Relazione Progetto Making

Antonio Sisinni

Septembre 2025

1 Introduzione

L'obiettivo del progetto è quello di mettere in pratica le conoscenze acquisite durante le lezioni, in particolare riguardo a:

- schede programmabili
- sensori e interruttori
- circuiti elettrici
- stampa 3D

Tali competenze devono essere applicate per realizzare un prototipo funzionante, in grado di svolgere un compito da noi definito.

2 Idea

L'idea alla base del progetto consiste nello sviluppo di un dispositivo in grado di integrare diverse tecnologie affrontate a lezione, con l'aggiunta di elementi di interesse personale. A tale scopo ho deciso di realizzare un dispositivo indossabile per la realtà aumentata.

Il sistema prevede l'impiego di una telecamera per acquisire informazioni dall'ambiente circostante. I dati raccolti vengono elaborati e successivamente presentati all'utente tramite un display trasparente, così da non compromettere la visibilità diretta.

L'interazione con il dispositivo è resa possibile attraverso dei pulsanti che consentono all'utente di filtrare o modificare le informazioni visualizzate.

Data la necessità di includere più componenti hardware, il design di riferimento scelto è lo *scouter* della serie *Dragon Ball*. Esso presenta dimensioni adeguate per ospitare i vari elementi, garantendo al contempo una buona ergonomia.

3 Studio di fattibilità

3.1 Schermo trasparente

In un primo momento avevo pensato di utilizzare un normale schermo abbinato a un prisma ottico, così da ridirezionare l'immagine e permettere all'utente di guardare attraverso.

Sebbene questa soluzione fosse funzionale, l'ingombro complessivo e il peso derivante dall'accoppiata schermo + prisma mi hanno spinto a optare per l'utilizzo di un display trasparente Figura 1.

Questa scelta, tuttavia, ha introdotto una nuova difficoltà: la libreria `u8g2` (utilizzata per disegnare sul display), il mio specifico display e la scheda *ESP32-S3*, non risultavano funzionanti assieme.

Per superare questo problema ho adottato una configurazione ibrida: la *ESP32-S3* per la potenza di elaborazione, affiancata da un *Arduino Nano* dedicato esclusivamente alla gestione della visualizzazione sul display.

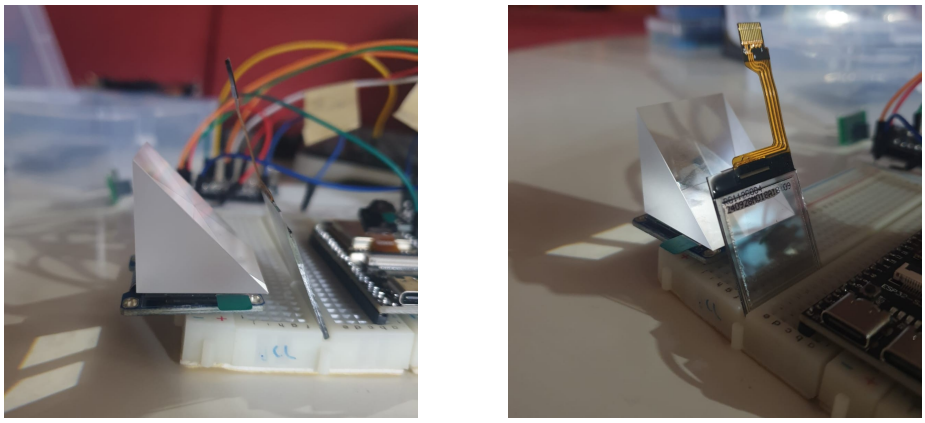


Figure 1: Confronto ingombri tra le due soluzioni per il display schermo

3.2 Elaborazione delle immagini

La mia idea iniziale era di sfruttare direttamente la *ESP32-S3* per l'elaborazione delle immagini, in quanto dotata di capacità di calcolo sufficienti a supportare modelli di *machine learning* leggeri (soluzioni come edgeimpulse o tensorflow lite nascono proprio per dispositivi del genere).

Tuttavia, la scheda da me acquistata presentava scarsa documentazione e non aveva compatibilità diretta con librerie apposite. Dopo alcuni tentativi ho quindi

deciso di utilizzare la *ESP32-S3* unicamente la cattura dei dati e la trasmissione di essi via *Wi-Fi*, demandando l'elaborazione delle immagini a un dispositivo esterno. In questo modo ho potuto ottenere le informazioni necessarie mantenendo il dispositivo portatile.

4 Struttura del Sistema

Le diverse parti del sistema collaborano per assolvere compiti distinti, così da garantire il corretto funzionamento complessivo e la fruibilità da parte dell'utente.

- **ESP32-s3** cuore del sistema, si occupa di acquisire le immagini e di gestire la comunicazione tra le varie componenti
- **Arduino Nano** gestisce il display *OLED* trasparente sul quale l'utente visualizza le informazioni elaborate. Inoltre, si occupa dell'acquisizione degli input dell'utente
- Un **PC** è usato come dispositivo esterno per eseguire l'elaborazione delle immagini tramite il modello di *computer vision* `YOLOv11`, che consente di individuare all'interno di immagini oggetti presenti, identificandone posizione e classe di appartenenza.

