Immagine che contiene Carattere, logo, testo, simbolo

Descrizione generata automaticamente



Relazione progetto “Controllo qualità pezzi”



*Fioritto Alessandro - matr. 152205*

*Zampieri Giacomo - matr. 153321*

# Obiettivi di progetto e componentistica

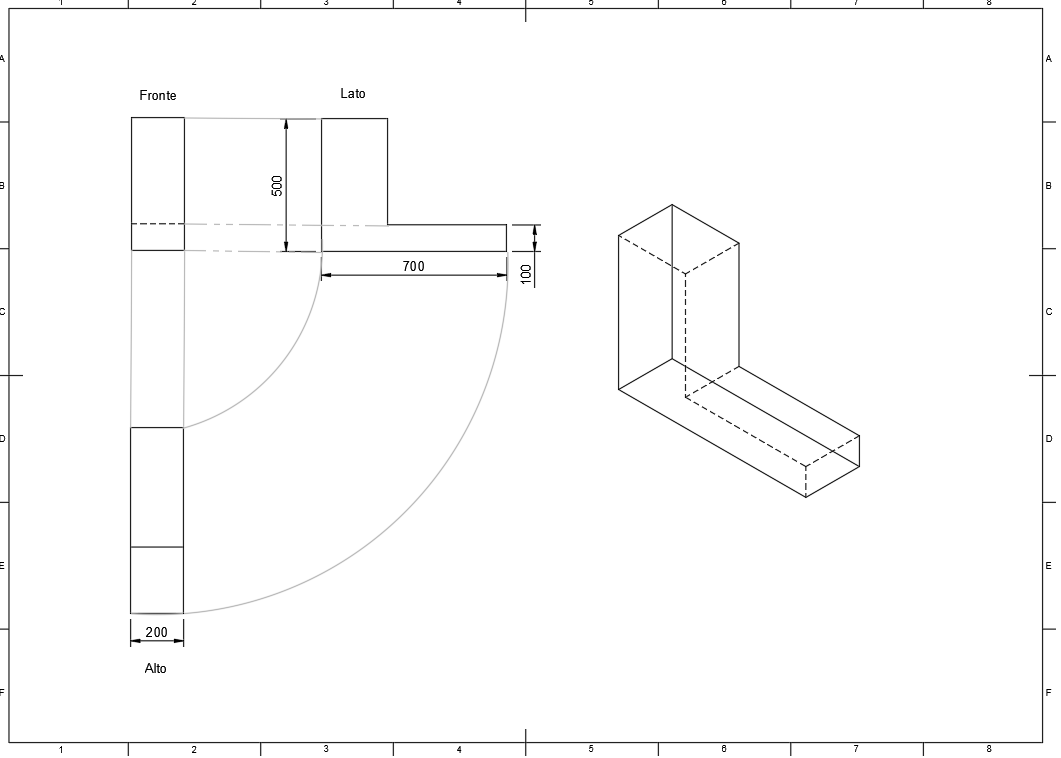
L’obiettivo del progetto è ideare, studiare e realizzare un circuito con la scheda Arduino che permetta di identificare dei componenti, verificando che siano conformi a delle dimensioni preposte, per poi smistarli, tramite dei nastri trasportatori (analogamente ad un sistema di controllo qualità industriale, seppur più semplice).

I componenti utilizzati nel progetto completo sono:

