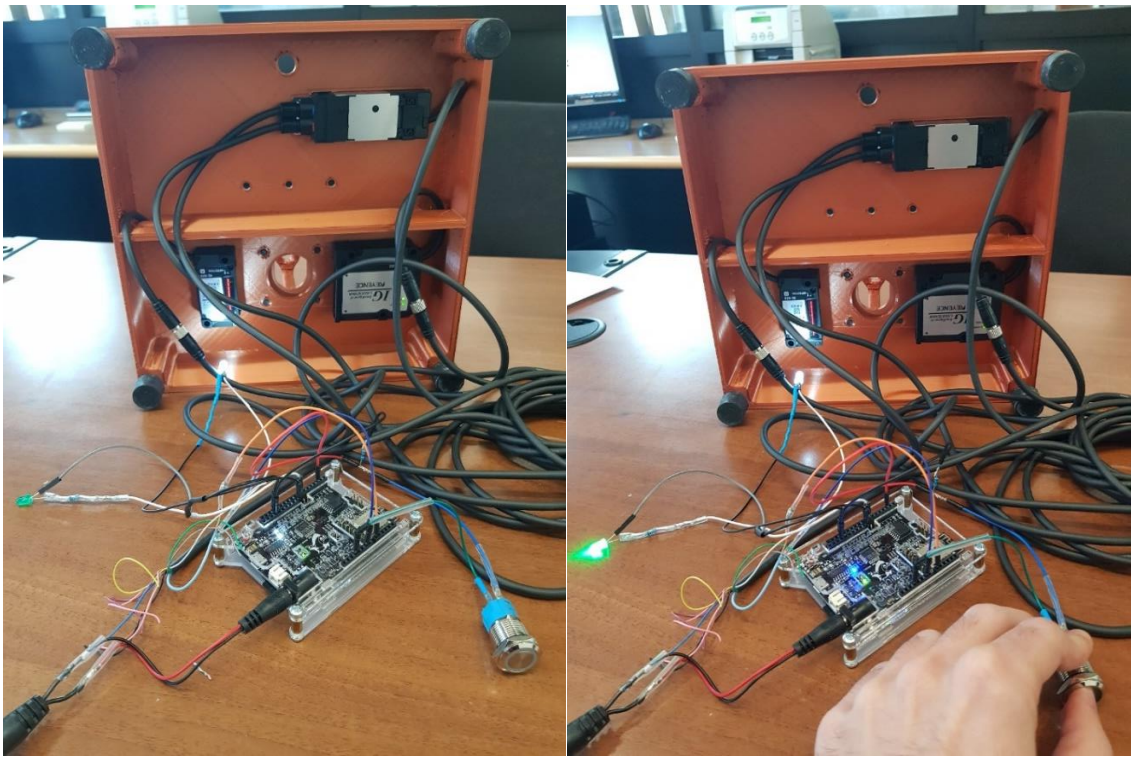


Figura 3.28: Stazione di controllo diametri. A sinistra con il solo LED bianco di connessione acceso, a destra con anche il LED verde acceso a seguito dell'invio del dato

Per quanto riguarda l'alimentazione del micrometro laser e del Fishino si utilizza un unico alimentatore da 12 V e 1000 mA.

3.4 Programma Fishino

Per quanto riguarda la parte di programmazione si utilizza l'ambiente di sviluppo Arduino IDE che utilizza un linguaggio di programmazione derivato dal C++. La schermata dell'ambiente di sviluppo si presenta come in Figura 3.29.

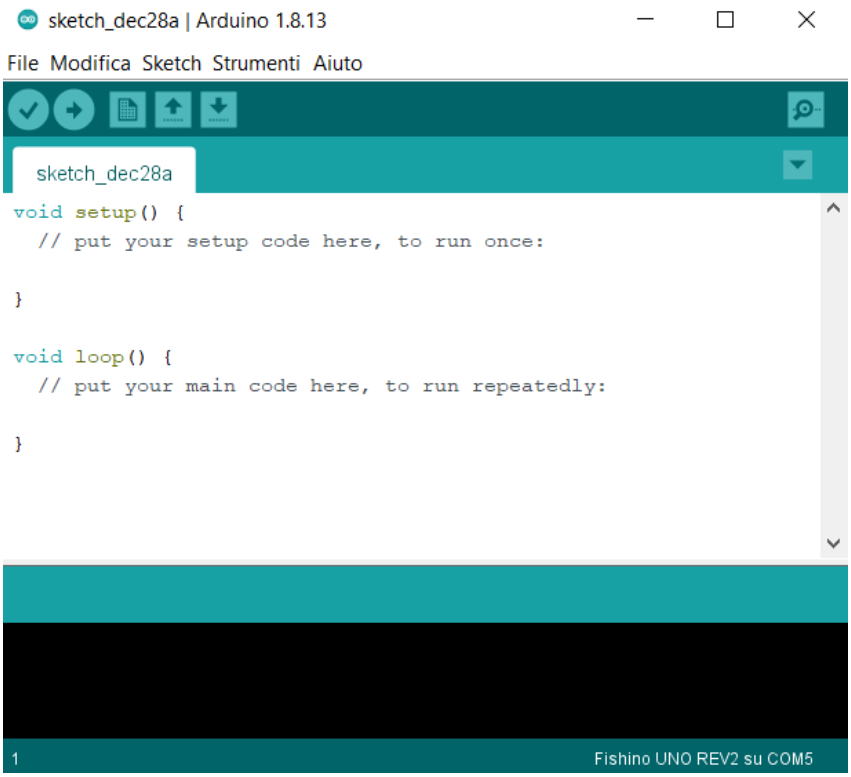


Figura 3.29: Ambiente di sviluppo Arduino IDE

La struttura base del linguaggio è costituita principalmente da due funzioni:

- `void setup()`: è la prima ad essere chiamata nello sketch ed è utilizzata ad esempio per inizializzare variabili o impostare lo stato dei pin;
- `void loop()`: permette di eseguire ciclicamente il programma definito al suo interno.

Al di fuori delle precedenti funzioni vengono definite le librerie e le variabili di tipo globale le quali possono essere utilizzate da ogni istruzione e funzione del programma.

Per quanto riguarda gli elementi principali di sintassi presenti nello sketch si evidenzia: la presenza del punto e virgola (;) inserito alla fine di un'istruzione per separare gli elementi del programma; la presenza delle parentesi graffe ({}) che definiscono l'inizio e la fine di un blocco di istruzioni; l'utilizzo delle linee di commento (//) per fornire una maggiore chiarezza e comprensione del programma.

Di seguito viene riportato e commentato lo sketch utilizzato.

```
//definizione librerie
#include <FishinoWebServer.h>
#include <Flash.h>
#include <FishinoUdp.h>
#include <Fishino.h>
#include <SPI.h>
#include <SD.h>
```

Tramite il comando `#include` vengono incluse le librerie esterne necessarie per utilizzare le funzioni volute all'interno dello sketch.

Con `#include <Fishino.h>` si include la libreria “Fishino” che contiene tutta la gestione del modulo WiFi.

```
//inizializzazione variabili
byte analogPin = A0;
const int buttonPin = 2;
const int led_con = 8;
const int led_press = 7;
```

Nelle precedenti righe di codice vengono definiti i pin di input e output utilizzati: il pin A0 viene utilizzato come pin di ingresso analogico per ricevere il segnale in uscita dal sensore; il pin 2 è adibito all’ingresso digitale corrispondente al segnale associato alla pressione del pulsante; i pin 7 ed 8 sono utilizzati per i segnali digitali di uscita rispettivamente per accendere il LED associato all’invio del dato e il LED che garantisce la presenza di connessione alla rete da parte del Fishino.

Per la definizione del pin di ingresso analogico viene utilizzato il tipo di dato “byte” che memorizza un numero senza segno a 8 bit.

Per la definizione dei pin di ingresso ed uscita digitali si utilizza `const int`: “const” è un qualificatore di variabile che sta per “costante” ed indica che il valore della variabile non può essere cambiato nel corso dello sketch; “int” è un tipo di dato che sta per “integer” e viene usato per memorizzare numeri interi utilizzando 2 byte.

```
int buttonState = 0;
```

Inizializzazione della variabile che assume livello alto o basso a seconda che il pulsante venga premuto o meno.

```
char server[] = "192.168.1.53";
```

Definisce la stringa di caratteri che identifica l’indirizzo IP del server.

```
unsigned int val = 0;
```

Variabile utilizzata per memorizzare il valore della funzione “analogread()” definita in seguito.

Il tipo di dato indica un intero senza segno memorizzato in due byte.

```
float V_misura = 0;
float mm_misura = 0;
float misura = 0;
```

Il tipo di dato “float” memorizza numeri decimali tramite la modalità a virgola mobile. Vengono inizializzate le variabili rispettivamente associate al valore espresso in Volt del segnale analogico in ingresso; al valore del segnale convertito in millimetri ed al valore che viene passato nella “query string” per l’invio del dato.

```
int const dim = 100;
```

Definizione della variabile associata alla dimensione del buffer utilizzato per effettuare la media mobile dei valori ricevuti in ingresso dal sensore di misura.

```
float fil[dim];
```

Definizione del buffer.

```
int i = 0;
```

Inizializzazione dell'indice utilizzato nel ciclo *for* per il calcolo della media.

```
FishinoClient client;
#define MY_SSID  "PsaPipes"
#define MY_PASS "pipes197052"
```

Tramite `#define` si definiscono le due costanti di tipo *string* che identificano rispettivamente ID e password della rete aziendale necessari per la connessione in rete del Fishino.

```
void setup(){
   pinMode(buttonPin, INPUT);
   pinMode(led_con, OUTPUT);
   pinMode(led_press, OUTPUT);
```

La funzione pinMode permette di definire il comportamento che devono assumere i pin utilizzati nello sketch: in particolare come primo parametro viene passata la variabile associata al pin e come secondo parametro il comportamento che deve assumere il pin (ingresso o uscita).

```
    Serial.begin(9600);
```

Tramite questa istruzione è possibile far trasmettere informazioni attraverso la porta seriale e leggerle sul PC (ovviamente quando il controllore ed il PC sono connessi via USB). La configurazione della porta seriale richiede che venga passato il parametro corrispondente alla velocità di comunicazione solitamente impostata a 9600 baud (1 baud indica una certa quantità di bit trasmessi in un secondo attraverso una linea digitale). I messaggi inviati verso il computer possono essere letti aprendo l'apposito *Serial Monitor*.

```
   // inizializza il modulo SPI
   SPI.begin();
   SPI.setClockDivider(SPI_CLOCK_DIV2);
   // resetta e testa il modulo WiFi
   while (!Fishino.reset())
     Serial << F("Fishino RESET FAILED, RETRYING....\n");
   Serial << F("Fishino WiFi RESET OK\n");
```

Si utilizza la funzione `Fishino.reset()` per inizializzare il modulo WiFi inviandogli un reset del software. Tale procedura è obbligatoria per garantire un avvio corretto del modulo.

```
   // imposta la modalità stazione
   Fishino.setMode(STATION_MODE);
```

Tale funzione imposta la modalità di funzionamento del modulo WiFi: in particolare si imposta la modalità stazione in quanto richiede la presenza di un router WiFi a cui connettersi.

```
   // Inizializza la connessione WiFi
   Serial << F("Setting up the Wifi connection...\n");
   // tenta la connessione finchè non riesce
   while (!Fishino.begin(MY_SSID, MY_PASS) == STATION_GOT_IP)
```

```
      Serial << F("ERROR CONNECTING TO AP, RETRYING....\n");
    Serial << F("SUCCESFULLY CONNECTED TO AP\n");
```

Tramite `Fishino.begin` si esegue la connessione all'access point passando l'ID e la password di rete precedentemente dichiarati.

```
    #ifdef IPADDR
      Fishino.config(ip);
    #else
      Fishino.staStartDHCP();
    #endif
    delay(1000);
}
void loop(){
    buttonState = digitalRead(buttonPin);
    val = analogRead(analogPin);
```

Per leggere lo stato del pin digitale si utilizza la funzione *digitalRead* che restituisce un valore numerico di tipo intero che può valere solo HIGH o LOW.

La funzione *analogRead* viene richiamata per leggere il segnale analogico in ingresso sul pin A0, restituendo un numero intero compreso tra 0 e 1023 che viene memorizzato all'interno della variabile *val*. Tale valore è quindi frutto della seguente relazione:

$$val = trunc(2^N \times \frac{V_{IN}}{V_{REF}}) \tag{3.7}$$

La funzione effettua così un troncamento tramite cui i valori del segnale compresi all'interno di 1 LSB vengono rappresentati dalla stessa codifica digitale.

```
    V_misura = ((float)val)*5.027/1024.0;
    mm_misura = V_misura*28/5;
```

Si effettua una stima della tensione in ingresso al convertitore che è pari a:

$$V_misura = val \times \frac{V_{REF}}{2^N} = val \times \frac{5{,}027}{1024} \tag{3.8}$$

dove V_{REF} indica la tensione operativa del Fishino che in linea teorica è pari a 5 V.

Si nota però sperimentalmente che, misurando la tensione tra il pin AREF e GND tramite un multimetro, essa è leggermente più alta rispetto alle condizioni nominali. Inoltre il valore impostato pari a 5,027 tiene conto anche della calibrazione del guadagno del segnale in tensione in ingresso sul pin A0.

Conoscendo la misura massima che è in grado di effettuare il micrometro laser (28 mm) ed avendo impostato sul condizionatore del sensore un'uscita in tensione compresa nel range $0 \div 5\ V$, si ricava il guadagno del sensore come:

$$G = \frac{28}{5}\left[\frac{mm}{V}\right] \tag{3.9}$$

che permette di stimare il corrispondente valore espresso in millimetri.

```
    fil[i] = mm_misura;
```

```
    if (i < (dim-1)) i++;
    else i = 0;
    for (int j = 0; j < dim; j++){
    misura += (float)fil[j];
    }
misura = (misura / (float)(dim+1))-0.063;
```

Come già anticipato nel Paragrafo 3.2 il valore in tensione letto dal Fishino sul pin A0 è influenzato da rumore che ne causa oscillazioni attorno al valore nominale in uscita dal sensore di misura. S'implementa quindi all'interno dello sketch un filtro a media mobile che consente di elaborare i segnali digitali: si definisce un buffer in grado di contenere un certo numero di campioni e in cui, una volta saturato, ad ogni ciclo viene tolto il campione più vecchio ed aggiunto uno nuovo. Ad ogni passo si somma così il valore dei campioni immagazzinati nel buffer dividendolo per il numero di posti. Si ottiene una media cosiddetta "mobile" poiché il segnale mediato insegue il segnale in ingresso (tramite il *Plotter Seriale* vengono rappresentati in Figura 3.30 gli andamenti del segnale affetto da rumore e quello mediato per una misurazione effettuata con il micrometro su un tubo di diametro pari a circa 6 mm).