5 Pipeline di Comunicazione

Affinché il sistema possa operare correttamente, le diverse componenti devono comunicare tra loro. La struttura della comunicazione è illustrata in Figura 2.

La connessione tra l'*esp32-s3* e il *PC* avviene in modalità wireless, così da garantire la mobilità dell'utente. La comunicazione tra l'*esp32-s3* e l'*Arduino Nano*, invece, utilizza un semplice collegamento seriale, per garantire bassa latenza.

5.1 Messaggi di Stato

Oltre a trasmettere le informazioni relative alle immagini acquisite, l'*esp32-s3* può inviare una serie di messaggi sullo stato del sistema. Questi messaggi risultano particolarmente utili per informare l'utente sul corretto funzionamento del dispositivo, soprattutto in caso di malfunzionamenti nella trasmissione dei dati.

I messaggi di stato previsti sono:

- `connecting`: indica che il dispositivo sta tentando di stabilire la connessione con il *PC*
- `com_err`: segnala un problema di comunicazione tra l'*esp32-s3* e il *PC*

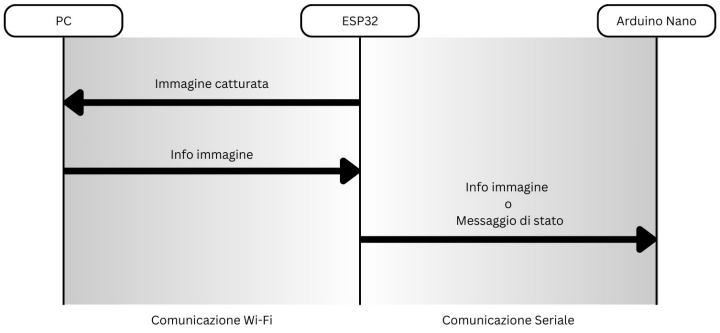


Figure 2: Schema della pipeline di comunicazione tra le componenti del sistema.

6 Hardware Utilizzato

6.1 Componenti

Per la realizzazione del prototipo ho utilizzato i seguenti componenti:

- ESP32-S3 WROOM N16R8 CAM
- Arduino Nano 3.0
- Display OLED Trasparente
- **Resistenza**
  Valore: $1\,M\Omega\ \pm 1\%$
- **Condensatori**
  Tre unità (la tipologia precisa verrà specificata successivamente).
- **Pulsanti**
  Tre unità, utilizzati come input utente.

6.2 Wiring

Lo schema di collegamento dei componenti è mostrato in Figura 3. Nel diagramma, i LED rappresentano simbolicamente i condensatori utilizzati nel circuito.
La *esp32-s3* è poi collegata ad *Arduino Nano* tramite connessione seriale.

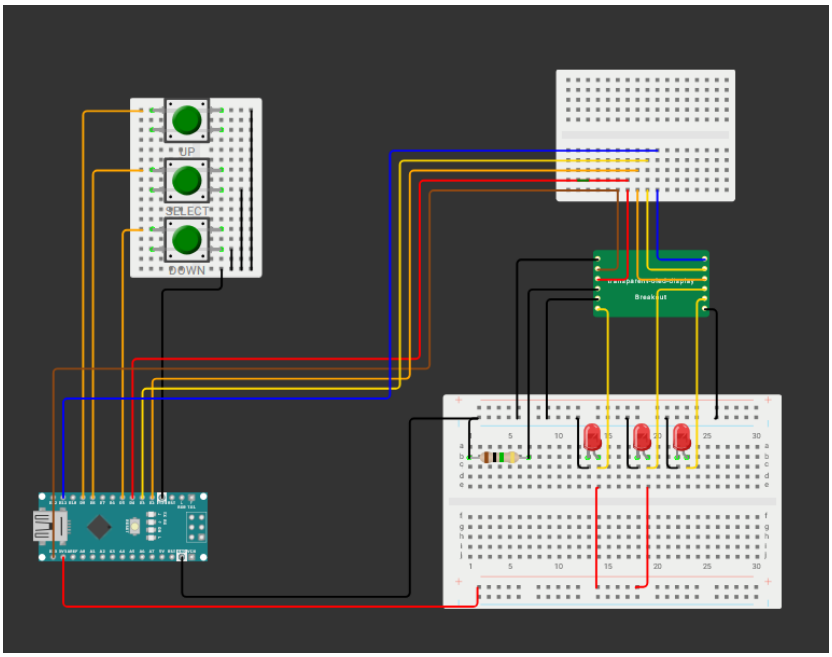


Figure 3: Schema di collegamento (*wiring*) dei componenti del sistema.

7 Elaborazione delle Immagini

Per ottenere informazioni sull'ambiente circostante utilizzo la fotocamera integrata nella *ESP32-S3*. Le immagini acquisite devono essere analizzate per estrarne i dati utili.