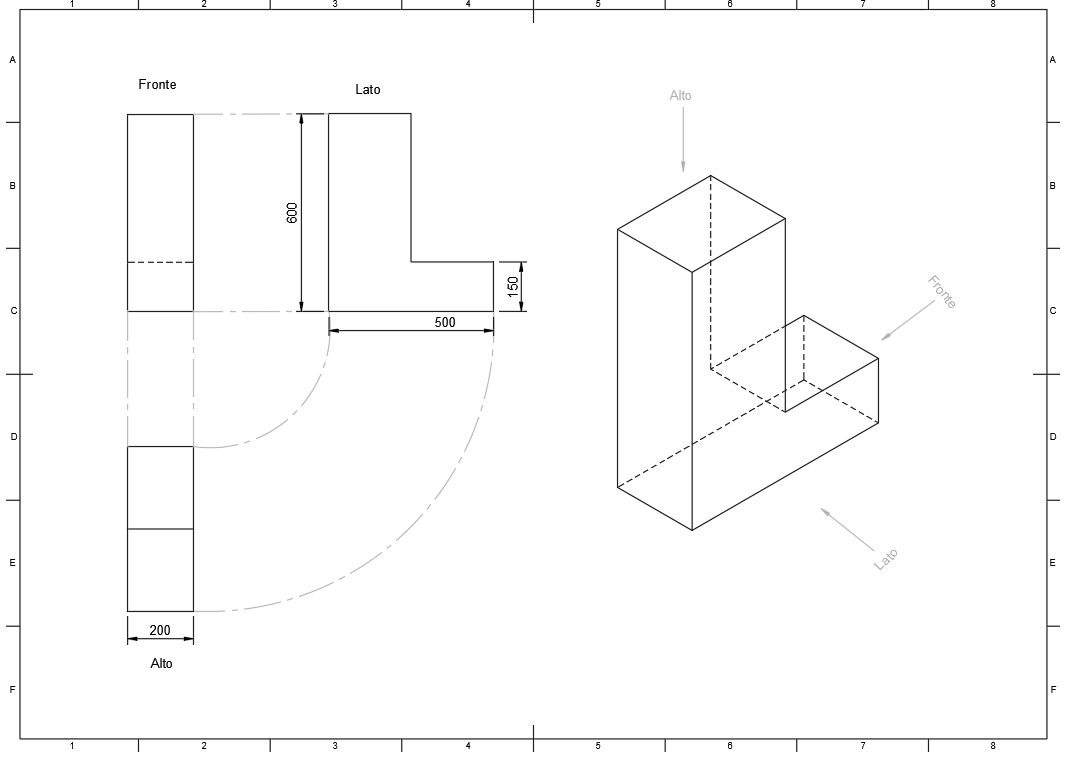
* Una scheda Arduino MEGA
* Quattro Breadboard
* Un numero di cavi di collegamento
* Un pulsante
* Un Buzzer
* Due diodi LED
* Sette coppie di emettitori/ricevitori infrarossi
* Due motorini che muovono altrettanti nastri trasportatori
* Componente L293D (ponte h) per la gestione dei due motori
* Alimentatore esterno da 5V
* Alimentatore esterno da 9V
* Un numero di resistenze (principalmente da 1000 Ohm)
* Una ventolina per il raffreddamento

Vengono inoltre utilizzate tre forme, denominate “pezzi” (Tipologia A, B o C) che rappresenteranno gli elementi rilevati dai sensori ad infrarossi.

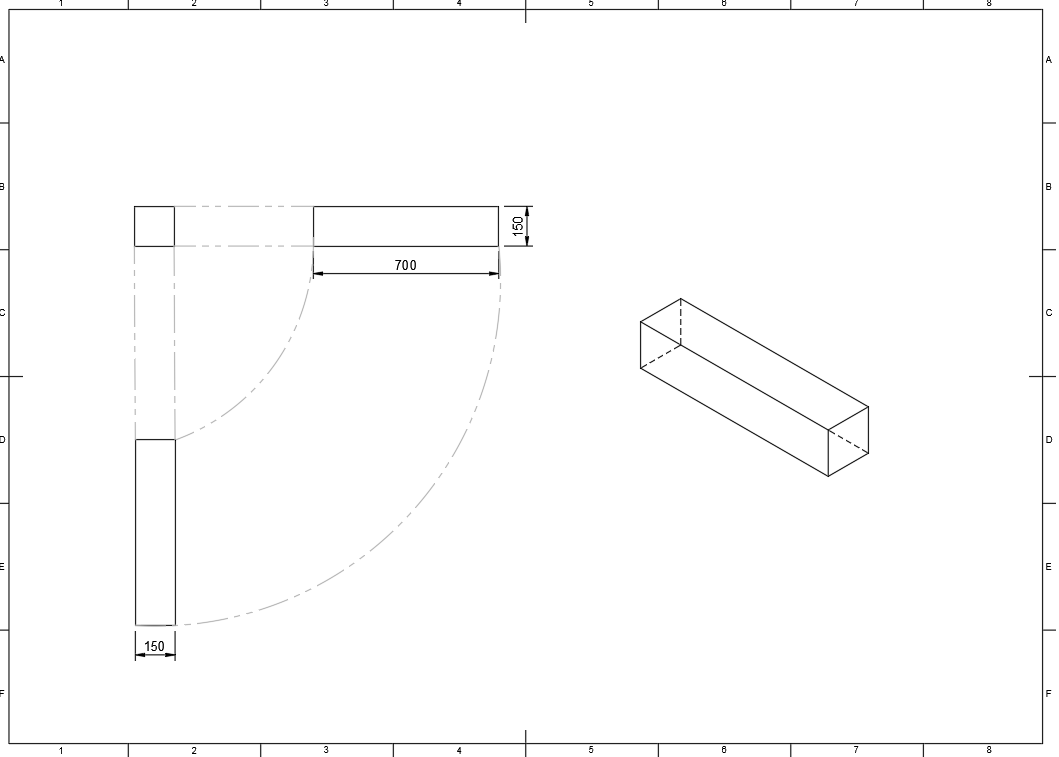
Il pezzo A avrà le misure “corrette” mentre il pezzo B e C avranno diversi gradi di imperfezione: il pezzo B avrà la base troppo corta ed il pezzo C non avrà la parte superiore, risultando “troppo bassa”.