Il valore mediato conservato all'interno della variabile *misura* viene depurato del valore di offset letto dal Fishino alla presenza di segnale nullo inviato dal micrometro laser. Per quanto concerne la dimensione del buffer si valuta in fase sperimentale se da una parte sia sufficiente a ridurre l'oscillazione del segnale in tensione e dall'altra se non sia eccessivamente elevata da non permettere un riempimento totale del vettore nel momento in cui venga premuto il pulsante per l'invio della misurazione.

```
    //Serial.println("Andamento_reale,Andamento_mediato");
            //Serial.print(mm_misura);          Serial.print(",");
Serial.println(misura);
```

Le due righe commentate vengono introdotte per plottare gli andamenti dei segnali riportati in Figura 3.30 e non aggiungono funzionalità allo sketch.

```
    // Chiude ogni eventuale connessione prima di inviare una nuova
    //richiesta
    client.stop();
    if (client.connect(server, 8080)){
      //Serial.println("connected");
      digitalWrite(led_con, HIGH);
      if (buttonState == HIGH){
         Serial.println(mm_misura);
         digitalWrite(led_press, HIGH);
         // Richiesta HTTP:
         Serial.print("GET
/phpmyadmin/prod/misurazioni.php?misura=");
         client.print("GET
/phpmyadmin/prod/misurazioni.php?misura=");
         Serial.println(misura);
```

```
            client.print(misura);
            client.print(" ");
            client.print("HTTP/1.1");
            client.println();
            client.println("Host: 192.168.1.53");
            client.println("User-Agent: FishinoWiFi/1.1");
            client.println("Connection: close");
            client.println();
            delay(1000);
        }
         else {
            digitalWrite(led_press, LOW);   //spegne il led
    }
     }
     else{
       digitalWrite(led_con, LOW);
     }
}
```

In questa parte finale del codice, presente all'interno della funzione *void loop()*, si inseriscono le istruzioni all'interno di un *if* che permette di eseguirle solamente se il Fishino è connesso alla rete aziendale. Se la connessione è avvenuta con successo rimane acceso un LED bianco tramite la funzione *digitalWrite* che, viceversa, lo farà spegnere una volta perduta la connessione. L'applicazione di questo LED permette all'operatore di capire se può inviare la misurazione effettuata e in caso segnalare eventuali problemi di connessione. Il secondo *if* permette di passare l'URL, in cui è presente il file in linguaggio PHP responsabile dell'invio del valore conservato all'interno di "misura" al database, quando la variabile *buttonState* è a livello logico alto ovvero quando viene premuto il pulsante. Inoltre viene acceso un LED verde identificativo dell'invio della misurazione effettuata. La funzione "delay" introduce una pausa nel flusso di esecuzione delle istruzioni (il valore passato all'interno della funzione è espresso in millisecondi).

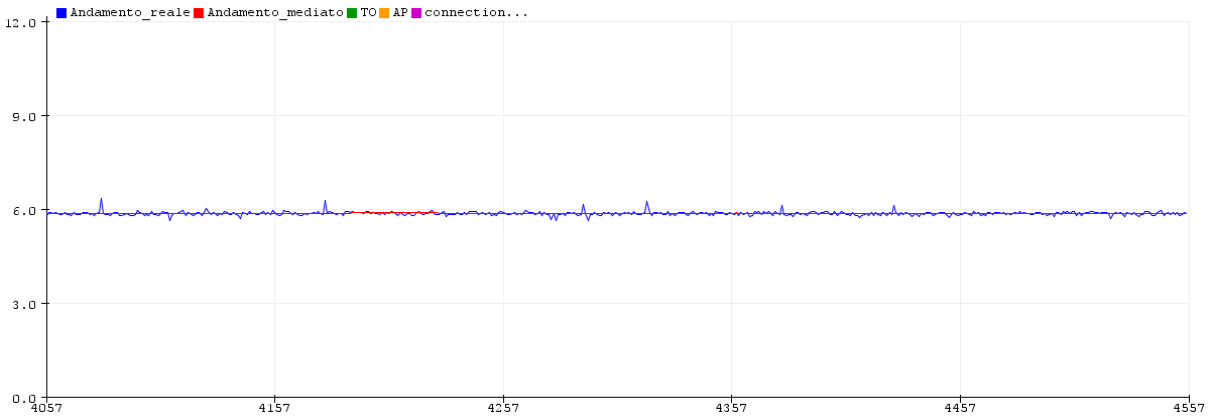


Figura 3.30: Andamenti del segnale a seguito della misurazione di un diametro pari a circa 6 mm. In blu l'andamento affetto da rumore, in rosso l'andamento mediato

3.5 Invio dei dati

Per quanto riguarda la parte di invio dei dati, viene creata la tabella "misura" all'interno del database "produzione". In Figura 3.31 sono riportati i campi che costituiscono la tabella.

	#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito	Commenti	Extra	Azione
☐	1	**id** 🔑	int(11)			No	*Nessuno*		AUTO_INCREMENT	Modifica Elimina Più
☐	2	**codice**	varchar(20)	latin1_swedish_ci		No	*Nessuno*			Modifica Elimina Più
☐	3	**misura**	decimal(10,2)			No	*Nessuno*			Modifica Elimina Più
☐	4	**time**	datetime			No	current_timestamp()			Modifica Elimina Più

Figura 3.31: Campi della tabella "misura" in phpMyAdmin

Sono presenti le seguenti colonne:

- id: numero di identificazione che definisce in modo univoco ciascuna riga di dati memorizzata. I valori assunti da tale colonna sono "int" (abbreviazione di "integer", che indica un numero intero). Alla voce "index" si sceglie "primary" in modo da identificare tale colonna come chiave primaria della tabella. Gli *id* per ciascuna registrazione vengono inoltre generati automaticamente con numeri progressivi tramite la spunta su "A_I".

- codice: indica il codice alfanumerico identificativo del pezzo sottoposto a misurazione. Il tipo di dato con cui è definita questa colonna è un *varchar*, cioè una sequenza di caratteri di lunghezza variabile (lunghezza massima impostata a 20 caratteri).

- misura: definisce la colonna adibita alla registrazione delle misurazioni effettuate che sono valori decimali con due cifre dopo la virgola.

- time: identifica la data corrispondente all'invio della misurazione.

Lo script in PHP è il seguente:

```
1. <?php
2. $connect=mysqli_connect("localhost","root","","produzione");
3. mysqli_query($connect,"INSERT INTO misura (misura) VALUES('".$_
GET["misura"]."')");
```

```
4. ?>
```

Commento:

Riga 1 e 4: contengono i tag di apertura e chiusura del linguaggio.

Riga 2: effettua la connessione al database.

Riga 3: tramite questa stringa viene selezionata la tabella "misura" e aggiunto alla colonna omonima il valore presente nella variabile "misura" all'interno dell'array super-globale `$_GET`. Tramite `INSERT` viene aggiunta una riga ogni qual volta si effettua una misurazione.

Per maggior chiarezza si descrive brevemente il procedimento con cui l'operatore effettua la misurazione. Nella postazione di controllo si effettuano infatti misurazioni su codici diversi tra loro. Ogni volta che l'operatore si trova a misurare un codice differente invia, tramite App su tablet, una nuova riga nella tabella "misura" contenente solamente la stringa del codice e la data. Quindi effettuando le misurazioni di quello stesso codice si vanno ad aggiungere all'interno della tabella tante righe quante sono le misurazioni effettuate.

3.6 Calibrazione del sistema di misurazione

Prima di procedere alla sperimentazione del sistema di misura si effettua una calibrazione per verificare che le misurazioni inviate abbiano degli scostamenti dai valori veri che possano essere trascurati per le applicazioni in esame. Per quanto riguarda la vera e propria parte di trasduzione, il micrometro laser KEYENCE IG-028 viene certificato e sottoposto a calibrazione secondo le normative vigenti. La periodicità con cui viene calibrato dipende sostanzialmente da quanto viene utilizzato: in particolare il dispositivo IG-028 è piuttosto nuovo e presenta la certificazione rilasciata durante l'acquisto. Dalle specifiche garantite del micrometro laser si identifica:

- un errore di linearità pari a $\pm 0{,}1\%$ del F.S ($\pm 28\ \mu m$);
- un errore di ripetibilità pari a $5\ \mu m$ per una distanza di installazione tra trasmettitore e ricevitore fino a 100 mm.

Si nota come errori dell'ordine del centesimo di millimetro siano accettabili per le applicazioni richieste in cui i range di tolleranza non si spingono oltre i decimi di millimetro. Per stabilire però se il dato che viene inviato al database, e quindi registrato, è anch'esso affetto da errori accettabili, si passa alla calibrazione del Fishino. Tale procedura viene effettuata tramite l'utilizzo dei blocchetti pian-paralleli di riscontro, detti anche blocchetti Johnson, andando a considerare come input il valore letto sul display dell'amplificatore del sensore e come output il dato inviato al database. Considerando che non si hanno le condizioni ottimali per poter posizionare i blocchetti in una posizione tale da garantire un allineamento con una precisione almeno dell'ordine del centesimo di millimetro, si decide di posizionare ciascun blocchetto tra trasmettitore e ricevitore fin quando non si trova il corrispettivo valore sul display: ad esempio si prende il blocchetto da 8 mm, lo si posiziona in prossimità del fascio di luce del sensore fino a quando quest'ultimo non restituisce un valore pari a 8,00 mm; quindi, premendo il pulsante, si registra il valore inviato. Se gli errori di linearità e ripetibilità calcolati fossero dell'ordine di grandezza degli errori del sensore, si riterrebbe il sistema valido poiché, pur essendo l'errore complessivo somma dei contributi dell'errore del sensore e del Fishino, si avrebbe comunque un ordine di grandezza trascurabile rispetto al decimo di millimetro.

Gli step della procedura adottata sono i seguenti:

- Si procede inizialmente alla regolazione del valore di zero e del guadagno del Fishino. Ad un segnale di 0 mm del sensore si va a sottrarre all'interno dello sketch un valore pari a 0,06 mm in modo da far corrispondere un valore nullo. Ad un segnale del sensore pari a 20 mm, considerato il massimo del range di calibrazione, si va ad aggiungere un fattore moltiplicativo all'interno dello sketch correggendo il valore in Volt della tensione di riferimento del Fishino (5,027 V).
- Una volta effettuata questa prima regolazione si effettuano tre cicli ciascuno dei quali con sei punti di misura in salita ed in discesa corrispondenti allo 0%, 20%, 40%, 60%, 80% e 100% del fondo scala costituenti i valori veri in ingresso (V_{Vi}).
- Si registrano i valori in uscita in salita (V_{Sij}) ed in discesa (V_{Dij}) per ciascuno dei tre cicli e si calcolano gli errori significativi al fine di avere dei parametri che possano servire a valutare la bontà delle misurazioni.

Di seguito vengono riportate le relazioni che permettono di calcolare i principali errori.

- Errore percentuale in salita:

$$E_{Sij} = \frac{V_{Sij} - V_{Vi}}{C} 100 \tag{3.10}$$

dove C è l'ampiezza del campo (*span*) pari a 20 mm.

Errore percentuale in discesa:

$$E_{Dij} = \frac{V_{Dij} - V_{Vi}}{C} 100 \tag{3.11}$$

- Errore percentuale medio in salita:

$$E_{MSi} = \frac{\sum_{j=1}^{n} E_{Sij}}{n} \tag{3.12}$$

dove n è il numero di cicli pari a 3.

- Errore percentuale medio in discesa:

$$E_{MDi} = \frac{\sum_{j=1}^{n} E_{Dij}}{n} \tag{3.13}$$

- Errore percentuale medio medio:

$$E_{MMi} = \frac{E_{MSi} + E_{MDi}}{2} \tag{3.14}$$

- Errore percentuale d'isteresi:

$$E_{ISTi} = |E_{MSi} - E_{MDi}| \tag{3.15}$$

- Errore percentuale in salita di ripetibilità:

$$E_{RSi} = \sqrt{\frac{\sum_{j=1}^{n} \left(E_{Sij} - E_{MSi}\right)^2}{n}} \tag{3.16}$$

- Errore percentuale in discesa di ripetibilità:

$$E_{RDi} = \sqrt{\frac{\sum_{j=1}^{n}\left(E_{Dij} - E_{MDi}\right)^2}{n}} \tag{3.17}$$

- Il valore medio medio:

$$V_{MMi} = \frac{\sum_{j=1}^{n} V_{Sij} + \sum_{j=1}^{n} V_{Dij}}{2n} \tag{3.18}$$

- Errore percentuale di linearità dipendente:

$$E_{LDi} = \frac{V_{LDi} - V_{MMi}}{C} 100 \tag{3.19}$$

dove $V_{LDi} = m_{LD} V_{Vi} + b_{LD}$ è l'equazione della retta di linearizzazione dipendente passante per i valori di 0% e 100% del valore di *span* ed in cui m_{LD} è il coefficiente angolare e b_{LD} è il termine noto della retta.

- Errore percentuale di linearità indipendente:

$$E_{LIi} = \frac{V_{LIi} - V_{MMi}}{C} 100 \tag{3.20}$$

dove $V_{LIi} = m_{LI} V_{Vi} + b_{LI}$ è l'equazione della retta di linearizzazione indipendente definita tramite il criterio dei minimi quadrati ed in cui m_{LD} è il coefficiente angolare e b_{LD} è il termine noto della retta.

Vengono riportate le tabelle (Tabella 3.1 e Tabella 3.2) prima e dopo la popolazione dei risultati ricavati dalle espressioni precedentemente descritte. I valori di input considerati ed espressi in mm sono pari a: 0, 4, 8, 12, 16 e 20 corrispondenti alle dimensioni dei blocchetti pian-paralleli (per la misurazione pari a 12 mm si riscontrano due blocchetti di dimensioni pari a 8 mm e 4 mm; per la misurazione pari a 16 mm si riscontrano due blocchetti di dimensioni pari a 10 mm e 6 mm).

Tabella 3.1: Procedura di calibrazione

Input		measured value [mm]						E%						E_M %		E_{MM} %	E_{IST} %	E_R %		V_{MM}	E_L %	
		j=1		j=2		j=3		j=1		j=2		j=3										
%	[mm]	rise	desc	rise	desc	rise	desc	rise	desc	rise	desc	rise	desc	rise	desc			rise	desc		ind	dep
0% i=1																						
20% i=2																						
40% i=3	V_{vi}			V_{Sij}	V_{Dij}					E_{Sij}	E_{Dij}			E_{MSi}	E_{MDi}	E_{MMi}	E_{ISTi}	E_{RSi}	E_{RDi}	V_{MMi}	E_{Li}	E_{LDi}
60% i=4																						
80% i=5																						
100% i=6																						

Tabella 3.2: Valori calcolati durante la calibrazione

Input		measured value [mm]						E%						E_M %		E_{MM} %	E_{IST} %	E_R %		V_{MM}	E_L %	
		j=1		j=2		j=3		j=1		j=2		j=3										
%	[mm]	rise	desc	rise	desc	rise	desc	rise	desc	rise	desc	rise	desc	rise	desc			rise	desc		ind	dep
0% (i=1)	0	0	-0,01	-0,01	0	0	0	0	-0,05	-0,1	0	0	0	-0,017	-0,017	-0,017	0	0,024	0,024	-0,003	0	0,017
20% (i=2)	4	3,99	3,99	3,99	4,01	4	4,01	-0,05	-0,05	-0	0,05	0	0,05	-0,033	0,0167	-0,008	0,05	0,024	0,047	3,9983	-0,003	0,008
40% (i=3)	8	8	8,01	8,01	8,01	8,01	8	0	0,05	0,05	0,05	0,05	0	0,033	0,0333	0,0333	0	0,024	0,024	8,0067	-0,04	-0,03
60% (i=4)	12	12,01	12,01	12,01	12	12,01	12	0,05	0,05	0,05	0	0,05	0	0,05	0,0167	0,0333	0,033	0	0,024	12,007	-0,035	-0,03
80% (i=5)	16	16,01	16	16,01	16	16,01	16,01	0,05	0	0,05	0	0,05	0,05	0,05	0,0167	0,0333	0,033	0	0,024	16,007	-0,03	-0,03
100% (i=6)	20	20	20	20	20	20,01	20	0	0	0	0	0,05	0	0,017	0	0,0083	0,017	0,024	0	20,002	0	-0,01

Si ricava così:

- l'accuratezza come valore minimo e massimo tra gli errori percentuali in salita ed in discesa;
- l'errore medio come valore minimo e massimo tra gli errori percentuali medi in salita ed in discesa;
- l'isteresi come valore massimo tra gli errori percentuali d'isteresi;
- la ripetibilità come valore massimo tra gli errori di ripetibilità;
- l'errore di linearità dipendente come valore massimo tra i corrispettivi errori;
- l'errore di linearità indipendente come valore massimo tra i corrispettivi errori.