7.1 Cattura

Per la gestione della fotocamera faccio uso della libreria `esp_camera.h`, che mette a disposizione la funzione:

```
camera_fb_t * fb = esp_camera_fb_get();
```

Tale funzione permette di catturare un frame (`fb`) che viene poi inviato per l'analisi tramite una *HTTP request*, realizzata con la libreria `HTTPClient.h`.

Per garantire un buon compromesso tra frequenza di acquisizione e qualità delle catture, ho utilizzato i seguenti parametri per la camera:

- `frame_size = FRAMESIZE_VGA`
- `jpeg_quality = 15`

7.2 Analisi

L'analisi delle immagini è affidata al modello di computer vision `YOLOv11`, che individua all'interno del frame gli oggetti presenti, specificandone posizione e classe di appartenenza.

Le informazioni estratte vengono poi impacchettate e inviate nuovamente alla *ESP32-S3* come risposta. In questa fase la comunicazione è gestita da un server *Flask*, che riceve le immagini, le elabora e restituisce i risultati in formato strutturato.

8 Interazione Utente

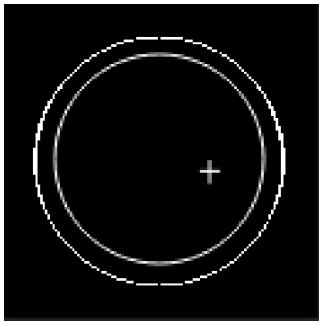
Il sistema è progettato per interagire in modo bidirezionale con l'utente.

8.1 Sistema → Utente

L'utente può visualizzare sul display trasparente le informazioni relative alle classi di oggetti selezionate. Per ciascuna classe mostro:

- il numero di occorrenze rilevate (Figure 4a);
- la posizione degli oggetti individuati all'interno dell'immagine (Figure 4b).

(a) Modalità **Conteggio** mostra il numero di occorrenze per ciascuna classe.



(b) Modalità **Localizzazione** mostra posizione degli oggetti individuati.

Per la creazione dell'interfaccia ho utilizzato la libreria `u8g2`, essa mette a disposizione primitive grafiche per il disegno su display e la possibilità di caricare e utilizzare bitmap per rappresentare elementi più complessi.

8.2 Utente → Sistema

L'utente interagisce con il dispositivo tramite i tre pulsanti integrati. In particolare, può:

- navigare tra le diverse classi disponibili (tasti su e giù)
- passare da modalità **conteggio** modalità **localizzazione** (tato select)

9 Difficoltà

Lo sviluppo di questo progetto ha presentato diverse difficoltà che mi hanno richiesto tempo e sperimentazione per essere affrontate.

9.1 Utilizzo della ESP32-S3 CAM

Ho incontrato problemi di compatibilità con i modelli di elaborazione delle immagini in esecuzione su dispositivi edge, oltre a difficoltà legate all'integrazione con il display trasparente (come già descritto nello *studio di fattibilità*).

9.2 Memoria insufficiente di Arduino Nano

La memoria ridotta della scheda ha limitato la complessità dell'interfaccia utente che potevo sviluppare. Questo problema potrebbe essere in parte risolto distribuendo meglio i compiti tra la *ESP32-S3* e l'*Arduino Nano*.

9.3 OLED Display Trasparente

L'assenza di una documentazione estesa ha reso complesso sia l'utilizzo sia il collegamento del display. Inoltre, le sue particolari caratteristiche tecniche hanno richiesto numerosi tentativi di configurazione.

9.4 Problemi ottici con il display

Ho dovuto calibrare attentamente la distanza del display dall'occhio e regolare la dimensione dei caratteri, così da ottenere una leggibilità adeguata e ridurre l'affaticamento visivo.

10 Peculiarità

Il progetto ha una struttura molto modulare, che mi permette di sostituire facilmente le sue componenti principali.

- **Modelli di analisi**
  Posso sostituire il modello di elaborazione delle immagini per ottenere prestazioni migliori o per estrarre informazioni differenti in base all'applicazione.
- **Interfaccia utente**
  L'attuale interfaccia basata su display può essere sostituita con altri sistemi di output, come ad esempio un'interfaccia audio, utile per supportare utenti non vedenti nella navigazione dell'ambiente.

11 Conclusioni

Il progetto offre una base solida per dispositivi *AR* indossabili, con ampi margini di miglioramento per quanto riguarda portabilità e ottimizzazione hardware.

Possibili sviluppi futuri:

- Ridurre le dimensioni complessive mediante schede dedicate per la gestione dei pulsanti e la connessione del display
- Spostare l'elaborazione delle immagini aumentare la portabilità, ad esempio:
  - su smartphone
  - su dispositivi edge (utilizzando *Edge Impulse* o *TensorFlow Lite*)

12 Reference

1. connessione scehermo SPI
2. modelli elaborazione immagini edge
3. modello elaborazione immagini PC
4. comunicazione seriale
5. cablaggio schermo
6. utilizzo bitmap 1 2