Rappresentazione grafica del pezzo A (dimensioni in mm), ovvero il pezzo corretto



Rappresentazione grafica del pezzo B (dimensioni in mm), “Figura troppo corta”



Rappresentazione grafica del pezzo C (dimensioni in mm), “Figura troppo bassa”

# Studio e analisi degli obiettivi di progetto

Verrà scelto un pezzo che andrà posizionato sul primo nastro trasportatore e verrà trasportato attraverso i sensori posizionati ad U lungo i lati del percorso: quando i sensori posti più a destra rileveranno una figura, il nastro trasportatore si fermerà per pochi secondi e tutti i sensori eseguiranno una lettura e trasmetteranno i dati ad Arduino, che li renderà visibili su una pagina web Javascript.

Dopo aver rilevato le dimensioni il nastro ripartirà, facendo poi cadere il pezzo sul secondo nastro trasportatore, posizionato parallelamente sotto al primo: questo si muoverà in avanti se il pezzo rispetta le misure (tipologia A) o all’ indietro se viene rilevata un’imperfezione (tipologia B o C), simulando uno scarto.

In contemporanea viene aggiornato in tempo reale il conteggio dei pezzi, tramite un semplice contatore di pezzi rilevati per ogni tipologia con un grafico a torta associato ed un valore percentuale dei pezzi corretti rispetto al totale delle rilevazioni.

# Analisi dei requisiti

Il progetto creato dovrà permettere di discernere e smistare dei componenti di diversa dimensione, utilizzando un insieme di sensori e nastri trasportatori.

Il primo nastro trasportatore (superiore) dovrà iniziare il movimento verrà premuto il pulsante e dovrà arrestarsi brevemente quando le cinque coppie di sensori superiori rileveranno un pezzo in transito e di seguito ripartire.

Le coppie di sensori superiori dovranno rimanere in attesa del transito di un pezzo e dovranno rilevare correttamente la forma assegnandola ad una delle tre tipologie possibili, rilasciando l’output tramite seriale.

Il secondo nastro trasportatore (inferiore) dovrà iniziare il movimento appena la rilevazione dei sensori superiori sarà conclusa e dovrà arrestarsi quando una delle due coppie, posizionate all’inizio e alla fine del nastro, non avrà rilevato un oggetto.

Le due coppie di sensori inferiori dovranno comunicare l’arresto del nastro inferiore quando rileveranno un pezzo in transito.

L’output seriale di Arduino dovrà essere reso in formato leggibile da un file JavaScript.

Dovrà essere creata una pagina web sviluppata in PHP e Javascript che dovrà presentare i dati raccolti.