Dai valori ottenuti mostrati in Tabella 3.3 si può constatare che gli scostamenti che si hanno tra il valore letto dal display del micrometro laser ed il valore inviato al database sono ben più piccoli dei decimi di millimetro che interessano le tolleranze richieste.

Tabella 3.3: Calcolo degli errori

Accuratezza (%)	-0,05	0,05
Errore medio (%)	-0,03	0,05
Isteresi (%)	0,05	
Ripetibilità (%)	0,047	
Linearità dipendente (%)	-0,033	
Linearità indipendente (%)	-0,04	

Si può concludere che il sistema per le misurazioni di controllo studiato è in grado di restituire misure affidabili per le applicazioni dell'azienda.

Si può inoltre affermare che la dimensione del buffer, responsabile della media mobile, imposta all'interno dello sketch del Fishino è sufficiente. Si verifica che una dimensione del buffer superiore introduce un tempo di esecuzione del programma per il riempimento del buffer troppo eccessivo causando un invio del dato errato. Infatti se, nell'intervallo di tempo che intercorre il posizionamento del pezzo e la pressione del pulsante, il buffer non fosse ancora saturo, il valore inviato risulterebbe più piccolo poiché la media effettuata avrebbe al numeratore una somma delle misurazioni sottostimata.

Di conseguenza si può procedere con alcune misurazioni allo scopo di evidenziare gli aspetti positivi e negativi del sistema realizzato.

4. Misurazioni ed analisi dei dati

L'obiettivo del sistema di misura è quello di poter esser affiancato a determinate macchine di produzione per poter effettuare misure di controllo sui pezzi prodotti ed avere uno storico di misurazioni che possa permettere di individuare nel tempo le cause di eventuali scostamenti dal campo di tolleranza.

Oggetto di questa tesi è quindi la sperimentazione del sistema realizzato: a tal proposito si prende in esame un lotto composto da 80 tubi con un diametro pari a 6 mm. Le motivazioni da ricondurre a questa scelta sono le seguenti:

- i tubi presentano una lavorazione sull'estremità che dev'essere oggetto di controllo;
- a fase di lavorazione ultimata i tubi non sono ancora piegati e si agevola così la procedura di misurazione;
- il diametro del tubo sia prima che dopo la lavorazione non supera i 28 mm (limite massimo di misura del micrometro laser);
- disponibilità di un lotto con un numero di tubi apprezzabile per poter effettuare qualche considerazione sul processo di lavorazione.

Il codice del tubo lavorato è X00610126. In Figura 4.1 ed in Figura 4.2 viene mostrato il tubo prima e dopo la lavorazione.

Figura 4.1: Tubo di diametro 6 mm prima della deformazione

Figura 4.2: Tubo di diametro 6 mm a seguito della formazione dell'ogiva

La lavorazione del tubo consiste in una deformazione dell'estremità per la formazione di un'ogiva oggetto della misurazione di controllo. Il processo di deformazione viene effettuato alla macchina 023: si tratta di una pressa frontale modello BEMA (mostrata in Figura 4.3 durante una fase di manutenzione).

Figura 4.3: Pressa frontale BEMA

La macchina è adibita alla deformazione assiale dei tubi ed è dotata di tre cilindri idraulici: il primo cilindro verticale si occupa del serraggio del pezzo durante la lavorazione; il secondo cilindro verticale è responsabile del movimento dell'utensile qualora si desiderasse effettuare una seconda lavorazione ed equipaggiando la macchina con un ulteriore utensile; il terzo cilindro orizzontale (non visibile in Figura) è responsabile della lavorazione assiale che avviene grazie alla presenza di una boccola sull'estremità dello stelo. Le misurazioni di controllo si effettuano sul diametro maggiore dell'ogiva che si realizza a seguito della lavorazione.

Per poter garantire una corretta misurazione del pezzo si utilizza un supporto di ancoraggio del pezzo, realizzato in stampa 3D (mostrato in Figura 4.4), che permette, anche grazie alla presenza di una calamita, di posizionare il pezzo perpendicolarmente al raggio di luce laser emessa dal micrometro.

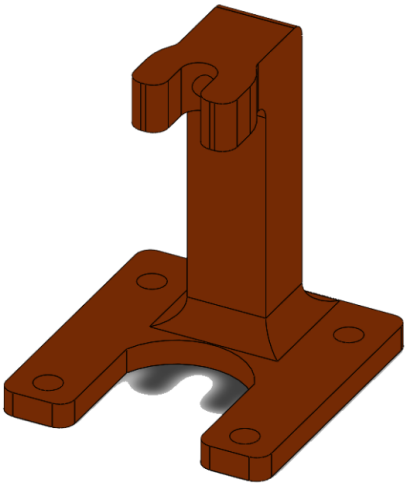


Figura 4.4: Supporto tubo Ø6 mm in PLA

Si realizzano inoltre le due parti fondamentali che vanno a costituire il riferimento per la misurazione: la piastra di centraggio in alluminio (Figura 4.5) viene avvitata al sensore e permette, grazie al foro filettato disposto coassialmente al tubo perpendicolare al fascio laser, di avvitare l'estremità filettata del cilindro in cui è presente il perno che si inserisce nel tubo durante la misurazione (Figura 4.6).

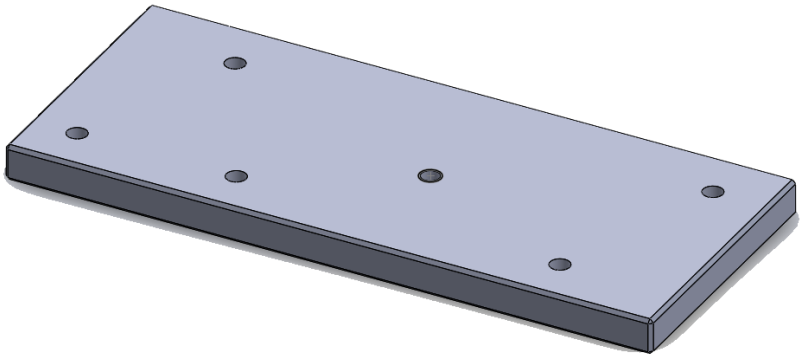


Figura 4.5: Piastra di centraggio in alluminio

Figura 4.6: Cilindro con perno in acciaio che costituisce il riferimento per il tubo

Il tubo preso in considerazione (mostrato in Figura 4.1 prima della lavorazione) ha una lunghezza pari a 113 mm e la parte del tubo non soggetta a deformazione ha un diametro di 6 mm. La base di appoggio, su cui è presente il perno, viene realizzata al tornio ed è costituita da un cilindro in acciaio con diametro pari a 12 mm ed una lunghezza pari a 7,1 mm in cui, da una parte è presente il perno di diametro pari a 3,5 mm che permette l'inserimento del tubo durante la misurazione e dall'altra il perno filettato che si avvita sulla base di appoggio a sua volta avvitata al trasmettitore ed al ricevitore del micrometro laser. La quota più importante che permette di effettuare la misurazione con successo è sicuramente l'altezza del cilindro d'appoggio (7,1 mm) che permette, una volta inserito il perno nel tubo, di allineare il diametro massimo dell'ogiva, realizzata dalla deformazione, con il fascio di luce emesso dal micrometro. Grazie all'accoppiamento filettato tra base di appoggio e cilindro si garantisce una certa versatilità al sistema potendo sostituire solamente la parte cilindrica per effettuare misurazioni su tubi differenti. In Figura 4.7 viene mostrata la stazione di controllo dei diametri durante la fase sperimentale.

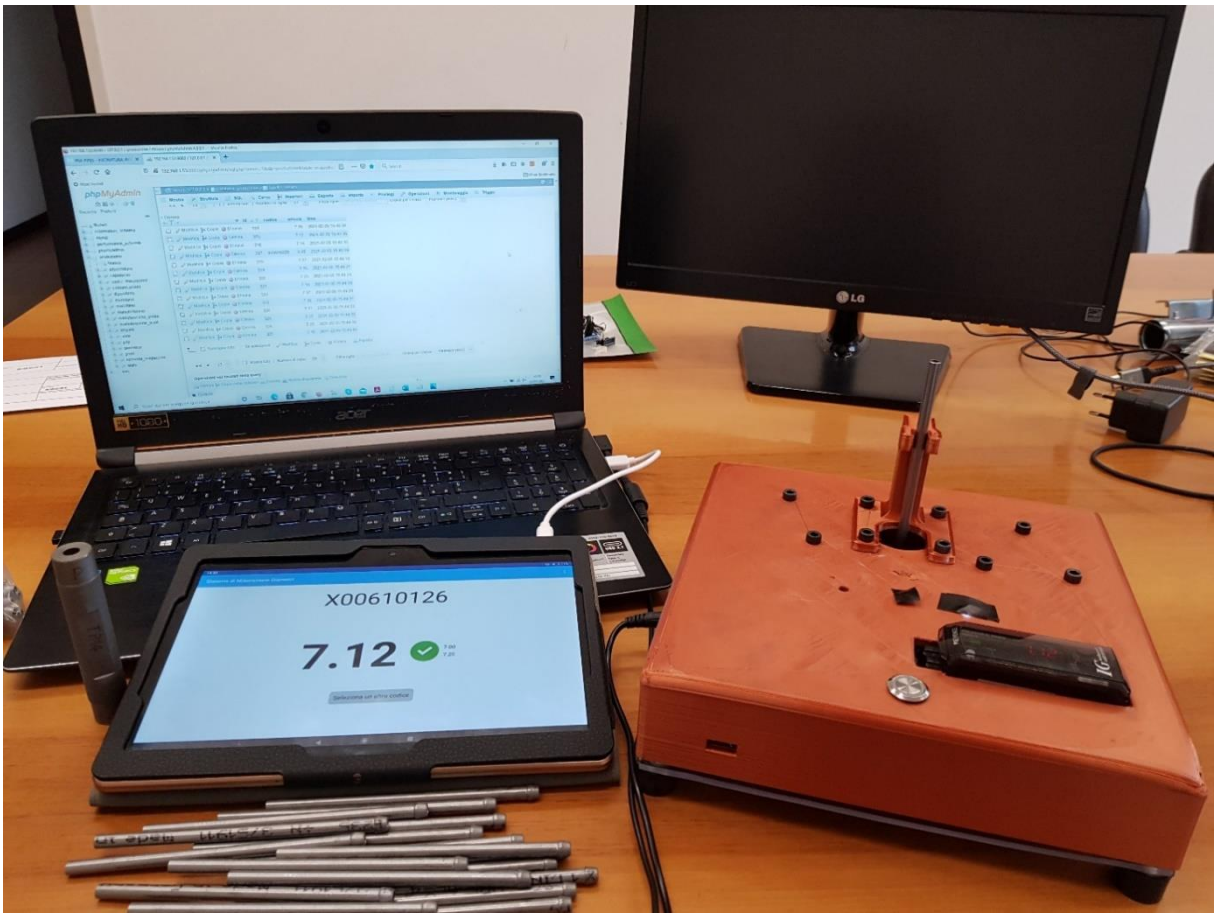


Figura 4.7: Fase sperimentale della stazione di controllo diametri

Le misurazioni vengono svolte su un lotto di 80 pezzi preso a lavorazione conclusa. Si decide di effettuare 10 misurazioni per ciascun codice per un totale di 800 misurazioni a causa della particolare lavorazione sul tubo. Infatti l'asse del cilindro che ha come diametro la dimensione maggiore dell'ogiva non è perfettamente coassiale all'asse del tubo, ma risulta leggermente inclinato determinando quindi un diverso valore di misurazione a seconda di come è ruotato l'asse. Di conseguenza ad ogni misurazione effettuata sullo stesso

tubo lo si fa ruotare di un certo angolo attorno al perno e si calcola una media sulle dieci misurazioni.

In Tabella 4.1 si riportano le misurazioni mediate effettuate sugli 80 tubi. (Le 800 misurazioni totali vengono riportate in Appendice).

Tabella 4.1: Misurazioni mediate degli 80 tubi (in rosso sono riportate le misure fuori tolleranza)

Lotto 80 pz							
Numero pz	**Misura [mm]**	**Numero pz**	**Misura [mm]**	**Numero pz**	**Misura [mm]**	**Numero pz**	**Misura [mm]**
1	7,10	**21**	7,08	**41**	7,10	**61**	7,11
2	7,13	**22**	7,04	**42**	7,12	**62**	7,00
3	6,98	**23**	7,02	**43**	6,99	**63**	7,03
4	7,05	**24**	7,14	**44**	7,13	**64**	7,07
5	7,11	**25**	7,13	**45**	7,00	**65**	7,21
6	7,04	**26**	7,07	**46**	7,19	**66**	7,09
7	7,13	**27**	6,97	**47**	7,05	**67**	6,98
8	6,93	**28**	7,02	**48**	7,04	**68**	7,11
9	7,10	**29**	7,01	**49**	7,03	**69**	7,05
10	6,96	**30**	7,17	**50**	7,04	**70**	7,05
11	7,04	**31**	7,03	**51**	7,11	**71**	7,11
12	6,89	**32**	7,16	**52**	7,04	**72**	7,12
13	7,10	**33**	7,16	**53**	7,05	**73**	7,10
14	7,05	**34**	7,18	**54**	7,05	**74**	7,12
15	7,02	**35**	7,19	**55**	7,18	**75**	6,95
16	7,06	**36**	7,02	**56**	7,07	**76**	7,03
17	7,10	**37**	7,13	**57**	7,07	**77**	7,15
18	7,02	**38**	7,14	**58**	7,07	**78**	7,01
19	7,00	**39**	6,99	**59**	7,22	**79**	6,94
20	7,02	**40**	7,10	**60**	7,04	**80**	6,94

Da disegno si ricavano le tolleranze imposte sul diametro dell'ogiva controllata: alla quota nominale pari a 7,1 mm è applicato un campo di tolleranza di 0,2 mm con uno scostamento superiore pari a $+0{,}1\,mm$ ed uno inferiore pari a $-0{,}1\,mm$ ($7{,}1^{\pm 0{,}1}$). In Tabella 4.1 vengono evidenziati in rosso i pezzi fuori tolleranza. Si riscontra un numero pari a 13 tubi fuori tolleranza che corrisponde ad un 16,25% dei pezzi totali. Si evidenzia inoltre come solamente due tubi superino l'estremo superiore di 7,2 mm: in particolare il tubo numero 59 va oltre di un valore di 0,02 mm ed il numero 65 di un valore di 0,01 mm.

In Figura 4.8 ed in Figura 4.9 vengono inoltre rappresentati rispettivamente l'istogramma e la distribuzione normale sulle 800 misurazioni effettuate. Si può notare un picco importante per valori compresi tra i 7,022 mm e i 7,051 mm corrispondente ad un numero di misurazioni pari a 190.

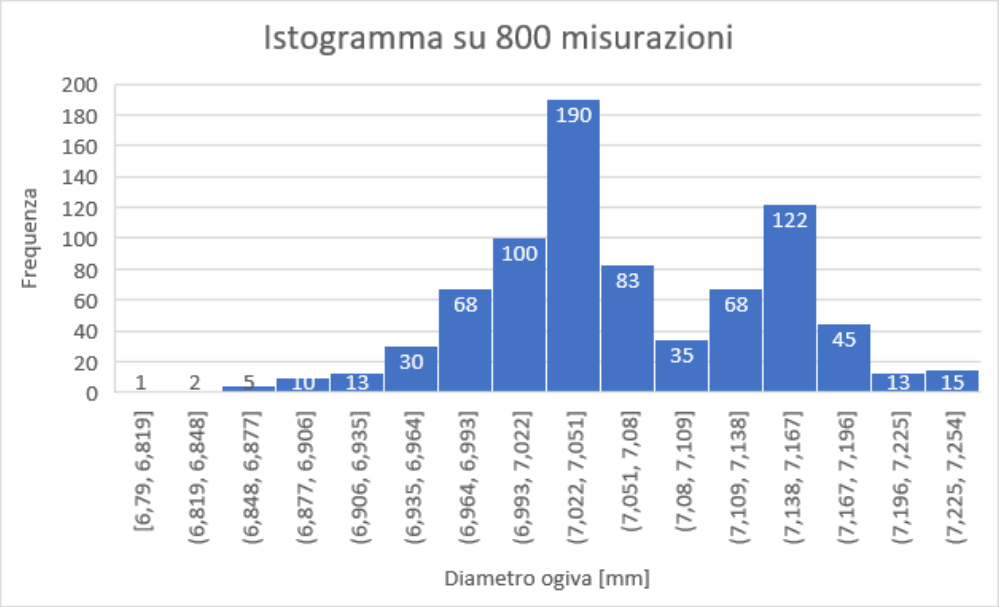


Figura 4.8: Istogramma delle 800 misurazioni effettuate sul codice X00610126

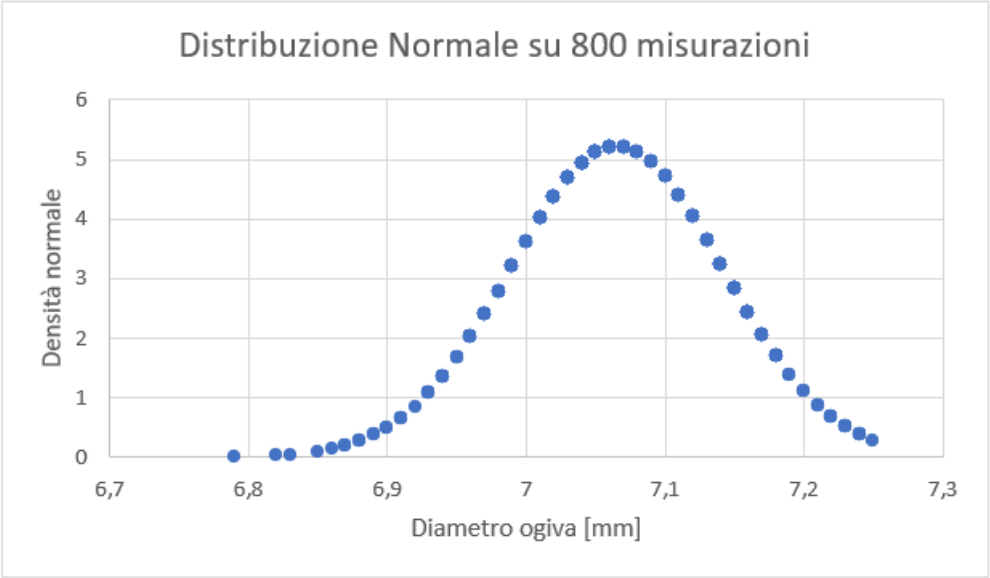


Figura 4.9: Curva Gaussiana delle 800 misurazioni effettuate sul codice X00610126

Sulla base dei primi risultati ottenuti si può già effettuare qualche considerazione:

- Dalla Tabella 4.1 si riscontra un considerevole numero di pezzi fuori tolleranza pari al 16,25% dei tubi del lotto (13 tubi). Di questa percentuale un buon 13,75% (11 tubi) corrisponde a pezzi al di sotto del diametro minimo accettabile (7,0 mm) e solamente un 2,5% (2 tubi) corrisponde a pezzi al di sopra del diametro massimo accettabile (7,2 mm). Questo fenomeno è prevedibile considerando il tipo di lavorazione che viene effettuata. Infatti poiché l'estremità del tubo prima di essere deformata dalla boccola dev'essere riscontrata appositamente su un riferimento, se ne deduce che se l'operatore non riscontra in modo ottimale il pezzo, durante la lavorazione si deformerà meno materiale scaturendo un diametro dell'ogiva inferiore.

- Dall'analisi dell'istogramma in Figura 4.8 si può facilmente notare un notevole numero di misurazioni (190) con valori compresi tra i 7,02 mm e 7,05 mm mentre si vorrebbe tendere ad avere un numero maggiore di misurazioni nell'intorno del valore nominale pari a 7,1 mm.
- Dalla curva Gaussiana di Figura 4.9 si evince come il massimo della campana sia attorno al valore medio di 7,06 mm, mentre la larghezza della campana sia dipendente dalla dispersione dei valori attorno alla media calcolata tramite la deviazione standard σ pari a 0,07. Per quanto riguarda il valore di deviazione standard, ad un primo sguardo, si potrebbe pensare ad un buon valore rispetto al valore medio: infatti il "*coefficient of variation*" ovvero il coefficiente di variazione (COV%), pari al rapporto tra deviazione standard e media, è pari all'1%. Considerando però che il campo di tolleranza è pari a 0,2 mm allora la deviazione standard diventa considerevole ed influente sulla riuscita del processo di lavorazione.

4.1 Considerazioni sui valori fuori tolleranza

Dopo aver individuato quali tubi hanno una media che è fuori tolleranza, si riportano in Tabella 4.2, 4.3, 4.4 e 4.5 le misure totali effettuate su alcuni di questi tubi in quanto rappresentano dei casi particolari.

Tabella 4.2: Totale delle misurazioni sul tubo 3 e 27

Tubo 3	Misura [mm]	Tubo 27	Misura [mm]
1	6,97	**1**	7,02
2	6,98	**2**	6,87
3	6,99	**3**	6,97
4	6,97	**4**	6,99
5	6,98	**5**	6,96
6	7,01	**6**	7,02
7	6,99	**7**	6,91
8	6,98	**8**	6,94
9	6,99	**9**	6,98
10	6,96	**10**	7,05
Media [mm]	6,98	**Media [mm]**	6,97
Min [mm]	6,96	**Min [mm]**	6,87
Max [mm]	7,01	**Max [mm]**	7,05

Tabella 4.3: Totale delle misurazioni sul tubo 39 e 43

Tubo 39	Misura [mm]	Tubo 43	Misura [mm]
1	6,99	**1**	7,01
2	6,95	**2**	7,01
3	6,97	**3**	7,01
4	7	**4**	7,01
5	7,02	**5**	6,98
6	6,99	**6**	6,97
7	6,96	**7**	6,96
8	6,96	**8**	6,97
9	7,03	**9**	6,97
10	7,04	**10**	7,03
Media [mm]	6,99	**Media [mm]**	6,99
Min [mm]	6,95	**Min [mm]**	6,96
Max [mm]	7,04	**Max [mm]**	7,03

Tabella 4.4: Totale delle misurazioni sul tubo 67 e 79

Tubo 67	Misura [mm]	Tubo 79	Misura [mm]
1	6,97	**1**	6,98
2	6,94	**2**	6,89
3	6,97	**3**	6,96
4	7,01	**4**	6,9
5	6,96	**5**	6,94
6	6,97	**6**	6,96
7	6,98	**7**	6,95
8	6,99	**8**	6,97
9	7,02	**9**	7
10	7,02	**10**	6,89
Media [mm]	6,98	**Media [mm]**	6,94
Min [mm]	6,94	**Min [mm]**	6,89
Max [mm]	7,02	**Max [mm]**	7

Tabella 4.5: Totale delle misurazioni sul tubo 80

Tubo 80	Misura [mm]
1	6,99
2	6,89
3	6,94
4	6,94
5	7,02
6	6,9
7	6,88
8	6,93
9	6,96
10	6,9
Media [mm]	6,94
Min [mm]	6,88
Max [mm]	7,02

Da come si evince dalle tabelle, che rappresentano i tubi con medie al di sotto del limite inferiore imposto, vi sono 7 tubi in cui almeno una misurazione sulle dieci condotte è al di sopra dei 7 mm. Questo fattore è causato, come già descritto, dalla non uniformità della lavorazione che in questo caso permette, seppur di poco, di far rientrate i pezzi all'interno del range accettabile. Si verifica infatti, tramite l'utilizzo del calibro a tampone passa-non passa, che tali diametri effettivamente non passano dal lato "non passa" e quindi non sono da scartare.

A valle di questa considerazione i tubi fuori tolleranza risultano essere 6 (7,5% dei tubi totali), 4 dei quali sotto il limite inferiore.

Se si volesse dare un'impronta più quantitativa sull'andamento del processo di lavorazione considerato si potrebbe fare un'analisi più approfondita tramite l'SPC (Statistical Process Control) ovvero il controllo di processo statistico.

4.2 Controllo di processo statistico

Il controllo di processo statistico, in generale, ha come principale obiettivo il miglioramento della qualità intesa come fattore determinante nelle decisioni del cliente per la scelta del prodotto. Un fattore sicuramente incisivo sulla qualità del prodotto è la variabilità: si può infatti definire la qualità come fattore inversamente proporzionale alla variabilità. A tal proposito l'SPC si occupa di stimare la variabilità nelle caratteristiche di un prodotto in modo di essere in grado di prevedere la percentuale di prodotti che soddisfino i limiti di specifica imposti.

La variabilità che un prodotto può avere all'interno del suo processo produttivo può essere attribuita a due tipi di cause:

- Cause comuni, le quali sono associate alla variabilità intrinseca al processo produttivo e sono spesso cause inevitabili;

- Cause speciali dovute ad eventi occasionali. Ne sono un esempio la regolazione non ottimale dei macchinari di produzione oppure gli errori commessi dall'operatore. Questi tipi di cause sono decisamente più influenti rispetto alle altre sulla variabilità del prodotto.

Si deduce quindi che un processo che funziona in assenza di cause speciali è un processo in *controllo statistico*; un processo che è soggetto a cause di tipo speciali è invece detto *fuori controllo statistico*.

Alla luce di queste definizioni si può facilmente stabilire che il processo di lavorazione che ha permesso la deformazione dei tubi su cui si sono condotte le misurazioni è un processo fuori controllo statistico. Le variabili che entrano in gioco durante la lavorazione, infatti, sono molteplici: la più importante è sicuramente l'accortezza dell'operatore nel posizionare il tubo prima della lavorazione che è responsabile della quantità di materiale che si deforma per formare l'ogiva. Inoltre anche una scarsa attenzione durante l'attrezzaggio della macchina potrebbe essere di fondamentale rilevanza sul risultato della lavorazione.

4.2.1 Capacità di processo

L'analisi della capacità di processo permette di stimare l'allineamento tra il processo (*Voice of Process*) e le aspettative del cliente in termini di limiti di specifica (*Voice of Customer*). Gli studi relativi alla capacità di processo sono quindi utilizzati per confrontare la variazione dei dati rispetto alle tolleranze ingegneristiche. Può essere utile avere dei parametri quantitativi che siano in grado di stabilire la capacità di uno specifico processo. S'individuano così due indici fondamentali: il C_p ed il C_{pk}. Le ipotesi necessarie al fine di calcolare correttamente tali indici sono:

- processo in controllo;
- istogramma dei dati gaussiano;
- limiti di specifica imposti (si indica con LSL, *lower specification limit*, il limite di specifica inferiore e USL, *upper specification limit*, il limite di specifica superiore).

Come già specificato il processo di lavorazione considerato è un processo fuori controllo statistico che non permette di soddisfare la prima ipotesi per il calcolo degli indici. Si decide comunque di trascurare questa ipotesi poiché, nonostante il valore numerico calcolato perda di significato, il concetto che sta alla base del calcolo di questi parametri è molto importante per avere una visione dell'evoluzione del processo di produzione in esame e individuare le cause responsabili di eventuali non conformità.

Una semplice misura dell'abilità del processo a produrre elementi all'interno dei limiti specificati è fornita dall'indice di capacità C_p definito come:

$$C_p = \frac{USL - LSL}{6\sigma} \tag{4.1}$$

dove 6σ è la variazione naturale del processo definita come sei volte la deviazione standard e quindi il range in cui finiscono quasi tutte le misurazioni effettuate (99,7% se l'andamento è gaussiano). Il risultato del C_p può cadere nelle seguenti tre possibilità:

- $C_p > 1$ indica che il processo è capace di produrre la quasi totalità dei prodotti entro i limiti di specifica;
- $C_p = 1$ indica che il processo è appena capace di produrre entro le tolleranze;
- $C_p < 1$ indica che il processo non è capace di produrre il 100% di misure accettabili.

In Figura 4.10 viene rappresentato un caso generico di distribuzione gaussiana in cui sono rappresentati i tre casi descritti.