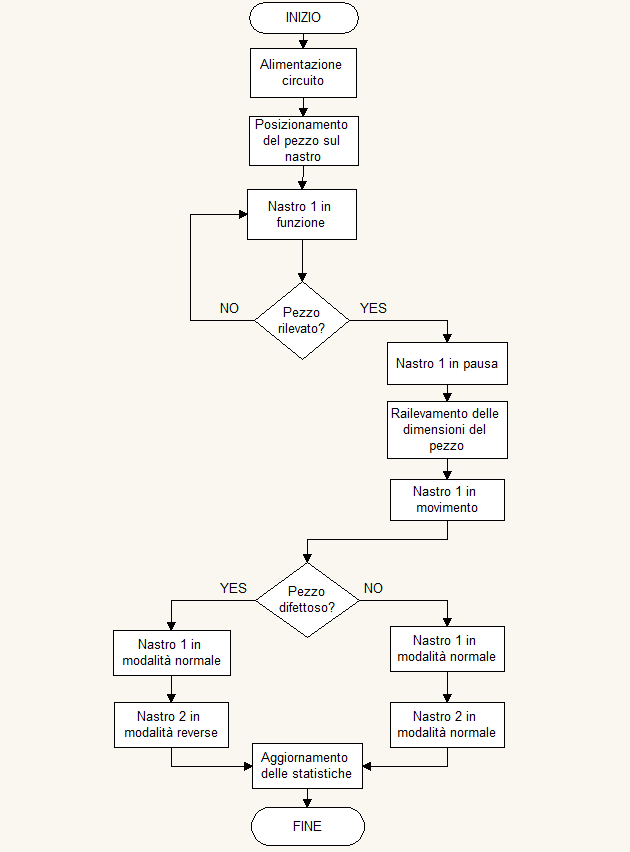
# Caratteristiche tecniche

La scelta dell’ambito di progetto è derivata dalla precedente conoscenza dei componenti, principalmente dei motori, utilizzati anche in altri progetti e dall’obiettivo prefissato di creare un circuito simulativo di un ambiente industriale reale.

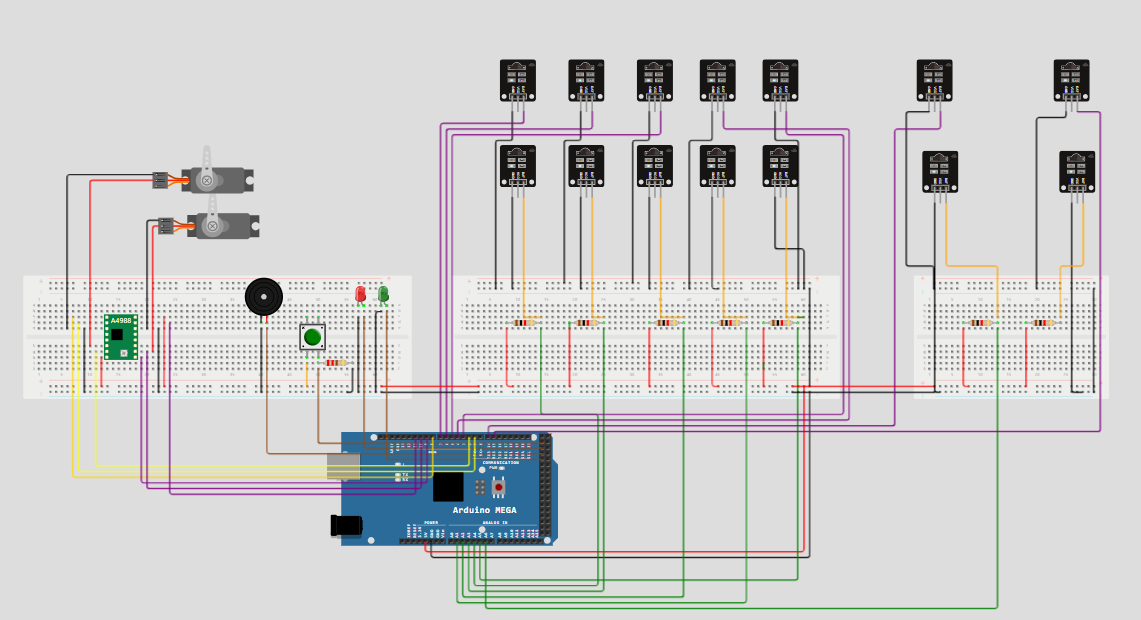
Abbiamo inoltre scelto di creare un sistema di presentazione dati, invece che utilizzare strumenti già pronti come Grafana, per sfruttare le competenze già in nostro possesso (rudimenti di javascript, php ed SQL) piuttosto che imparare un software nuovo e ci sembrava più interessante tentare di risolvere questo problema in autonomia e sviluppare qualcosa da zero.