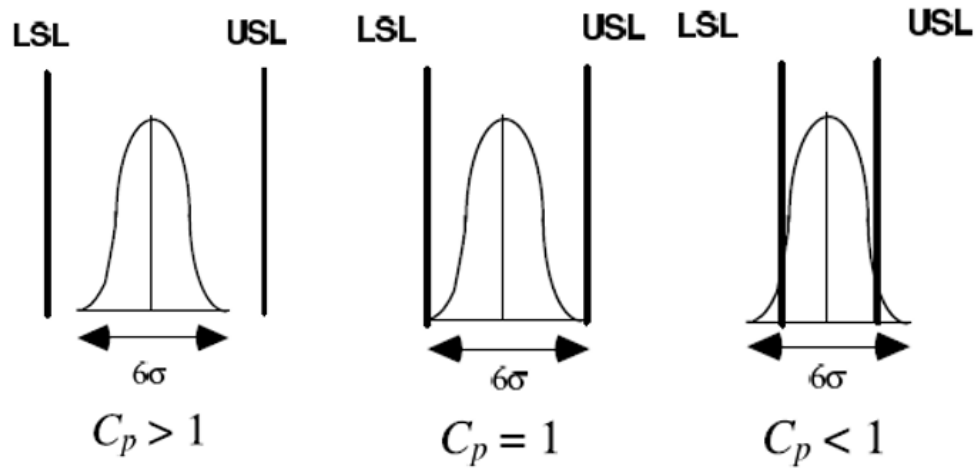


Figura 4.10: Distribuzione normale nei casi di C_p>1, C_p=1, C_p<1

Per un processo esistente il valore raccomandato di C_p è pari a 1,33. Da come si evince dalla sua definizione, tale indice non tiene conto della posizione della media delle misurazioni rispetto alle specifiche imposte e quindi determina un'analisi non completa del processo.

Per tale motivo s'introduce l'indice C_{pk} che è in grado di tenere conto sia della variabilità del processo che della centratura. Esso è definito come:

$$C_{pk} = min\left\{\frac{USL - \mu}{3\sigma}, \frac{\mu - LSL}{3\sigma}\right\} \tag{4.2}$$

dove μ è la media delle misurazioni. Anche in questo caso si possono riscontrare i seguenti casi:

- $C_{pk} > 1$ indica che il processo è capace di produrre entro le tolleranze;
- $C_{pk} = 1$ indica che il processo è appena capace di produrre entro i limiti;
- $C_{pk} < 1$ indica che il processo non è capace di produrre la quasi totalità dei pezzi entro le tolleranze imposte;
- $C_{pk} < 0$ indica che la media del processo è all'esterno dei limiti di specifica.

In Figura 4.11 viene rappresentato graficamente quanto descritto per un caso di curva gaussiana generica.

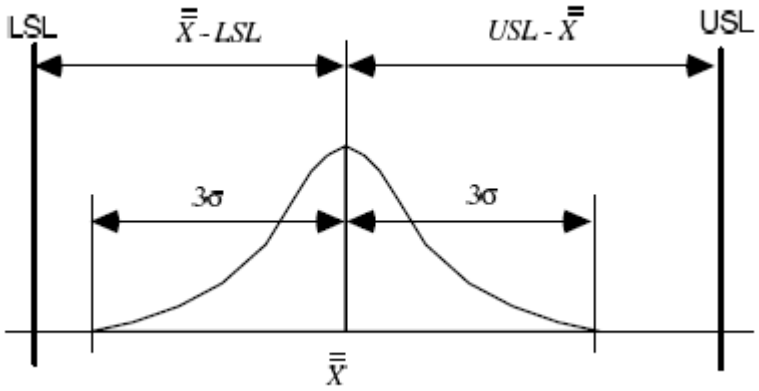


Figura 4.11: Significato grafico dei parametri coinvolti nel calcolo di C_{pk}

Per le misurazioni effettuate sugli 80 tubi si calcolano prima i coefficienti nel caso degli 80 valori mediati, poi sostituendo a quei tubi, le cui medie risultano al di sotto del limite inferiore ma che hanno almeno una tra le dieci misurazioni al di sopra di 7,00 mm, il valore massimo tra le misure.

Per il primo caso si procede quindi al calcolo di media μ e deviazione standard σ e si trova:

- $\mu = 7{,}07\ mm$
- $\sigma = 0{,}069$

Quindi considerando un $USL - LSL = 0{,}2\ mm$ si ricava:

- $C_p = 0{,}483$
- $C_{pk} = 0{,}320$

Valori che, come ci si aspettava, sono decisamente al di sotto di quelli raccomandati stabilendo che il processo non è in grado di produrre la quasi totalità dei pezzi entro le tolleranze imposte. Dalla Gaussiana riportata in Figura 4.12 si ha il riscontro dei valori di C_p e C_{pk} calcolati: si evidenzia come alcuni tubi siano al di fuori del range di tolleranza e come ci siano troppi diametri prossimi al limite inferiore.

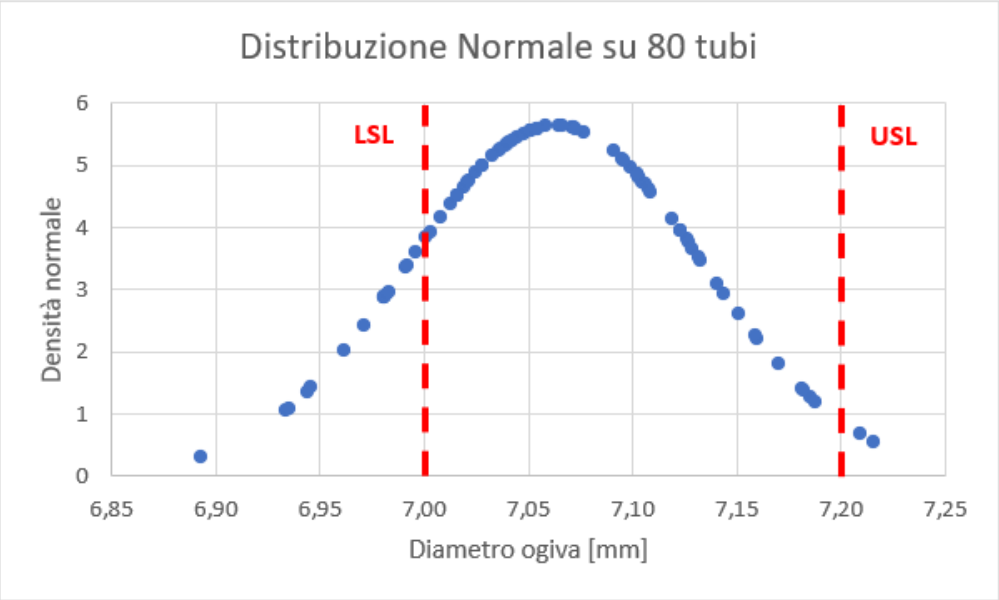


Figura 4.12: Curva Gaussiana su 80 tubi in relazione ai limiti di specifica imposti

Per il secondo caso si sostituisce:

- Al tubo 3 un valore di 7,01 mm;
- Al tubo 27 un valore di 7,05 mm;
- Al tubo 39 un valore di 7,04 mm;
- Al tubo 43 un valore di 7,03 mm;
- Al tubo 67 un valore di 7,02 mm;
- Al tubo 79 un valore di 7,00 mm;
- Al tubo 80 un valore di 7,02 mm.

I valori così trovati sono:

- $\mu = 7{,}07\ mm$;
- $\sigma = 0{,}064$;
- $C_p = 0{,}522$;
- $C_{pk} = 0{,}370$.

Si evince un valore medio invariato, una deviazione standard leggermente inferiore e dei valori degli indici C_p e C_{pk} di poco superiori ma comunque sempre inferiori all'unità.

La curva gaussiana che ne deriva dalle misure così ottenute viene rappresentata in Figura 4.13.

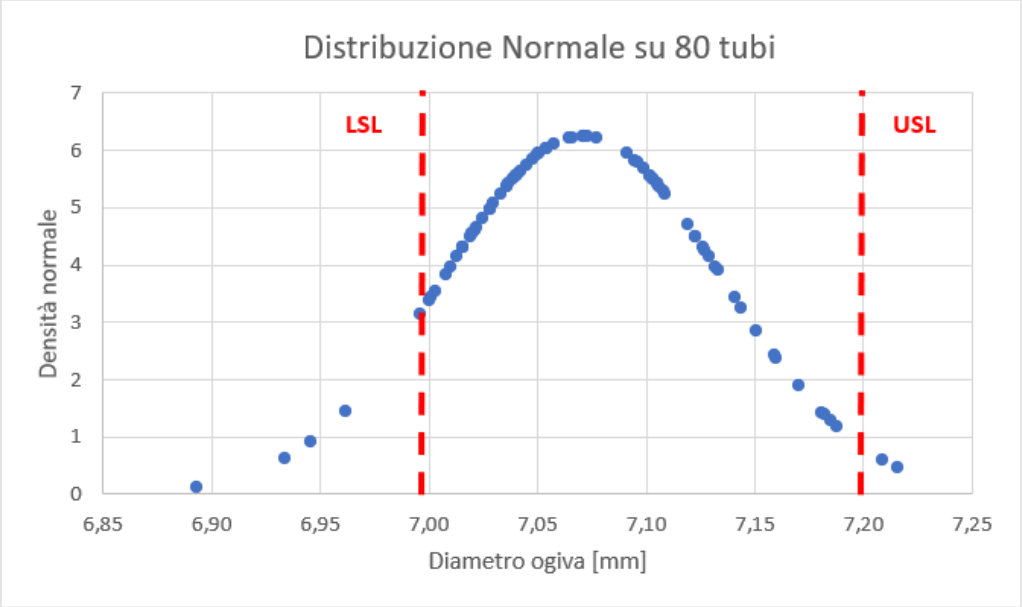


Figura 4.13: Curva Gaussiana su 80 tubi in relazione ai limiti di specifica considerando i valori massimi per i sette tubi con media fuori tolleranza

4.3 Considerazioni sui risultati

Dall'analisi dei risultati ottenuti si ottengono importanti parametri di valutazione sul processo di lavorazione. I risultati numerici ottenuti dal calcolo degli indici C_p e C_{pk} sono evidentemente al di sotto dei valori accettabili. Non ci si sofferma però sui valori numerici poiché, come già detto in precedenza, il processo è fuori controllo statistico ed i valori che si ottengono non possono essere paragonati a quelli che si otterrebbero per processi in controllo statistico. Tuttavia questo tipo di analisi ha permesso comunque di avere una visione sul processo produttivo limitatamente agli 80 tubi deformati. Si è potuto infatti constatare che il numero di tubi con dimensione del diametro dell'ogiva prossima al limite inferiore è troppo elevato e questo è già sufficiente a garantire un primo campanello d'allarme. La causa principale da ricondurre a tale fenomeno è l'errato posizionamento del tubo da parte dell'operatore: si può confermare che se durante la lavorazione il tubo non è tenuto correttamente in posizione la deformazione subisce delle variazioni. Si è riscontrata una maggiore probabilità che il pezzo venga mantenuto più distaccato rispetto al riscontro rappresentato da una barra in acciaio scaturendo, una volta che la barra si solleva, una deformazione minore dell'ogiva. Solo in pochi casi si è visto come il diametro sia prossimo al limite superiore imposto: anche in questo caso tale fenomeno è dovuto all'errato posizionamento del tubo da parte dell'operatore che spinge troppo il tubo contro la barra (che presenta un piccolo gioco) determinando una deformazione maggiore dell'ogiva.

Da sottolineare è senza ombra di dubbio la complessità del controllo del diametro massimo dell'ogiva: infatti da una parte si ha la difficoltà nel far coincidere il raggio di luce del sensore con quella che è la sezione con diametro massimo dell'ogiva; dall'altra la non uniformità della deformazione che porta ad effettuare più misurazioni su uno stesso tubo e quindi un tempo di controllo maggiore. Nonostante tali difficoltà il sistema di misura si è rivelato comunque all'altezza del compito in quanto i 6 tubi che sono risultati fuori tolleranza dall'analisi delle misure non hanno neppure soddisfatto il controllo tramite il calibro a tampone passa-non passa.

5. Conclusioni

L'ideazione e la realizzazione dei sistemi descritti permettono di avere non solo informazioni sul processo che fino ad ora non si avevano, ma soprattutto un controllo in tempo reale dei parametri coinvolti. Al giorno d'oggi è di fondamentale importanza essere a conoscenza di più dati possibili sulla produzione di modo che si possa avere una visione più ampia sui processi ed elaborare statistiche sul medio-lungo termine allo scopo di portare benefici sia dal punto di vista produttivo che economico.

Il sistema *contapezzi* progettato ha lo scopo di fornire dati riguardanti il numero di pezzi e di scarti prodotti rappresentando un passo in avanti rispetto ai dispositivi di conteggio dei cicli di lavorazione già presenti: in particolare con il sistema ideato si riesce ad avere un immagazzinamento digitale delle informazioni ed un conteggio veritiero dello scarto. In quest'ottica il linguaggio di programmazione Python si è rilevato molto efficiente nel gestire questa piccola applicazione di automazione. L'introduzione del modulo *RPi.GPIO* ha permesso agevolmente di interfacciare il Raspberry Pi con il segnale proveniente dal sensore KEYENCE PZ-G51P soprattutto permettendo di ottimizzare il conteggio dei pezzi tramite la funzione *GPIO.wait_for_edge()*. Grazie a questa semplice implementazione, responsabile del riconoscimento del fronte di discesa del segnale, si è reso, di fatto, migliore il sistema in quanto il conteggio viene incrementato solo una volta che il tubo esce dal campo di luce del sensore. Tramite lo *storage* del numero di pezzi prodotti si possono ricavare indirettamente alcune importanti informazioni come il tempo di lavoro effettivo della macchina e quindi si possono elaborare strategie manutentive in funzione di questo parametro. Sicuramente un'ulteriore applicazione che può essere studiata al fine di adempiere allo stesso scopo è quella di ricavare il segnale proveniente dal PLC della macchina andando a registrare i dati ricavati in modo analogo. Si è scelto comunque di intraprendere lo studio del sensore fotoelettrico poiché può essere un sistema più versatile potendo essere disposto anche in quelle postazioni in cui non è presente un macchinario.

Per quanto riguarda la stazione di controllo dei diametri progettata si è introdotto un sistema digitale che permette di avere una registrazione delle misurazioni effettuate e quindi avere a disposizione dati di cui fino ad ora non si era in possesso. La parte software realizzata in linguaggio C++ ha permesso di trasformare l'informazione analogica del micrometro KEYENCE IG-028 nel dato digitale di misura. Si è verificato tramite un'opportuna calibrazione che il valore pari a 100, caratterizzante la dimensione del buffer responsabile della media mobile necessaria a ridurre il rumore del segnale, è sufficiente per avere un valore di misura in uscita affetto da errori trascurabili rispetto alle tolleranze dell'ordine dei decimi di millimetro. Si è inoltre potuto constare che tale valore del buffer non è così eccessivo da introdurre un tempo troppo elevato di compilazione dovuto al riempimento del vettore. Un'altra strategia che si può adottare al fine di ridurre il rumore del segnale in ingresso è quella di alimentare separatamente il Fishino ed il micrometro laser, in quanto la tensione di alimentazione derivante da un unico alimentatore potrebbe avere delle oscillazioni causate dalla presenza di più carichi collegati. Non si decide comunque di intraprendere questa strada in quanto si dovrebbe avere a disposizione un alimentatore aggiuntivo ed inoltre si è comunque già verificato uno scostamento massimo tra il valore della misura ricavata dal micrometro ed il dato inviato pari a 1 centesimo di millimetro quindi soddisfacente per le tolleranze coinvolte nei tubi considerati.

L’importanza di questo secondo caso di studio si riflette nella possibilità di elaborare statistiche fondamentali sui processi di lavorazione sulla base delle misurazioni registrate. L’ultima parte del lavoro adibita alla sperimentazione del sistema di misura serve ad evidenziare questo concetto. Dall’analisi dei risultati ottenuti a seguito delle misurazioni effettuate su un lotto di 80 tubi si è potuto stabilire come il diametro dell’ogiva, in una buona parte dei tubi considerati, sia prossima al limite inferiore del campo di tolleranza imposto. Questo è un risultato importante che fa emergere come il processo responsabile della lavorazione debba essere preso in osservazione. Si è potuto infatti stabilire che quasi sicuramente la tendenza da parte dei diametri verso valori quasi sotto-tolleranza sia dovuta al posizionamento errato del pezzo da parte dell’operatore. Il tubo infatti, prima della lavorazione, non viene posizionato fino a battuta contro la barra di riferimento scaturendo una deformazione non completa del pezzo.

Nonostante la difficoltà del controllo dovuta alla particolare lavorazione dell’ogiva che non presenta una completa uniformità lungo il diametro, il sistema di misura si è rivelato in grado di adempiere in modo soddisfacente al proprio compito. Sicuramente può essere oggetto di studio più approfondito la parte di supporto dei tubi: in particolare si può pensare a come poter supportare tubi già curvati rendendo il sistema di misurazione più versatile e completo.

In conclusione questo lavoro di tesi ha permesso di avere un controllo più dinamico sulla produzione tramite il flusso di dati acquisiti in modalità wireless ed in tempo reale. Tramite la realizzazione dei due sistemi si è potuto da una parte eliminare gli errori di registrazione manuale riguardanti il conteggio dei pezzi e degli scarti, dall’altra avere uno storage di misurazioni di cui fino ad ora non si era in possesso. Tramite la logica con cui si sono ideati i sistemi si potrà espandere lo studio, adottando sensori differenti, allo scopo di monitorare diversi processi e poter avviare una manutenzione predittiva in grado di portare sicuramente benefici significativi in termini economici.

6. Bibliografia e sitografia

KEYENCE, 2010, *IG Series User's Manual*, Japan.

KEYENCE, 2010, *PZ-G Series Instruction Manual*, Japan.

Sorli M., 2020, Slide *Servosystem structure*, Politecnico di Torino.

Sorli M., 2020, Slide *Static characteristics of the systems*, Politecnico di Torino.

Sorli M. e Quaglia G., 2017, *Meccatronica*, Epics Edizioni, Torino.

Viktorov V. e Colombo F., 2018, *Automazione dei sistemi meccanici – Corso di base – Quinta edizione*, CLUT, Torino.

Sito Fishino: http://www.fishino.it/home-it.html

Sito KEYENCE: https://www.keyence.it/

Sito Introduzione a PHP: https://www.html.it/pag/16673/cos-php/

Sito MIT App Inventor: https://appinventor.mit.edu/

Sito Process Capability: https://isoarcadia.it/process-capability/

Sito Richieste HTTP: https://www.html.it/pag/62463/le-richieste-http-get-e-post/

Sito Raspberry: https://it.wikipedia.org/wiki/Raspberry_Pi

7. Appendice

Si riporta di seguito il programma completo del Raspberry:

```
import urllib.request
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(23, GPIO.IN)
GPIO.setup(16, GPIO.IN)
GPIO.setup(24, GPIO.IN)
GPIO.setup(17, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(22, GPIO.OUT, initial=GPIO.LOW)
count_1 = 0
scarto_1 = 0
while True:
    pin1 = GPIO.input(23)
    pin2 = GPIO.input(16)
    pin0 = GPIO.input(24)
    if(pin0):
        count_1 = 0
        scarto_1 = 0
    if(pin1):
        GPIO.wait_for_edge(23, GPIO.FALLING)
        count_1 = count_1 + 1
        print("contatore:")
        print(count_1)
        count= "%d" %count_1
        with urllib.request.urlopen("http://192.168.1.53:8080/\
        phpmyadmin/prod/aggiorna.php?count="+count+"")as response:
            html = response.read()
        with urllib.request.urlopen("http://192.168.1.53:8080/\
        phpmyadmin/prod/datafine.php") as response:
            html = response.read()
        GPIO.output(17, GPIO.HIGH)
        time.sleep(1)
```

```
            GPIO.output(17, GPIO.LOW)
            time.sleep(1)
        time.sleep(.1)
        if(pin2):
            scarto_1 = scarto_1 + 1
            print("scarto:")
            print(scarto_1)
            count_1 = count_1 - 1
            count= "%d" %count_1
            print("contatore:")
            print(count_1)
            scarto = "%d" %scarto_1
            with urllib.request.urlopen("http://192.168.1.53:8080/\
            phpmyadmin/prod/aggiornascarto.php?\
            count="+count+"&scarto="+scarto+"") as response:
                html = response.read()
            GPIO.output(22, GPIO.HIGH)
            time.sleep(1)
            GPIO.output(22, GPIO.LOW)
            time.sleep(1)
        time.sleep(.2)
```

Si riporta di seguito il programma completo del Fishino:

```
//definizione librerie
#include <FishinoWebServer.h>
#include <Flash.h>
#include <FishinoUdp.h>
#include <Fishino.h>
#include <SPI.h>
#include <SD.h>
//inizializzazione variabili
byte analogPin = A0;
unsigned int val = 0;
float V_misura = 0;
float mm_misura = 0;
float misura = 0;
const int buttonPin = 2;
const int led_con = 8;
```

```
const int led_press = 7;
int buttonState = 0;
char server[] = "192.168.1.53";
int const dim = 100;
float fil[dim];
int i = 0;
FishinoClient client;
#define MY_SSID  "PsaPipes"
#define MY_PASS "pipes197052"
void setup(){
   pinMode(buttonPin, INPUT);
   pinMode(led_con, OUTPUT);
   pinMode(led_press, OUTPUT);
   Serial.begin(9600);
   // inizializza il modulo SPI
   SPI.begin();
   SPI.setClockDivider(SPI_CLOCK_DIV2);
   // resetta e testa il modulo WiFi
   while (!Fishino.reset())
     Serial << F("Fishino RESET FAILED, RETRYING....\n");
   Serial << F("Fishino WiFi RESET OK\n");
   // imposta la modalità stazione
   Fishino.setMode(STATION_MODE);
   // Inizializza la connessione WiFi
   Serial << F("Setting up the Wifi connection...\n");
   // tenta la connessione finchè non riesce
   while (!Fishino.begin(MY_SSID, MY_PASS) == STATION_GOT_IP)
     Serial << F("ERROR CONNECTING TO AP, RETRYING....\n");
   Serial << F("SUCCESFULLY CONNECTED TO AP\n");
   #ifdef IPADDR
     Fishino.config(ip);
   #else
     Fishino.staStartDHCP();
   #endif
   delay(1000);
}
void loop(){
   buttonState = digitalRead(buttonPin);
```

```
    val = analogRead(analogPin);
    V_misura = ((float)val)*5.0275/1024.0;
    mm_misura = V_misura*28/5;
    fil[i] = mm_misura;
    if (i < (dim-1)) i++;
    else i = 0;
    for (int j = 0; j < dim; j++){
    misura += (float)fil[j];
    }
    misura = (misura / (float)(dim+1))-0.063;
   //Serial.println("Andamento_reale,Andamento_mediato");
   //Serial.print(mm_misura); Serial.print(",");
Serial.println(misura);
    // Chiude ogni eventuale connessione prima di inviare una
nuova richiesta
    client.stop();
    if (client.connect(server, 8080)){
      //Serial.println("connected");
      digitalWrite(led_con, HIGH);
      if (buttonState == HIGH){
        Serial.println(mm_misura);
        digitalWrite(led_press, HIGH);
        // Richiesta HTTP:
        Serial.print("GET
/phpmyadmin/prod/misurazioni.php?misura=");
        client.print("GET
/phpmyadmin/prod/misurazioni.php?misura=");
        Serial.println(misura);
        client.print(misura);
        client.print(" ");
        client.print("HTTP/1.1");
        client.println();
        client.println("Host: 192.168.1.53");
        client.println("User-Agent: FishinoWiFi/1.1");
        client.println("Connection: close");
        client.println();
        delay(1000);
      }
       else {
```

```
        digitalWrite(led_press, LOW);   //spegne il led
  }
   }
   else{
     digitalWrite(led_con, LOW);
   }
}
```

Le 800 misurazioni effettuate sugli 80 tubi vengono riportate nelle tabelle seguenti:

Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]
1	7,12	**41**	7,15	**81**	6,87	**121**	7,13	**161**	7,03
2	7,05	**42**	7,12	**82**	6,98	**122**	7,07	**162**	7,04
3	7,06	**43**	7,08	**83**	6,97	**123**	7,03	**163**	7,02
4	7,13	**44**	7,16	**84**	6,97	**124**	7,03	**164**	7,01
5	7,06	**45**	7,12	**85**	6,98	**125**	7,04	**165**	7,04
6	7,14	**46**	7,12	**86**	6,96	**126**	7,03	**166**	7,02
7	7,06	**47**	7,13	**87**	6,96	**127**	7,06	**167**	7,02
8	7,17	**48**	7,17	**88**	6,98	**128**	7,02	**168**	7,01
9	7,16	**49**	7,13	**89**	6,97	**129**	7,06	**169**	6,99
10	7,07	**50**	7,15	**90**	6,98	**130**	7,01	**170**	7,03
11	7,14	**51**	6,94	**91**	7,03	**131**	7,04	**171**	7,09
12	7,12	**52**	6,96	**92**	7,01	**132**	6,99	**172**	7,04
13	7,15	**53**	6,88	**93**	7,07	**133**	7,03	**173**	7,13
14	7,13	**54**	6,91	**94**	7,03	**134**	7,02	**174**	7,05
15	7,12	**55**	6,97	**95**	7,03	**135**	7,02	**175**	7,14
16	7,14	**56**	6,97	**96**	7,04	**136**	6,95	**176**	7,12
17	7,05	**57**	6,86	**97**	7,04	**137**	7,05	**177**	7,01
18	7,12	**58**	6,93	**98**	7,03	**138**	6,99	**178**	7,08
19	7,15	**59**	6,95	**99**	7,07	**139**	7,01	**179**	7,07
20	7,14	**60**	6,97	**100**	7,02	**140**	7,06	**180**	7,04
21	6,97	**61**	7,06	**101**	6,97	**141**	7,02	**181**	7,06
22	6,98	**62**	7,04	**102**	6,83	**142**	7	**182**	7,07
23	6,99	**63**	7,14	**103**	6,82	**143**	7,04	**183**	7,02
24	6,97	**64**	7,03	**104**	6,97	**144**	6,93	**184**	7,14
25	6,98	**65**	7,02	**105**	6,93	**145**	6,94	**185**	7,04
26	7,01	**66**	7,04	**106**	6,85	**146**	6,99	**186**	6,99
27	6,99	**67**	7,03	**107**	6,95	**147**	6,93	**187**	7,01
28	6,98	**68**	7,06	**108**	6,94	**148**	7,06	**188**	6,99
29	6,99	**69**	7,12	**109**	6,79	**149**	7,1	**189**	7,03
30	6,96	**70**	7,04	**110**	6,88	**150**	7	**190**	7,04
31	6,92	**71**	7,1	**111**	7,05	**151**	7,1	**191**	7
32	7	**72**	7,14	**112**	7,14	**152**	7,05	**192**	7,02
33	6,99	**73**	7,14	**113**	7,1	**153**	7,08	**193**	7,04
34	6,99	**74**	7,11	**114**	7,06	**154**	7,14	**194**	6,91
35	7,01	**75**	7,05	**115**	7,13	**155**	7,08	**195**	7,12
36	6,94	**76**	7,14	**116**	7,14	**156**	7,17	**196**	7,03
37	6,99	**77**	7,05	**117**	7,03	**157**	7,14	**197**	7,03
38	6,96	**78**	7,09	**118**	7,12	**158**	7,07	**198**	6,99
39	7,01	**79**	7,06	**119**	7,06	**159**	7,17	**199**	7,02
40	7	**80**	7,15	**120**	7,16	**160**	7,08	**200**	7,03

Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]
201	7,15	**241**	7,04	**281**	7,18	**321**	6,99	**361**	7,15
202	7,07	**242**	7,02	**282**	7,17	**322**	7,01	**362**	7,02
203	7,15	**243**	7,01	**283**	7,21	**323**	7,01	**363**	7,04
204	7,15	**244**	7,04	**284**	7,13	**324**	7,02	**364**	7,11
205	7,14	**245**	7,03	**285**	7,16	**325**	6,97	**365**	7,12
206	7,16	**246**	7	**286**	7,16	**326**	7,03	**366**	7,06
207	7,19	**247**	7,03	**287**	7,16	**327**	7,06	**367**	7,05
208	7,18	**248**	7,03	**288**	7,08	**328**	7,03	**368**	7,13
209	7,1	**249**	7,02	**289**	7,19	**329**	7,03	**369**	7,14
210	7,15	**250**	7	**290**	7,16	**330**	7,11	**370**	7,05
211	7,15	**251**	7,03	**291**	7,17	**331**	7,12	**371**	7,09
212	7,06	**252**	6,91	**292**	7,16	**332**	7,11	**372**	7,12
213	7,12	**253**	7,04	**293**	7,16	**333**	7,12	**373**	7,11
214	7,14	**254**	7,04	**294**	7,16	**334**	7,07	**374**	7,06
215	7,13	**255**	7,02	**295**	7,16	**335**	7,16	**375**	7,05
216	7,17	**256**	7,03	**296**	7,16	**336**	7,14	**376**	7,1
217	7,15	**257**	7,03	**297**	7,15	**337**	7,14	**377**	7,15
218	7,15	**258**	6,99	**298**	7,17	**338**	7,17	**378**	7,08
219	7,16	**259**	7,02	**299**	7,14	**339**	7,15	**379**	7,15
220	7,04	**260**	7,17	**300**	7,17	**340**	7,04	**380**	7,12
221	7,05	**261**	7,16	**301**	7,15	**341**	7,11	**381**	7,09
222	7,03	**262**	7,22	**302**	7,23	**342**	7,16	**382**	7,11
223	7,07	**263**	7,17	**303**	7,16	**343**	7,16	**383**	7,1
224	7,05	**264**	7,16	**304**	7,17	**344**	7,13	**384**	7,09
225	7,04	**265**	7,17	**305**	7,16	**345**	7,15	**385**	7,16
226	7,04	**266**	7,16	**306**	7,17	**346**	7,14	**386**	7,15
227	7,14	**267**	7,15	**307**	7,21	**347**	7,17	**387**	7,14
228	7,14	**268**	7,16	**308**	7,23	**348**	7,18	**388**	7,12
229	7,13	**269**	7,18	**309**	7,16	**349**	7,17	**389**	7,15
230	7,02	**270**	7,04	**310**	7,22	**350**	6,99	**390**	7,01
231	7,02	**271**	7	**311**	7,23	**351**	6,95	**391**	7,01
232	6,87	**272**	7,02	**312**	7,16	**352**	6,97	**392**	7,01
233	6,97	**273**	7,02	**313**	7,17	**353**	7	**393**	7,01
234	6,99	**274**	7,04	**314**	7,18	**354**	7,02	**394**	6,98
235	6,96	**275**	7,04	**315**	7,18	**355**	6,99	**395**	6,97
236	7,02	**276**	7,05	**316**	7,18	**356**	6,96	**396**	6,96
237	6,91	**277**	7	**317**	7,18	**357**	6,96	**397**	6,97
238	6,94	**278**	7,04	**318**	7,17	**358**	7,03	**398**	6,97
239	6,98	**279**	7,03	**319**	7,18	**359**	7,04	**399**	7,03
240	7,05	**280**	7,16	**320**	7,01	**360**	7,13	**400**	7,15

Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]
401	7,07	**441**	7,03	**481**	7,05	**521**	7,12	**561**	7,03
402	7,15	**442**	7,01	**482**	7,11	**522**	7,08	**562**	7,12
403	7,14	**443**	7,02	**483**	7,04	**523**	7,06	**563**	7,15
404	7,14	**444**	7,07	**484**	7,05	**524**	7,05	**564**	7,07
405	7,07	**445**	7,08	**485**	7,12	**525**	7,04	**565**	7,05
406	7,14	**446**	7,05	**486**	7,08	**526**	7,05	**566**	7,06
407	7,15	**447**	7,05	**487**	7,06	**527**	7,05	**567**	7,09
408	7,16	**448**	7,01	**488**	7,05	**528**	7,03	**568**	7,06
409	7,15	**449**	7,04	**489**	7,04	**529**	7,02	**569**	7,05
410	7,02	**450**	7,04	**490**	7,05	**530**	7,01	**570**	7,04
411	6,98	**451**	7,03	**491**	7,05	**531**	7,05	**571**	7,06
412	6,97	**452**	7,04	**492**	7,03	**532**	7	**572**	7,05
413	7,01	**453**	7,04	**493**	7,02	**533**	7,02	**573**	7,06
414	7,04	**454**	7,01	**494**	7,01	**534**	7,1	**574**	7,09
415	7,03	**455**	7,03	**495**	7,04	**535**	7,09	**575**	7,1
416	6,99	**456**	7	**496**	7,02	**536**	7,01	**576**	7,06
417	6,97	**457**	7,01	**497**	7,01	**537**	7,02	**577**	7,06
418	6,99	**458**	7,03	**498**	7,04	**538**	7,04	**578**	7,07
419	7,03	**459**	7,02	**499**	7,03	**539**	7,07	**579**	7,05
420	7,22	**460**	7,01	**500**	7,03	**540**	7,14	**580**	7,06
421	7,21	**461**	7,03	**501**	7,03	**541**	7,18	**581**	7,22
422	7,24	**462**	7,05	**502**	7,03	**542**	7,18	**582**	7,18
423	7,17	**463**	7,05	**503**	7,02	**543**	7,18	**583**	7,25
424	7,16	**464**	7,08	**504**	7	**544**	7,18	**584**	7,2
425	7,12	**465**	7,02	**505**	7,11	**545**	7,23	**585**	7,2
426	7,15	**466**	7,03	**506**	7,08	**546**	7,2	**586**	7,17
427	7,15	**467**	7,05	**507**	7,13	**547**	7,18	**587**	7,23
428	7,24	**468**	7,03	**508**	7,11	**548**	7,15	**588**	7,23
429	7,22	**469**	7,05	**509**	7,15	**549**	7,16	**589**	7,24
430	7,04	**470**	7,11	**510**	7,1	**550**	7,18	**590**	7,24
431	7,04	**471**	7,13	**511**	7,02	**551**	7,09	**591**	7,03
432	7,04	**472**	7,11	**512**	7,04	**552**	7,03	**592**	7,05
433	7,03	**473**	7,06	**513**	7,05	**553**	7,05	**593**	7,03
434	7,02	**474**	7,08	**514**	7,04	**554**	7,04	**594**	7,04
435	7,02	**475**	7,09	**515**	7,13	**555**	7,05	**595**	7,05
436	7,13	**476**	7,03	**516**	6,99	**556**	7,14	**596**	7,05
437	7,04	**477**	7,05	**517**	6,98	**557**	7,05	**597**	7,02
438	7,05	**478**	7,04	**518**	7,04	**558**	7,11	**598**	6,99
439	7,04	**479**	7,05	**519**	7,06	**559**	7,04	**599**	7,04
440	7,05	**480**	7,14	**520**	7,07	**560**	7,05	**600**	7,06

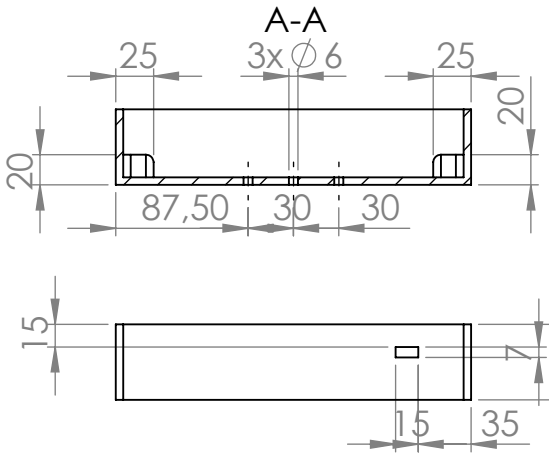
Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]	Numero pz	Misura [mm]
601	7,05	**641**	7,17	**681**	7,08	**721**	7,07	**761**	7,13
602	7,16	**642**	7,22	**682**	7,05	**722**	7,12	**762**	7,14
603	7,09	**643**	7,23	**683**	7,05	**723**	7,15	**763**	7,15
604	7,14	**644**	7,24	**684**	7,03	**724**	7,13	**764**	7,14
605	7,05	**645**	7,19	**685**	7,04	**725**	7,17	**765**	7,16
606	7,05	**646**	7,18	**686**	7,07	**726**	7,13	**766**	7,15
607	7,06	**647**	7,21	**687**	7,05	**727**	7,04	**767**	7,15
608	7,15	**648**	7,24	**688**	7,07	**728**	7,03	**768**	7,17
609	7,15	**649**	7,24	**689**	7,04	**729**	7,08	**769**	7,16
610	7,15	**650**	7,17	**690**	7,03	**730**	7,1	**770**	7,16
611	6,98	**651**	7,07	**691**	7,05	**731**	7,16	**771**	6,94
612	6,96	**652**	7,05	**692**	7,06	**732**	7,14	**772**	6,98
613	7,01	**653**	7,16	**693**	7,07	**733**	7,12	**773**	7,05
614	6,99	**654**	7,05	**694**	7,03	**734**	7,12	**774**	7
615	7	**655**	7,06	**695**	7,01	**735**	7,14	**775**	7
616	7,04	**656**	7,11	**696**	7,07	**736**	7,13	**776**	7
617	7,01	**657**	7,11	**697**	7,09	**737**	7,1	**777**	7,01
618	7,02	**658**	7,08	**698**	7,1	**738**	7,06	**778**	7,01
619	7,02	**659**	7,07	**699**	7,03	**739**	7,09	**779**	7,05
620	6,93	**660**	7,15	**700**	7,03	**740**	7,13	**780**	7,04
621	7,04	**661**	6,97	**701**	7,16	**741**	6,98	**781**	6,98
622	7,05	**662**	6,94	**702**	7,07	**742**	6,87	**782**	6,89
623	7	**663**	6,97	**703**	7,1	**743**	6,88	**783**	6,96
624	6,97	**664**	7,01	**704**	7,12	**744**	6,91	**784**	6,9
625	7,03	**665**	6,96	**705**	7,06	**745**	6,98	**785**	6,94
626	7,04	**666**	6,97	**706**	7,05	**746**	6,99	**786**	6,96
627	7,06	**667**	6,98	**707**	7,14	**747**	6,99	**787**	6,95
628	7,06	**668**	6,99	**708**	7,1	**748**	6,99	**788**	6,97
629	7,04	**669**	7,02	**709**	7,15	**749**	6,94	**789**	7
630	7,04	**670**	7,02	**710**	7,14	**750**	6,93	**790**	6,89
631	7,1	**671**	7,11	**711**	7,13	**751**	7,04	**791**	6,99
632	7,09	**672**	7,11	**712**	7,15	**752**	7,04	**792**	6,89
633	7,04	**673**	7,07	**713**	7,14	**753**	7,02	**793**	6,94
634	7,13	**674**	7,15	**714**	7,15	**754**	7,03	**794**	6,94
635	7,05	**675**	7,15	**715**	7,16	**755**	7,02	**795**	7,02
636	7,11	**676**	7,16	**716**	7,15	**756**	7,02	**796**	6,9
637	7,06	**677**	7,06	**717**	7,06	**757**	7,03	**797**	6,88
638	7,06	**678**	7,08	**718**	7,1	**758**	7,03	**798**	6,93
639	7,05	**679**	7,09	**719**	7,09	**759**	7,03	**799**	6,96
640	7,04	**680**	7,1	**720**	7,1	**760**	7,02	**800**	6,9

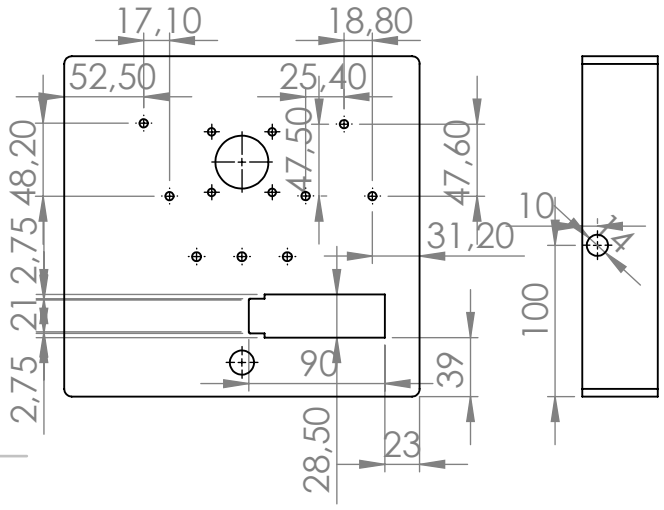
8. Ringraziamenti

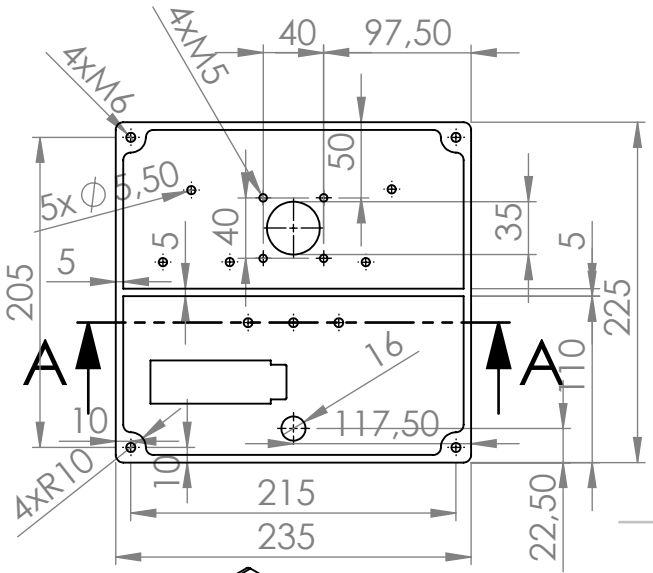
Un ringraziamento particolare va al mio tutor aziendale che mi ha affidato il lavoro di tesi riponendo in me fiducia in questi mesi. Un ringraziamento speciale va alla disponibilità e pazienza del fresatore Arian che ha permesso la realizzazione di alcune parti della stazione per le misurazioni.

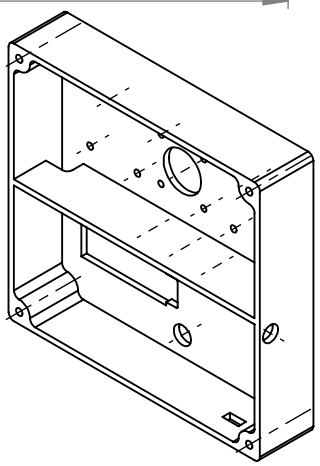
Il ringraziamento più grande va infine alla mia famiglia, ai miei nonni ed a mio zio che mi hanno sempre supportato negli anni per il raggiungimento di questo piccolo grande obiettivo.

9. Allegati









Raccordi non quotati R5

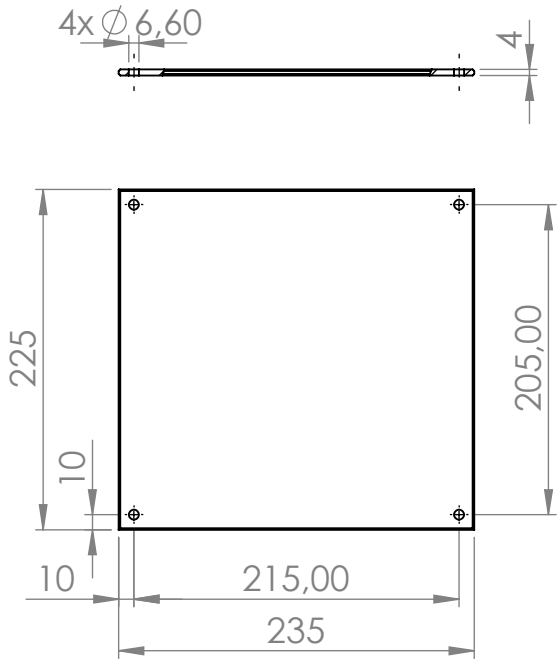
POLITECNICO DI TORINO	FINITURA: Ra 3.2	Toll. generali ISO 22081	Argomento: Stazione di controllo diametri
			Allievo: Davide Battezzati — Matr. s262798

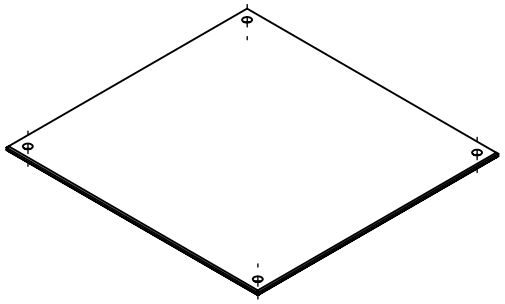
	NOME	FIRMA	DATA
DISEG.			
VERIF.			
APPR.			
FABB.			
Qual.			

Descrizione: **Coperchio**

MATERIALE: PLA	N. DISEGNO 1	A4
PESO:	SCALA:1:5	FOGLIO 1 DI 1

SOLIDWORKS Educational Product. Solo per uso didattico.





Smussi non quotati 1x45°

POLITECNICO DI TORINO	FINITURA: Ra 3.2	Toll. generali ISO 22081	Argomento: Stazione di controllo diametri
			Allievo: Davide Battezzati — Matr. s262798

	NOME	FIRMA	DATA
DISEG.			
VERIF.			
APPR.			
FABB.			
Qual.			