Inoltre, per il controllo dei due motori distinti dei due nastri è necessario utilizzare il componente L293D, chiamato anche ponte H, che permette l’andamento in modalità normale e reverse.

# Diagramma funzionale a blocchi



# Schema elettrico



Schema elettrico del circuito finale: sulla sinistra il buzzer, il pulsante di arresto, il ponte H ed i motori dei nastri trasportatori. A destra i due gruppi di sensori infrarossi.

# Implementazione

## Fasi di collaudo

Inizialmente abbiamo costruito una rudimentale struttura di compensato per sostenere i due nastri e permettere la realizzazione

I nastri trasportatori che abbiamo acquistato non nascono naturalmente compatibili con Arduino ed è stato quindi necessario modificarli, sostituendo i motorini con componenti nativi di Arduino e quindi controllabili dalla scheda.

Dopo aver assemblato il circuito ed aver alimentato la scheda Arduino sono stati attivati i due nastri.

È stato posizionato il primo pezzo (tipologia A, corretto) sul nastro che si è arrestato una volta che le prime coppie di sensori hanno rilevato il componente: avendolo rilevato come tipologia A il nastro è ripartito, facendolo cadere sul nastro sottostante per lo smistamento.

Nel frattempo, il nastro per lo smistamento ha correttamente attivato la modalità normale, muovendosi in avanti per un determinato tempo, ed ha fatto cadere il pezzo segnalando la fine del procedimento.

La prova è stata ripetuta con entrambi i pezzi difettosi: entrambe le tipologie di pezzi (B e C) sono stati riconosciuti come errate e smistate dal nastro sottostante in modalità reverse.

Sono sorti però dei problemi non trascurabili durante la prima fase di collaudo, dove il primo problema era la mancanza di sufficiente amperaggio per l’alimentazione corretta dei motori in test prolungati: la batteria da 9V fornita dal kit di Arduino risultava insufficiente vista la velocità con la quale si scaricava. Sono stati quindi introdotti due alimentatori (rispettivamente da 5V e 9V) direttamente collegati al ponte H.

Un ulteriore difficoltà è sorta dopo la modifica precedente visto che il ponte H si surriscaldava eccessivamente e portava a non far muovere il nastro correttamente: il problema era forse derivato dalla fonte di alimentazione troppo potente ed inserendo delle resistenze per diminuire l’apporto di corrente si ricadeva nel problema dei nastri troppo lenti o con movimenti erratici; è stata quindi posizionata una semplice ventola per il raffreddamento sopra al controllore.

Inoltre, per inserire altre funzionalità nel progetto, la scheda Arduino UNO era insufficiente ad ospitare tutti i componenti necessari, ed è stata sostituita con una scheda Arduino MEGA per avere un maggior numero di pin a disposizione.

Abbiamo anche inserito una funzione di “panic button” sul bottone di accensione principale che permette l’arresto completo del sistema durante il funzionamento, qualora fosse necessario.

Dopo aver risolto i problemi abbiamo effettuato un nuovo collaudo, nel quale è stato posizionato il pezzo tipologia A sul nastro come precedentemente, che è stato riconosciuto ed è stato correttamente smistato dal nastro sottostante: la differenza è che il pezzo è stato rilevato anche tramite un ulteriore sensore, posizionato alla fine del nastro di smistamento con la funzione di disattivare i nastri all’occorrenza (per limitare il tempo di funzionamento, diminuire il surriscaldamento dei componenti e “migliorare” il throughput dei pezzi ).

# Rappresentazione dati

Come già accennato in precedenza la parte di rappresentazione dati abbiamo optato per la trasformazione dell’output di Arduino da testo sulla porta seriale ad una pagina web creata in JavaScript e PHP.

Per far comunicare la scheda Arduino con la pagina web abbiamo utilizzato il framework Node-RED, che nel nostro caso prende i dati in formattazione JSON che arrivano dalla porta seriale collegata all’Arduino e li carica su un database realizzato con phpMyAdmin.

Successivamente la pagina web prenderà i dati da questo database per popolare delle tabelle e realizzare un grafico.