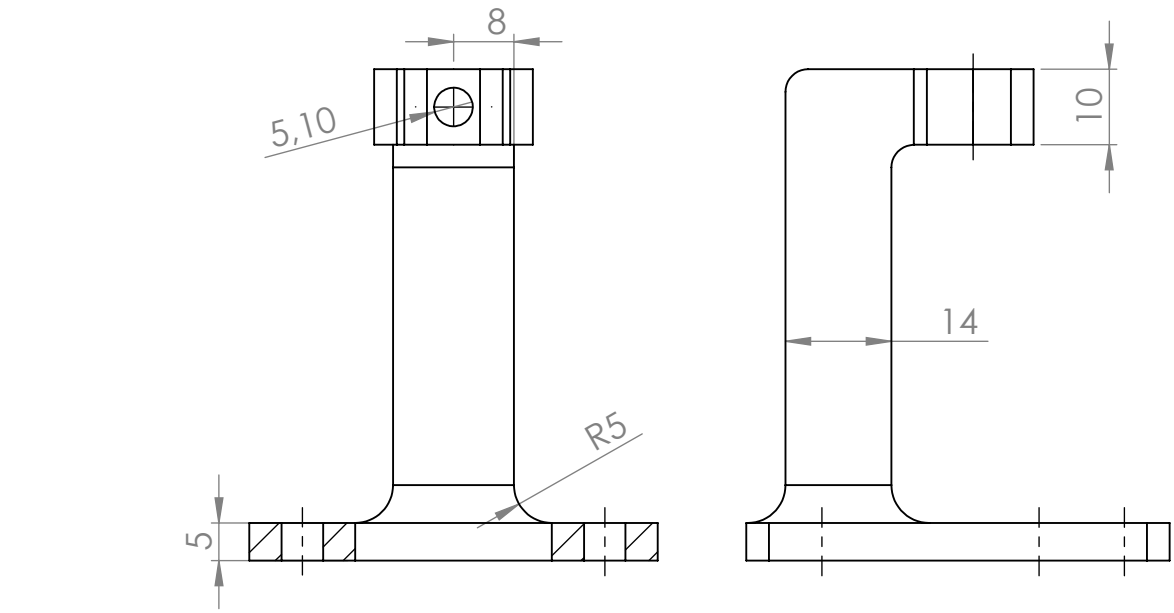
Descrizione: **Base**

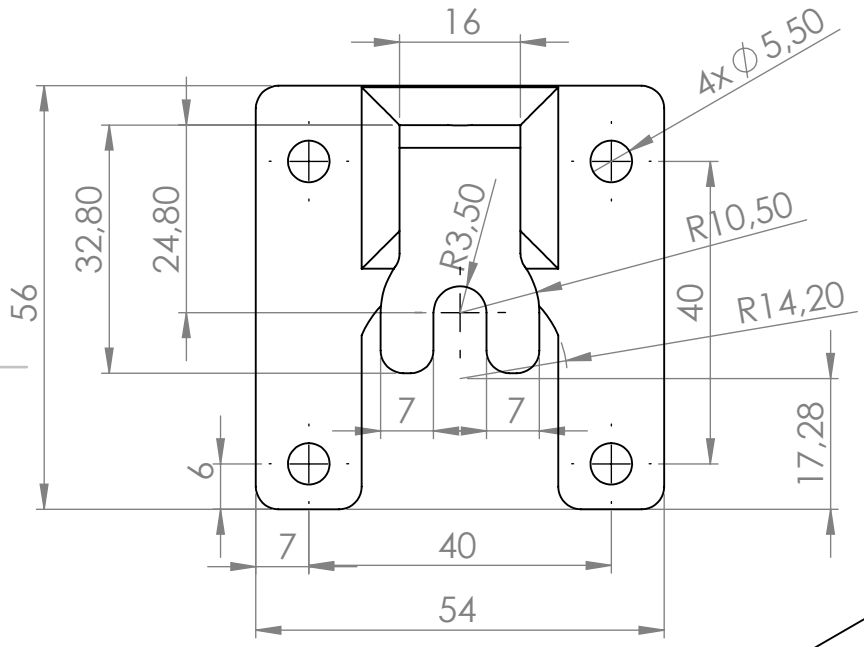
MATERIALE: PMMA

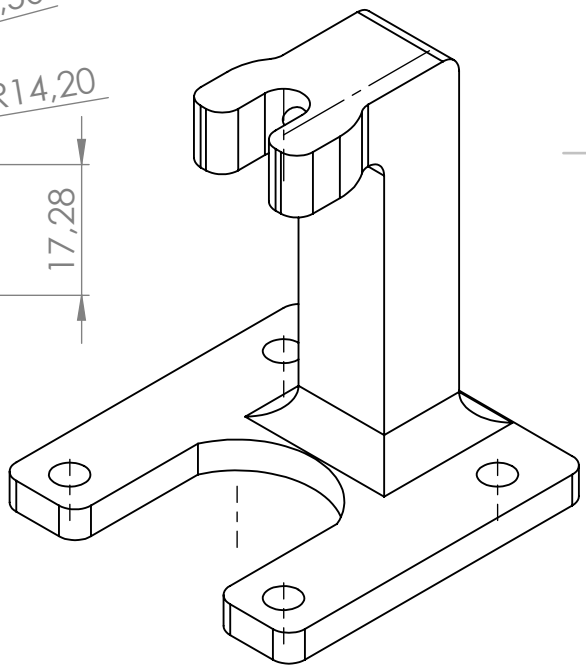
N. DISEGNO 2 — A4

PESO: — SCALA:1:5 — FOGLIO 1 DI 1

SOLIDWORKS Educational Product. Solo per uso didattico.







Raccordi non quotati R3

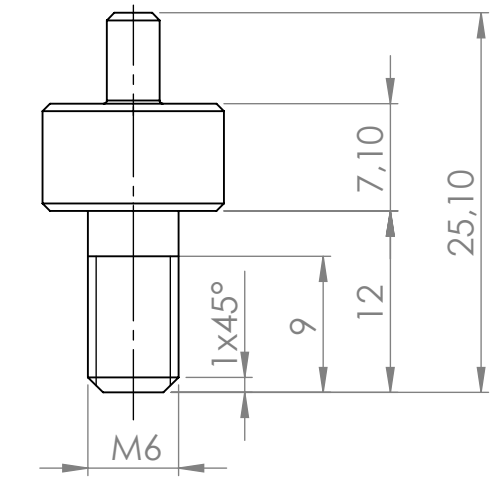
POLITECNICO DI TORINO	FINITURA: Ra 3.2	Toll. generali ISO 22081	Argomento: Stazione di controllo diametri
			Allievo: Davide Battezzati — Matr. s262798

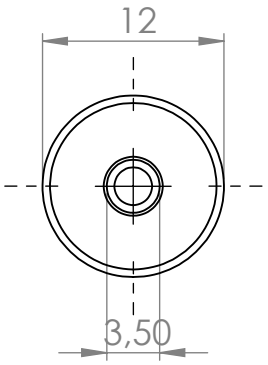
	NOME	FIRMA	DATA
DISEG.			
VERIF.			
APPR.			
FABB.			
Qual.			

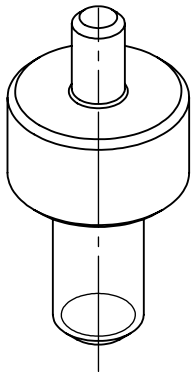
Descrizione: **Supporto tubo**

MATERIALE: PLA	N. DISEGNO 3	A4
PESO:	SCALA:1:1	FOGLIO 1 DI 1

SOLIDWORKS Educational Product. Solo per uso didattico.







Smussi non quotati 0,5x45°

Raccordi non quotati R0,5

POLITECNICO DI TORINO	FINITURA: Ra 3.2	Toll. generali ISO 22081	Argomento: Stazione di controllo diametri		
			Allievo: Davide Battezzati	Matr. s262798	

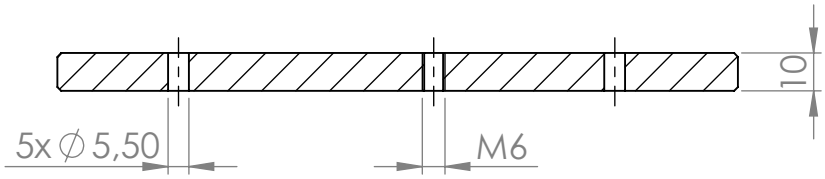
	NOME	FIRMA	DATA
DISEG.			
VERIF.			
APPR.			
FABB.			
Qual.			

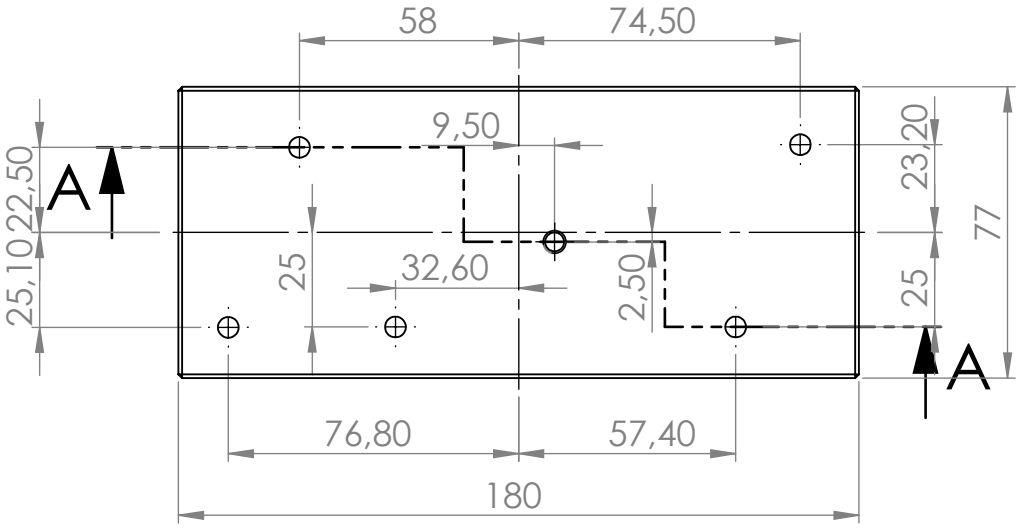
Descrizione: **Perno**

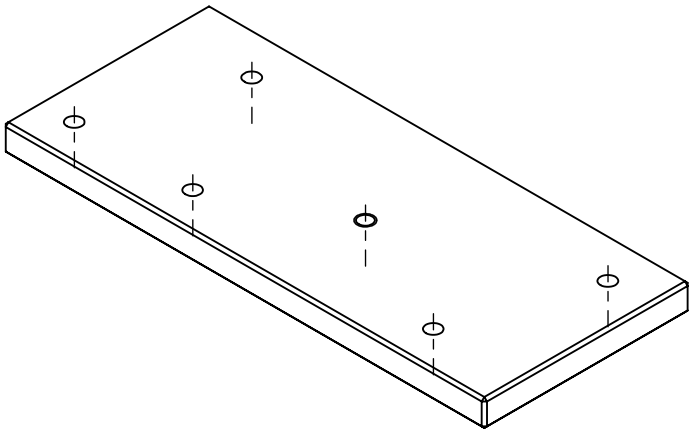
MATERIALE:	N. DISEGNO	
Acciaio	4	A4
PESO:	SCALA:2:1	FOGLIO 1 DI 1

SOLIDWORKS Educational Product. Solo per uso didattico.

A-A







Smussi non quotati 1x45°

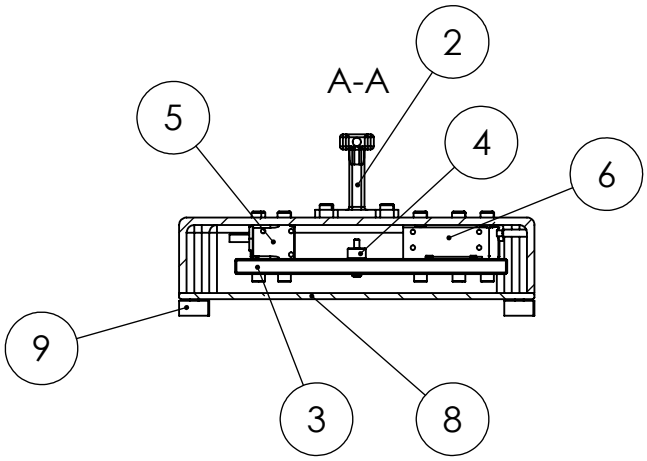
POLITECNICO DI TORINO	FINITURA: Ra 3.2	Toll. generali ISO 22081	Argomento: Stazione di controllo diametri
			Allievo: Davide Battezzati — Matr. s262798

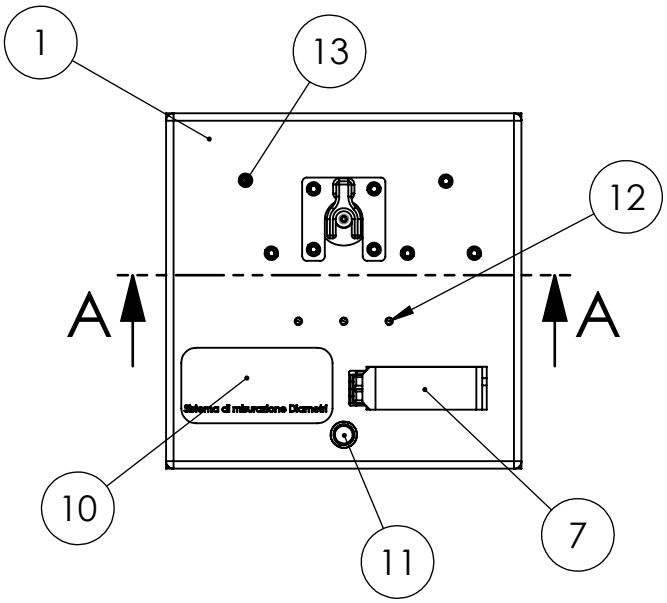
	NOME	FIRMA	DATA
DISEG.			
VERIF.			
APPR.			
FABB.			
Qual.			

Descrizione: **Piastra di centraggio**

MATERIALE: Alluminio	N. DISEGNO 5	A4
PESO:	SCALA:1:2	FOGLIO 1 DI 1

SOLIDWORKS Educational Product. Solo per uso didattico.





Num.	Descrizione	Qtà
1	Coperchio	1
2	Supporto tubo	1
3	Piastra di centraggio	1
4	Perno	1
5	Ricevitore	1
6	Trasmettitore	1
7	Condizionatore	1
8	Base	1
9	Piedino	4
10	Targhetta	1
11	Pulsante	1
12	LED	3
13	Vite testa cilindrica	19

POLITECNICO DI TORINO

FINITURA: Ra 3.2

Toll. generali ISO 22081

Argomento: Stazione di controllo diametri

Allievo: Davide Battezzati | Matr. s262798

	NOME	FIRMA	DATA
DISEG.			
VERIF.			
APPR.			
FABB.			
Qual.			

Descrizione: Assieme stazione di controllo

MATERIALE:

N. DISEGNO 6

A4

PESO:

SCALA:1:5

FOLGLIO 1 DI 1

SOLIDWORKS Educational Product. Solo per uso didattico.