Immagine che contiene testo, Carattere, bianco, ricevuta

Descrizione generata automaticamente

Sopra possiamo vedere un esempio di output di Arduino: partendo da 0, ogni volta che viene rilevata una figura il contatore viene aggiornato (4 rilevazioni totali).

Immagine che contiene testo, schermata, diagramma, linea

Descrizione generata automaticamente

* Il primo nodo è quello relativo alla porta seriale COM5, che è quella collegata ad Arduino e sui cui vengono ricevuti i dati provenienti dalla scheda.
* Il nodo successivo trasforma la stringa in arrivo dalla seriale, in formato JSON, in un oggetto Javascript che verrà utilizzato nel nodo successivo.
* Il terzo nodo contiene una funzione scritta in Javascript che estrae i valori dei tre campi della stringa, ovvero il numero di figure OK, il numero di figure basse e il numero di figure corte e restituisce il nuovo messaggio contenente i tre dati.
* Il quarto nodo permette di accedere al database MySQL denominato “iot”, per lo storage dei dati*.*

Schema del flusso dei nodi di Node-RED.

*Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente*

Funzione contenuta nel nodo function 1.

Il seguente codice descrive la parte di caricamento dei dati presenti nel database, all’interno della tabella della pagina web.

<!-- Creazione della tabella in cui vengono mostrati gli attuali valori del numero di figure rilevate -->

<table id="FigTab">

<thead>

  <tr>

            <th>NumFigureOk</th>

                  <th>NumFigureCorte</th>

                  <th>NumFigureBasse</th>

            </tr>

</thead>

      <tbody>

      <?php

            //Connessione al database MySQL

                  $conn = mysqli\_connect("localhost","root","","iot");

                  //Gestione dell'errore di connessione

                if($conn -> connect\_error){

                  die('connection failed:'. $conn -> connect\_error);

                  }

                  //Prendo l'ultima riga del database

                  $sql = "SELECT NumFigureOk,NumFigureCorte,NumFigureBasse from data ORDER BY NumFigureOk DESC LIMIT 1";

                  $result = $conn -> query($sql);

                  if($result -> num\_rows > 0){

                //Inserisco i dati presi dal database nella tabella HTML

                      while($row = $result -> fetch\_assoc()){

                      echo "<tr><td id='OK'>".$row["NumFigureOk"]."</td><td id='Corte'>".$row["NumFigureCorte"]."</td><td id='Basse'>".$row["NumFigureBasse"]."</td></tr>";

            }

                  echo "</table>";

                    }

      ?>

</tbody>

</table>

Il database verrà poi interpellato tramite queste funzioni, che andranno a creare i contatori delle varie tipologie di pezzo

 <!--PARTE DI OUTPUT GRAFICO -->

        <script>

            ok =  document.getElementById('OK').innerHTML;

            corte =  document.getElementById('Corte').innerHTML;

            basse =  document.getElementById('Basse').innerHTML;

            tot = + ok + + corte + + basse;

            percentualeCorrette = Math.round((ok / tot)\*100);

            if(percentualeCorrette > 0){

                document.getElementById('percentualeCorrette').innerHTML = "Percentuale di figure corrette: " + percentualeCorrette + "%."; }

            new Chart(document.getElementById('pie-chart'), {

                type: 'pie', data: {

                    labels: ["Figure Corrette", "Figure Corte", "Figure Basse"],

                    datasets: [{ data: [ok, corte, basse] }]

Queste funzioni invece sono finalizzate alla creazione del grafico a torta, rappresentativo dei dati visualizzati in precedenza.

Il risultato finale è il seguente:

Immagine che contiene testo, schermata, diagramma, Carattere

Descrizione generata automaticamente

Pagina web per la visualizzazione dei dati.

Oltre ad aver realizzato manualmente la pagina web per la visualizzazione dati, abbiamo deciso di creare anche una dashboard con *Grafana* come abbiamo visto a lezione. La dashboard creata contiene un grafico a torta come quello realizzato per la pagina web:

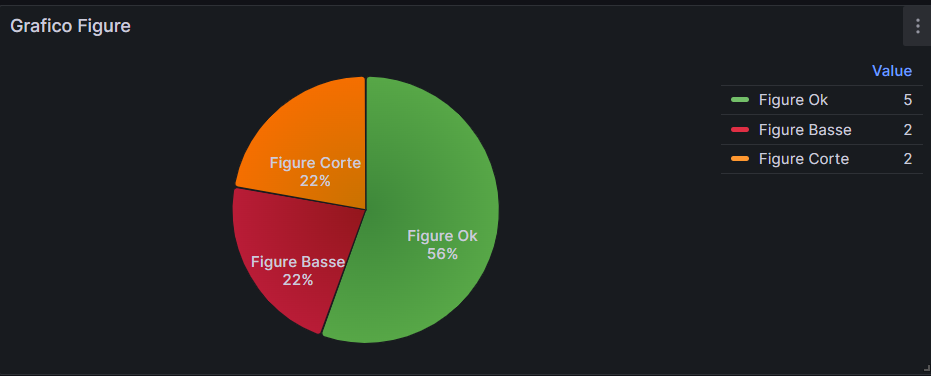


Grafico a torta realizzato con Grafana.

# Considerazioni conclusive

## Risultati raggiunti e miglioramenti

L’idea di progetto iniziale è riuscita ed il prototipo ha le funzioni preposte in fase di progettazione; sono state inoltre aggiunte funzionalità non previste inizialmente, come il panic button e le coppie inferiori di sensori, che hanno portato però al miglioramento del progetto e all’avvicinamento all’obiettivo di simulazione di un ambiente industriale reale.

Nonostante il progetto nel suo complesso sia risultato coerente con le aspettative iniziali ci sono comunque aspetti critici migliorabili:

(P rappresenta il PROBLEMA ed S la SOLUZIONE proposta)

**P**: Il codice è tarato per il riconoscimento di errori predeterminati.

**S**: Utilizzare più sensori (posizionati in modo da verificare le tre dimensioni dell’oggetto simultaneamente) o altri tipi di sensori più adatti, come i sensori ad ultrasuoni.

**P**: Lo smistamento dei pezzi rallenta il funzionamento generale del sistema.

**S**: Aggiungendo un piccolo pistone (o un elemento umano) che rimuova il pezzo difettoso dal nastro principale si potrebbe evitare la fase di smistamento dei pezzi difettosi velocizzando il funzionamento generale del sistema.

**P**: Il ponte H si surriscalda troppo velocemente, portando il sistema ad avere un tempo di funzionamento molto basso, anche implementando la ventola.

**S**: L’aggiunta di un dissipatore sopra al componente potrebbe migliorarne il funzionamento e rendere superflua la ventola; in alternativa si potrebbe sostituire il componente con uno più adatto, meno prono a surriscaldarsi.

Un ulteriore miglioramento, non legato necessariamente a problematiche incontrate, potrebbe essere l’aggiunta di un metodo per simulare un allarme, inserendo un timer di alcuni secondi allo scadere del quale, se non dovesse essere stato rilevato nulla dai sensori inferiori, accenda un led rosso, a significare che il pezzo è incastrato e che andrebbe premuto l’arresto di emergenza per evitare danni alla catena.

Un ulteriore proposta di miglioramento sarebbe l’ampliamento della pagina web di output dei dati, aggiungendo un contatore che rappresenti la serie di pezzi corretti rilevati attuale e un giudizio in base alla percentuale di pezzi corretti globali.

Infine, dopo aver completato il progetto, è sorto un dubbio sull’utilizzo di memoria del database: visto che ad ogni rilevazione viene mandata in output la riga di codice completa di tutti i dati passati e che questa viene direttamente aggiunta al database a lungo andare si riempirebbe di dati sostanzialmente inutili.

La soluzione sarebbe creare delle query specifiche per l’aggiornamento dei soli dati interessati dalla rilevazione o l’eliminazione delle ridondanze nel database dopo un determinato lasso di tempo.

# Tabella per la suddivisione dei compiti

|  |  |  |
| --- | --- | --- |
|  | Fioritto Alessandro | Zampieri Giacomo |
| Stesura relazione | X | X |
| Codice gestione nastri | X |  |
| Codice Rilevamento pezzi |  | X |
| Codice analisi dati |  | X |
| Schemi logici ed elettrici | X |  |
| Adattamento dei nastri ad Arduino |  | X |
| Implementazione “panic button” | X |  |
| Realizzazione circuito | X | X |

Link al progetto presente su GitHub

<https://github.com/GiacomoZampieri/Progetto-IoT>